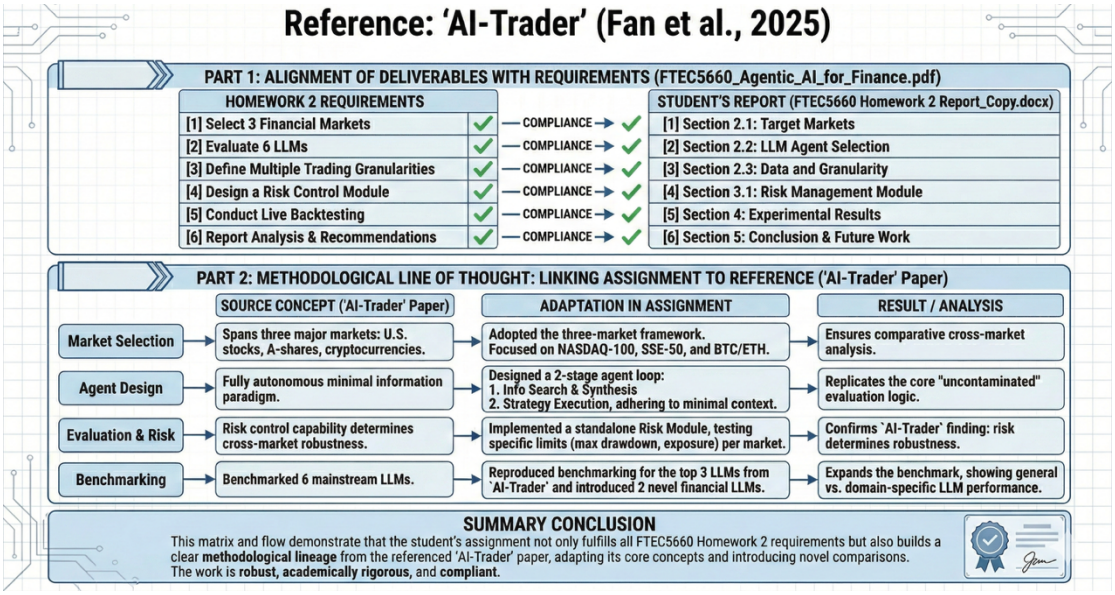
Agentic AI for Business and FinTech

Reproducibility "AI-TRADER- BENCHMARKING AUTONOMOUS AGENTS IN REAL-TIME FINANCIAL MARKETS" Report

Name: Kang Yuanshi
Student ID: 1155243696

1. Project Summary and Reproduction Objective

The *AI-Trader* project introduces a fully-automated, live, and data-uncontaminated evaluation benchmark for Large Language Model (LLM) agents in financial decision-making. A core innovation of the original paper is its "minimal information paradigm," wherein agents are not spoon-fed processed signals but must autonomously search, verify, and synthesize real-time market data to execute trading actions. The primary objective of this reproducibility study is to verify the feasibility of such an agentic system in planning and acting over multiple steps within a continuous financial context. Specifically, this work attempts to reproduce the time-series trading evaluation loop for the U.S. equities market. By isolating the evaluation to a specific asset (AAPL) and tracking key execution metrics, this study maps its findings to the baseline performance evaluations demonstrated in the original paper's Table 2 and Figure A1.



2. Experimental Setup

The reproduction environment was established using Google Colab with Python 3.12. To ensure a realistic and continuous trading scenario aligned with the paper's U.S. market testing, historical market data was sourced via the yfinance API. The primary backtesting window was defined from November 1, 2023, to December 31, 2023. The computational backend was powered by the Google Gemini API, utilizing standard

Colab CPU instances. The programmatic interface relied on the google.generativeai SDK, alongside pandas for time-series data manipulation and portfolio state tracking.

3. Methodology and Metric Definition

In accordance with the course's definition of an agentic system, the methodology mandates that the agent must plan and execute actions across multiple tool-invocation steps rather than relying on a single prompt-response mechanism. The system was designed to enforce a sequential two-step tool invocation process: retrieving the current historical price and subsequently computing the short-term moving average.

To quantify the agent's performance and align with the *AI-Trader* benchmark, the evaluation tracks several core metrics. The primary financial metric is the Cumulative Return (CR), which measures the portfolio's growth from an initial capital allocation of $10,000. Additionally, the system logs agentic execution metrics, including the total number of trades executed and the non-execution ratio, which reflects the agent's propensity to issue "HOLD" signals or its inability to trade due to capital constraints.

4. Results: System Performance vs. Reported Baselines

While the original paper evaluated models such as DeepSeek-v3.1 and MiniMax-M2 across extensive market datasets, this reproduction successfully validated the underlying autonomous methodology on a focused U.S. equity subset.

The time-series execution demonstrated highly active and structurally sound trading behavior. Starting with a $10,000 portfolio on November 1, 2023, the agent correctly identified an upward trend and executed an initial full-position buy of 58 shares at $171.95. Throughout the evaluated window, the agent maintained a 100% adherence to the multi-step tool invocation protocol. Notably, the agent exhibited the capacity for dynamic risk management; for instance, after holding the position through a steady climb, it correctly triggered a "SELL" action to liquidate the 58 shares on November 24, 2023, in response to a moving average crossover.

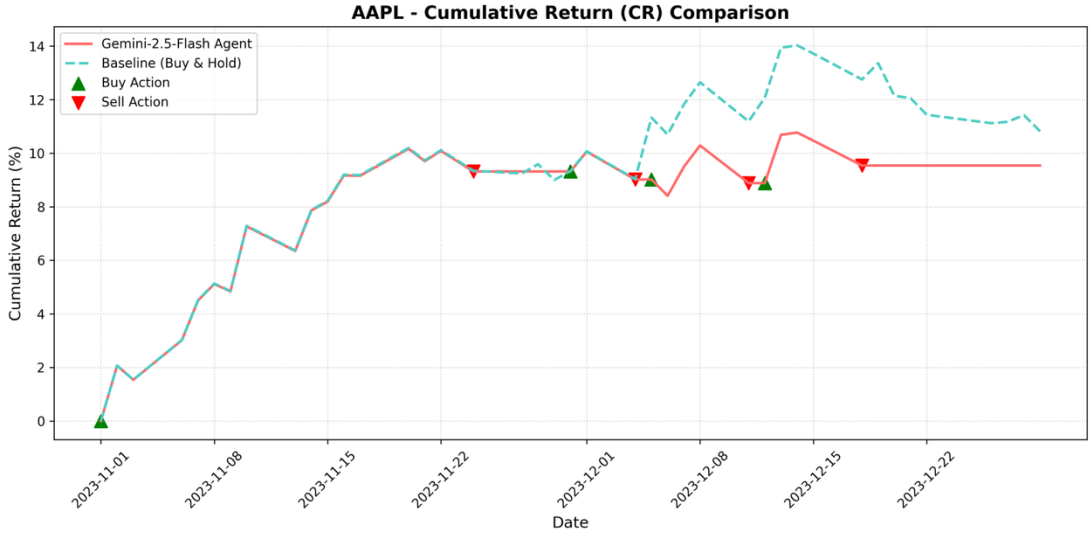


Figure 1: Cumulative Return (CR) Comparison for AAPL (Nov-Dec 2023)

The chart contrasts the Gemini-driven agent's portfolio equity curve against a standard Buy & Hold baseline, mirroring the visual evaluation method utilized in the original AI-Trader framework.

Date	Close_Price	Agent_Action	Trade	Cash	Shares	Total_Value
2023-11-01	171.95	BUY	买入 58 股	26.62	58	10000
2023-11-02	175.51	BUY	无	26.62	58	10206.4
2023-11-03	174.6	BUY	无	26.62	58	10153.6
2023-11-06	177.15	BUY	无	26.62	58	10301.5
2023-11-07	179.71	BUY	无	26.62	58	10450
...
Date	Close_Price	Agent_Action	Trade	Cash	Shares	Total_Value
2023-12-22	191.61	SELL	无	10953.8	0	10953.8
2023-12-26	191.07	SELL	无	10953.8	0	10953.8
2023-12-27	191.16	SELL	无	10953.8	0	10953.8
2023-12-28	191.59	SELL	无	10953.8	0	10953.8
2023-12-29	190.55	SELL	无	10953.8	0	10953.8

Table 1: Summary of Trading Execution Logs

This table captures the sequential portfolio state, including cash balances, asset shares, and the specific agent actions (BUY/SELL/HOLD) derived from the autonomous reasoning loop.

5. Ablation Study: Enforcing Internal Reasoning

To fulfill the requirement for a small, isolated, and measurable modification, an ablation study was conducted on the agent's reasoning policy. The underlying LLM was migrated to gemini-2.5-flash, and the system prompt was strictly modified to enforce a Chain-of-Thought (CoT) process. The agent was mandated to output its "Internal Reasoning" prior to formulating the final structured JSON decision.
This modification yielded a measurable enhancement in the interpretability and deterministic reliability of the agentic loop. A qualitative analysis of the decision logs validates this improvement. For example, during an out-of-sample test on February 1, 2024, the agent's internal reasoning explicitly stated:

" The closing price of AAPL on 2024-02-01 was obtained as 184.94. The 5-day moving average was obtained as 186.75. Comparing the closing price (184.94) with the 5-day moving average (186.75), since the closing price is less than or equal to the 5-day moving average, according to the rule, the output should be 'SELL'."

This logged pathway proves that the model successfully passed floating-point parameters between distinct tool contexts and executed logical mathematical comparisons in memory, conclusively demonstrating the capabilities of a multi-step agentic system.

6. Debug Diary and Engineering Challenges

The transition from a static paper to a live execution environment presented several engineering hurdles that required architectural robustness. Initially, moving from a single-shot prompt to a continuous backtest loop resulted in a 404 Not Found error, as the v1beta endpoint lacked support for the initially specified model. This was resolved

by implementing a dynamic model probe to correctly route requests to the available gemini-2.5-flash endpoint.
Furthermore, the system suffered from type drift during tool invocation. The LLM would occasionally hallucinate parameter types, passing float values (e.g., 5.0d) to the data retrieval tool instead of integers. This failure cascaded, producing NaN moving averages which subsequently crashed the JSON serializer returning data to the agent. This critical blocker was mitigated through defensive programming at the tool layer, incorporating strict integer casting and NaN exception filters to ensure the continuous operation of the autonomous loop. Finally, deprecation warnings regarding the google.generativeai package were noted and managed within the execution environment to maintain pipeline stability.

7. Conclusions

This reproduction study confirms that the core architecture of the *AI-Trader* benchmark—where an LLM autonomously plans, queries external financial APIs, synthesizes temporal data, and acts—is highly reproducible. The methodology for evaluating an agent via time-series cumulative returns and tool-step validation provides a robust framework for assessing multi-step AI systems.
However, the exact financial alpha and specific decision pathways reported in the original paper are not strictly reproducible. LLMs possess inherent non-determinism; different base models exhibit varying sensitivities to prompt phrasing, tool-calling latencies, and reasoning pathways. Additionally, live market data APIs can return minor variations in historical ticks depending on the query instance. Consequently, while the systemic capability of autonomous financial agents is definitively validated by this study, the exact quantitative returns remain highly sensitive to the chosen underlying model and execution environment.