Chain-of-Alpha: Unleashing the Power of Large Language Models for Alpha Mining in Quantitative Trading

Lang Cao Zekun Xi Long Liao Ziwei Yang Zheng Cao

SCITIX (SGP) TECH PTE. LTD.

Abstract

Alpha factor mining is a fundamental task in quantitative trading, aimed at discovering interpretable signals that can predict asset returns beyond systematic market risk. While traditional methods rely on manual formula design or heuristic search with machine learning, recent advances have leveraged Large Language Models (LLMs) for automated factor discovery. However, existing LLM-based alpha mining approaches remain limited in terms of automation, generality, and efficiency. In this paper, we propose *Chain-of-Alpha*, a novel, simple, yet effective and efficient LLM-based framework for fully automated formulaic alpha mining. Our method features a dual-chain architecture, consisting of a *Factor Generation Chain* and a *Factor Optimization Chain*, which iteratively generate, evaluate, and refine candidate alpha factors using only market data, while leveraging backtest feedback and prior optimization knowledge. The two chains work synergistically to enable high-quality alpha discovery without human intervention and offer strong scalability. Extensive experiments on real-world A-share benchmarks demonstrate that *Chain-of-Alpha* outperforms existing baselines across multiple metrics, presenting a promising direction for LLM-driven quantitative research.

Introduction

Quantitative trading refers to a data-driven approach to trading that leverages mathematical modeling, statistical analysis, and algorithmic execution. Despite its systematic nature, it entails significant challenges, such as ensuring data quality, managing the complexity of model design and validation, mitigating overfitting risks, and continuously adapting to dynamic market (Hou, Karolyi, and Kho 2011).

Formulaic alpha factor mining (hereafter referred to as ***alpha mining***) is a fundamental component of quantitative trading research, dedicated to the systematic discovery of return-predictive signals. The primary objective of alpha mining is to identify *alpha factors*, signals that can explain or forecast asset returns beyond what is captured by broad market risk exposures (Ng, Engle, and Rothschild 1992). By definition, alpha represents asset-specific sources of return that are largely uncorrelated with general market movements, in contrast to beta, which reflects systematic risk and overall market trends (Sharpe 1964). Alpha mining is typically conducted through cross-sectional analysis to uncover predictive patterns across assets at a given point in time. Unlike neural-based approaches to alpha mining (Duan et al. 2022; Xu et al. 2021), which implicitly learn complex alpha signals through deep learning and capture intricate patterns in financial data, formula-based methods aim to discover alpha factors expressed as explicit mathematical formulas. Traditionally, such alphas are manually designed by domain experts and often reflect well-established market insights (Fama and French 2004). Compared to neural-based ones, formula-based alphas offer greater interpretability, improved robustness, stronger generalization across datasets, and typically require less data to be effective.

Mining formulaic alpha factors is both time-consuming and labor-intensive, often requiring significant domain expertise and manual effort. Machine learning (ML) techniques have been widely used in finance to assist human (Yang, Liu, and Wang 2023; Cao, Zhang, and Chen 2021). In particular, numerous studies have explored automated alpha mining techniques based on traditional ML methods (Lin et al. 2019; Zhang et al. 2020; Cui et al. 2021; Yu et al. 2023a). Recently, Large Language Models (LLMs) have emerged as a promising tool for alpha mining (Cao et al. 2025), owing to their powerful capabilities in reasoning, pattern recognition, and natural language understanding (OpenAI 2024; Gunasekar et al. 2023; Plaat et al. 2024). By leveraging LLMs, it becomes feasible to generate interpretable and high-quality alpha factors with significantly reduced human intervention. Compared to traditional ML methods, LLM-based approaches offer greater flexibility and expressiveness. Several recent works (Wang et al. 2023a; Tang et al. 2025; Li et al. 2024; Kou et al. 2025; Wang et al. 2024; Shi, Duan, and Li 2025) have explored this direction, demonstrating the potential of LLMs to automate the discovery of formulaic alpha signals.

Despite these promising developments, existing LLM-based alpha mining approaches still face substantial limitations. (1) ***Not Fully Automated***: Some methods, such as AlphaGPT (Wang et al. 2023a) and AlphaAgent (Tang et al. 2025), rely heavily on human feedback. Others depend on pre-existing alpha factors for in-context learning, as in FAMA (Li et al. 2024), thereby limiting autonomy and hindering truly from-scratch discovery. (2) ***Limited Generality***: Certain approaches rely on multi-modal inputs (Kou et al. 2025), involving data sources and modalities that are

not widely available or generally applicable to alpha mining. Others focus on broader strategy generation, such as QuantAgent (Wang et al. 2024), instead of purely targeting alpha factor discovery, thereby reducing their generality in the context of comprehensive alpha mining. (3) ***Inefficiency***: The *LLM+MCTS* approach (Shi, Duan, and Li 2025) improves generation quality but remains inefficient. It suffers from limited parallelizability due to the inherently sequential nature of tree-based search, which constrains its efficiency and scalability in large-scale applications.

To overcome these challenges, we propose ***Chain-of-Alpha***, a novel, simple, yet effective and efficient LLM-based framework for automated alpha mining. First, our method is fully automated, requiring no human intervention. It autonomously generates seed factors, optimizes them, and selects promising candidates to construct a high-quality alpha pool. Second, the framework is exclusively designed to operate on market data, which is more widely available and broadly applicable to alpha mining tasks. Third, we introduce a novel dual-chain mechanism within the framework: the *Factor Generation Chain* and the *Factor Optimization Chain*. These two synergistic chains operate collaboratively, leveraging backtest feedback and prior optimization knowledge, to ensure both the efficiency and quality of the generated alpha factors.

In summary, our key contributions are as follows:

- We propose *Chain-of-Alpha*, a novel, simple, yet effective and efficient LLM-based framework for automated alpha mining.
- We design a novel dual-chain architecture comprising the *Factor Generation Chain* and the *Factor Optimization Chain*, which work synergistically to efficiently produce high-quality candidate alpha factors.
- We conduct extensive experiments on real-world A-share stock benchmarks, demonstrating the effectiveness of *Chain-of-Alpha* and providing comprehensive analyses that offer deeper insights for alpha mining.

Related Work

Reasoning with Large Language Models. Large language models (LLMs) exhibit emergent reasoning abilities when scaled to a sufficient size (Wei et al. 2022; Suzgun et al. 2022). One effective technique to elicit such reasoning is chain-of-thought prompting, which guides the model to solve problems step by step and has been shown to substantially improve performance on complex tasks (Wei et al. 2023). Building on this foundation, methods such as self-consistency (Wang et al. 2023b) and structured reasoning frameworks, including tree-based (Yao et al. 2023a) and graph-based approaches (Besta et al. 2024; Cao 2024), further enhance the model's capability to handle more sophisticated reasoning challenges.

Recent advances have shown that integrating reasoning and action in LLMs, through interaction with external tools and environments, significantly enhances their ability to reason over complex, open-ended tasks. Notable approaches include ReAct (Yao et al. 2023b), which interleaves chain-of-thought reasoning with tool use, and Toolformer (Schick et al. 2023), which allows LLMs to learn when and how to invoke APIs through self-supervised training. SciAgent (Ma et al. 2024) focuses on scientific problem solving by incorporating domain-specific tools, while RAG-Star (Jiang et al. 2024) enhances multi-step reasoning via retrieval and structured planning. In this work, we extend this paradigm to quantitative finance by enabling LLMs to reason over historical market data for alpha mining.

Large Language Models for Quantitative Trading. LLMs have been widely adopted in quantitative trading to assist humans in making better decisions due to their strong reasoning abilities. LLMs have primarily been used to generate trading decisions and act as autonomous agents in financial markets (Xiao et al. 2025; Yu et al. 2023b; Liu and Lo 2025), or to collaborate with human traders by providing investment suggestions (Rao et al. 2025; Guo et al. 2025). On the other hand, several studies have explored the use of LLMs for alpha mining (Li et al. 2024; Shi, Duan, and Li 2025), aiming to discover more effective trading signals and strategies. In this work, we follow the latter direction and aim to develop a more general and effective framework for alpha mining using LLMs.

Automatic Mining of Formulaic Alpha Factors. Traditional formulaic alpha factor mining has primarily relied on Genetic Programming (GP), as seen in methods such as GPLearn (Lin et al. 2019), AutoAlpha (Zhang et al. 2020), and AlphaEvolve (Cui et al. 2021). AlphaGen (Yu et al. 2023a) optimizes sets of alpha factors using reinforcement learning. AlphaForge (Shi et al. 2024) features a generative-predictive architecture to mine alpha factor. However, traditional methods often lack the general reasoning capabilities and flexible knowledge integration that LLMs can provide. For example, AlphaGPT (Wang et al. 2023a; Yuan, Wang, and Guo 2024) and AlphaAgent (Tang et al. 2025) propose human-AI interactive frameworks for alpha mining and optimization. Automate Strategy Finding with LLM(Kou et al. 2025) introduces a system for discovering alpha signals from diverse modality data. QuantAgent (Wang et al. 2024) features a two-loop LLM refinement architecture augmented with a knowledge base for learning trading strategies. FAMA (Li et al. 2024) leverages LLMs to discover alpha factors through in-context learning by synthesizing diverse existing alphas and prior mining experiences. The integration of LLMs with Monte Carlo Tree Search (MCTS) (Shi, Duan, and Li 2025) has been proposed to optimize the generation of alpha factors. Despite these promising advances, many existing methods still suffer from notable limitations, as previously discussed. Consequently, developing a general, efficient, and fully automated framework for alpha mining with LLMs remains an open challenge, which we aim to address in this work.

Preliminary

Task 1 (*Formulaic Alpha Factor Mining for Daily Cross-Sectional Stock Return Prediction*)

In this task, we consider a stock pool consisting of n stocks observed over T trading days in the financial market. For

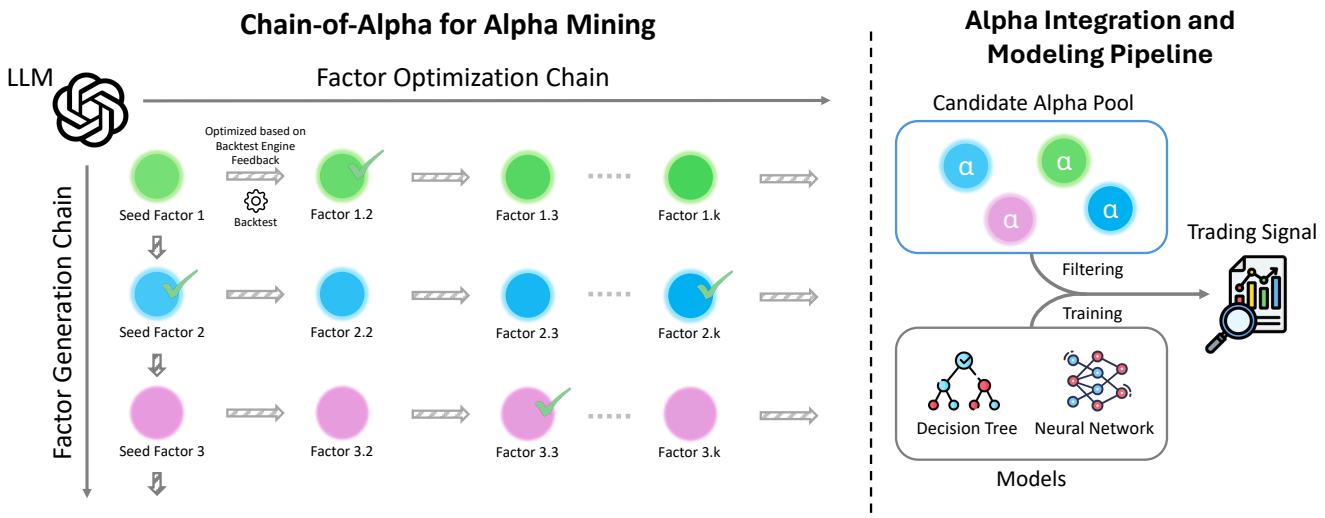


Figure 1: The *Chain-of-Alpha* framework for alpha mining. The left part illustrates the *Chain-of-Alpha* process, which comprises two interconnected chains: the *Factor Generation Chain*, where a large language model (LLM) sequentially generates diverse seed factors, and the *Factor Optimization Chain*, where each factor is iteratively optimized based on feedback from a backtest engine. Selected factors from these chains are aggregated into a candidate alpha pool. The right part depicts the integration pipeline, where candidate factors are filtered and used to train predictive models (e.g., decision trees, neural networks), ultimately producing actionable trading signals.

each stock $i \in \{1, \ldots, n\}$ and day $t \in \{1, \ldots, T\}$, a raw feature vector $x_{i,t} \in \mathbb{R}^m$ is available, comprising daily open, high, low, and close prices, trading volume, and other relevant market features. The complete market history is represented as a tensor $\mathbf{X} \in \mathbb{R}^{T \times n \times m}$. Correspondingly, the realized future returns over a prediction horizon h form a matrix $\mathbf{Y} \in \mathbb{R}^{T \times n}$, where $y_{i,t}$ denotes the realized return of stock i over the h-day period following day t.

An alpha factor f maps historical feature data to a predictive signal $\mathbf{v}_t = f(\mathbf{X}_{t-\tau+1:t}) \in \mathbb{R}^n$, where τ denotes the lookback window capturing temporal patterns. Each $v_{i,t}$ represents the alpha's prediction for the future return of stock i at time t. The goal of alpha mining is to discover a diverse set of K alpha factors, denoted by $\mathcal{F} = \{f_1, \ldots, f_K\}$. The outputs of these individual alphas, $\{\mathbf{v}_{k,t} = f_k(\mathbf{X}_{t-\tau+1:t})\}_{k=1}^K$, are typically aggregated by a combination model g (e.g., a decision tree or neural network) into a composite predictive signal: $\mathbf{z}_t = g(\{\mathbf{v}_{k,t}\}_{k=1}^K; \theta_g) \in \mathbb{R}^n$, where θ_g denotes the model parameters. Collecting these signals across T days yields a composite signal matrix $\mathbf{Z}(\mathcal{F}, \theta_g) \in \mathbb{R}^{T \times n}$. The quality of the composite signal $\mathbf{Z}$ is evaluated against the realized returns $\mathbf{Y}$ using a predefined performance metric (e.g., Information Coefficient), which we seek to maximize.

This setup constitutes a bilevel optimization problem: the inner level learns the optimal combination parameters θ_g^* for a given alpha set $\mathcal{F}$, while the outer level searches for the optimal alpha set $\mathcal{F}^*$ that maximizes the performance of the resulting combined signal. To simplify the optimization process under a fixed alpha integration and modeling pipeline, our objective is to identify a set of high-quality (e.g., high-IC) and diverse alpha factors that collectively yield the best overall predictive performance.

Methodology

As illustrated in Figure 1, we propose the *Chain-of-Alpha* framework for automated alpha mining. It primarily consists of two components: the *Factor Generation Chain* and the *Factor Optimization Chain*. The two chains operate synergistically, and effective alpha factors are subsequently selected and passed to the alpha integration and modeling pipeline.

Factor Selection

Prior to detailing the procedures for factor generation and optimization, we first describe the factor selection process. Each generated factor is evaluated by testing it on historical market data through backtesting to obtain its performance results. During the operation of both chains, each newly generated factor is immediately evaluated:

$$\textbf{Score} = \text{Evaluate}(f), \tag{1}$$

where $\textbf{Score} = [S, C, E, D]$ is a four-dimensional evaluation score representing the factor's performance across four key dimensions:

1. **Strength (S):** Reflects the predictive power of the factor. It is measured by the cross-sectional Rank Information Coefficient (RankIC), which captures the correlation between factor values and subsequent realized returns.
2. **Consistency (C):** Captures the temporal stability of the factor's performance. It is measured by the RankIC Information Ratio (RankICIR), defined as the mean RankIC divided by its standard deviation over time.
3. **Efficiency (E):** Represents the trading cost implications of the factor. It is measured by the turnover rate, which quantifies how frequently portfolio positions change

when using the factor. Lower turnover implies more stable signals and reduced transaction costs.

4. **Diversity (D):** Encourages independence among selected factors. It is quantified as the minimum of $1-\text{Corr}(f, f_k)$ over all existing effective factors $f_k \in \mathcal{F}^e$, promoting low redundancy and complementarity in the factor pool. A higher score indicates greater dissimilarity from existing factors, which is preferred.

Once the score $\mathbf{S}$ is obtained, the factor is further checked against predefined thresholds to determine its effectiveness:

$$\mathbf{E} = \text{Check}(f, \mathbf{S}) \tag{2}$$

where $\mathbf{E}$ is a binary indicator denoting whether the factor f is considered effective. Valid alpha factors are selected and added to the candidate alpha pool $\mathcal{F}^e = \{f_1^e, \ldots, f_K^e\}$. This candidate pool is maintained as a reference for subsequent factor generation, ensuring diversity and avoiding redundancy, while also serving as a valuable source of guidance for generating additional high-quality factors. Non-effective factors are also recorded in a deprecated pool $\mathcal{F}^d = \{f_1^d, \ldots, f_M^d\}$, which serves as a negative reference to discourage the LLM from generating similar low-quality factors in the future.

Factor Generation

The *Factor Generation Chain* aims to produce a diverse collection of seed alpha factors f_k^{seed} using a large language model (LLM), with ***diversity*** being the highest priority. At each iteration, the LLM proposes a new formulaic alpha candidate f based on the current candidate alpha pool $\mathcal{F}^e = \{f_1^e, \ldots, f_K^e\}$, the deprecated pool $\mathcal{F}^d = \{f_1^d, \ldots, f_M^d\}$, and external prompts $\mathcal{P}_{\text{generation}}$:

$$f^{\text{seed}} = \text{LLM}(\mathcal{F}^e, \mathcal{F}^d \mid \mathcal{P}_{\text{generation}}), \tag{3}$$

where $\text{LLM}(\cdot)$ denotes the inference process of the large language model using chain-of-thought reasoning. The output f is initially in text form and is subsequently parsed and transformed into executable code representing the factor's mapping function. The prompt $\mathcal{P}_{\text{generation}}$ typically includes descriptions of available data fields, a list of mathematical operators, and task-specific instructions guiding the LLM on how to construct interpretable and high-quality alpha factors.

The operation of the *Factor Generation Chain* can be formalized as:

$$f_{k+1}^{\text{seed}} = \textit{Chain-of-Alpha}_{\text{generation}}(f_1^{\text{seed}}, f_2^{\text{seed}}, \ldots, f_k^{\text{seed}}), \tag{4}$$

where each factor f_i is evaluated to determine its effectiveness. This process resembles a self-evolving chain, in which previously generated factors guide the generation of new ones, enabling continuous exploration of novel and diverse alpha candidates.

The *Factor Generation Chain* can operate autonomously and indefinitely, and this iterative procedure can be repeated to produce a broad range of potentially useful seed alpha factors with varied mathematical structures and behavioral characteristics.

Factor Optimization

To further enhance the effectiveness of seed alpha factors, the *Factor Optimization Chain* performs iterative refinement based on backtesting feedback, with LLMs guiding the optimization process. In this stage, ***effectiveness*** is the highest priority.

Given a seed factor f_k^{seed}, the LLM generates a sequence of optimized variants $\{f_k^{(1)}, f_k^{(2)}, \ldots, f_k^{(m)}\}$, guided by backtesting results $\mathcal{B}$. For example, if the Information Coefficient (IC) is low, the LLM attempts to enhance the signal strength; if the RankICIR is low, it focuses on improving temporal stability. This feedback-aware generation enables the LLM to refine factors progressively.

For each seed factor, an optimization history H_k is maintained to record all intermediate variants and their evaluation results. This history is continuously updated and used within the current optimization chain. Alongside H_k and $\mathcal{B}$, the LLM also receives an external prompt $\mathcal{P}_{\text{optimization}}$ that specifies the optimization objective and available operators:

$$f_k^{(m+1)} = \text{LLM}(f_k^{\text{seed}}, f_k^{(m)}, \mathcal{B}, H_k \mid \mathcal{P}_{\text{optimization}}), \tag{5}$$

Optimized factors that meet performance criteria are added to the candidate pool $\mathcal{F}^e$, while ineffective variants are stored in the deprecated pool $\mathcal{F}^d$ to avoid redundant exploration in the future.

The operation of the *Factor Optimization Chain* can be formalized as:

$$f_k^{(m+1)} = \textit{Chain-of-Alpha}_{\text{optimization}}(f_k^{\text{seed}}, f_k^{(1)}, \ldots, f_k^{(m)}), \tag{6}$$

where $\{f_k^{\text{seed}}, f_k^{(1)}, \ldots, f_k^{(m)}\}$ represents the optimization history H_k of the seed factor f_k^{seed}. Each new optimized factor is generated based on accumulated optimization experience, forming a chain-like refinement process guided by prior iterations.

The *Factor Optimization Chain* operates autonomously, but with a capped number of iterations. If no effective variant is produced within a predefined limit, the chain is terminated early and the corresponding seed alpha is discarded, indicating that the initial direction may not be promising. This mechanism ensures computational efficiency while enabling continuous refinement of alpha factors toward higher predictive strength, stability, and lower trading costs.

Meanwhile, the optimization chains for different seed factors can operate in parallel and independently, without mutual interference. This allows the main *Factor Generation Chain* and multiple *Factor Optimization Chains* to run synergistically, significantly improving the scalability and efficiency of large-scale alpha discovery.

Alpha Integration and Modeling

Given a set of candidate alpha factors $\mathcal{F} = \{f_1, \ldots, f_K\}$, each factor f_k produces a predictive signal at each time step t:

$$\mathbf{v}_{k,t} = f_k(\mathbf{X}_{t-\tau+1:t}) \in \mathbb{R}^n, \tag{7}$$

where τ denotes the lookback window and n is the number of stocks in the universe. These signals represent the expected returns for the next prediction horizon h.

To construct a unified trading signal, the outputs of individual alpha factors are integrated using a combination model g, such as a decision tree or neural network. The candidate alpha factors are first filtered according to predefined rules, and a specified number of them are selected for integration. The composite signal at time t is defined as:

$$\mathbf{z}_t = g(\{\mathbf{v}_{k,t}\}_{k=1}^{K}; \theta_g) \in \mathbb{R}^n, \tag{8}$$

where θ_g denotes the trainable parameters of the integration model. Collectively, the signals over the trading period form a matrix:

$$\mathbf{Z}(\mathcal{F}, \theta_g) \in \mathbb{R}^{T \times n}. \tag{9}$$

The performance of the integrated signal is evaluated against the realized future return matrix $\mathbf{Y} \in \mathbb{R}^{T \times n}$ using a predefined metric $\mathcal{P}$ (e.g., Information Coefficient, Sharpe ratio, or portfolio return). The optimal integration parameters are learned by solving:

$$\theta_g^*(\mathcal{F}) = \arg\max_{\theta_g} \mathcal{P}(\mathbf{Z}(\mathcal{F}, \theta_g), \mathbf{Y}). \tag{10}$$

This integration process ensures that the most informative and complementary signals from multiple alpha factors are effectively combined to produce a robust and reliable predictive signal. The alpha mining framework of *Chain-of-Alpha* serves as the foundation for this *Alpha Integration and Modeling* pipeline, supplying a rich and diverse set of high-quality factors that enhance the final signal construction and downstream trading performance.

Experiments

Settings

Datasets. We evaluate the *Chain-of-Alpha* framework using historical data from the China A-share market, focusing on two representative stock pools: CSI 500 (mid-cap) and CSI 1000 (small-cap). These indices represent distinct segments of the Chinese equity market—CSI 500 consists of mid-sized companies, while CSI 1000 captures smaller-cap stocks with higher market granularity.

The full dataset spans from 2010-01-01 to 2025-06-30. Specifically, the training set covers the period from 2010-01-01 to 2019-12-31, the validation set spans from 2020-01-01 to 2021-12-31, and the test set includes data from 2022-01-01 to 2025-06-30. For alpha mining, we use both the training and validation sets to guide the factor generation and optimization processes. For alpha integration, the model is trained on the training set, validated on the validation set, and evaluated on the test set. We adopt a prediction horizon of 10 trading days. On each trading day t, alpha factors are computed based on historical features up to time t, while the prediction target is the forward return from t to $t + 10$.

Evaluation metrics are reported in the Appendix. All results are benchmarked relative to the corresponding market index (i.e., excess return over the index).

Baselines. To evaluate the effectiveness of our proposed *Chain-of-Alpha* framework, we compare it against a diverse set of baseline methods spanning three major categories:

Classic Factors. These include widely used hand-crafted factor libraries such as Alpha 101 (Kakushadze 2016) (with inapplicable formulas removed), Alpha 158, and Alpha 360 (Yang et al. 2020), which are commonly employed in quantitative investment strategies.

Traditional Methods. This category includes automatic factor mining approaches based on Genetic Programming (GP) (Lin et al. 2019), Differentiable Symbolic Optimization (DSO) (Landajuela et al. 2022), reinforcement learning-based AlphaGen (Yu et al. 2023a), and AlphaForge (Shi et al. 2024), a generative framework.

LLM-based Methods. We also compare against traditional LLM reasoning baselines, including Chain-of-Thought prompting (*LLM+CoT*), Tree-of-Thoughts (*LLM+ToT*), LLM guided Monte Carlo Tree Search (*LLM+MCTS*) (Shi, Duan, and Li 2025). All LLM baselines leverage backtest feedback during factor generation. Specifically, CoT continuously updates the most recent node, ToT updates the global node with the highest score, and MCTS follows the original paper by using the UCT algorithm to select nodes.

To ensure a fair comparison, we benchmark performance under a controlled alpha search budget. All methods are allowed to generate up to 1,000 candidate factors, from which the top 100 are selected based on their *RankIC* scores.

Further details on experimental settings, including alpha selection criteria, model configurations, backtesting strategy, and demo prompts, can be found in the Appendix.

Main Results

Table 1 presents a comprehensive performance comparison across classic factors, traditional methods, and LLM-based methods on the CSI 500 and CSI 1000 indices. The evaluation includes six metrics: IC, RankIC, ICIR, RankICIR, annualized return (AR), and information ratio (IR). Among these, AR and IR are the most critical as they directly reflect the investment performance of strategies driven by the mined alphas, while IC, RankIC, ICIR, and RankICIR indicate the correlation between factor signals and future returns. Our proposed method, *Chain-of-Alpha*, consistently ranks among the top performers and achieves the best results in 10 out of 12 metrics across the two benchmarks.

On the CSI 500 index, *Chain-of-Alpha* achieves the highest AR of 0.1324 and IR of 1.4178, outperforming both traditional methods such as AlphaForge and LLM-based baselines like *LLM+MCTS*. The improvement is particularly significant in return-based metrics, highlighting the effectiveness of our dual-chain design in capturing return-predictive alpha signals. On the CSI 1000 index, our method remains highly competitive, achieving top performance across all evaluation metrics. In particular, it attains an AR of 0.1471 and an IR of 1.4043, outperforming all classic, traditional, and LLM-based baselines. These results demonstrate that Chain-of-Alpha generalizes well across different market universes and benefits from both robust factor generation and effective optimization.

These results also demonstrate that complex tree-based exploration with LLMs strategies, such as those used in ToT or MCTS, are not necessary. Tree-based exploration is generally more complex and less efficient compared to chain-based approaches. Moreover, we consider such methods of-

Method	CSI 500						CSI 1000					
	IC	RankIC	ICIR	RankICIR	AR	IR	IC	RankIC	ICIR	RankICIR	AR	IR
Classic Factors												
Alpha 101	0.0345	0.0617	0.2170	0.4239	0.0568	0.7311	0.0615	0.0832	0.3845	0.5391	0.1006	1.1219
Alpha 158	0.0477	0.0686	0.3202	0.4685	0.0989	1.0424	0.0591	0.0800	0.4420	0.5817	0.1205	1.2307
Alpha 360	0.0457	0.0524	**0.3345**	0.3975	0.1092	1.1017	0.0551	0.0649	0.4384	0.5175	0.0965	0.9801
Traditional Method												
GP	0.0351	0.0659	0.2185	0.4308	0.0792	0.9535	0.0602	0.0823	0.3741	0.5281	0.1116	1.2457
DSO	0.0436	0.0638	0.3140	0.4716	0.0984	1.2640	0.0616	0.0765	0.4583	0.5304	0.1235	1.3079
AlphaGen	0.0460	0.0769	0.2786	0.4711	0.1150	1.2751	0.0655	0.0889	0.4224	0.5573	0.1247	1.2043
AlphaForge	0.0463	0.0638	0.3291	0.4630	0.0989	1.1918	0.0617	0.0768	0.4602	0.5327	0.1325	1.2657
LLM-based Method												
LLM + CoT	0.0404	0.0711	0.2558	0.4870	0.0759	0.9659	0.0620	0.0847	0.4464	0.6152	0.1181	1.2625
LLM + ToT	0.0292	0.0607	0.2227	0.4883	0.0994	1.2693	0.0597	0.0876	0.4169	0.6024	0.1267	1.3258
LLM + MCTS	0.0347	0.0595	0.3083	**0.5268**	0.0815	1.0736	0.0465	0.0713	0.4320	0.5930	0.1235	1.3342
Chain-of-Alpha (Ours)	**0.0485**	**0.0771**	0.3047	0.5013	**0.1324**	**1.4178**	**0.0672**	**0.0902**	**0.4630**	**0.6228**	**0.1471**	**1.4043**

Table 1: Performance comparison across CSI 500 and CSI 1000. Metrics include IC, RankIC, ICIR, RankICIR, annualized return (AR), and information ratio (IR). The best result in each column is highlighted in bold.

ten originate from a single root node, which can limit the diversity of candidate factors. As a result, the final integrated alpha signal may still underperform. In contrast, our dual-chain framework provides a highly effective and substantially more efficient alternative, especially when compared to single-chain CoT methods.

Ablation Study

Method	IC	RankIC	ICIR	RankICIR	AR	IR
Chain-of-Alpha	**0.0672**	**0.0902**	**0.4630**	**0.6228**	**0.1471**	**1.4043**
Factor Generation Chain	0.0586	0.0867	0.4078	0.6203	0.1346	1.3492
Factor Optimization Chain	0.0620	0.0847	0.4464	0.6152	0.1181	1.2625

Table 2: Ablation study of the *Chain-of-Alpha* framework and its individual components on the CSI 1000 index.

Table 2 reports the ablation results of the *Chain-of-Alpha* framework by isolating its two main components: the *Factor Generation Chain* and the *Factor Optimization Chain*. Removing either component leads to a noticeable performance drop across most evaluation metrics. Specifically, using only the optimization chain reduces the RankIC from 0.0902 to 0.0847 and the AR from 0.1471 to 0.1211. Conversely, using only the generated factors without refinement also lowers the AR to 0.1346 and the IR to 1.4492. The complete dual-chain framework achieves the best trade-off between factor diversity and the quality of individual alphas in a unified optimization direction. These results confirm that both chains are essential for maximizing overall performance.

Impact of Backbone LLM Choice

In addition to the default model *GPT-4o*, we evaluate our framework using *DeepSeek-V3* and *Qwen3-32B* as alternative LLM backbones with different model architectures and sizes. Table 3 presents the robustness results of *Chain-of-Alpha* across these three models. In all cases, *Chain-of-Alpha* consistently outperforms the best of baselines in Table 1 across all evaluation metrics, demonstrating the effectiveness of our chain-based framework regardless of the underlying language model. Notably, the *DeepSeek-V3* backbone achieves the highest AR of 0.1517, suggesting that more powerful LLMs may further enhance performance. Meanwhile, both *GPT-4o* and *Qwen3-32B* also yield strong results. These findings confirm that *Chain-of-Alpha* is backbone-agnostic and can benefit from increasingly capable language models.

Method	Backbone	IC	RankIC	ICIR	RankICIR	AR	IR
Best of Baselines		0.0655	0.0889	0.4602	0.6152	0.1325	1.3342
Chain-of-Alpha	GPT-4o	**0.0672**	0.0902	**0.4630**	0.6228	0.1471	1.4043
	DeepSeek-V3	0.0671	**0.1011**	0.4492	0.6063	**0.1517**	1.4020
	Qwen3-32B	0.0653	0.0939	0.4597	**0.6342**	0.1365	**1.5804**

Table 3: Performance comparison on the CSI 1000 index using *Chain-of-Alpha* across different LLM backbones. The best result in each column is highlighted in bold.

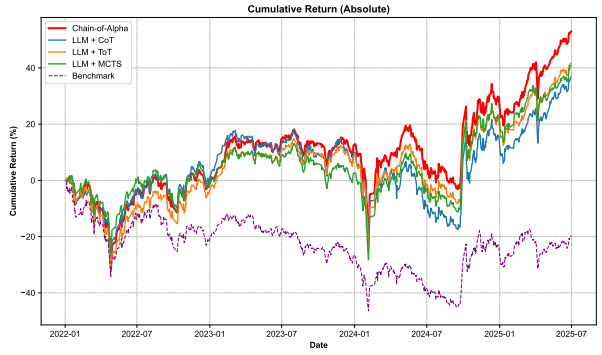
Visualization of Strategy Returns Using Mined Alphas

Figure 2 compares the cumulative returns of strategies constructed using alphas mined by four methods: *Chain-of-Alpha*, *LLM+CoT*, *LLM+ToT*, and *LLM+MCTS*. In Figure 2(a), *Chain-of-Alpha* achieves the highest cumulative absolute return, consistently outperforming all other strategies for the majority of the evaluation period. The performance gap becomes increasingly significant after mid-2023, demonstrating the superior robustness and profitability of *Chain-of-Alpha*. In contrast, the benchmark remains in a persistent drawdown, underscoring the advantage of active alpha mining over passive exposure.

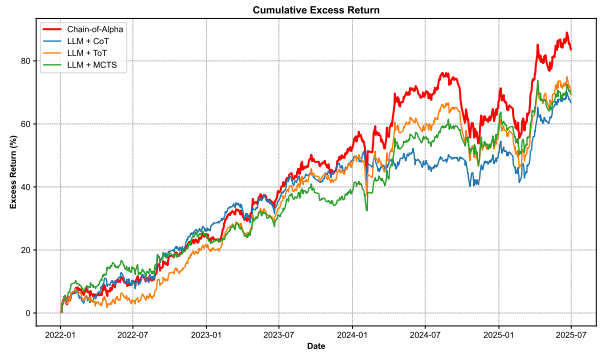
Figure 2(b) shows cumulative excess returns relative to the benchmark. Again, *Chain-of-Alpha* leads throughout the backtest period, followed by *LLM+ToT*, while *LLM+MCTS* and *LLM+CoT* lag behind. This performance indicates that the alphas generated by *Chain-of-Alpha* are more predictive and stable. The consistent superiority observed in both absolute and excess return plots underscores the robustness and

Name	Expression	Description	RankIC	RankIR
VWAP_Stability_Enhance	`Div(Sub($close, Mean($vwap, 2)), Std($amount, 5))`	This factor evaluates the deviation of the closing price from the 2-day average VWAP, normalized by the standard deviation of the trading amount over five days. It aims to capture stable price zones by reflecting short-term VWAP trends and accounting for varying market sentiment through trading amount fluctuations.	0.0688	0.7051
Volume_Adjusted_Mean_Corr	`Corr(Rank($close, 5), Rank($amount, 5), 5)`	This factor evaluates the correlation between the 5-day ranked returns of the closing price and trading amount, suggesting potential price movement adjustments related to the trading amount.	0.0375	0.5084
VWAP_Flow_Variance_Optimization	`Div(Abs(Sub($close, $vwap)), Add(Sum(Var($amount, 2), 4), 1))`	This factor assesses the deviation of the closing price from the VWAP, scaled by a summation of the 2-day variance of trading amount over a 4-day period. It captures significant fluctuations in price relative to capital flow variability, highlighting potential market reversals.	0.0838	0.7590

Table 4: Example alpha factors generated by the *Chain-of-Alpha* framework, along with their expressions, descriptions, and evaluation metrics.



(a) Cumulative absolute return.



(b) Cumulative excess return.

Figure 2: Visualization of cumulative returns using alphas mined by different methods. The strategy performance from *Chain-of-Alpha* is highlighted in bold red.

practical tradability of the proposed framework.

Case Study

We present a selection of representative alpha factors automatically generated through our proposed *Chain-of-Alpha* on the CSI 1000 index. As shown in Table 4, each factor is presented in symbolic form, accompanied by a natural language description and its corresponding evaluation metrics.

These case studies illustrate that the framework can produce interpretable and diverse alpha factors with statistically significant predictive power. Notably, the selected factors capture a range of market behaviors, including price deviations from VWAP, volume-adjusted volatility dynamics, and rank-based correlations between price and trading activity. The consistently strong RankIC and RankIR values highlight the framework's capability to mine alpha signals that are both effective and temporally stable.

Efficiency Analysis

A key advantage of the *Chain-of-Alpha* framework lies in its efficiency across the entire alpha mining pipeline. Unlike previous approaches that rely on heavy human involvement (e.g., AlphaGPT, AlphaAgent) or computationally expensive reasoning procedures (e.g., *LLM+ToT*, *LLM+MCTS*), our framework adopts a modular dual-chain design that enables end-to-end automation, scalability, and cost-effectiveness.

In particular, the *Factor Optimization Chain* is designed for high parallelism. Each seed factor is independently refined based on backtesting feedback, allowing multiple optimization chains to run concurrently without mutual interference. Let K denote the number of generated seed factors and m the maximum number of optimization steps per factor. The overall complexity of the optimization stage is $\mathcal{O}(Km)$, which is linear and embarrassingly parallel. In practice, m is kept small (e.g., $m \leq 5$) to ensure rapid convergence with minimal overhead.

By contrast, tree-based exploration methods such as *LLM+ToT* require sequential traversal of a reasoning tree with branching factor b and depth d, resulting in a time complexity of $\mathcal{O}(b^d)$ per factor. This not only increases the runtime exponentially but also limits parallelism due to dependency between nodes.

Overall, our dual-chain design is not only effective but also highly efficient. Its parallelizable optimization chains make it especially suitable for large-scale factor generation.

Conclusion

In this paper, we introduce *Chain-of-Alpha*, a LLM-powered framework designed to automatically mine alpha factors for quantitative financial trading. The framework is novel, simple, yet highly effective and efficient. Unlike traditional LLM-based reasoning methods, as introduced in previous related work such as CoT, ToT, and MCTS, our dual-chain design offers a more scalable and streamlined approach to alpha discovery.

Experimental results demonstrate the effectiveness of the *Chain-of-Alpha* and highlight the potential of LLMs in quantitative finance, marking a significant step toward the emerging field of LLM-driven quantitative research.

References

Besta, M.; Blach, N.; Kubicek, A.; Gerstenberger, R.; Podstawski, M.; Gianinazzi, L.; Gajda, J.; Lehmann, T.; Niewiadomski, H.; Nyczyk, P.; and Hoefler, T. 2024. Graph of Thoughts: Solving Elaborate Problems with Large Language Models. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(16): 17682–17690.

Cao, B.; Wang, S.; Lin, X.; Wu, X.; Zhang, H.; Ni, L. M.; and Guo, J. 2025. From Deep Learning to LLMs: A survey of AI in Quantitative Investment. arXiv:2503.21422.

Cao, L. 2024. GraphReason: Enhancing Reasoning Capabilities of Large Language Models through A Graph-Based Verification Approach. In Dalvi Mishra, B.; Durrett, G.; Jansen, P.; Lipkin, B.; Neves Ribeiro, D.; Wong, L.; Ye, X.; and Zhao, W., eds., *Proceedings of the 2nd Workshop on Natural Language Reasoning and Structured Explanations (@ACL 2024)*, 1–12. Bangkok, Thailand: Association for Computational Linguistics.

Cao, L.; Zhang, S.; and Chen, J. 2021. CBCP: A method of causality extraction from unstructured financial text. In *Proceedings of the 2021 5th International Conference on Natural Language Processing and Information Retrieval*, 135–140.

Cui, C.; Wang, W.; Zhang, M.; Chen, G.; Luo, Z.; and Ooi, B. C. 2021. AlphaEvolve: A learning framework to discover novel alphas in quantitative investment. In *Proceedings of the 2021 ACM SIGMOD International Conference on Management of Data*, 2208–2216.

Duan, Y.; Wang, L.; Zhang, Q.; and Li, J. 2022. 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, 4468–4476.

Fama, E. F.; and French, K. R. 2004. The capital asset pricing model: Theory and evidence. *Journal of economic perspectives*, 18(3): 25–46.

Gunasekar, S.; Zhang, Y.; Aneja, J.; Mendes, C. C. T.; Giorno, A. D.; Gopi, S.; Javaheripi, M.; Kauffmann, P.; de Rosa, G.; Saarikivi, O.; Salim, A.; Shah, S.; Behl, H. S.; Wang, X.; Bubeck, S.; Eldan, R.; Kalai, A. T.; Lee, Y. T.; and Li, Y. 2023. Textbooks Are All You Need. arXiv:2306.11644.

Guo, T.; Shen, H.; Huang, J.; Mao, Z.; Luo, J.; Chen, Z.; Liu, X.; Xia, B.; Liu, L.; Ma, Y.; and Zhang, M. 2025. MASS: Multi-Agent Simulation Scaling for Portfolio Construction. arXiv:2505.10278.

Hou, K.; Karolyi, G. A.; and Kho, B.-C. 2011. What factors drive global stock returns? *The Review of Financial Studies*, 24(8): 2527–2574.

Jiang, J.; Chen, J.; Li, J.; Ren, R.; Wang, S.; Zhao, W. X.; Song, Y.; and Zhang, T. 2024. RAG-Star: Enhancing Deliberative Reasoning with Retrieval Augmented Verification and Refinement. *arXiv preprint arXiv:2412.12881*.

Kakushadze, Z. 2016. 101 Formulaic Alphas. arXiv:1601.00991.

Ke, G.; Meng, Q.; Finley, T.; Wang, T.; Chen, W.; Ma, W.; Ye, Q.; and Liu, T.-Y. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In Guyon, I.; Luxburg, U. V.; Bengio, S.; Wallach, H.; Fergus, R.; Vishwanathan, S.; and Garnett, R., eds., *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Kou, Z.; Yu, H.; Luo, J.; Peng, J.; Li, X.; Liu, C.; Dai, J.; Chen, L.; Han, S.; and Guo, Y. 2025. Automate Strategy Finding with LLM in Quant Investment. arXiv:2409.06289.

Landajuela, M.; Lee, C. S.; Yang, J.; Glatt, R.; Santiago, C. P.; Aravena, I.; Mundhenk, T.; Mulcahy, G.; and Petersen, B. K. 2022. A Unified Framework for Deep Symbolic Regression. In Koyejo, S.; Mohamed, S.; Agarwal, A.; Belgrave, D.; Cho, K.; and Oh, A., eds., *Advances in Neural Information Processing Systems*, volume 35, 33985–33998. Curran Associates, Inc.

Li, Z.; Song, R.; Sun, C.; Xu, W.; Yu, Z.; and Wen, J.-R. 2024. Can Large Language Models Mine Interpretable Financial Factors More Effectively? A Neural-Symbolic Factor Mining Agent Model. In Ku, L.-W.; Martins, A.; and Srikumar, V., eds., *Findings of the Association for Computational Linguistics: ACL 2024*, 3891–3902. Bangkok, Thailand: Association for Computational Linguistics.

Lin, X.; Chen, Y.; Li, Z.; and He, K. 2019. Stock alpha mining based on genetic programming. Technical report, Huatai Securities Research Center.

Liu, K.-M.; and Lo, M.-C. 2025. LLM-Based Routing in Mixture of Experts: A Novel Framework for Trading. arXiv:2501.09636.

Ma, Y.; Gou, Z.; Hao, J.; Xu, R.; Wang, S.; Pan, L.; Yang, Y.; Cao, Y.; Sun, A.; et al. 2024. SciAgent: Tool-augmented Language Models for Scientific Reasoning. *Proceedings of EMNLP*.

Ng, V.; Engle, R. F.; and Rothschild, M. 1992. A multi-dynamic-factor model for stock returns. *Journal of Econometrics*, 52(1-2): 245–266.

OpenAI. 2024. GPT-4 Technical Report. arXiv:2303.08774.

Plaat, A.; Wong, A.; Verberne, S.; Broekens, J.; van Stein, N.; and Back, T. 2024. Reasoning with Large Language Models, a Survey. arXiv:2407.11511.

Rao, V. N.; Agarwal, E.; Dalal, S.; Calacci, D.; and Monroy-Hernández, A. 2025. QuaLLM: An LLM-based Framework to Extract Quantitative Insights from Online Forums. arXiv:2405.05345.

Schick, T.; Dwivedi-Yu, J.; Dessì, R.; Raileanu, R.; Lomeli, M.; Zettlemoyer, L.; Cancedda, N.; and Scialom, T. 2023. Toolformer: Language Models Can Teach Themselves to Use Tools. *arXiv preprint arXiv:2302.04761*.

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

Shi, H.; Song, W.; Zhang, X.; Shi, J.; Luo, C.; Ao, X.; Arian, H.; and Seco, L. 2024. AlphaForge: A Framework to Mine and Dynamically Combine Formulaic Alpha Factors. arXiv:2406.18394.

Shi, Y.; Duan, Y.; and Li, J. 2025. Navigating the Alpha Jungle: An LLM-Powered MCTS Framework for Formulaic Factor Mining. arXiv:2505.11122.

Suzgun, M.; Scales, N.; Schärli, N.; Gehrmann, S.; Tay, Y.; Chung, H. W.; Chowdhery, A.; Le, Q. V.; Chi, E. H.; Zhou, D.; and Wei, J. 2022. Challenging BIG-Bench Tasks and Whether Chain-of-Thought Can Solve Them. arXiv:2210.09261.

Tang, Z.; Chen, Z.; Yang, J.; Mai, J.; Zheng, Y.; Wang, K.; Chen, J.; and Lin, L. 2025. AlphaAgent: LLM-Driven Alpha Mining with Regularized Exploration to Counteract Alpha Decay. arXiv:2502.16789.

Wang, S.; Yuan, H.; Ni, L. M.; and Guo, J. 2024. QuantAgent: Seeking Holy Grail in Trading by Self-Improving Large Language Model. arXiv:2402.03755.

Wang, S.; Yuan, H.; Zhou, L.; Ni, L. M.; Shum, H.-Y.; and Guo, J. 2023a. Alpha-GPT: Human-AI Interactive Alpha Mining for Quantitative Investment. arXiv:2308.00016.

Wang, X.; Wei, J.; Schuurmans, D.; Le, Q.; Chi, E.; Narang, S.; Chowdhery, A.; and Zhou, D. 2023b. Self-Consistency Improves Chain of Thought Reasoning in Language Models. arXiv:2203.11171.

Wei, J.; Tay, Y.; Bommasani, R.; Raffel, C.; Zoph, B.; Borgeaud, S.; Yogatama, D.; Bosma, M.; Zhou, D.; Metzler, D.; Chi, E. H.; Hashimoto, T.; Vinyals, O.; Liang, P.; Dean, J.; and Fedus, W. 2022. Emergent Abilities of Large Language Models. arXiv:2206.07682.

Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E.; Le, Q.; and Zhou, D. 2023. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. arXiv:2201.11903.

Xiao, Y.; Sun, E.; Luo, D.; and Wang, W. 2025. TradingAgents: Multi-Agents LLM Financial Trading Framework. arXiv:2412.20138.

Xu, W.; Liu, W.; Xu, C.; Bian, J.; Yin, J.; and Liu, T.-Y. 2021. Rest: Relational event-driven stock trend forecasting. In *Proceedings of the web conference 2021*, 1–10.

Yang, H.; Liu, X.-Y.; and Wang, C. D. 2023. FinGPT: Open-Source Financial Large Language Models. arXiv:2306.06031.

Yang, X.; Liu, W.; Zhou, D.; Bian, J.; and Liu, T.-Y. 2020. Qlib: An AI-oriented Quantitative Investment Platform. arXiv:2009.11189.

Yao, S.; Yu, D.; Zhao, J.; Shafran, I.; Griffiths, T. L.; Cao, Y.; and Narasimhan, K. 2023a. Tree of Thoughts: Deliberate Problem Solving with Large Language Models. arXiv:2305.10601.

Yao, S.; Zhao, J.; Yu, D.; Du, N.; Shafran, I.; Narasimhan, K.; and Cao, Y. 2023b. ReAct: Synergizing Reasoning and Acting in Language Models. In *International Conference on Learning Representations (ICLR)*.

Yu, S.; Xue, H.; Ao, X.; Pan, F.; He, J.; Tu, D.; and He, Q. 2023a. Generating Synergistic Formulaic Alpha Collections via Reinforcement Learning. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 5476–5486.

Yu, Y.; Li, H.; Chen, Z.; Jiang, Y.; Li, Y.; Zhang, D.; Liu, R.; Suchow, J. W.; and Khashanah, K. 2023b. FinMem: A Performance-Enhanced LLM Trading Agent with Layered Memory and Character Design. arXiv:2311.13743.

Yuan, H.; Wang, S.; and Guo, J. 2024. Alpha-GPT 2.0: Human-in-the-Loop AI for Quantitative Investment. arXiv:2402.09746.

Zhang, T.; Li, Y.; Jin, Y.; and Li, J. 2020. AutoAlpha: an efficient hierarchical evolutionary algorithm for mining alpha factors in quantitative investment. *arXiv preprint arXiv:2002.08245*.

Additional Experimental Settings

Alpha Selection Criteria. To assess the quality of generated alpha factors, we apply the following quantitative criteria during evaluation:

- **Strength**: RankIC ≥ 0.015, indicating reliable predictive power.
- **Consistency**: RankICIR ≥ 0.2, ensuring temporal stability of performance.
- **Efficiency**: Turnover ≤ 1.5, controlling for excessive trading costs.
- **Diversity**: Diversity ≥ 0.2, promoting factor novelty and complementarity.

Based on these criteria, we select the top-K alpha factors with the highest *RankIC* from the candidate pool to construct the final alpha set for integration and prediction model training. By default, K is set to 100 in our experiments, though both the threshold values and K can be adjusted depending on specific applications or deployment settings.

Model Settings. For all LLM-based operations, we utilize *GPT-4o* (`gpt-4o-2024-11-20`) from Azure OpenAI by default. The temperature is set to 1.0 to encourage response diversity, while all other generation parameters remain at their default settings.

For the alpha integration and prediction modeling stage, we adopt the LightGBM (Ke et al. 2017) framework with the following hyperparameters: mean squared error (MSE) as the loss function, early stopping after 200 rounds, 24 leaves per tree, a maximum of 2000 estimators, maximum tree depth of 8, learning rate of 0.005, L_1 regularization coefficient $\alpha = 0.1$, L_2 regularization coefficient $\lambda = 0.1$, and evaluation every 10 steps. These settings are chosen to ensure stable training and robust generalization performance.

Backtesting Strategy. We primarily use Qlib (Yang et al. 2020) to support both alpha factor evaluation and final strategy backtesting. The backtests adopt a top-k/drop-n portfolio construction strategy configured as follows. On each trading day, an equal-weighted portfolio is constructed by selecting the top k stocks based on the predictive scores produced by the trained model, where k corresponds to the top 10% of the stock universe. To control turnover and mitigate transaction costs, at most n stocks are allowed to be bought or sold per day, with n computed as $n = k/w$, where w is the prediction horizon. All trades are executed at the closing price. Transaction costs are accounted for using an opening cost of 0.03% and a closing cost of 0.1% per trade. These settings ensure a realistic evaluation of strategy performance under practical trading conditions.

Evaluation Metrics

To quantitatively assess the predictive performance and trading quality of the mined alpha factors and their integrated portfolio, we adopt several widely used evaluation metrics. These include the Information Coefficient (IC), Rank Information Coefficient (RankIC), their corresponding Information Ratios (ICIR, RankICIR), the Annualized Return (AR), and the Information Ratio (IR) of the trading strategy.

Information Coefficient (IC). The IC assesses the degree of linear association between model-generated alpha scores and the actual realized returns on a cross-sectional basis. At each time step t, let $f_{i,t}$ denote the predicted signal for stock i and $r_{i,t+1}$ the corresponding realized return in the next period. The IC at time t is calculated as:

$$\text{IC}t = \text{Corr}(f1 : N_t, t,\ r_{1:N_t,t+1}), \tag{11}$$

where N_t is the number of available stocks at time t, and $\text{Corr}(\cdot)$ denotes the Pearson correlation. The average IC across T trading periods is given by:

$$\text{IC} = \frac{1}{T}\sum_{t=1}^{T}\text{IC}_t. \tag{12}$$

Rank Information Coefficient (RankIC). The RankIC evaluates the monotonic relationship between predicted and realized returns, making it less sensitive to outliers. It is computed as the Spearman rank correlation at each time point:

$$\text{RankIC}t = \text{Corr}(\text{rank}(f1 : N_t, t),\ \text{rank}(r_{1:N_t,t+1})), \tag{13}$$

and averaged across the evaluation window:

$$\text{RankIC} = \frac{1}{T}\sum_{t=1}^{T}\text{RankIC}_t. \tag{14}$$

Information Coefficient Information Ratio (ICIR). To assess the consistency of IC over time, we compute the ICIR as the mean IC normalized by its temporal standard deviation:

$$\text{ICIR} = \frac{\text{IC}}{\text{Std}(\text{IC}_t)}, \tag{15}$$

where $\text{Std}(\text{IC}_t)$ denotes the standard deviation of IC across the T periods. A higher ICIR indicates more stable predictive strength across time.

Rank Information Coefficient Information Ratio (RankICIR). Analogously, RankICIR measures the temporal robustness of RankIC by evaluating its signal-to-noise ratio:

$$\text{RankICIR} = \frac{\text{RankIC}}{\text{Std}(\text{RankIC}_t)}, \tag{16}$$

where $\text{Std}(\text{RankIC}_t)$ is the standard deviation of RankIC over time. Higher values imply stronger and more consistent ordinal predictive relationships.

Annualized Return (AR). AR reflects the average yearly return achieved by a trading strategy. For a top-k long-only approach, where k stocks with the highest predicted returns are equally weighted at each rebalancing time t, the portfolio return for the subsequent period is:

$$R_{p,t+1} = \frac{1}{k}\sum_{i\in\text{Top}k} ri, t+1. \tag{17}$$

The annualized return over T_p holding periods is then computed as:

$$\text{AR} = \left(\frac{1}{T_p}\sum_{j=1}^{T_p} R_{p,j}\right) \times P, \tag{18}$$

where P is the number of trading intervals in a year (e.g., $P = 252$ for daily rebalancing).

Information Ratio (IR). IR evaluates the return-to-risk efficiency of the strategy by dividing the annualized return by its annualized volatility:

$$\text{IR} = \frac{\text{AR}}{\sigma(R_p)\sqrt{P}}, \tag{19}$$

where $\sigma(R_p)$ denotes the standard deviation of portfolio returns over the T_p periods. A higher IR implies better risk-adjusted performance.

Factor Turnover (Efficiency). Turnover measures the frequency of changes in the portfolio composition and reflects trading intensity. For a top-k long-only strategy, the daily turnover is computed as the proportion of assets that enter or exit the top-k set between consecutive trading days. Formally, let M_t be the set of top-k assets at time t, the turnover at time t is given by:

$$\text{Turnover}t = \frac{|\text{Top}k_t \triangle \text{Top}kt-1|}{k}, \tag{20}$$

where $\triangle$ denotes the symmetric difference between the top-k sets at time t and $t-1$. The average turnover over the evaluation period reflects the expected trading volume per rebalancing. Lower turnover is generally preferred, as it implies reduced transaction costs and greater operational efficiency.

Factor Diversity. Diversity quantifies how different a candidate alpha factor is from existing ones. We define it based on the average of the top-k absolute Spearman correlations between the candidate factor f and a set of reference factors $g_1, ..., g_n$, then subtract from 1 to represent dissimilarity:

$$\text{Diversity}(f) = 1 - \frac{1}{k}\sum_{i=1}^{k} |\rho(f, g_i)|, \tag{21}$$

where $\rho(f, g_i)$ is the average Spearman correlation between f and g_i over all trading days. A higher diversity score indicates greater uniqueness and potential orthogonality with respect to the existing factor pool, which is beneficial for portfolio diversification.

List of Data Fields

Table 5 presents a complete list of data fields available for constructing formulaic alpha factor expressions.

Field Symbol	Field Name	Description
`$open`	Open	Opening price of the stock
`$high`	High	Highest price of the stock
`$low`	Low	Lowest price of the stock
`$close`	Close	Closing price of the stock
`$volume`	Volume	Trading volume of the stock
`$amount`	Amount	Trading amount of the stock
`$change`	Change	Price change of the stock
`$vwap`	VWAP	Volume-weighted average price of the stock

Table 5: List of available data fields for constructing formulaic alpha factor expressions.

List of Operators

Table 6 presents a complete list of data fields available for constructing formulaic alpha factor expressions.

Prompts

Figure 4 and Figure 5 illustrate the demo versions of the prompt templates used for seed factor generation and factor optimization, respectively. These prompts are designed to guide the large language model in generating effective and interpretable alpha expressions. To ensure consistently high-quality factor generation, more carefully crafted prompts are needed. In each template, the blue-highlighted text indicates variable slots that are dynamically filled at runtime based on user input, contextual information, or previously generated results.

Category	Operator Symbol	Operator Name	Description
Mathematical	`Add(x, y)`	Addition	Element-wise addition of x and y
	`Sub(x, y)`	Subtraction	Element-wise subtraction of y from x
	`Mul(x, y)`	Multiplication	Element-wise multiplication of x and y
	`Div(x, y)`	Division	Element-wise division of x by y
	`Log(x)`	Logarithm	Natural logarithm of x
	`Abs(x)`	Absolute Value	Absolute value of x
	`Power(x, n)`	Exponentiation	Raise x to the power of n
	`Sign(x)`	Sign	Sign of x (+1, -1, or 0)
Time Series (rolling)	`Mean(x, N)`	Rolling Mean	Mean of x over past N days
	`Std(x, N)`	Rolling Std	Standard deviation over N days
	`Var(x, N)`	Rolling Variance	Variance over N days
	`Sum(x, N)`	Rolling Sum	Sum over N days
	`Max(x, N)`	Rolling Max	Maximum value over N days
	`Min(x, N)`	Rolling Min	Minimum value over N days
	`Med(x, N)`	Median	Median over N days
	`Mad(x, N)`	Mean Abs Dev	Mean absolute deviation over N days
	`Rank(x, N)`	Percentile Rank	Percentile rank in N-day window
	`Quantile(x, N, q)`	Quantile	q-quantile over N days
	`Count(x, N)`	Valid Count	Number of valid values in N days
	`Ref(x, N)`	Lag	Value N days ago
	`Delta(x, N)`	Change	Difference from N days ago
	`IdxMax(x, N)`	Index of Max	Position of max value in window
	`IdxMin(x, N)`	Index of Min	Position of min value in window
Regression (rolling)	`Resi(x, N)`	Residual	Residual of regression of x over N days
	`Slope(x, N)`	Slope	Regression slope over N days
	`Rsquare(x, N)`	R^2	Coefficient of determination
Statistical (rolling)	`Skew(x, N)`	Skewness	Skewness over N days
	`Kurt(x, N)`	Kurtosis	Kurtosis over N days
	`Corr(x, y, N)`	Correlation	Correlation between x and y
	`Cov(x, y, N)`	Covariance	Covariance between x and y
Conditional	`If(cond, x, y)`	Conditional	If condition is true, return x, else y
	`Gt(x, y)`	Greater Than	1 if $x > y$, else 0
	`Lt(x, y)`	Less Than	1 if $x < y$, else 0
	`Ge(x, y)`	Greater Equal	1 if $x \geq y$, else 0
	`Le(x, y)`	Less Equal	1 if $x \leq y$, else 0
	`Eq(x, y)`	Equal	1 if $x = y$, else 0
	`Ne(x, y)`	Not Equal	1 if $x \neq y$, else 0
Logical	`And(x, y)`	Logical AND	Logical AND between x and y
	`Or(x, y)`	Logical OR	Logical OR between x and y
	`Not(x)`	Logical NOT	Logical NOT of x

Table 6: Categorized list of supported operators for constructing formulaic alpha factors.

Prompt of Seed Factor Generation

 # Objective
You are an expert in alpha factor generation for quantitative trading.

*Your task is to design a new **cross-sectional alpha factor**—a mathematical expression that assigns a score to each stock on a given trading day based on its recent market data.*
This score will be used to rank and select stocks on the same day (i.e., across the market cross-section).

You are provided with:
- A list of available data fields and operators.
*- Example sets of **effective** and **non-effective** factors.*

Your goal is to generate a factor expression that is:
- Distinct from the examples
- Inspired by effective ones
- Potentially more predictive than the non-effective ones

 # Available Data Fields
You may use the following data fields to construct the expression.
Make sure to include a dollar sign ($) before each field name in your expression:
{available_data_fields}

 # Available Operators
You may use the following operators to construct your factor expression:
{available_operators}

 # Reference Factors
Use the following for reference:
Effective factors:
{effective_factors}

Non-effective factors:
{non_effective_factors}

 # Requirements
1. Take inspiration from effective factors and avoid patterns seen in non-effective ones (e.g., poor RankIC or IR).
2. The factor should produce a unitless value, independent of price scale or volume units.
*3. Use **only** the provided data fields and operators.*
*4. You are **not** required to use all data fields. The factor expression should have NO MORE than 3 data fields in total.*
5. Avoid excessive operator nesting or overly complex expressions to reduce overfitting risk.
6. The expression should be concise, readable, and interpretable.

 # Response Format
You need to return a JSON object with the following fields:
- factor_name: a brief name of the factor
- factor_expression: the expression of the factor
- description: a 1-2 sentence description of the factor's intuition or financial meaning

Figure 3: The prompt of seed factor generation.

Prompt of Factor Optimization

 # Objective
You are an expert in alpha factor optimization for quantitative trading.

*Your task is to optimize an existing **cross-sectional alpha factor**—a mathematical expression that assigns a score to each stock on a given trading day based on its recent market data.*
This score will be used to rank and select stocks on the same day (i.e., across the market cross-section).

You are provided with:
- A list of available data fields and operators.
- The existing factor expression and its performance (e.g., RankIC, RankIR, Turnover, Diversity).
- The optimization history of the factor.

Your goal is to optimize the factor expression to improve its performance.

 # Available Data Fields
You may use the following data fields to construct the expression.
Make sure to include a dollar sign ($) before each field name in your expression:
{available_data_fields}

 # Available Operators
You may use the following operators to construct your factor expression:
{available_operators}

 # Existing Factor Information
The factor name is: {factor_name}
The factor expression is: {factor_expression}
The factor description is: {description}
The RankIC is: {rankic}
The RankIR is: {rankicir}
The Turnover is: {turnover}
The Diversity is: {diversity}

 # Optimization History
The optimization history is:
{optimization_history}

 # Requirements
1. A good factor should have: RankIC > 0.015, RankIR > 0.2, Turnover < 1.5, Diversity < 0.8.
2. Your goal is to optimize the factor based on these performance metrics. Typically, improving RankIC and RankIR is the primary focus; turnover and diversity are secondary.
3. The factor should produce a unitless value, independent of price scale or volume units.
*4. Use **only** the provided data fields and operators.*
*5. You are **not** required to use all data fields. The factor expression should have NO MORE than 3 data fields in total.*
6. Avoid excessive operator nesting or overly complex expressions to reduce overfitting risk.
7. The expression should be concise, readable, and interpretable.
8. Review the optimization history to avoid repeating previous attempts.
9. Inspect the factor expression to guard against overfitting.

 # Response Format
You need to return a JSON object with the following fields:
- factor_name: a brief name of the factor
- factor_expression: the expression of the factor
- description: a 1-2 sentence description of the factor's intuition or financial meaning
- reason: a short explanation of why this optimization was made, such as which metric was improved or which weakness in the previous version was addressed

Figure 4: The prompt of factor optimization.