STAR Analyst: Self-Tuning Alpha Research

William F. Shen*, **Alex Iacob***, **Zichen Zhang**, **Daoheng Wang**, **Wenyi Wang**, **Rui Liang**, **Yulong Zhang**, **Xinchi Qiu**, **Nicholas D. Lane**

University of Cambridge, Freeride AI
*Equal contribution.

Discovering effective alphas from noisy, non-stationary financial data remains a challenging open problem. Recent LLM-based alpha discovery systems improve search automation, but they still largely operate within human-specified research protocols and the evolving object is typically the alpha formulas or mining trajectory, rather than the researcher itself. To address these gaps, we introduce STAR, the first self-tuning alpha researcher that elevates the object of evolution from individual alpha formulas to the alpha researcher itself. STAR couples an LLM-based alpha researcher with a metacognitive module that can revise the research scaffold, while fixed and isolated validation/test data and a host-side evaluator preserve the integrity of the utility channel, and the search policy prioritizes lineages with stronger descendants. Under a leakage-controlled forward protocol on the most recent 2026Q1 deployment window on CSI300, evolved STAR researchers generate substantially higher quality alphas than the base model, shifting median IR from **1.6** to **3.9**. It also significantly outperforms a comprehensive set of strong baselines in downstream performance, with **17**% quarterly excess return, IR of **1.54**, and maximum drawdown **below 3**%. Qualitative analysis further indicates that these gains arise from genuine researcher evolution, including a **6.9**× expansion of operator vocabulary, new tools, and improved research protocols. These results highlight the effectiveness of self-evolving agents in improving capabilities in noisy, weak-supervision domains, and position them as a promising pathway toward building more capable financial researchers.

Correspondence: fs604@cam.ac.uk, xq227@cam.ac.uk

1 Introduction

Alpha discovery, the search for predictive, economically grounded, and interpretable signals of future asset returns, lies at the heart of quantitative investing. High-quality signal libraries not only explain substantial variation in asset returns, but also shape academic asset-pricing theory and real-world portfolio construction. As a result, investors who identify robust signals early and deploy them effectively can earn meaningful risk-adjusted returns.

Historically, alpha discovery has evolved through several paradigms. Early work relied primarily on human expertise and hypothesis-driven factor design, including economic factor models, accounting characteristics, technical rules, and manually engineered signal libraries (Sharpe, 1964; Fama and French, 1993; Jegadeesh and Titman, 1993; Carhart, 1997; Kakushadze, 2016). Attempts at automation then turned to heuristic search, symbolic regression, and genetic programming to explore interpretable rule spaces directly (Neely et al., 1997; Allen and Karjalainen, 1999; Zhang et al., 2020; Cui et al., 2021). Additionally, machine learning methods showed that nonlinear predictor interactions can materially improve return forecasting, while symbolic and formulaic approaches sought to preserve interpretability by searching directly over compositional expressions (Gu et al., 2020; Xu et al., 2021; Duan et al., 2022; Zhang et al., 2020; Cui et al., 2021; Yu et al., 2023; Ren et al., 2024; Zhao et al., 2025).

More recently, the emergence of large language models (LLMs) has provided an alternative approach to this problem. Rather than searching only over symbolic trees or opaque function approximators, LLM-based systems can formulate qualitative market hypotheses, translate textual and domain knowledge into executable factor expressions, and revise candidate signals using code-level and backtest feedback (Li et al., 2024; Wang et al., 2023). This shift has been further accelerated by increasingly autonomous agentic systems

for quantitative research (Tang et al., 2025; Shi et al., 2026b; Li et al., 2025; Wang et al., 2026; Han et al., 2026; Guo et al., 2026). Collectively, these studies suggest that LLMs are useful not merely as natural-language interfaces, but as components of end-to-end research systems that operate across the full cycle of factor ideation, implementation, evaluation, and iterative refinement.

Despite this progress, most existing alpha discovery systems still operate within a predominantly human-specified research protocol. The object of evolution is typically limited to alpha factors or their trajectories, while the surrounding scaffold, including the agent prompt, mutation or recombination rules, quality checks, filtering criteria, and overall workflow, is fixed *ex ante*. The underlying operator library is likewise usually fixed or expanded only through manual intervention during search. In other words, prior systems aim to improve outcome within a pre-determined agentic scaffold rather than improving the scaffold itself. As a result, these systems more closely resemble highly capable assistants operating inside a fixed laboratory protocol than autonomous researchers capable of expanding their own methodology.

Recent work on recursive self-improvement makes this limitation explicit. Self-improving agents inspired by Darwinian (DGM (Zhang et al., 2025)) and Huxleyan (HGM (Wang et al., 2025)) views of evolution have been shown to discover stronger harnesses to perform coding tasks. Hyperagents (Zhang et al., 2026) argued that sustained progress may require the meta-level mechanism that generates future modifications to be editable as well. This distinction is especially important for alpha discovery: if the agent cannot expand its own toolbox, revise its own search policy, or alter the mathematical ingredients from which factors are composed, then its autonomy remains fundamentally bounded by human design.

Motivated by this gap, we present STAR, the first *self-tuning alpha researcher*. The key idea is to extend the target of evolution from individual factors to the research process itself. Rather than evolving only candidate alphas, STAR allows the agent to revise the code and procedures that govern future discovery, including search, retrieval, memory, tool use, operator construction, and the meta-level mechanism that converts past evidence into researcher improvements. This enables the system to improve not only *what* factors it discovers, but also *how* it discovers them.

We evaluate STAR under a leakage-controlled forward protocol designed to test deployable alpha discovery. Rather than using prolonged test windows that can smooth over heterogeneous regimes, our setting asks a stricter and more realistic question: can a researcher use a recent data panel to generate alphas that transfer to the immediately subsequent deployment window? We empirically show that, with the foundation model fixed, evolved STAR researchers substantially improve generated-alpha quality relative to the base model, shifting the held-out IR distribution upward: median IR increases from approximately $\mathbf{1.6}$ to $\mathbf{3.9}$, with all paired seed comparisons positive on the CSI300 2026Q1 forward test. On the same held-out period, STAR also achieves the strongest reported IR, excess return, and drawdown profile among a comprehensive set of baselines.

Qualitative analysis further indicates that these gains come from genuine researcher evolution rather than shallow node selection. The strongest policies emerge deep within productive clades of the search tree, supporting the role of clade-level credit assignment. The evolved researcher also expands its operator vocabulary by approximately $\mathbf{6.9}\times$ within the search budget and introduces new protocols and tools for memory retrieval, candidate validation, mechanism balancing, and failure-aware generation. Together, these results show that self-evolving agents are not confined to coding and mathematics: they can improve capability in noisy, weak-supervision domains including where LLMs' ability to discover predictive signal from tabular market data remains far less established (Tang et al., 2025; Shi et al., 2026a; Luo et al., 2026; Ding et al., 2025). STAR also highlights self-evolving frameworks as a promising pathway toward building more capable financial researchers.

2 Related work

Automated alpha research. Automated alpha discovery has progressed from hand-crafted factor libraries and manually designed formulaic signals (Kakushadze, 2016; Tulchinsky, 2019) to programmatic search over interpretable expressions (Zhang et al., 2020; Cui et al., 2021), sequential symbolic decision making (Yu et al., 2023; Ren et al., 2024), and dynamic factor combination or structure-aware exploration (Shi et al., 2025;

Chen et al., 2025). More recently, LLM-based systems have shifted the interface toward language-guided search: AlphaAgent regularizes LLM exploration for novelty, complexity, and hypothesis alignment (Tang et al., 2025), while other systems combine LLM reasoning with dual-chain refinement, full-stack research loops, or code-based evolution (Shi et al., 2026b; Cao et al., 2025; Li et al., 2025; Liu et al., 2025). Closely related work further introduces modular skills, trajectory evolution, graph-structured factor pools, and reproducible constrained generation (Wang et al., 2026; Han et al., 2026; Guo et al., 2026; Shi et al., 2026a). Across these systems, however, the outer research protocol, such as mutation and crossover rules, memory updates, filtering mechanism, is largely specified by hand before the search. Relative to prior alpha agents, our contribution is therefore not only another alpha generation scaffold, but an evolutionary protocol that allows the research scaffold to revise how future alphas are generated.

Self-improving agents. STAR also relates to self-improving program search. SWE-agent and ADAS show that LLMs can operate and modify software workflows, while FunSearch and AlphaEvolve demonstrate program discovery under external evaluator loops in mathematics, coding, and algorithmic design (Yang et al., 2024; Hu et al., 2024; Romera-Paredes et al., 2024; Novikov et al., 2025). More directly, the Darwin–Gödel Machine maintains an archive of self-modified coding agents and accepts descendants according to empirical performance, while the Huxley–Gödel Machine allocates expansion according to clade-level metaproductivity (Zhang et al., 2025; Wang et al., 2025). HyperAgents further argues that the mechanism generating future modifications should itself be editable (Zhang et al., 2026). In contrast to these primarily coding- or math-oriented settings, STAR studies self-improvement in financial research, where supervision is noisy, delayed, regime-dependent, and vulnerable to leakage or adaptive overfitting. Moreover, the ability of LLM agents to discover predictive signal from tabular market data itself is less established than their coding or mathematical reasoning capabilities. STAR broadens self-improving agents beyond coding and math, showing that it can operate in noisy, weak-supervision domains such as financial research.

3 Method

We introduce STAR, a self-tuning alpha research framework in which an LLM-based alpha researcher is iteratively improved by a metacognitive module. Unlike prior alpha-mining systems that evolve candidate factors inside a fixed, human-specified workflow, STAR treats large parts of the research process itself, such as prompting, memory construction, tool use, operator exposure, and search protocol, as editable program state, while keeping the evaluator and the held-out data boundary fixed and isolated to preserve integrity. The object of search is therefore not merely the alpha formula, but the alpha researcher itself. This asymmetry is the central design principle: in finance, where supervision is scalar, noisy, and easy to overfit, open-ended self-improvement is credible only when the measurement channel is less mutable than the search process it supervises.

3.1 Researcher

STAR evolves *programmatic financial researchers*: task-level agents that translate financial hypotheses into executable, machine-evaluable candidate signals. This abstraction is not tied to a particular data modality or asset class. A researcher may operate on OHLCV $\rightarrow$ (`open`, `high`, `low`, `close`, `volume`) panels, fundamentals, sentiment, order-book states, cross-asset panels, or other financial inputs, as long as its outputs conform to a fixed evaluation interface. In this paper, we instantiate this abstraction in a stock OHLCV-based alpha research setting, which provides a concrete and reproducible testbed for studying self-tuning researcher evolution.

During evolution, STAR maintains an archive tree of researcher snapshots. Each archive node a corresponds to a frozen researcher state. The researcher at node a, denoted A_a, is defined by the fixed input/output research interface, the frozen repository state R_a, and the available tool surface $\mathcal{T}_a$. Given search data D_{tr}, A_a performs end-to-end factor research and outputs a bounded set of candidate alpha programs:

$$\mathcal{C}_a = A_a(D_{\text{tr}}; R_a, \mathcal{T}_a) = \{\alpha_{a,1}, \ldots, \alpha_{a,m_a}\},$$

where each $\alpha_{a,j}$ is an executable signal generator that conforms to the standardized interface and can be evaluated by the host evaluator. In our experiments, these candidates are Python modules that map a daily

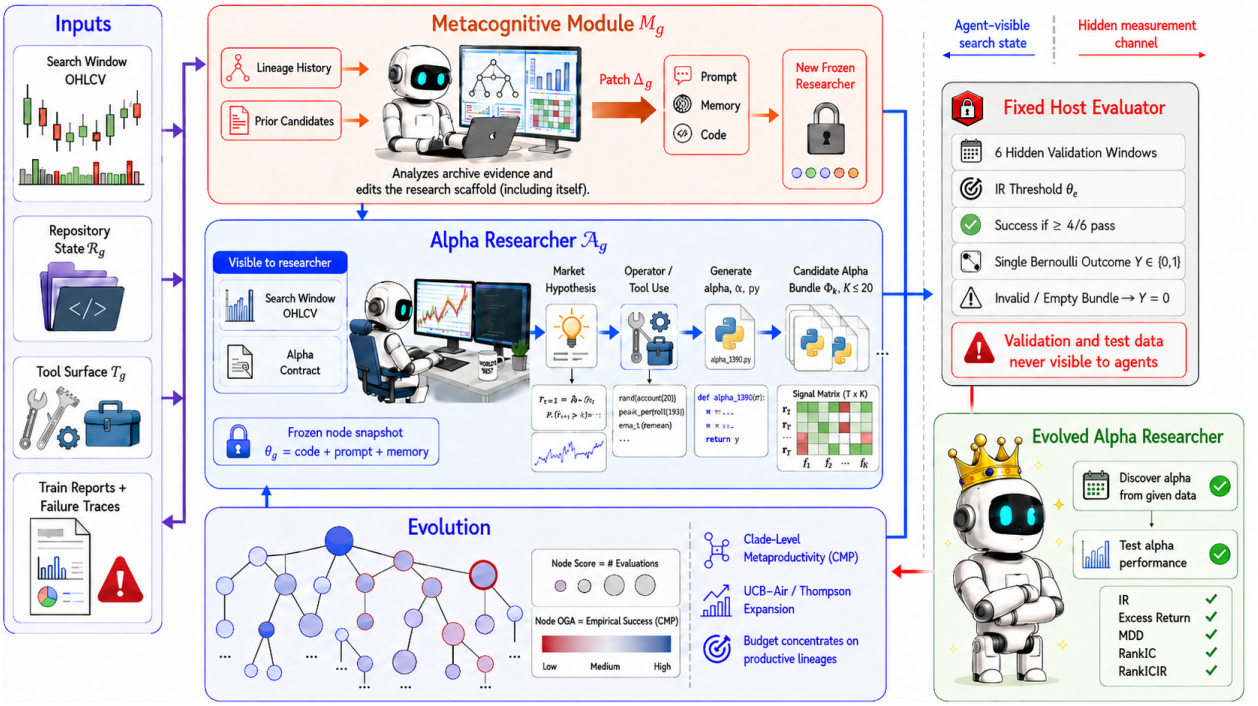


Figure 1 Overview of STAR. STAR evolves an alpha researcher rather than only individual alpha formulas. At each archive node, a frozen researcher snapshot (consists of $(\texttt{code}_a, \texttt{prompt}_a, \texttt{memory}_a)$) , receives search-window OHLCV data and alpha contract, then emits candidate alpha programs. A fixed host evaluator scores bounded candidate bundles on hidden validation windows and converts each admitted call into a single binary success observation. A metacognitive module reads only train-side lineage information and reports, then patches the state of the researcher or itself to create a child node. Clade-level metaproductivity allocates evaluation and modification budget toward lineages whose descendants repeatedly produce better alphas. Final evolved alpha research policies at budget exhaustion obtain stronger capability in generating high quality alphas.

OHLCV panel to a $T \times N$ signal matrix, accompanied by the corresponding market hypothesis and working traces. The repository state R_a contains the task prompt, memory system, and alpha contract that determine how financial hypotheses are translated into executable alphas. Thus, while our empirical instantiation focuses on stock-OHLCV alphas, the conceptual object being evolved is broader: STAR provides a recipe for evolving programmatic financial researchers under any setting where candidate research outputs can be expressed, executed, and scored through a fixed interface.

3.2 Meta-cognition

We augment the researcher with a metacognitive module, which generates higher-order revisions that create descendant archive nodes. To create a child node a from parent $p(a)$, the metacognitive module consumes a sanitized meta-view $H_{<a}$ of the archive, including train-side lineage information, prior alpha candidates, train-side reports, and failure traces, and emits a patch Δ_a:

$$\Delta_a = M_{p(a)}(H_{<a}; R_{p(a)}), \qquad R_a = \Delta_a(R_{p(a)}).$$

The child node a then freezes its own repository state R_a at creation.

Following (Zhang et al., 2026), the patch is intentionally broad. It may revise the entire workspace of the alpha researcher and the metacognitive module itself, saving only for the evaluator (including metric definition) and the held-out data boundary which remain outside the editable surface to maintain rigorous of the evaluation pipeline. This design follows the central intuition that long-horizon self-improvement requires not only better task outputs, but also the ability to improve the mechanism that generates future improvements. For alpha

discovery, the bottleneck is often procedural rather than purely combinatorial. For instance, fixed workflow may repeatedly rediscover crowded motifs, under-explore useful operator families, or reject promising candidates because of brittle novelty heuristics (Tang et al., 2025; Wang et al., 2026). Metacognition gives the system a way to discover unintended research habits and revise them accordingly without pre-defined scope or human intervention, shifting the evolutionary object from the alpha formula alone to include the research policy that produces it.

3.3 The Evolution

Conceptually, financial researcher together with its metacognitive module is a Gödel-machine-style self-referential program: a Turing-complete process that may rewrite parts of its own code in pursuit of a fixed utility, and that, in the idealized setting, executes only those self-rewrites whose value can be *proved* under a fixed axiom system (Schmidhuber, 2003). Two recent lines of work relax this proof requirement to make the schema operational. The Darwin–Gödel Machine (DGM) (Zhang et al., 2025) replaces formal proofs with empirical performance improvements: it maintains an archive of agent variants, samples parents to mutate, and accepts descendants based on downstream benchmark performance. The Huxley–Gödel Machine (HGM) (Wang et al., 2025) further sharpens DGM's exploration by recognizing that immediate benchmark score may not be a reliable predictor of long-run improvement value, and therefore expands lineages whose *clade-level metaproductivity*, the probability that the subtree of descendants will eventually contain a strong agent, is high (see Appendix A).

HGM, however, rests on four core assumptions (Wang et al., 2025) (details in Appendix A). Taken literally, these assumptions do not hold in alpha discovery. Financial returns are non-stationary across calendar regimes. Validation rewards are scalar and noisy. Backtests can be expensive rather than cost-free proof certificates. Additionally, an LLM-generated researcher may occasionally fail to compile or emit alpha that strictly adhere to the formatting contract.

We therefore adapt the schema by intervening on each challenging point. Most importantly, we do not assume that markets are stationary, but, instead, make the *search interface* stationary by freezing the researcher, the data distribution, and the evaluator used to define one atomic trial. The HGM assumptions then hold for the induced Bernoulli process observed by the scheduler.

Specifically, each archive node a is created with an immutable researcher snapshot

$$\sigma_a = (\texttt{code}_a, \texttt{prompt}_a, \texttt{memory}_a).$$

Search-evaluation outcomes are never written back into σ_a. The host also pins all non-editable components of the evaluation environment, including the evaluator image, package versions, data hashes, tradability mask, inference configuration, and decoding parameters. Thus, repeated evaluations of the same node call the same researcher distribution under the same evaluator, rather than a researcher whose prompt, memory, dependencies, or validation rule have drifted over time.

Then, we define an atomic evaluation of node a as follows. The host first sample a search window w_{tr} and pairwise disjoint validation windows $\mathbf{w}_k = \{(w_{1,k}, \ldots, w_{6,k})\}_{k=1}^{K_{\max}}$ from the fixed search pool. Only w_{tr} is exposed to the researcher for within-call discovery, while six validation windows are sampled independently from the same distribution and are hidden to agents. The frozen researcher then emits a bounded candidate bundle $\Phi_a(\xi) = \{\phi_k\}_{k=1}^{K}$, with $0 \le K \le K_{\max} = 20$. The host evaluates each valid ϕ_k on its corresponding hidden validation tuple. Invalid code, failed compilation, malformed output, or an empty alpha bundle is represented by $K = 0$. For a candidate ϕ_k, define

$$g_\ell(\phi_k, \mathbf{w}_k) = \mathbb{1}\left[\sum_{j=1}^{6} \mathbb{1}\left[\text{IR}_{w_{j,k}}(\phi_k) > \theta_\ell\right] \ge 4\right],$$

where $\text{IR}_{w_{j,k}}(\phi_k)$ is the information ratio (IR)[1] on validation window $w_{j,k}$, and θ_ℓ is the level-specific success threshold. Because IR distributions can vary substantially across models and datasets, a fixed success

[1]IR is the natural objective for training an alpha researcher because the economic target is persistent risk-adjusted active return, not raw return. Raw returns can be increased by loading on leverage. IR instead measures expected active return per unit of active risk, aligning the search criterion with the value of a deployable alpha signal (Sharpe, 1966; Grinold and Kahn, 2000).

threshold may either be too permissive, yielding little selection signal, or too stringent, causing nearly all candidates to fail. We therefore evaluate candidates across curriculum levels ℓ, each with a level-specific threshold θ_ℓ, allowing the search to adaptively balance exploration and exploitation. A candidate is successful if it clears the threshold on at least four of the six windows.

The full researcher call is then collapsed to a single Bernoulli outcome,

$$Y_a^{(\ell)} = \mathbb{1}\left[\max_{1\le k\le K} g_\ell(\phi_k, \mathbf{w}_k) = 1\right],$$

with the convention that the maximum over an empty bundle is zero. Thus every call, including failed compiles and empty bundles, produces exactly one binary outcome.

In aggregate, the system satisfies the four HGM assumptions. First, the search policy is evaluated only through the researcher returned at budget exhaustion, and hidden held-out test data are not available to the search procedure. Second, each node's repository snapshot is frozen at creation time. Hence, for a fixed curriculum level ℓ, selecting node a induces a fixed host-side measurement kernel. Conditional on the non-anticipating scheduler selecting (a, ℓ), each admitted atomic evaluation draws fresh host randomness and produces a Bernoulli outcome with success probability $p_a^{(\ell)}$. The resulting active outcomes therefore contribute the standard Bernoulli likelihood under adaptive scheduling (Appendix B.1). When the curriculum advances, previous threshold-dependent utility evidence is discarded to avoid polluting the HGM search process. Third, the system maintains a strict separation between research context and utility measurement. Hidden validation measurements, held-out test data, posterior counts, clade accumulators, curriculum-trigger statistics, and final selection scores are not exposed to the researcher or metacognitive module. Training-side evidence and sanitized failure traces may influence future search only by shaping the creation of descendant repository snapshots; they are not used directly as utility observations in the evaluator, CMP posterior, curriculum trigger, or final selection score. Fourth, STAR implements an HGM-compatible action accounting interface using typed host-side ledgers. Each admitted search-time evaluation consumes one unit of evaluation budget and contributes exactly one binary outcome to the active posterior. Each admitted modification attempt, including an invalid or failed patch, consumes one unit of modification budget and may be retained only as a sanitized failure trace. Thus, no self-modification attempt receives a free retry – it is charged in the overall action accounting even though it does not create a node-level posterior observation. Lastly, admitted candidates that fail during measurement are recorded as $Y = 0$. The bounded bundle size $K_{\max}$ prevents unbounded within-call search from being hidden inside a single budget unit, and single-bit aggregation prevents a node from receiving multiple posterior observations from one atomic evaluation. Thus, the cost model exposed to the system matches the actual search interface.

4 Experiments

We present the method of STAR, a self-improving alpha-discovery system, in §3. We evaluate its effectiveness under a leakage-controlled financial supervision protocol, and attempt to answer the following research questions:

- **RQ1: Policy improvement.** Can alpha research policies evolved under STAR generate higher quality alphas than the base model?
- **RQ2: Downstream performance.** Do alphas discovered by evolved research policies yield stronger cost-aware held-out portfolio performance than baseline-derived alphas?
- **RQ3: Lineage-level search dynamics.** Does STAR allocate budget toward productive clades, and do strong researchers emerge through multi-step refinement rather than lucky shallow nodes?
- **RQ4: Anatomy of capability gain.** How does STAR reshape the researcher's internal scaffold to support stronger alpha discovery?

4.1 Setup

Dataset and Model We evaluate on CSI300, a point-in-time universe covering the liquid large-cap segment of the China A-share market. We choose this setting because alphas in canonical markets such as the S&P 500 have been found to be more heavily arbitraged and fragile (Fama and French, 2010; Chordia et al., 2014; Hou et al., 2017). In contrast, CSI300 has a distinct market structure, higher retail participation, and stronger regime- and sentiment-driven effects (Qiu et al., 2023; Han and Shi, 2022; Hou et al., 2023), and therefore provides a liquid but less saturated testbed for assessing whether the researcher can discover robust signals.

Existing LLM-based alpha discovery methods typically mine signals from and evaluate their returns over long historical windows (Tang et al., 2025; Han et al., 2026; Liu et al., 2025). However, multi-year test windows can average out heterogeneous regimes and mask regime-conditional failures (Welch and Goyal, 2008; OrientSecurities, 2026), amplify data-snooping bias in alpha discovery (Sullivan et al., 1999), and reward slow regime reversion rather than signals that remain actionable immediately after discovery (Tashman, 2000; Giacomini and White, 2006). The setting is also inconsistent with evidence that alpha information decays over sub-year horizons and with industry practice of frequent, often quarterly, reassessment and rebalancing (Di Mascio et al., 2017; Flint and Vermaak, 2023; Zhou and Lin, 2017; Shrivastava et al., 2018). Additionally and critically, if the evaluation period overlaps with data that may have appeared in the model's pretraining corpus, performance can be confounded by temporal leakage rather than genuine forward discovery.

To mitigate these issues, the alpha research policy is evolved using the 2019–2024 search and validation pool, which provides cross-regime breadth across COVID stress, reopening, policy-driven rotations, liquidity cycles, and factor/style reversals. After evolution, we freeze the researcher and give it only 2025 data to generate and select deployable alphas, which are then evaluated on the strictly disjoint 2026Q1 window. This protocol mirrors the practical setting in which an alpha researcher is trained on past regimes, deployed using the most recent data, and judged on the next short forward period. Finally, we consistently use GPT-5.1 for `STAR` and all LLM-based alpha discovery baselines, and its September 30, 2024 knowledge cutoff precedes our 2025–2026Q1 evaluation period (OpenAI, 2025). The final forward test mitigates the risk that the LLM is exploiting memorized market outcomes rather than discovering signals from the provided data. More details in Appendix C.1.

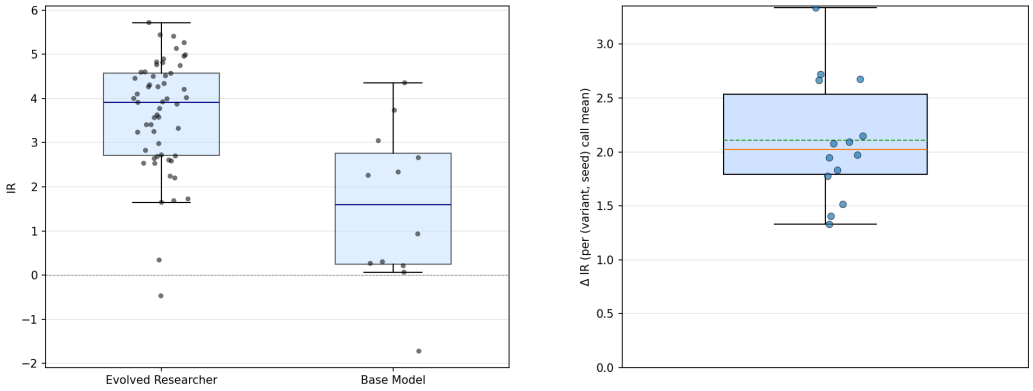
Metrics. We report the information ratio (IR) of the net-of-cost long-only TopK-Dropout sleeve as the primary metric, since it directly matches our objective of evolving an alpha researcher that discovers signals with stronger deployable portfolio performance. We also report strategy-level metrics, including benchmark excess return (ER) and maximum drawdown (MDD). Additionally, we include RankIC and RankICIR as secondary factor-level diagnostics against the same 10-day open-to-open return target. These cross-sectional metrics capture a different notion of predictive quality from the portfolio-level IR and are therefore informative but not directly comparable to the optimized objective. Appendix C.2.

4.2 Results

Improvement of alpha researcher. First, we evaluate whether the alpha researcher, augmented by the evolved scaffold, is better able to discover predictive signal from tabular market data.

Figures 2 isolate the contribution of the evolved alpha researcher state relative to base GPT-5.1 model. Figure 2a compares the per-alpha IR distribution, pooling generated alphas across variants and seeds. The evolved researcher shift the entire distribution upward: median IR increases from approximately 1.6 for the base model to 3.9 for the evolved researcher, representing a 2.4x increase. Additionally, the evolved interquartile range not only lies significantly above, but also more tightly than, the baseline distribution. This indicates that the evolved researcher does not merely produce a few lucky outliers, but improves the typical quality and consistency of generated alphas. Figure 2b confirms this effect under a paired design: for each matched seed pair, we subtract the mean IR of the corresponding no-scaffold baseline call from the mean IR of the evolved researcher call. All 15 valid paired differences are positive, with ΔIR ranging from approximately 1.3 to 3.3, a mean and median of approximately 2.1 and 2.0 respectively, and a one-sided Wilcoxon signed-rank of $p = 0.0001$.

With the foundation model fixed, the results show that alpha-discovery capability depends not only on the



(a) IR distribution. **(b)** Paired ΔIR.

Figure 2 Evolved alpha research policies versus the base model. Left: IR distribution of alphas generated by the top-5 evolved research policies compared with the base model across three seeds. **Right:** Paired IR improvement under matched seeds, computed as the mean IR of alphas generated by evolved policies minus the mean IR of alphas generated by the base model. Box edges indicate the 25th/75th percentiles, whiskers span the bulk of the data, and jittered dots show individual alphas or paired samples.

base model, but also on the researcher scaffold around it. STAR improves this scaffold through search-derived memory, operators, tools, and executable state, yielding stronger forward alphas than the base model alone.

Improvement of downstream alpha performance. We next conduct a comprehensive downstream evaluation of whether the discovered frontier improves final alpha quality. We compare STAR against four complementary baseline families spanning standard supervised predictors, deep learning models, public factor libraries, and recent LLM-based alpha-mining agents (Table 1). Specifically, the comparison includes: (1) machine learning models commonly used in quantitative finance: AdaBoost (Freund and Schapire, 1997), Random Forest (Breiman, 2001), XGBoost (Chen and Guestrin, 2016), LightGBM (Ke et al., 2017), and CatBoost (Prokhorenkova et al., 2018); (2) representative deep learning models: LSTM (Hochreiter and Schmidhuber, 1997), Transformer (Vaswani et al., 2017), and TRA (Lin et al., 2021), a finance-specific temporal routing model with mixture-style, state-conditioned heads over an LSTM backbone; (3) public alpha libraries: Alpha158 and Alpha360 (Han et al., 2026); and (4) recent LLM-based alpha discovery methods: AlphaAgent (Tang et al., 2025), QuantaAlpha (Han et al., 2026), and CogAlpha (Liu et al., 2025). This coverage is designed to test STAR against both conventional return-prediction pipelines and the most directly related agentic alpha discovery systems. Details of the baseline methods are provided in Appendix C.3.

As discussed in §4.1, prior LLM-based alpha discovery studies often generate, filter, and evaluate alphas over extended multi-year windows that average across many regimes and overlap with the base model's pretraining period. We instead use a stricter forward protocol: the LLM observes only the post-cutoff 2025 OHLCV panel for alpha discovery, and performance is evaluated on the disjoint 2026Q1 held-out window. This shorter forward horizon better reflects practical deployment and reassessment cycles in quantitative investment. For STAR and all LLM-based alpha-mining baselines, we use the same base model, GPT-5.1, to isolate the effect of the research scaffold rather than differences in model capability.

As shown in Table 1, alphas generated by STAR achieve the strongest held-out performance, with an IR of 1.54, compared with 1.31 for the closest baseline, LightGBM.[2] The advantage is also visible in realized excess return: STAR achieves 17% over the test period, whereas most baselines remain below 10%. Notably, existing LLM-based agentic baselines appear brittle under this regime-dependent forward test, likely because they mainly expand and filter candidate alpha pools using multi-year historical metrics. Such filtering can mask regime sensitivity and transfer less robustly to realistic short deployment and rebalancing windows. In

[2]As discussed in §4.1, IR is the primary search target as it matches the long-only TopK-Dropout strategy, while RankIC and RankICIR are full-cross-sectional diagnostics. The two need not align: RankIC can be positive without improving the traded top-K sleeve, and weak full-universe RankIC can still coincide with strong localized top-K performance. We therefore treat IR, ER, and MDD as primary metrics, with RankIC and RankICIR reported as secondary diagnostics.

Table 1 CSI300 2026Q1 held-out results. Forward performance comparing STAR with various baseline methods.

Category	Method	IR	ER	MDD	RankIC	RankICIR
Machine Learning	XGBoost	0.985	0.084	0.082	0.003	0.017
	LightGBM	1.306	0.128	0.049	0.011	0.086
	RandomForest	0.845	0.076	0.057	−0.056	−0.339
	CatBoost	1.001	0.091	0.073	0.004	0.028
	AdaBoost	1.134	0.084	0.031	−0.043	−0.252
Deep Learning	LSTM	0.711	0.089	0.044	−0.069	−0.422
	Transformer	0.358	0.052	0.064	−0.061	−0.329
	TRA	0.374	0.054	0.052	−0.068	−0.323
Alpha Library	Alpha158	0.432	0.032	0.046	−0.105	−0.447
	Alpha360	0.348	0.017	0.029	−0.044	−0.261
LLM-based Agent	AlphaAgent	−0.293	−0.015	0.030	0.053	**0.404**
	QuantaAlpha	−0.023	−0.005	0.030	0.040	0.257
	CogAlpha	−0.201	−0.009	0.040	−0.002	−0.012
Self Evolving Agent	**STAR**	**1.544**	**0.170**	**0.029**	**0.065**	0.359

contrast, STAR evolves the alpha researcher itself, producing a search-derived scaffold rather than relying on a fixed filtering rule over output alphas.

Anatomy of the search tree To examine both the effectiveness of clade-level metaproductivity with Thompson sampling and the possibility that strong evolved researchers arise from a few lucky shallow nodes, we analyze how STAR traverses the search tree during evolution.

Figure 3 visualizes the search archive at budget exhaustion and reveals three qualitative patterns. First, the top mature alpha researcher, marked by red borders, appear at substantial depth rather than near the root, suggesting that strong researchers emerge through multi-step metacognitive refinement. Second, node area shows that evaluations are concentrated in a small number of productive clades instead of being spread uniformly across the tree. This is consistent with the lineage-level credit assignment in §3.3, which allocates budget to branches whose descendants repeatedly produce positive outcomes. Third, node color varies substantially across nearby nodes, indicating heterogeneous researcher quality across the search frontier. Together, these patterns suggest that STAR is not simply selecting a lucky shallow prompt, but progressively directing compute toward lineages that produce stronger mature policies.

Anatomy of capability gain We next inspect the internal changes in the evolved policy to understand how they support the improved alpha generation capability and downstream performance observed above.

Figure 4 quantifies one aspect of this process by tracking the operator vocabulary invoked by generated alphas. We define an operator as either a DSL primitive or a helper symbol imported into the alpha namespace, and measure the cumulative union of operators used by any alpha emitted up to each node. Starting from a compact 21-operator seed at the root, the vocabulary expands rapidly during the early exploratory phase and ultimately reaches 144 operators over the full run. This indicates that the search does not collapse into repeated use of a fixed hand-written factor template; instead, it continually broadens the constructive language available for alpha generation, including ranking transformations, rolling statistics, masking logic, normalization routines, and reusable helper abstractions.

The evolution also changes the research protocol itself. Along the productive trajectory, the researcher adds routines for retrieving and summarizing strong alphas, fragile passers, novel but underperforming candidates, non-novel variants, and overused operators from memory. It develops a keyword-based taxonomy of alpha families spanning multiple mechanism classes, such as orderly drift, sponsorship/participation, session pressure, compression, and gap/shock effects. This taxonomy is then used to structure each alpha-generation batch: some slots exploit mechanisms that previously worked, while others explore under-sampled or orthogonal families. The result is a more deliberate exploration–exploitation protocol than

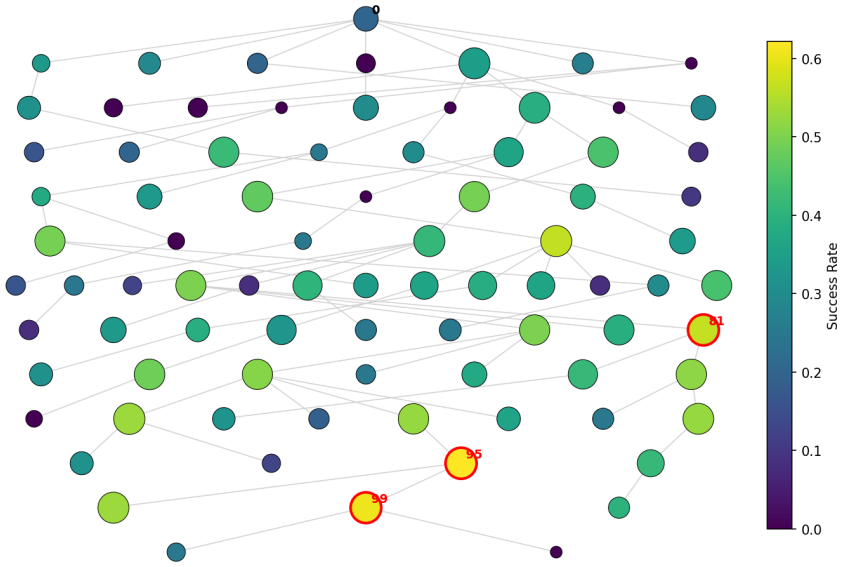


Figure 3 STAR search tree illustraions. Each node is a frozen researcher snapshot. Edges denote metacognitive parent–child patches. Node color encodes empirical success rate at the terminal curriculum level, and node area scales with the number of validation trials, so heavily evaluated lineages appear larger. Red borders mark the top-performing evolved alpha research policies. The top researchers emerge deep in the tree, while evaluation budget concentrates in a small number of productive clades, illustrating lineage-level rather than purely node-local credit assignment.

one-shot generation. In addition, the researcher adds validation routines that check candidate format and contract compliance, reducing uninformative failures caused by malformed code or invalid alpha outputs.

The metacognitive module evolves in parallel. As evaluation reports and lineage histories grow longer, prompt-only inspection becomes increasingly brittle. The module therefore introduces tools to parse lineage reports, inspect novelty, summarize parent-specific failure modes, and prune invalid or obsolete memory entries. These changes turn the meta-loop from a generic patch generator into a more systematic mechanism for reading accumulated evidence and converting it into targeted revisions of the descendant researcher.

Overall, the evolved policy is no longer a one-shot alpha generator. It becomes a closed-loop alpha-construction system that reads memory, conditions search on lineage-specific failures, allocates mechanism slots, audits generated code, repairs invalid batches, and expands the operator vocabulary available to future generations. This open-ended scaffold evolution is central to STAR: it produces research policy improvements that fixed, non-evolving alpha search procedures cannot access, and helps explain why the evolved researcher scaffold yields stronger forward alpha discovery.

5 Conclusion

We introduced STAR, a framework for self-tuning alpha research policies rather than only individual alpha formulas. STAR couples an LLM-based alpha researcher with a metacognitive module that can freely revise the system, while keeping the validation/test data, and host-side evaluator fixed and isolated to preserve the integrity of the utility channel. Search is guided by clade-level metaproductivity, which allocates evaluation and modification budget toward lineages whose descendants repeatedly produce stronger alphas.

Under a leakage-controlled forward protocol, evolved STAR researchers generate substantially higher-quality alphas than the base model and achieve stronger downstream performance than a comprehensive set of baselines. Qualitative analysis further indicates that these gains arise from genuine researcher evolution, including an expanded operator vocabulary, new tools, and improved research protocols. More broadly, STAR shows that self-improving LLM agents can operate in domains where supervision is noisy and weak. It also

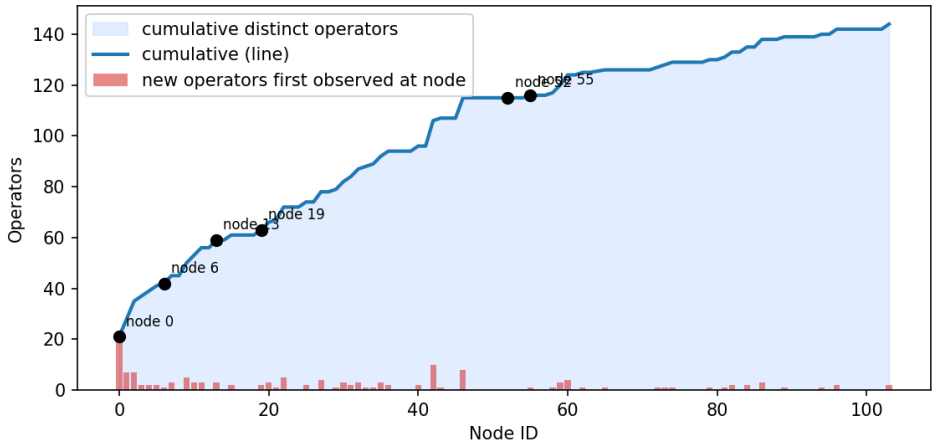


Figure 4 Operator vocabulary growth during evolution. Nodes are ordered by creation time. The blue curve shows the cumulative number of distinct operators used by generated alphas; red bars mark first appearances; black markers denote the productive backbone.

highlights self-evolving scaffolds as a critical pathway toward building more capable financial researchers.

References

Franklin Allen and Risto Karjalainen. Using genetic algorithms to find technical trading rules. *Journal of Financial Economics*, 51(2):245–271, 1999.

Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

Lang Cao, Zekun Xi, Long Liao, Ziwei Yang, and Zheng Cao. Chain-of-alpha: Unleashing the power of large language models for alpha mining in quantitative trading. *arXiv preprint arXiv:2508.06312*, 2025.

Mark M. Carhart. On persistence in mutual fund performance. *The Journal of Finance*, 52(1):57–82, 1997.

Binqi Chen, Hongjun Ding, Ning Shen, Jinsheng Huang, Taian Guo, Luchen Liu, and Ming Zhang. Alphasage: Structure-aware alpha mining via gflownets for robust exploration. *arXiv preprint arXiv:2509.25055*, 2025.

Tianqi Chen and Carlos Guestrin. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794, 2016.

Tarun Chordia, Avanidhar Subrahmanyam, and Qing Tong. Have capital market anomalies attenuated in the recent era of high liquidity and trading activity? *Journal of Accounting and Economics*, 58(1):41–58, 2014.

Can Cui, Wei Wang, Meihui Zhang, Gang Chen, Zhenjie Luo, and Beng Chin Ooi. Alphaevolve: A learning framework to discover novel alphas in quantitative investment. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2021.

Rick Di Mascio, Anton Lines, and Narayan Y Naik. Alpha decay and institutional trading. *Available at SSRN 2580551*, 2017.

Hongjun Ding, Binqi Chen, Jinsheng Huang, Taian Guo, Zhengyang Mao, Guoyi Shao, Lutong Zou, Luchen Liu, and Ming Zhang. Alphaeval: A comprehensive and efficient evaluation framework for formula alpha mining. *arXiv preprint arXiv:2508.13174*, 2025.

Yitong Duan, Lei Wang, Qizhong Zhang, and Jian Li. Factorvae: A probabilistic dynamic factor model based on variational autoencoder for predicting cross-sectional stock returns. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 4468–4476, 2022. doi: 10.1609/aaai.v36i4.20369.

Eugene F. Fama and Kenneth R. French. Common risk factors in the returns on stocks and bonds. *Journal of Financial Economics*, 33(1):3–56, 1993.

Eugene F Fama and Kenneth R French. Luck versus skill in the cross-section of mutual fund returns. *The journal of finance*, 65(5):1915–1947, 2010.

Emlyn Flint and Rademeyer Vermaak. Factor information decay: A global study. *Journal of Portfolio Management*, 49(2), 2023.

Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

Raffaella Giacomini and Halbert White. Tests of conditional predictive ability. *Econometrica*, 74(6):1545–1578, 2006.

Richard C Grinold and Ronald N Kahn. *Active portfolio management*. McGraw Hill New York, 2000.

Shihao Gu, Bryan Kelly, and Dacheng Xiu. Empirical asset pricing via machine learning. *The Review of Financial Studies*, 33:2223–2273, 2020.

Taian Guo, Haiyang Shen, Junyu Luo, Binqi Chen, Hongjun Ding, Jinsheng Huang, Luchen Liu, Yun Ma, and Ming Zhang. Alphaprobe: Alpha mining via principled retrieval and on-graph biased evolution. *arXiv preprint arXiv:2602.11917*, 2026.

Chunmao Han and Yongdong Shi. Chinese stock anomalies and investor sentiment. *Pacific-Basin Finance Journal*, 73: 101739, 2022.

Jun Han, Shuo Zhang, Wei Li, Zhi Yang, Yifan Dong, Tu Hu, Jialuo Yuan, et al. Quantaalpha: An evolutionary framework for llm-driven alpha mining. *arXiv preprint arXiv:2602.07085*, 2026.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Kewei Hou, Chen Xue, and Lu Zhang. Replicating anomalies. Technical report, National Bureau of Economic Research, 2017.

Kewei Hou, Fang Qiao, and Xiaoyan Zhang. Finding anomalies in china. *Fisher College of Business Working Paper*, 2023-03: 002, 2023.

Shengran Hu, Chris Lu, and Jeff Clune. Automated design of agentic systems. *arXiv preprint*, 2024.

Huaan Securities. The alpha behind limit-ups: A systematic exploration and empirical study of first-limit strategy. Research Report, March 2026. In Chinese.

Narasimhan Jegadeesh and Sheridan Titman. Returns to buying winners and selling losers: Implications for stock market efficiency. *The Journal of Finance*, 48(1):65–91, 1993.

Zura Kakushadze. 101 formulaic alphas. *Wilmott*, 2016(84):72–81, 2016.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in neural information processing systems*, 30, 2017.

Yuante Li, Xu Yang, Xiao Yang, Minrui Xu, Xisen Wang, Weiqing Liu, and Jiang Bian. R&d-agent-quant: A multi-agent framework for data-centric factors and model joint optimization. *arXiv preprint arXiv:2505.15155*, 2025.

Zhiwei Li, Ran Song, Caihong Sun, Wei Xu, Zhengtao Yu, and Ji-Rong Wen. Can large language models mine interpretable financial factors more effectively? a neural-symbolic factor mining agent model. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 3891–3902, 2024. doi: 10.18653/v1/2024.findings-acl.233.

Hengxu Lin, Dong Zhou, Weiqing Liu, and Jiang Bian. Learning multiple stock trading patterns with temporal routing adaptor and optimal transport. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, KDD '21, pages 1017–1026. Association for Computing Machinery, 2021. doi: 10.1145/3447548.3467358.

Fengyuan Liu, Huang Yi, Sichun Luo, Yuqi Wang, Yazheng Yang, Xinye Li, Zefa Hu, Junlan Feng, and Qi Liu. Cognitive alpha mining via llm-driven code-based evolution. *arXiv preprint arXiv:2511.18850*, 2025.

Haochen Luo, Ho Tin Ko, Jiandong Chen, David Sun, Yuan Zhang, and Chen Liu. Alphabench: Benchmarking large language models in formulaic alpha factor mining. *International Conference on Learning Representations*, 2026.

Christopher Neely, Paul Weller, and Rob Dittmar. Is technical analysis in the foreign exchange market profitable? a genetic programming approach. *Journal of Financial and Quantitative Analysis*, 32(4):405–426, 1997.

Alexander Novikov, Ngân Vũ, Marvin Eisenberger, Emilien Dupont, Po-Sen Huang, Adam Zsolt Wagner, Sergey Shirobokov, et al. Alphaevolve: A coding agent for scientific and algorithmic discovery. Technical report, Google DeepMind, 2025. arXiv:2506.13131.

OpenAI OpenAI. Gpt-5.1: A smarter, more conversational chatgpt—openai, 2025.

OrientSecurities. Optimal factor timing in a high-dimensional setting (quantitative research reference series no.3), 2026-04-21, 2026.

Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. Catboost: unbiased boosting with categorical features. *Advances in neural information processing systems*, 31, 2018.

Jiayan Qiu, Wei Huang, and Ying Jiang. When do they trade? heterogeneous investors in china. *Finance Research Letters*, 54:103729, 2023.

Tao Ren, Ruihan Zhou, Jinyang Jiang, Jiafeng Liang, Qinghao Wang, and Yijie Peng. Riskminer: Discovering formulaic alphas via risk seeking monte carlo tree search. In *Proceedings of the 5th ACM International Conference on AI in Finance*, pages 752–760, 2024. doi: 10.1145/3677052.3698613.

Bernardino Romera-Paredes, Mohammadamin Barekatain, Alexander Novikov, Matej Balog, M. Pawan Kumar, Emilien Dupont, Francisco J. R. Ruiz, Jordan S. Ellenberg, Pengming Wang, Omar Fawzi, Pushmeet Kohli, and Alhussein Fawzi. Mathematical discoveries from program search with large language models. *Nature*, 625:468–475, 2024.

Jürgen Schmidhuber. Gödel machines: Self-referential universal problem solvers making provably optimal self-improvements. *arXiv preprint cs/0309048*, 2003.

William F. Sharpe. Capital asset prices: A theory of market equilibrium under conditions of risk. *The Journal of Finance*, 19 (3):425–442, 1964.

William F Sharpe. Mutual fund performance. *The Journal of business*, 39(1):119–138, 1966.

Hao Shi, Weili Song, Xinting Zhang, Jiahe Shi, Cuicui Luo, Xiang Ao, Hamid Arian, and Luis Angel Seco. Alphaforge: A framework to mine and dynamically combine formulaic alpha factors. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(12):12524–12532, 2025. doi: 10.1609/aaai.v39i12.33365.

Runze Shi, Shengyu Yan, Yuecheng Cai, and Chengxi Lv. Hubble: An llm-driven agentic framework for safe, diverse, and reproducible alpha factor discovery. *arXiv preprint arXiv:2604.09601*, 2026a.

Yu Shi, Yitong Duan, and Jian Li. Navigating the alpha jungle: An llm-powered mcts framework for formulaic alpha factor mining. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2026b.

Rohit Shrivastava, J Lee, and George D Mussalli. Factor investing in the china a-shares market: Revelations from a contextual alpha model. *White paper, PanAgora*, 2018.

Ryan Sullivan, Allan Timmermann, and Halbert White. Data-snooping, technical trading rule performance, and the bootstrap. *The journal of Finance*, 54(5):1647–1691, 1999.

Ziyi Tang, Zechuan Chen, Jiarui Yang, Jiayao Mai, Yongsen Zheng, Keze Wang, Jinrui Chen, et al. Alphaagent: Llm-driven alpha mining with regularized exploration to counteract alpha decay. *arXiv preprint arXiv:2502.16789*, 2025.

Leonard J Tashman. Out-of-sample tests of forecasting accuracy: an analysis and review. *International journal of forecasting*, 16(4):437–450, 2000.

Igor Tulchinsky. *Finding Alphas: A Quantitative Approach to Building Trading Strategies*. Wiley, 2 edition, 2019.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Saizhuo Wang, Hang Yuan, Leon Zhou, Lionel M. Ni, Heung-Yeung Shum, and Jian Guo. Alpha-gpt: Human-ai interactive alpha mining for quantitative investment. *arXiv preprint arXiv:2308.00016*, 2023.

Yanlong Wang, Jian Xu, Hongkang Zhang, Shao-Lun Huang, Danny Dongning Sun, and Xiao-Ping Zhang. Factorminer: A self-evolving agent with skills and experience memory for financial alpha discovery. *arXiv preprint arXiv:2602.14670*, 2026.

Yifei Wang, Sicheng Du, Yuxin Liu, Bingchen Zhao, Yang Liu, Ping Luo, Yuanchen Bei, Ting Hu, and Anima Anandkumar. Huxley-gödel machine: Human-level coding agent development by an approximation of the optimal self-improving machine. *arXiv preprint arXiv:2510.21614*, 2025.

Ivo Welch and Amit Goyal. A comprehensive look at the empirical performance of equity premium prediction. *The Review of Financial Studies*, 21(4):1455–1508, 2008.

Wentao Xu, Weiqing Liu, Lewen Wang, Yingce Xia, Jiang Bian, Jian Yin, and Tie-Yan Liu. Hist: A graph-based framework for stock trend forecasting via mining concept-oriented shared information. *arXiv preprint arXiv:2110.13716*, 2021.

John Yang, Carlos E. Jimenez, Alexander Wettig, Kilian Lieret, Shunyu Yao, Karthik Narasimhan, and Ofir Press. Swe-agent: Agent-computer interfaces enable automated software engineering. In *Advances in Neural Information Processing Systems*, 2024.

Shuo Yu, Hongyan Xue, Xiang Ao, Feiyang Pan, Jing He, Dandan Tu, and Qing He. Generating synergistic formulaic alpha collections via reinforcement learning. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2023.

Jenny Zhang, Shengran Hu, Chris Lu, Robert Lange, and Jeff Clune. Darwin gödel machine: Open-ended evolution of self-improving agents. *arXiv preprint arXiv:2505.22954*, 2025.

Jenny Zhang, Bingchen Zhao, Wannan Yang, Jakob Foerster, Jeff Clune, Minqi Jiang, Sam Devlin, and Tatiana Shavrina. Hyperagents. *arXiv preprint arXiv:2603.19461*, 2026.

Tianping Zhang, Yuanqi Li, Yifei Jin, and Jian Li. Autoalpha: An efficient hierarchical evolutionary algorithm for mining alpha factors in quantitative investment. *arXiv preprint arXiv:2002.08245*, 2020.

Junjie Zhao, Chengxi Zhang, Chenkai Wang, and Peng Yang. Learning from expert factors: Trajectory-level reward shaping for formulaic alpha mining. *arXiv preprint arXiv:2507.20263*, 2025.

Yi Zhou and Jianwu Lin. The alpha life cycle of quantitative strategy. In *2017 IEEE/SICE International Symposium on System Integration (SII)*, pages 53–59. IEEE, 2017.

Appendix

Table of Contents

A Preliminaries

We are inspired by and build on the Huxley–Gödel Machine (HGM) (Wang et al., 2025), which formalizes self-improvement as tree search guided by clade-level metaproductivity (CMP), and HyperAgents (Zhang et al., 2026), which introduces metacognitive self-modification. We now reintroduce the core mathematical machinery underlying the setup.

Following Wang et al. (2025), let $\mathcal{T}_t$ denote the archive tree after t host-admitted HGM actions, initialized at $\mathcal{T}_0 = \{a_0\}$. At each HGM action time, the scheduler chooses an action from

$$\mathcal{A}_t = \mathcal{M}_t \cup \mathcal{V}_t, \qquad \mathcal{M}_t = \{m_a : a \in \mathcal{T}_t\}, \qquad \mathcal{V}_t = \{v_a : a \in \mathcal{T}_t\}.$$

A modification action m_a attempts to create a self-modified child of a. If the attempted patch is invalid, the action returns $\perp$ and the tree is unchanged, but the attempt is still charged. An evaluation action v_a evaluates a and appends a binary outcome $Y \in \{0, 1\}$ to the active search record. After a budget of B_{HGM} admitted HGM actions, the policy returns

$$a_{\mathrm{final}} = \operatorname*{arg\,max}_{a \in \mathcal{T}_{B_{\mathrm{HGM}}}} \mathrm{Score}_\pi(a),$$

and optimizes

$$J(\pi) = \mathbb{E}[U(a_{\mathrm{final}})],$$

where $U(a)$ is the fixed utility of agent a under the active task distribution.

For the theoretical analysis of the Gödel Machine (Schmidhuber, 2003) applied to self-improving coding agents HGM, (Wang et al., 2025) made four main assumptions:

- **H1. Final-agent objective.** The objective depends only on the final returned agent. Intermediate measurements may guide search, but they are not themselves rewards in $J(\pi)$.
- **H2. Repeatable stationary trials.** For each fixed agent–task pair, the expected binary outcome is independent of evaluation time and prior search history.
- **H3. Free proof/accounting updates.** Proofs or posterior-accounting operations do not consume search budget. Costly evidence-generating acts are charged separately.
- **H4. Unit-cost self-modification.** Each admitted self-modification attempt has the same unit cost in the HGM action clock. Failed modifications do not receive free retries.

Clade-level metaproductivity. Let $\mathcal{C}_{\mathcal{T}}(a)$ be the clade rooted at a, i.e., a together with all descendants of a in tree $\mathcal{T}$. When the tree is clear from context, write $\mathcal{C}(a)$. HGM defines the clade-level metaproductivity of a under policy π as the expected utility of the best future agent in the clade after spending the remaining budget:

$$\mathrm{CMP}_\pi(\mathcal{T}, a) = \mathbb{E}_{\mathcal{T}_{B_{\mathrm{HGM}}} \sim p_\pi(\cdot|\mathcal{T},a)} \left[U\left(\operatorname*{arg\,max}_{a' \in \mathcal{C}_{\mathcal{T}_{B_{\mathrm{HGM}}}}(a)} \mathrm{Score}_\pi(a') \right) \right].$$

When $\mathrm{Score}_\pi = U$, this reduces to

$$\mathrm{CMP}_\pi(\mathcal{T}, a) = \mathbb{E}_{\mathcal{T}_{B_{\mathrm{HGM}}} \sim p_\pi(\cdot|\mathcal{T},a)} \left[\max_{a' \in \mathcal{C}_{\mathcal{T}_{B_{\mathrm{HGM}}}}(a)} U(a') \right].$$

The HGM oracle theorem states that, under H1–H4, access to the CMP oracle is sufficient to implement the Gödel-machine selection principle (Wang et al., 2025). Empirical HGM replaces this oracle by an accumulator over binary clade outcomes. Let

$$S_{\mathcal{C}}(a) = \sum_{b \in \mathcal{C}(a)} S_b, \qquad F_{\mathcal{C}}(a) = \sum_{b \in \mathcal{C}(a)} F_b,$$

where S_b and F_b are the numbers of successful and failed active search evaluations recorded at node b. The empirical clade score is

$$\widehat{\mathrm{CMP}}(a) = \frac{S_{\mathcal{C}}(a)}{S_{\mathcal{C}}(a) + F_{\mathcal{C}}(a)} \quad \text{when } S_{\mathcal{C}}(a) + F_{\mathcal{C}}(a) > 0.$$

For Thompson-style expansion, the scheduler may draw

$$\vartheta_a \sim \mathrm{Beta}(\tau_t(1 + S_{\mathcal{C}}(a)), \tau_t(1 + F_{\mathcal{C}}(a))),$$

where τ_t is the HGM temperature, and expand the node with the largest sample subject to the expansion gate.

B Method Details

In this section, we show that although STAR evolves a programmatic financial researcher, its domain-specific mechanisms preserve the HGM interface. Specifically, STAR satisfies the HGM operating conditions within each fixed-threshold curriculum epoch, while curriculum transitions reset threshold-dependent evidence and restart HGM-valid accounting on the inherited frozen archive.

As a recap, we have introduced the STAR framework in §3. We restate the list of invariants in STAR, which are strictly enforced by the host boundary rather than softly by prompts.

S1. Frozen snapshots. Once a node a is created, its researcher state σ_a is immutable.

S2. Fixed measurement kernel and independent trial draws. Within each curriculum level, the measurement kernel is fixed, including the evaluator, threshold, environment, model versions, data snapshot, and validation-window sampler. Each atomic evaluation draws an independent validation tuple from this fixed distribution, with the six within-tuple windows constrained to be pairwise disjoint.

S3. Isolated fresh trials. Conditional on selecting node a at level ℓ, the host draws fresh randomness ξ independently of the prior transcript. The researcher sees only search data and its frozen repository while hidden validation and held-out test data are accessible only to the host evaluator.

S4. Single bounded observation. Each admitted atomic evaluation produces exactly one binary outcome $Y \in \{0, 1\}$, and no unbounded within-call search hidden inside one posterior observation.

S5. Charged action accounting. Fixed host-side budgets for both action types, including one unit of modification budget consumption for each admitted modification attempt.

S6. Utility-channel isolation. Training-side artifacts may guide descendant construction, but never enter the active search record, CMP estimates, curriculum triggers, or final selection. Hidden validation outcomes are never written into editable researcher memory.

S7. Curriculum erasure. Threshold-dependent evidence is restarted at threshold transition.

B.1 Bernoulli Validity and Clade-Level Estimation

We first show that atomic STAR evaluations induce Bernoulli trials (Lemma 1) and that adaptive scheduling preserves the corresponding Bernoulli likelihood (Lemma 2). We then show the exact node-level Beta posterior and the associated empirical capability estimate. Aggregating these outcomes over descendants yields the standard empirical HGM clade score within the same curriculum threshold (Proposition 1).

Lemma 1. *Fix a node a and curriculum level ℓ, $Y_a^{(\ell)}$ is a Bernoulli random variable with fixed success probability*

$$p_a^{(\ell)} = \mathbb{P}\left(Y_a^{(\ell)} = 1 \mid \sigma_a, \theta_\ell\right).$$

Proof. The host evaluator is deterministic conditional on a fixed candidate program, window tuple, pinned measurement kernel and random host draw ξ (S1-S3). Therefore $Y_a^{(\ell)}(\xi)$ is a measurable map from the trial randomness to $\{0, 1\}$. Its expectation is the constant $p_a^{(\ell)}$ above. Any failed measurement (e.g., invalid code) is mapped to $Y = 0$, so it does not create missing data. The search-window randomness, hidden validation-window randomness, LLM sampling randomness, and within-call dependence among the six validation windows are all integrated into the fixed law of ξ. Hence the call-level outcome is Bernoulli with fixed mean $p_a^{(\ell)}$. □

Lemma 2. *Let $\mathcal{F}_{t-1}$ be the full search transcript before atomic evaluation call t, including the tree, retained active outcomes, failure traces, curriculum level, and scheduler state. Let A_t and L_t denote the node and curriculum level selected by the scheduler. If (A_t, L_t) is $\mathcal{F}_{t-1}$-measurable and the trial randomness at time t is fresh and independent of $\mathcal{F}_{t-1}$, then*

$$\mathbb{P}\left(Y_t = 1 \mid \mathcal{F}_{t-1}, A_t = a, L_t = \ell\right) = p_a^{(\ell)}.$$

Moreover, for any realized active transcript containing outcomes

$$D_a^{(\ell)} = \{Y_{a,1}^{(\ell)}, \ldots, Y_{a,n_a^{(\ell)}}^{(\ell)}\}$$

for a fixed existing node–level pair (a, ℓ), the likelihood contribution of these outcomes factors as

$$\mathcal{L}\left(p_a^{(\ell)}; D_a^{(\ell)}\right) = \prod_{i=1}^{n_a^{(\ell)}} \left(p_a^{(\ell)}\right)^{Y_{a,i}^{(\ell)}} \left(1 - p_a^{(\ell)}\right)^{1 - Y_{a,i}^{(\ell)}},$$

up to scheduler-dependent factors that do not depend on $p_a^{(\ell)}$.

Proof. Conditioning on $\mathcal{F}_{t-1}$ and the event $(A_t, L_t) = (a, \ell)$ fixes the selected frozen snapshot and threshold. By S3, the next trial randomness is independent of $\mathcal{F}_{t-1}$ and has the same law as in Lemma 1. Hence the conditional success probability is $p_a^{(\ell)}$.

For the likelihood statement, write the probability of a realized transcript as the product over time of two factors: the scheduler's probability of selecting an action given the past transcript, and the host evaluation probability for the resulting outcome when an evaluation occurs. The scheduler factor is a function of the observed past transcript, prior hyperparameters, and deterministic host policy; it does not depend on the unknown value of $p_a^{(\ell)}$ except through outcomes already conditioned on in the transcript. Each time the scheduler selects (a, ℓ), the evaluation factor is Bernoulli with parameter $p_a^{(\ell)}$ by the first part of the lemma. Multiplying these factors over all selections of (a, ℓ) gives the displayed Bernoulli likelihood, times scheduler factors independent of $p_a^{(\ell)}$. This is the standard non-anticipating adaptive-sampling factorization; it does not require the scheduler to select nodes independently of past outcomes. □

Proposition 1. *For any fixed existing node a and active curriculum level ℓ, let*

$$S_a^{(\ell)} = \sum_{i=1}^{n_a^{(\ell)}} Y_{a,i}^{(\ell)}, \qquad F_a^{(\ell)} = n_a^{(\ell)} - S_a^{(\ell)}.$$

With prior $p_a^{(\ell)} \sim \mathrm{Beta}(\alpha_0, \beta_0)$, the exact posterior induced by the active search record is

$$p_a^{(\ell)} \mid D_a^{(\ell)} \sim \mathrm{Beta}\left(\alpha_0 + S_a^{(\ell)}, \beta_0 + F_a^{(\ell)}\right).$$

Proof. By Lemma 2, the likelihood contribution of the active outcomes for (a, ℓ) is

$$\left(p_a^{(\ell)}\right)^{S_a^{(\ell)}} \left(1 - p_a^{(\ell)}\right)^{F_a^{(\ell)}},$$

up to factors independent of $p_a^{(\ell)}$. Multiplying by the Beta prior density gives

$$\left(p_a^{(\ell)}\right)^{\alpha_0 + S_a^{(\ell)} - 1} \left(1 - p_a^{(\ell)}\right)^{\beta_0 + F_a^{(\ell)} - 1},$$

which is the kernel of

$$\mathrm{Beta}\left(\alpha_0 + S_a^{(\ell)}, \beta_0 + F_a^{(\ell)}\right).$$

The result holds under adaptive node selection because the selection rule is non-anticipating and contributes no additional likelihood term involving $p_a^{(\ell)}$.

The node-level empirical capability estimate is therefore

$$\widehat{p}_a^{(\ell)} = \frac{S_a^{(\ell)}}{n_a^{(\ell)}}, \qquad n_a^{(\ell)} > 0.$$

The clade-level HGM accumulator at level ℓ is obtained by summing active successes and failures over descendants:

$$S_{\mathcal{C}}^{(\ell)}(a) = \sum_{b \in \mathcal{C}(a)} S_b^{(\ell)}, \qquad F_{\mathcal{C}}^{(\ell)}(a) = \sum_{b \in \mathcal{C}(a)} F_b^{(\ell)}.$$

□

B.2 Curriculum Transitions and Level-Local Stationarity

We next describe how STAR implements curriculum transitions and show that curriculum erasure restores level-local stationarity (Proposition 2).

Let T_ℓ be the stopping time at which STAR advances from level ℓ to level $\ell+1$. A curriculum transition is the operation

$$(\mathcal{T}_{T_\ell}, \theta_{\ell+1}, \mathsf{ERASE}_{\ell \to \ell+1}),$$

where $\theta_{\ell+1}$ is fixed before any level-$(\ell+1)$ posterior update is admitted, and

$$\mathsf{ERASE}_{\ell \to \ell+1}$$

removes from every active scheduler quantity all search outcomes generated under θ_ℓ. In particular, old-threshold outcomes are removed from node posteriors, clade accumulators, expansion scores, curriculum triggers, and final search-time selection scores. Tree topology and frozen node snapshots may be preserved. Training reports, failure traces, and audit logs may also be preserved only subject to S6: they must be sanitized search-side artifacts and must not expose hidden validation measurements, hidden validation-window identities, old threshold labels, posterior counts, clade accumulators, curriculum-trigger statistics, or final selection scores. Preserved artifacts may affect future search only through non-anticipating scheduling or through the creation of descendant frozen snapshots; they are never treated as level-$(\ell+1)$ utility observations.

For example, with deterministic tie-breaking, STAR may define the current leader by

$$\bar{a}_\ell = \arg\max_{a : n_a^{(\ell)} > 0} \widehat{p}_a^{(\ell)}$$

and advance the curriculum when

$$\widehat{p}_{\bar{a}_\ell}^{(\ell)} > \rho \quad \text{and} \quad n_{\bar{a}_\ell}^{(\ell)} \geq n_{\min}.$$

After the transition, active counts restart at level $\ell+1$.

Proposition 2. *Suppose* STAR *transitions from curriculum level ℓ to $\ell+1$ at stopping time T_ℓ, applies* $\mathsf{ERASE}_{\ell \to \ell+1}$ *before admitting any level-$(\ell+1)$ posterior update, and preserves only artifacts allowed by S6.Then every retained active search outcome after the transition is generated under threshold $\theta_{\ell+1}$. Consequently, post-erasure search records are stationary with respect to the level-local utility $U_{\ell+1}(a) = p_a^{(\ell+1)}$.*

Proof. Before the transition, active outcomes are generated under θ_ℓ. The success event depends explicitly on θ_ℓ through indicators of the form

$$\mathbf{1}\left[\mathrm{IR}_{W_j}(\phi_k) > \theta_\ell\right].$$

If $\theta_{\ell+1} \neq \theta_\ell$, these outcomes are observations of a different binary random variable from the one defining $Y_a^{(\ell+1)}$. The erasure operation removes all such old-threshold outcomes from every active posterior, clade accumulator, expansion score, curriculum trigger, and final search-time selection score. Immediately after erasure, no retained active outcome was generated under the old threshold. Subsequent outcomes are generated only by the fixed host sampler, frozen evaluator, and active threshold $\theta_{\ell+1}$. By Lemma 2, for each selected node a, the conditional success probability of a new outcome is $p_a^{(\ell+1)}$ and is independent of prior search history. Therefore every post-erasure utility observation counted by the active search record is generated by the level-$(\ell+1)$ measurement kernel and has the level-local Bernoulli law $Y_a^{(\ell+1)} \sim \mathrm{Bernoulli}(p_a^{(\ell+1)})$ conditional on selecting node a. Preserved sanitized artifacts may influence which nodes are selected, but by S6 and Lemma 2 this non-anticipating adaptivity does not alter the level-$(\ell+1)$ Bernoulli likelihood. □

C Experimental Details

C.1 Configuration

Dataset. We use daily OHLCV data for point-in-time verified stock universes, adjusted for splits and dividends. All OHLCV fields are drawn from a single data source (Sina) to ensure consistent adjustment, suspension, and field conventions. For per-stock suspensions within an instrument's observed trading range, we carry forward the previous close and set turnover to zero.

We implement warm-up as context-only lookback. Each split prepends 60 trading days of OHLCV history so that rolling alpha operators have sufficient past observations, but labels and metrics are computed only on in-window dates. Warm-up rows are assigned missing forward returns and are excluded from all metric calculations. To prevent boundary leakage, we apply a 13-trading-day tail embargo to every split, corresponding to the 10-day label horizon, one-day execution lag, and a two-day safety buffer for calendar/trading-day slippage. Labels are pre-shifted to represent tradable open-to-open returns from $t+1$ to $t+11$. During search phase, both training and validation windows are sampled exclusively from the 2019–2024 search pool, while the 2025–2026 held-out data is stored separately and made inaccessible to all search and evaluation sandboxes.

Our protocol does not explicitly model daily price-limit constraints, to remain comparable with prior academic factor-evaluation settings. This simplification is less consequential for CSI300 because its constituents are highly liquid and less exposed to extreme liquidity events than small-cap or speculative segments. Existing evidence suggests that one-day limit-up events are concentrated mainly outside the large-cap universe (Huaan Securities, 2026), and over our experimental period, price-limit untradability affects only a small fraction of CSI300 trading days and is unlikely to change the main conclusions.

Sandbox isolation. We enforce leakage control through isolated agent-facing sandboxes and a separate host-side evaluator. The researcher sandbox exposes only the search window, a read-only researcher repository, and writable scratch/output paths – validation and test data are not mounted. The metacognitive sandbox exposes only the parent snapshot, train-side evaluation reports, and a writable staged copy of the researcher state. During search, the host evaluator loads emitted alpha files and validation data to produce per-alpha scores, but the held-out test pool remains inaccessible to all agent-facing processes. Final test evaluation is performed only after evolution terminates, in a separate host-side pass with search-time artifacts removed. This isolation ensures that reported test performance reflects genuine out-of-sample generalization rather than accidental access to future data.

C.2 Metrics

We evaluate each candidate alpha with a frozen evaluator. After observing the daily bar ending at date t, the alpha emits scores $s^{\alpha}_{t,i}$. Predictive metrics use future open-to-open returns, while portfolio metrics execute trades at the next open. We follow prior work (Tang et al., 2025; Han et al., 2026) and use a long-only TopK-Dropout sleeve. For CSI300, we set target book size $K = 30$ and daily dropout count $N_{\text{drop}} = 3$. Entry and exit transaction costs are $c_{\text{open}} = 5$ bps and $c_{\text{close}} = 15$ bps, respectively, and are deducted before computing any return statistic.

Let $\mathcal{U}_t$ denote the point-in-time tradable universe. All computations are restricted to the valid set

$$\mathcal{V}_t = \{i \in \mathcal{U}_t : s^{\alpha}_{t,i},\ \mathsf{open}_{t+1,i},\ \mathsf{open}_{t+2,i},\ \mathsf{open}_{t+11,i} \text{ are finite and tradable}\}.$$

Predictive metrics use the ten-trading-day open-to-open forward return

$$y_{t,i} = \frac{\mathsf{open}_{t+11,i}}{\mathsf{open}_{t+1,i}} - 1. \tag{1}$$

Dates whose labels cross a split boundary are embargoed. All reported metrics are computed on the remaining valid evaluation dates T. For any time series x_t, let $\overline{x}$ denote its mean over T, and let $\sigma(x)$ denote its sample standard deviation. A small ε is added to standard-deviation denominators for numerical stability.

RankIC and RankICIR. For each date t, RankIC is the Spearman rank correlation between alpha scores and future returns:

$$\mathrm{RankIC}_t(\alpha) = \rho_{\mathrm{Spearman}}\left(\{s^{\alpha}_{t,i}\}_{i\in\mathcal{V}_t}, \{y_{t,i}\}_{i\in\mathcal{V}_t}\right). \tag{2}$$

Dates with insufficient valid cross-sectional coverage are excluded. We report the time-series average

$$\mathrm{RankIC}(\alpha) = \overline{\mathrm{RankIC}_t(\alpha)}, \tag{3}$$

and its information ratio

$$\mathrm{RankICIR}(\alpha) = \frac{\overline{\mathrm{RankIC}_t(\alpha)}}{\sigma(\mathrm{RankIC}_t(\alpha)) + \varepsilon}. \tag{4}$$

Strategy-level metrics. First, we show topK-dropout portfolio construction as follows. On each date t, instruments are ranked by $s^{\alpha}_{t,i}$. Starting from the previous holdings, the evaluator removes non-tradable names and replaces up to N_{drop} held names with the highest-ranked eligible instruments not already held, until the book reaches

$$K_t = \min\{K, |\mathcal{V}_t|\}.$$

This keeps the sleeve concentrated in the top-ranked names while inducing controlled daily turnover. Let H_t denote the post-rebalance holdings selected using information available at date t. Trades execute at open_{t+1}, and the portfolio is held with equal weights

$$w_{t,i} = \begin{cases} |H_t|^{-1}, & i \in H_t, \\ 0, & i \notin H_t. \end{cases} \tag{5}$$

Let $\tau^{\mathrm{buy}}_{t+1}$ and $\tau^{\mathrm{sell}}_{t+1}$ denote the bought and sold notional at the rebalance. The transaction cost is

$$C_{t+1} = c_{\mathrm{open}}\tau^{\mathrm{buy}}_{t+1} + c_{\mathrm{close}}\tau^{\mathrm{sell}}_{t+1}.$$

The net one-day portfolio return is

$$r^{\mathrm{port}}_{t+1}(\alpha) = \sum_{i\in H_t} w_{t,i}\left(\frac{\mathsf{open}_{t+2,i}}{\mathsf{open}_{t+1,i}} - 1\right) - C_{t+1}. \tag{6}$$

Let r^{bench}_{t+1} denote the corresponding benchmark return. The cost-adjusted active return is

$$e_{t+1}(\alpha) = r^{\mathrm{port}}_{t+1}(\alpha) - r^{\mathrm{bench}}_{t+1}. \tag{7}$$

IR. Information ratio is the risk-adjusted performance metric, measuring active return over a benchmark over tracking error.

$$\mathrm{IR}(\alpha) = \sqrt{\mathrm{N}}\,\frac{\overline{e_t(\alpha)}}{\sigma(e_t(\alpha)) + \varepsilon}, \tag{8}$$

where N is the number of trading days.

ER. Excess return over benchmark during the trading period. Reported ER is not annualized.

MDD. Maximum Drawdown is the largest peak-to-trough decline in cumulative excess returns.

C.3 Downstream Performance Baselines

STAR differs from conventional alpha research baselines in that it improves the *alpha research policy* itself, rather than optimizing only the final alpha formulas. Nevertheless, to evaluate downstream utility, we compare the alphas discovered by the evolved research policy against four families of baselines: machine learning models, deep learning models, public factor libraries (Alpha158 and Alpha360), and LLM-driven alpha-discovery agents. All baselines are evaluated under the same universe, prediction horizon, transaction-cost model, portfolio construction rule, and 2026Q1 test protocol.

To evaluate classical machine learning baselines, we train XGBoost, LightGBM, Random Forest, AdaBoost, and CatBoost on the official Qlib Alpha158 feature panel under the matched chronological split. All baseline hyperparameters are selected on the same 2025 validation split, with the 2026Q1 test period used exactly once. To evaluate deep learning baselines, we consider LSTM, Transformer, and TRA, where TRA is a finance-specific temporal routing model with mixture-style, state-conditioned heads over an LSTM backbone. Following the standard Qlib deep learning convention, these models are trained on 60-day windows of price-volume features normalized by the same-day close, with per-feature z-score standardization fitted on the training split and clipped to $\pm 5\sigma$.

Additionally, we evaluate LLM-driven alpha discovery systems, including AlphaAgent (Tang et al., 2025), QuantaAlpha (Han et al., 2026), and CogAlpha (Liu et al., 2025). AlphaAgent and QuantaAlpha are run using their official released repositories. Since CogAlpha does not provide public source code, we re-implement it from the published description. CogAlpha specifies large search-population hyperparameters, including the initial-pool, parent-pool, and child-pool sizes, which can yield substantially more candidate alphas than the other agentic baselines. To prevent the comparison from being dominated by brute-force proposal volume, we evaluate CogAlpha under a budget-normalized configuration with reduced population sizes. This configuration produces roughly 5k candidate alphas, which is larger than the alpha-pool scale of the other LLM-based agentic baselines but remains within the same order of magnitude. We then apply the filtering procedure described in the paper to obtain the qualified alpha set. This setup tests whether CogAlpha can produce high-quality alphas under a comparable proposal budget, rather than benefiting primarily from a much larger candidate pool.