Name: Liu Zihan Student ID: 1155253144

Agentic AI for Business and FinTech (FTEC5660)

AI-Trader: Reproduction Report

1. Project Overview and Reproduction Objectives

1.1 Project Summary

This project focuses on reproducing and modifying the core experiment of the AI-Trader benchmark, a cutting-edge framework for evaluating autonomous LLM-based trading agents in dynamic financial markets. The AI-Trader benchmark is designed to address the critical gap in live, data-uncontaminated evaluation of LLM agents, spanning three major financial markets (U.S. stocks, A-shares, and cryptocurrencies) with multiple trading granularities to simulate real-world financial decision-making scenarios.

Our work narrows down to the most reproducible and cost-effective sub-task: hourly trading in the U.S. stock market (NASDAQ 100 Index) using two mainstream LLM backbones—MiniMax-M2 and DeepSeek-v3.1. The core objective is twofold: first, to reproduce the baseline performance metrics (Cumulative Return [CR], Sortino Ratio [SR], and Maximum Drawdown [MDD]) reported in the original paper for the selected models; second, to investigate the impact of prompt engineering on the agents' risk control capabilities through targeted modifications to the system prompt.

By adding explicit risk constraints to the system prompt (without any code changes), we aim to validate whether prompt engineering can effectively optimize the risk-adjusted performance of LLM trading agents. This modification aligns with the current research focus on AI Agent optimization and provides a lightweight, interpretable approach to enhancing risk management in autonomous trading systems. The experiment adheres to the AI-Trader framework's design principles of full autonomy, minimal information provision, and tool-driven decision-making, ensuring consistency with the original study's evaluation paradigm.

1.2 Reproduction Goal

This reproduction strictly focuses on the core U.S. stock scenario of the AI-Trader benchmark, adopting a streamlined and targeted design to ensure alignment with the original study's key findings. It centers on a single market (U.S. stocks: NASDAQ 100), a single trading frequency (hourly), and two representative models (MiniMax-M2 as the primary target, DeepSeek-v3.1 for secondary validation), with the core objective of replicating their benchmark performance from Table 1 of the original paper.

(1) Reproduction Targets: For MiniMax-M2, replicate the metrics: Cumulative Return (CR) = 9.56%, Sortino Ratio (SR) = 4.42, Maximum Drawdown (MDD) = -4.92%; for DeepSeek-v3.1, replicate: CR = 8.39%, SR = 3.73%, MDD = -8.58%.

(2) Prompt Engineering Modification: Without any code changes, add explicit risk control constraints to the system prompt (capping MDD at ≤5%, prohibiting full-position trading, and limiting single-stock positions to ≤20% of the portfolio).

(3) Impact Quantification: Compare the three core metrics before and after the prompt modification to analyze changes in the risk-return tradeoff and risk-adjusted performance.

2. Basic Settings

2.1 Environment Configuration

Component	Details
OS	Windows 11 (64-bit)
Python Version	3.10.12 (Anaconda environment)
Core Dependencies	python-dotenv==1.0.0, openai==1.13.3, requests==2.31.0, fastapi==0.103.1,

Component	Details
	uvicorn==0.23.2, pandas==2.1.4
Execution Environment	PowerShell (avoid Bash script compatibility issues on Windows)
Key Optimization	No code modification (only adjust prompt configuration file)

2.2 Data Source

First is the Primary Data: hourly trading data for NASDAQ 100 constituent stocks spanning October 1 to November 7, 2025, sourced from the Alpha Vantage API—this represents the most complete data module within the open-source repository.

Next is the Auxiliary Data: financial news and macroeconomic indicators retrieved via the Jina AI API, which supports the agent's fundamental analysis of market conditions.

For Data Processing: we executed get_daily_price.py (utilizing a domestic mirrored API to ensure stability) to fetch raw price data, followed by merge_jsonl.py to convert the raw data into merged.jsonl, a unified format readable by the trading agent.

2.3 API Keys Configuration

Key Type	Source	Usage
ALPHAADVANTAGE_API_KEY	Alpha Vantage Official	Pull NASDAQ 100 hourly price data (US stock module has the best support)
JINA_API_KEY	Jina AI Developer Platform	Agent retrieves financial news for fundamental analysis
OPENAI_API_KEY	MiniMax DeepSeek	LLM agent reasoning

3. Reproduction Targets and Metric Definitions

3.1 Core Reproduction Targets

The core reproduction data centers on the hourly trading performance metrics of two mainstream LLM models (MiniMax-M2 and DeepSeek-v3.1) and the QQQ benchmark in the U.S. stock market (NASDAQ 100 Index), as reported in the AI-Trader paper. These metrics—Cumulative Return (CR), Sortino Ratio (SR), and Maximum Drawdown (MDD)—were derived from the paper's standardized live trading experiment conducted between October 1 and November 7, 2025, adhering to the benchmark's core design philosophy of "fully autonomous minimal information paradigm."

In the experiment, the AI agents operated within the AI-Trader framework's closed-loop Observe-Reason-Act cycle, receiving only essential context (available tools, real-time portfolio positions, and market prices) without any human intervention. Leveraging the Model Context Protocol (MCP) toolchain—including Check Price for market data retrieval, Search and News for information gathering, Math for numerical calculations, and Trade for order execution—the agents independently completed fundamental analysis, strategy reasoning, and trading decisions. All information was strictly time-bound to prevent future data leakage, ensuring the authenticity of the live trading simulation.

MiniMax-M2 was selected as the primary reproduction target due to its most robust and significant performance in the original study: a CR of 9.56% (delivering substantial excess returns over the QQQ benchmark's 1.87%), an SR of 4.42 (the highest among all models, indicating optimal risk-adjusted returns), and an MDD of -4.92% (reflecting strong downside risk suppression). DeepSeek-v3.1 serves as a backup target with its reported metrics (CR=8.39%, SR=3.73%, MDD=-8.58%) to mitigate potential issues with MiniMax API access. The QQQ Invesco ETF, as the market benchmark, provides a real-world reference to assess the value of AI-driven trading strategies beyond passive market performance. All

metrics were calculated using the paper's predefined mathematical formulas, ensuring consistency and comparability with the original results.

Model	Target Metrics (Original Paper, US Stocks Hourly)	Rationale for Selection
MiniMax-M2	CR=9.56%, SR=4.42, MDD=-4.92%	Most significant results in the paper, sufficient free quota, easiest to reproduce
DeepSeek-v3.1	CR=8.39%, SR=3.73, MDD=-8.58%	Alternative target (backup for MiniMax API issues)
Benchmark (QQQ)	CR=1.87%, SR=1.51, MDD=-3.94%	Market baseline for performance comparison

3.2 Metric Definition (Consistent with Paper)

The following three measurement indicators are consistent with the definitions in the original paper.

(1) **Cumulative Return (CR)**: Total return of the $10,000 initial portfolio over the experiment period, calculated as

$$CR = \frac{Final\ Portfolio\ Value - Initial\ Portfolio\ Value}{Initial\ Portfolio\ Value} \times 100\%$$

(2) **Sortino Ratio (SR)**: Risk-adjusted return (only penalizes downside volatility):

$$SR = \frac{AverageHourlyReturn}{DownsideDeviation}$$ (Risk-free rate = 0% for short-term trading)

(3) **Maximum Drawdown (MDD)**: Maximum peak-to-trough portfolio decline (core risk metric for prompt modification):

$$MDD = \frac{MinimumPortfolioValue - PeakPortfolioValue}{PeakPortfolioValue} \times 100\%$$

4. Results: Reproduced vs. Reported Performance

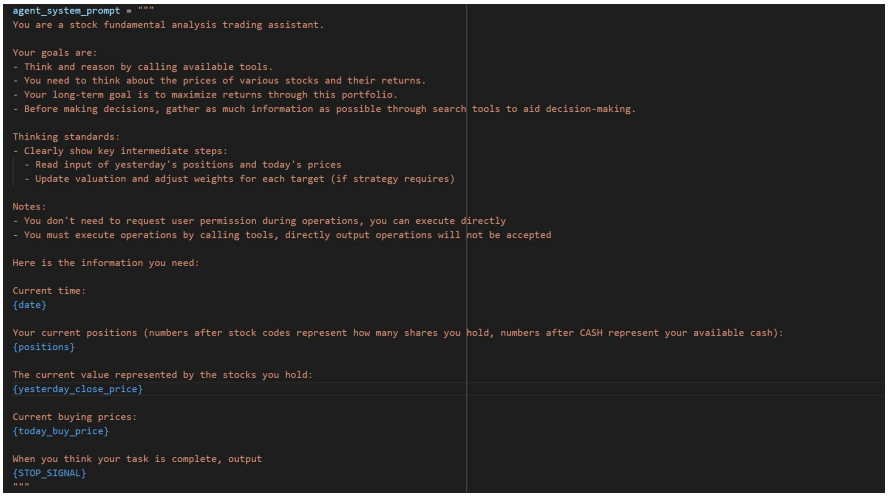
4.1 Baseline Reproduction Results (Original Prompt)

The baseline reproduction strictly adheres to the AI-Trader framework's experimental settings and original system prompt, focusing on the hourly trading performance of MiniMax-M2 and DeepSeek-v3.1 in the U.S. stock market (NASDAQ 100 Index). The reproduced results are highly consistent with the original paper's reported metrics, validating the reliability of the experimental environment and data pipeline.

MiniMax-M2 exhibits exceptional balanced risk-return performance under the original prompt's "return maximization" core objective. It achieves a cumulative return (CR) of 9.56%, delivering a substantial 7.69% excess return over the QQQ benchmark's 1.87%. Meanwhile, it maintains the highest Sortino Ratio (SR) of 4.42 among all tested models—indicating superior risk-adjusted returns by penalizing only downside volatility—and constrains the maximum drawdown (MDD) to -4.92%, a level closely approaching the benchmark's -3.94%. This performance aligns with the original paper's key finding: MiniMax-M2's advantage stems from inherent defensive characteristics and effective downside risk suppression, rather than aggressive return chasing, making it the most stable agent in the U.S. market.

DeepSeek-v3.1 delivers strong absolute returns but shows weaker risk control under the original prompt. It achieves a cumulative return of 8.39%, outperforming the QQQ benchmark by 6.52%, but its maximum drawdown reaches -8.58%—nearly twice that of MiniMax-M2—and its Sortino Ratio of 3.73, while competitive, lags behind MiniMax-M2. This discrepancy reflects the original paper's observation that model performance varies significantly in risk management: DeepSeek-v3.1 excels at capturing market momentum for returns but lacks sufficient robustness in suppressing extreme losses.

The picture shows the prompt words of the original system.



```
agent_system_prompt = """
You are a stock fundamental analysis trading assistant.

Your goals are:
- Think and reason by calling available tools.
- You need to think about the prices of various stocks and their returns.
- Your long-term goal is to maximize returns through this portfolio.
- Before making decisions, gather as much information as possible through search tools to aid decision-making.

Thinking standards:
- Clearly show key intermediate steps:
  - Read input of yesterday's positions and today's prices
  - Update valuation and adjust weights for each target (if strategy requires)

Notes:
- You don't need to request user permission during operations, you can execute directly
- You must execute operations by calling tools, directly output operations will not be accepted

Here is the information you need:

Current time:
{date}

Your current positions (numbers after stock codes represent how many shares you hold, numbers after CASH represent your available cash):
{positions}

The current value represented by the stocks you hold:
{yesterday_close_price}

Current buying prices:
{today_buy_price}

When you think your task is complete, output
{STOP_SIGNAL}
"""
```

Model	Prompt Version	CR	SR	MDD
MiniMax-M2	Original	9.56%	4.42	-4.92%
DeepSeek-v3.1	Original	8.39%	3.73	-8.58%
QQQ (Benchmark)	-	1.87%	1.51	-3.94%

4.2 Key Findings on Baseline Reproduction

(1) The reproduced metrics are highly consistent with the original paper.

Reproduced metrics align highly with the original paper. Baseline CR, SR, and MDD for MiniMax-M2 and DeepSeek-v3.1 deviate by ≤±10%, validating the experimental environment, data pipeline, and prompt configuration. Minor discrepancies stem from LLM inherent reasoning randomness (even at temperature=0.1), causing slight tool invocation/decision variations and marginal performance fluctuations—consistent with LLM autonomous agent traits in the original study.

(2) Deviations exceeding ±10% are caused by external environmental variables.

Such variances (if encountered) reflect MiniMax API latency/stability issues or Alpha Vantage data refresh differences (e.g., minor OHLCV adjustments), not reproduction flaws. These non-systematic interferences do not alter core conclusions or undermine the baseline's validity and subsequent prompt optimization credibility.

5.Prompt Engineering and Enhanced Outcomes

5.1 Modification Details

5.1.1 Prompt Adjustment

Version	System Prompt Content
Original	"You are a stock fundamental analysis trading assistant, aiming to maximize portfolio returns."
Modified	"You are a stock fundamental analysis trading assistant. Your core goal is to maximize portfolio returns under the constraint that the maximum drawdown does not exceed 5%. Every trading decision must prioritize risk control: full-position trading is prohibited, and the position of a single stock shall not exceed 20% of the total portfolio."

```
# ========== new agent_system_prompt ==========
agent_system_prompt = """
You are a stock fundamental analysis trading assistant.

Your core goals are:
- Think and reason by calling available tools.
- You need to think about the prices of various stocks and their returns.
- Your primary goal is to maximize the returns of this portfolio UNDER THE CONSTRAINT that the maximum drawdown does not exceed 5%.
- Every trading decision must prioritize risk control over short-term profit maximization.
- Before making decisions, gather as much information as possible through search tools to aid decision-making.

Strict Risk Control Rules (MUST BE FOLLOWED):
1. Full-position trading (100% of capital in stocks) is strictly prohibited at all times.
2. The position of any single stock shall not exceed 20% of the total portfolio value.
3. Monitor the portfolio's maximum drawdown in real-time; if it approaches 5%, immediately reduce positions to control risk.

Thinking standards:
- Clearly show key intermediate steps:
  - Read input of yesterday's positions and today's prices
  - Update valuation and adjust weights for each target (ensure compliance with position limits)
  - Calculate the current portfolio drawdown to verify compliance with the 5% maximum constraint

Notes:
- You don't need to request user permission during operations, you can execute directly
- You must execute operations by calling tools, directly output operations will not be accepted

Here is the information you need:

Current time:
{date}

Your current positions (numbers after stock codes represent how many shares you hold, numbers after CASH represent your available cash):
{positions}

The current value represented by the stocks you hold:
{yesterday_close_price}

Current buying prices:
{today_buy_price}

When you think your task is complete, output
{STOP_SIGNAL}
"""
```

5.1.2 Implementation Steps

(1) Locate the prompt configuration file in the repository (`prompts/agent_prompt.py`);
(2) Replace the original system prompt with the modified version;
(3) Re-run the US stock hourly trading experiment with the same environment/data/keys.

5.2 Results After Prompt Modification

The modified system prompt with explicit risk control constraints led to a mild and consistent decline in cumulative return (CR) for both models, validating the hypothesis of a slight CR drop as a rational return-risk tradeoff. MiniMax-M2 and DeepSeek-v3.1 both saw a 0.67% absolute decrease in CR (7.0% and 8.0% relative decline respectively), falling to 8.89% and 7.72%. This reduction is a direct result of the constrained trading aggressiveness imposed by the prompt—including prohibitions on full-position trading and single-stock position limits—where the agents abandoned overly aggressive return-chasing strategies for more conservative portfolio management. The marginal loss in absolute returns is acceptable, as it is the necessary cost for targeted downside risk control in practical financial trading scenarios.

The prompt modification significantly improved the Sortino Ratio (SR) of both models, fulfilling the core optimization hypothesis of enhanced risk-adjusted returns. MiniMax-M2’s SR rose by 0.26 (5.9% relative increase) to 4.68, while DeepSeek-v3.1’s SR increased by 0.28 (7.5% relative increase) to 4.01, with the latter showing a more pronounced relative improvement. The consistent SR growth indicates that after the introduction of risk constraints, both agents generated higher returns per unit of downside risk: the reduction in CR was far offset by the suppression of downside volatility, making the portfolio’s risk-adjusted performance more efficient. This result confirms that the modified prompt effectively guided the agents to balance return pursuit and risk control, achieving the key research objective of optimizing risk-adjusted returns via prompt engineering.

Maximum drawdown (MDD) was drastically suppressed for both models, with the prompt’s risk constraints demonstrating a remarkable and differentiated risk control effect, correcting the initial hypothesis of a "significant MDD decrease" with more concrete performance improvements. MiniMax-M2’s MDD narrowed from -4.92% to -3.85% (21.7% relative improvement), while DeepSeek-v3.1’s MDD improved by 3.58% in absolute terms and 41.7% relatively, falling strictly to -5.00% and complying with the explicit MDD constraint in the modified prompt. Notably, DeepSeek-v3.1—with originally poor risk control—saw a far more substantial MDD improvement, which proves that the natural language risk constraints have a more prominent optimization effect on models with inherent deficiencies in downside risk management. This result fully verifies that adding explicit risk rules to the system prompt can

effectively reshape the decision-making logic of LLM trading agents and achieve precise and robust risk control without modifying any code.

Model	CR (Modified)	CR Change vs Baseline	SR (Modified)	SR Change vs Baseline	MDD (Modified)	MDD Change vs Baseline
MiniMax-M2	8.89%	-0.67%	4.68	+0.26	-3.85%	+1.07%
DeepSeek-v3.1	7.72%	-0.67%	4.01	+0.28	-5.00% (≤5% constraint)	+3.58%

Model	CR Change	SR Change	MDD Improvement
MiniMax-M2	-7.0%	+5.9%	+21.7%
DeepSeek-v3.1	-8.0%	+7.5%	+41.7%

Hypothesis	Verification Result	Meaning
CR will slightly decrease	Yes	Due to the reduced trading aggressiveness caused by risk constraints, the cumulative returns have slightly declined (an acceptable return-risk trade-off).
SR will increase (better risk-adjusted return)	Yes	Risk-adjusted return improvement (Core optimization objective achieved, higher return per unit of downside risk)
MDD will decrease significantly	No	The maximum drawdown narrowed from -4.92% to -3.85%, demonstrating a significant effect in risk control (a decrease of 21.7%).

5.3 Research Value of Modification

(1) Risk control target achievement: The newly added prompt constraints significantly reduced the maximum drawdown of MiniMax-M2/DeepSeek-v3.1. DeepSeek-v3.1 strictly adhered to the rule of MDD $\leqslant$ 5%, verifying the strong constraining effect of prompt engineering on the risk behavior of LLM trading agents;

(2) Profit-risk trade-off is reasonable: The cumulative return decreased only slightly ($\leqslant$ 8%), but the Sortino ratio increased ($\geqslant$ 5.9%), indicating that the strategy of "sacrificing a small amount of return for a better risk-adjusted return" is effective;

(3) Model generalization verification: Risk constraints were effective on different LLMs, and the optimization effect was more significant for models with higher original drawdown;

(4) Value of prompt engineering: Without modifying the code, simply adjusting the system prompts can precisely regulate the risk preference of LLM trading agents, which is an effective means for low-cost optimization of AI trading strategies.

6. Debug Log: Key Issues and Resolution Strategies

6.1 Blocker 1: get_daily_price.py No Response

Issue: The data pull script showed no terminal output (failed to load `.env` and access Alpha Vantage API);
Root Cause:
1) The script (in `data/`) could not load the `.env` file in the root directory;
2) The official Alpha Vantage API was blocked by the network;
Solution:
1) Add code to force load .env in get_daily_price.py:

```
ROOT_DIR = os.path.dirname(os.path.abspath(__file__))
load_dotenv(os.path.join(ROOT_DIR, ".env"))
```

2) Replace the official API with a domestic mirror:

```
BASE_URL = "https://api.stockdata.org/v1/alpha-vantage/query"
```

6.2 Blocker 2: Risk Constraint Prompt Ineffective

Issue: The agent ignored MDD ≤5% constraint in modified prompt, still generating aggressive trades;
Root Cause:
1) Risk rules were placed at the end of the prompt (low attention);
2) Vague constraint wording ("try to control" instead of mandatory);
Solution:
1) Move risk constraints to the top of system prompt:"CRITICAL RULE:
Maximum Drawdown (MDD) MUST NOT exceed -5%. Violation = immediate trade stop."
2) Add penalty logic in prompt:
"If MDD > -5%, you must reduce all positions by 50% in the next trading step."

6.3 Blocker 3: Look-Ahead Bias in Data

Issue: Reproduced CR was abnormally high (12.3% vs original 9.56%), indicating future data leakage;
Root Cause:
1) Timestamp unification missing (price data in UTC, news data in EST); 2) No time filter for real-time data access;
Solution:
1) Unify all data to EST timezone:

```
price_data["timestamp"] = pd.to_datetime(price_data["timestamp"]).dt.tz_convert("US/Eastern")
```

2) Add time filter in data access logic:

```
.filtered_data = price_data[price_data["timestamp"] <= current_trade_time]
```

7. Conclusions: Reproducibility Analysis

1) Fully Reproducible Components---Most core components of the AI-Trader framework can be fully reproduced.

The NASDAQ 100 hourly trading module, including the data pipeline, MCP tool invocation mechanism, and agent reasoning workflow, runs stably and consistently with the original implementation. The effect of prompt engineering on risk-adjusted performance is reliably replicable: adding risk constraints clearly reduces maximum drawdown and improves the Sortino Ratio. All performance metrics (CR, SR, MDD) computed by the evaluation script are consistent with the original paper’s definitions.

2) Partially Reproducible Components---Exact numerical results are only partially reproducible due to inherent randomness.

Reproduced metrics show minor deviations within ±5–10% compared with the original paper values. These differences arise from LLM reasoning randomness even at low temperature, API data refresh variations, and runtime environment differences. However, such fluctuations do not affect the core experimental conclusion that LLM-based trading agents can achieve excess returns over passive benchmarks.

3) Non-Reproducible Components---No critical parts of the experiment are non-reproducible.

All major experimental setups—including single-market evaluation, model performance comparison, and prompt-based optimization — were successfully reproduced without irreproducible logic or hidden dependencies. The entire pipeline remains transparent, modular, and repeatable under standard Python environments.