

ARTICLE TYPE

FinRL Contests: Benchmarking Data-driven Financial Reinforcement Learning Agents

Keyi Wang¹ | Nikolaus Holzer² | Ziyi Xia² | Yupeng Cao⁴ | Jiechao Gao^{1,5} |
Anwar Walid^{1,3} | Kairong Xiao⁶ | Xiao-Yang Liu Yanglet^{1,3}

¹SecureFinAI Lab, Columbia University, New York, USA

²Computer Science, Columbia University, New York, USA

³Electrical Engineering, Columbia University, New York, USA

⁴Electrical and Computer Engineering, Stevens Institute of Technology, New Jersey, USA

⁵Center for SDGC, Stanford University, California, USA

⁶Columbia Business School, Columbia University, New York, USA

Correspondence

Corresponding author Xiao-Yang Liu Yanglet
Email: XL2427@columbia.edu

Abstract

Financial reinforcement learning (FinRL) is now a practical paradigm for financial engineering. However, applying RL strategies to real-world trading tasks remains a challenge for individuals, as it is error-prone and engineering-heavy. The non-stationarity of financial data, low signal-to-noise ratios, and various market frictions require deep accumulations. Although numerous FinRL methods have been developed for tasks such as stock/crypto trading and portfolio management, the lack of standardized task definitions, real-time high-quality datasets, close-to-real market environments, and robust baselines has hindered consistent reproduction in both open-source community and FinTech industry. To bridge this gap, we organized a series of FinRL Contests from 2023 to 2025, covering a diverse range of financial tasks such as stock trading, order execution, crypto trading, and the use of large language model (LLM)-engineered signals. These contests attracted 200+ participants from 100+ institutions over 20+ countries. To encourage participations, we provided starter kits featuring GPU-optimized parallel market environments, ensemble learning, and comprehensive instructions. In this paper, we summarize these benchmarking efforts, detailing task formulations, data curation pipelines, environment implementations, evaluation protocols, participant performance, and organizational insights. It guides our follow-up FinRL contests, and also provides a reference for FinAI contests alike.

KEYWORDS

Financial reinforcement learning, FinRL, LLM, benchmark, FinAI, financial engineering

1 | INTRODUCTION

Financial reinforcement learning (FinRL)^{1,2,3,4,5,6} is an interdisciplinary field that applies reinforcement learning algorithms to financial tasks, such as algorithmic trading, portfolio management, option pricing, hedging, and market making^{7,8,9,10,11}. FinRL aims to train trading agents to interact with dynamic financial environments and to optimize their financial decisions autonomously. As FinRL methods continue to advance, two practical challenges have become increasingly apparent. One is policy instability, where small changes in training settings or market environments can lead to large variation in performance. The other is the sampling bottleneck, as collecting high-quality trajectories is often expensive and limited by access to financial data. Additionally, it remains difficult to compare the performance of these RL strategies due to inconsistencies in task definitions, dataset choices, environment implementations, and baselines. These challenges make it difficult to obtain reliable and consistent results, emphasizing the need for reproducible benchmarks that enable fair and systematic evaluation across tasks.

FinRL-Meta^{5,6} partially addressed this issue by transforming financial data into gym-style market environments, which built an automated data curation pipeline to deal with dynamic market data. Such a framework enables different FinRL algorithms to be implemented and evaluated under standardized conditions across various tasks. Still, there is no widely accepted benchmark for evaluating RL agents in financial applications¹⁰, which hinders reproducibility and makes it difficult for the community to compare results and build on prior works. There is a strong need for an open and collaborative platform where researchers can

FinRL Contests	Starter Kit	Website
2025 @ IEEE CSCloud*	Github Repo 2025	Website 2025
2025 @ IEEE IDS	Github Repo 2025	Website 2025
2024 @ ACM 2024	Github Repo 2024	Website 2024
2023 @ ACM 2023	Github Repo 2023	Website 2023
Documentation	FinAI Contests documentation	

TABLE 1 Links to the starter kits and documentation provided for FinRL Contests 2023-2025. * The FinAI Contest 2025 @ IEEE CSCloud is ongoing.

systematically evaluate their methods under realistic financial conditions. To fill this gap, we initiated the FinRL Contests as a community-driven effort to promote reproducibility, transparency, and benchmarking in financial reinforcement learning.

From 2023 to 2025, we have organized a series of FinRL contests to support reproducible experimentation and systematic benchmarking. These contests cover diverse tasks such as stock trading, order execution, crypto trading, and LLM-engineered signals for FinRL. Excluding the ongoing FinAI Contest 2025, the FinRL contests attracted 200+ participants from 100+ institutions over 20+ countries. They formed 46, 21, and 85 teams across 2023 to 2025, respectively. To ensure transparency and reproducibility, all participant teams were encouraged to open-source the solutions. To support the development of solutions for the FinRL Contest tasks, we provide starter kits for participants, including tutorials, market data, GPU-optimized parallel market environments, and baseline methods. The full list of starter kits and contest websites from 2023 to 2025 is summarized in Table 1. In addition, we maintain a continuously updated documentation, serving as a detailed guide and resource hub for participants. This documentation includes a detailed FinRL introduction, task descriptions, and explanations of baseline solutions.

In this paper, we summarize the FinRL Contests 2023-2025 that are our benchmarking efforts for financial reinforcement learning. We detail the formulation of financial tasks as Markov Decision Processes (MDPs), the design of an automated data curation pipeline, the implementation of GPU-optimized parallel market environments, and the establishment of standardized evaluation protocols. In addition, we provide a summary of participant performance and the organizational experience gained from running these contests.

The remainder of this paper is organized as follows. Section 2 reviews related works. Section 3 describes tasks. Section 4 describes dynamic market data in FinRL. Section 5 describes the benchmarking pipeline. Section 6 shows performance. Section 7 summarizes organizational aspects. Finally, we conclude this paper in Section 8.

2 | RELATED WORK

2.1 | Applications of Financial Reinforcement Learning

Financial markets are inherently complex, dynamic, and high-dimensional, making deep reinforcement learning (DRL) a compelling approach for sequential decision-making⁹. DRL has been successfully applied to a variety of financial tasks⁸, including algorithmic trading, portfolio optimization, and option pricing, where traditional rule-based or supervised methods often fall short in capturing long-term rewards under uncertainty. Beyond conventional reward-driven learning, recent advances in preference-based reinforcement learning offer alternative supervision signals. For example, Direct Preference Optimization (DPO)¹² formulates learning objectives based on pairwise comparisons between trajectories, enabling policy improvement in settings where explicit reward design is difficult or noisy. Such methods hold promise for financial applications that involve subjective goals, human feedback, or implicit utility functions, and may further enhance the robustness and flexibility of DRL in real-world markets.

2.2 | Methods of Financial Reinforcement Learning

A wide range of reinforcement learning methods^{7,8} have been explored for financial decision-making. Among these, algorithms based on policy gradients and actor-critic architectures¹³ have received particular attention due to their effectiveness in handling continuous control, high-dimensional action spaces, and non-stationary dynamics. Notable examples include proximal policy optimization (PPO), deep deterministic policy gradient (DDPG), soft actor-critic (SAC), and twin-delayed DDPG (TD3), which are frequently adopted for their robustness and sample efficiency. Ying et al.¹⁴ proposed the CVaR Proximal Policy Optimization (CPPO) algorithm, which integrates Conditional Value-at-Risk (CVaR) into its objective based on PPO.

Key Components	Attributes	FinRL Contest 2023	FinRL Contest 2024		FinRL Contest 2025		FinAI Contest 2025
		Task 1 Data-Centric Stock Trading	Task 1 Crypto Trading with Ensemble Learning	Task 2 LLM-Engineered Signals with RLHF	Task 1 FinRL-DeepSeek for Stock Trading	Task 2 FinRL-AlphaSeek for Crypto Trading	Task 1 FinRL-DeepSeek for Crypto Trading
State	Balance; Shares	✓	✓	✓	✓	✓	✓
	Open/high/low/close/volume (OHLCV) data	✓		✓	✓		
	Limit order book (LOB) data		✓			✓	✓
	Technical indicators	✓	✓	✓	✓	✓	✓
	Fundamental indicators						
	Social data; Sentiment data			✓	✓		✓
Action	Smart beta indexes, etc.						
	Buy/Sell/Hold	✓	✓		✓	✓	✓
	Short/Long						
	Portfolio weights			✓			
Rewards	Sentiment score						
	Change of portfolio value	✓	✓		✓	✓	✓
	Portfolio log-return						
	Sharpe ratio				✓		
	Penalize for high risk						
	Market feedback			✓			

FIGURE 1 Tasks in FinRL Contests 2023-2025.

Building on these algorithmic foundations, many studies have explored DRL applications in quantitative finance. For example, Deng et al.¹⁵ propose a recurrent deep reinforcement learning framework that jointly learns market representations and trading policies. Their model demonstrates robust performance across stock and commodity markets by integrating deep feature extraction with sequential decision-making. Avramelou et al.¹⁶ propose a multi-modal embedding approach to integrate both price and sentiment data in DRL-based trading agents. Their method improves trading performance while enabling dynamic adjustment of modality importance without retraining. To address the practical challenges of applying DRL in quantitative trading, Li et al.⁴ propose FinRL-Podracar, a scalable cloud-based framework for efficient training and deployment of DRL agents. The system integrates high-performance GPU optimization with automated training workflows, significantly improving both trading performance and development efficiency. While recent studies have advanced DRL applications across various financial domains, systematic evaluation and comparison of different approaches remain an important direction, motivating further efforts toward benchmarking financial reinforcement learning.

2.3 | Benchmarks of Financial Reinforcement Learning

Recent research works have applied deep reinforcement learning (DRL) to quantitative finance^{17,18,19,20,21} by developing customized market environments. Although these open-source libraries provide useful environments and datasets from sources like Yahoo Finance and WRDS, there is still a lack of standardized benchmarks for systematic evaluation. FinRL^{1,2,3} offers a full pipeline with environments for stock trading, portfolio allocation, and crypto trading; however, these environments are increasingly insufficient to meet the growing demands of the community. To address the challenges of data quality, survivorship bias, and model overfitting, Liu et al.^{5,6} further introduced FinRL-Meta. FinRL-Meta follows a DataOps paradigm to automatically collect dynamic datasets and construct hundreds of gym-style market environments, reproduces popular DRL trading papers to facilitate research reproducibility, and supports cloud-based deployment and visualization for community-driven performance evaluation. Despite these advances, establishing a unified, large-scale, and competitive benchmark for data-driven financial reinforcement learning remains an open direction, motivating the development of FinRL Contests.

3 | TASKS

Based on projects FinRL^{1,2,3,4} and FinRL-Meta^{5,6}, and feedback from our open-source community, we select several stable and representative trading tasks for the FinRL Contests. In this section, we describe the tasks in the language of Markov Decision Processes (MDPs), discuss the parallelism in the estimate of policy gradients, and describe the specific tasks featured in the FinRL Contests.

3.1 | Tasks in FinRL Contests 2023-2025

We summarize trading tasks of FinRL Contests 2023-2025 in Fig. 1, including daily stock trading with OHLCV data and second-level crypto trading with LOB data. The stock trading tasks include:

- **Data-Centric Stock Trading @ FinRL Contest 2023**^{1,2,3}. It is to train a stock trading agent by applying novel data and feature engineering based on market data. It is a multi-asset trading task for 30 constituent stocks in the Dow Jones Index. The task has been clearly defined and well studied in FinRL^{1,2,3} and FinRL-Meta^{5,6}. The task is stable and reproducible, making it ideal for benchmarking in a competition setting. In addition, this task focuses on data and feature engineering, encouraging participants to tackle the challenge of dynamic market data. This aligns with the community's ongoing interest in exploring dynamic financial data and developing stable multi-asset environments.
- **LLM-Engineered Signals with RLMF @ FinRL Contest 2024**^{22,23}. It is to develop LLMs that can generate actionable trading signals from financial news by using reinforcement learning from market feedback (RLMF). LLMs have been applied in many financial tasks²³, but general-purpose LLMs trained on Internet data cannot capture the intrinsic dynamics of financial markets. To align LLMs with financial markets, this task encourages adapting LLMs using Reinforcement Learning from Market Feedback, as a counterpart to Reinforcement Learning from Human Feedback (RLHF)²⁴. RLMF utilizes feedback from the financial market as reward signals, enabling LLMs to learn from financial market behaviors. This task was selected for its novelty and potential to push the frontier of LLM-enhanced financial decision-making. It aligns with the growing interest in the FinRL and FinLLM communities in combining FinRL and LLMs.
- **FinRL-DeepSeek for Stock Trading @ FinRL Contest 2025**^{22,25}. It is to develop trading agents by combining FinRL and LLM-engineered signals. Unstructured financial texts, such as news and SEC filings, contain valuable information about market sentiment, risk, and events. LLMs, such as DeepSeek-V3, have been applied to extract signals from financial documents²³, which are then integrated into FinRL. This task was selected because it reflects real-world decision-making, where financial analysts and traders rely on both structured market data and unstructured financial text. It also aligns with the growing interest in the FinRL and FinLLM communities in LLM-engineered signals and multimodal data exploration.

Different from stock markets, crypto trading faces greater volatility due to drastic price fluctuations and market sentiment shifts. crypto markets operate 24/7, demanding adaptable strategies in a continuous trading environment without traditional market open and close cycles. It is more challenging to develop robust and profitable trading strategies. To reflect these unique challenges, we include crypto trading tasks in the FinRL Contests. To differentiate from the multi-asset stock trading tasks above, the crypto trading tasks are designed to trade single-asset Bitcoin at the second level. The crypto trading tasks include:

- **crypto Trading with Ensemble Learning @ FinRL Contest 2024**¹⁷ is to train a trading agent for Bitcoin by using ensemble methods. RL policies are unstable, whose performance is sensitive to hyperparameters, market noise, and random seeds. Policy instability can also come from value function approximation errors. These issues are amplified in crypto markets with high volatility. Ensemble methods have shown effectiveness in mitigating policy instability¹⁷. This task extends these ideas to the crypto trading tasks. In addition, collecting high-quality financial trajectories is often expensive and limited by data access. It causes the sampling bottleneck, especially during training multiple FinRL agents for an ensemble. Therefore, this task encourages participants to employ novel ensemble methods and utilize the provided GPU-optimized parallel environments to improve sampling speed. It also aligns with the need in the financial industry for robust and efficient trading strategies in crypto markets.
- **FinRL-AlphaSeek for Crypto Trading @ FinRL Contest 2025**¹⁷ extends the previous task of crypto trading by introducing an additional focus on factor mining. Factors such as momentum play a critical role in driving trading decisions, enabling traders to design efficient, data-driven strategies. This task encourages participants to develop a two-stage pipeline: 1) factor engineering and selection and 2) ensemble learning to build robust trading agents. It aligns with the industry need for effective alpha and beta signal extraction in crypto markets.
- **FinRL-DeepSeek for Crypto Trading @ FinAI Contest 2025**¹⁷ extends the previous two tasks of crypto trading by introducing LLM-engineered signals. crypto markets are highly sensitive to news headlines, tweets, regulatory shifts, and viral narratives. The timely interpretation of market sentiment is critical for crypto trading. LLMs can extract actionable signals from financial news, tweets, and filings. In this task, we encourage participants to explore LLM-engineered signals and integrate them into a FinRL trading agent for crypto trading.

To address the sampling bottleneck of the training stage, we develop massively parallel market environments on GPUs. We encourage contestants to use ensemble methods to mitigate policy instability and improve models' robustness. These will be described in the following section.

3.2 | MDP Formulation

We formulate stock and crypto trading tasks as Markov Decision Processes (MDPs)⁶.

- **State** $\mathbf{s}_t = [b_t, \mathbf{p}_t, \mathbf{h}_t, \mathbf{f}_t] \in \mathbb{R}^{K(I+2)+1}$ represents market conditions at time t . $b_t \in \mathbb{R}_+$ is the account balance. $\mathbf{p}_t \in \mathbb{R}_+^K$ is the prices of stocks or cryptocurrencies, where K is the number of assets. $\mathbf{h}_t \in \mathbb{R}_+^K$ is the holding positions. $\mathbf{f}_t \in \mathbb{R}^{KI}$ is the feature vector, where there are I features for each asset.
- **Action** $\mathbf{a}_t \in \mathbb{R}^K$ represents trading actions at time t , such that $\mathbf{h}_{t+1} = \max(\mathbf{h}_t + \mathbf{a}_t, 0)$ (making \mathbf{h}_{t+1} nonnegative). An entry $a_t^i > 0, i = 1, \dots, K$ indicates buying a_t^i shares of assets i , while $a_t^i < 0$ indicates selling and $a_t^i = 0$ indicates holding the current position. Each entry of \mathbf{h}_{t+1} is nonnegative, which means the agent cannot sell more shares than it currently holds.
- **State-transition function** $\delta(\mathbf{s}_{t+1}|\mathbf{s}_t, \mathbf{a}_t)$ defines the probability distribution over the next state \mathbf{s}_{t+1} given the current state \mathbf{s}_t and action \mathbf{a}_t . Transitions are governed by real-world asset price movements and trading outcomes. In simulation with historical market data, the transitions may be deterministic during training. In real-time trading, the state transition function is stochastic.
- **Reward** $r_{t+1} = R(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1})$ is an incentive signal to encourage or discourage the trading agent to perform action \mathbf{a}_t at state \mathbf{s}_t . We use r_{t+1} instead of r_t to reflect the delayed reward in financial markets. It emphasizes that the reward r_{t+1} is determined at the next state \mathbf{s}_{t+1} after taking action \mathbf{a}_t . The reward can be defined as the change in the total asset value, i.e., $R(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}) = v_{t+1} - v_t$, where $v_t = b_t + \mathbf{p}_t^T \mathbf{h}_t$ is the total asset value at time t . The reward can also be penalized for high-risk signals.
- **Discount factor** $\gamma \in (0, 1)$ is used to evaluate discounted expected return at time t . The discount factor γ determines the present value of future rewards.

Note that a policy $\pi(\cdot | \mathbf{s}_t)$ is a probability distribution over actions at state \mathbf{s}_t . It determines the likelihood of each possible trading action given the current market conditions. The trading agent's objective is to learn a policy π that maximizes cumulative positive rewards while minimizing losses over time.

In the state \mathbf{s}_t , the holding positions \mathbf{h}_t are typically constrained to be nonnegative in long-only trading settings, where the agent is not allowed to short assets. The feature vector \mathbf{f}_t can include market indicators (e.g., moving average convergence divergence), factors learned by neural networks, and LLM-engineered signals from financial news or regulatory filings. The action \mathbf{a}_t can be discrete or continuous, depending on the task. It may support long-only or long-short strategies. The reward function R can use the Sharpe ratio in addition to the change in asset value. It can be further adjusted by risk or market sentiment to better reflect real-world trading scenarios.

For example, a trading task is to develop a trading agent for all 30 constituent stocks in the Dow Jones Index with an initial investment of \$1 million. In this setting, the number of assets is $K = 30$. The state includes the account balance of $b_0 = 100,000$, prices of the 30 stocks, holding positions of the 30 stocks, and $I = 4$ technical indicators for each stock. The state space has a dimension $K(I + 2) + 1 = 181$. The action space has dimension $K = 30$. An entry, for example, $a_t^1 = 5$ means buying 5 shares of the first stock in the portfolio. The reward function is the change in the total asset value, with $v_0 = 100,000$. Taking the first stock as an example, assume at time t , the account balance is $b_t = 100,000$, the price of the first stock is $p_t^1 = 200$, and the shares of the first stock are $h_t^1 = 10$. The agent buys 5 shares of the first stock ($a_t^1 = 5$) and the price increases to $p_{t+1}^1 = 201$ at time $t + 1$. Assuming holding all other 29 stocks, the state is updated where $b_{t+1} = 99,000$, $p_{t+1}^1 = 201$, and $h_{t+1}^1 = 15$. The return $r_t = v_{t+1} - v_t = (b_{t+1} + \mathbf{p}_{t+1}^T \mathbf{h}_{t+1}) - (b_t + \mathbf{p}_t^T \mathbf{h}_t) = b_{t+1} - b_t + p_{t+1}^1 h_{t+1}^1 - p_t^1 h_t^1 = -1000 + 201 \cdot 15 - 200 \cdot 10 = 15$. In this example, the agent only operates on the first stock.

3.3 | Policy Gradient Estimate and Its Parallelism

In the FinRL tasks, the policy π_θ is typically a neural network (e.g., transformer network or diffusion model), where θ denotes the learnable parameters. A trajectory, $\tau = (\mathbf{s}_0, \mathbf{a}_0, r_1, \mathbf{s}_1, \mathbf{a}_1, r_2, \mathbf{s}_2, \mathbf{a}_2, \dots, \mathbf{s}_T)$, is a sequence of trading actions and states observed

over a period, where \mathbf{s}_0 is the initial state and T is the number of steps. The probability of the trajectory τ is

$$P(\tau \mid \pi_\theta) = \rho_0(\mathbf{s}_0) \prod_{t=0}^{T-1} \pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t) P(\mathbf{s}_{t+1} \mid \mathbf{s}_t, \mathbf{a}_t), \quad (1)$$

where $\rho_0(s_0)$ is the initial state distribution. The financial results are quantified as rewards along τ to calculate the cumulative return $R(\tau) = \sum_{t=1}^T \gamma^{t-1} r_t$. The goal is to learn a policy π_θ that maximizes the following expected return

$$J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} [R(\tau)] = \int_{\tau} P(\tau \mid \pi_\theta) R(\tau), \quad (2)$$

The gradient of $J(\theta)$ with respect to θ is ²⁶

$$\nabla J(\theta) = \mathbb{E}_{\tau \sim \pi_\theta} \left[(R(\tau) - b) \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(\mathbf{a}_t \mid \mathbf{s}_t) \right], \quad (3)$$

where $b \in \mathbb{R}$ is a constant, and subtracting b can reduce the variance in practice.

In the FinRL tasks, the trading strategy is governed by π_θ to maximize $J(\theta)$. When dealing with complex and noisy financial datasets, achieving a low-variance gradient estimation $\nabla J(\theta)$ is crucial for stable and reliable policy updates, reducing the risk of suboptimal trading strategies.

To estimate $\nabla J(\theta)$ in (3), we can use the Monte Carlo method²⁷

$$\nabla J(\theta) = \frac{1}{n} \sum_{j=1}^n \left((R(\tau^{(j)}) - b) \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(\mathbf{a}_t^{(j)} \mid \mathbf{s}_t^{(j)}) \right), \quad (4)$$

where n trajectories are used. The Law of Large Numbers guarantees that as the sample size n increases, the estimation of $\nabla J(\theta)$ will converge to its expected value. According to the Central Limit Theorem, increasing n leads to a reduction in the variance of the estimate.

As shown in (4), a large number n of trajectories sampled during the simulation phase is required to reduce the variance of $\nabla J(\theta)$. In (4), each j in the outer sum from 1 to n corresponds to a separate trajectory $\tau^{(j)}$, which can be considered as a complete and independent simulation of the policy π_θ in the environment. Therefore, each trajectory $\tau^{(j)}$ can be simulated in parallel, allowing for a high degree of parallelism. In FinRL, parallel simulation involves executing the trading strategy in multiple market scenarios simultaneously. The parallelism accelerates the simulation phase, allowing for more rapid updates and iterations of the policy π_θ . Therefore, the trading strategy governed by π_θ can be quickly updated and adapted to changing market conditions.

4 | FINRL ENVIRONMENTS

Market data and the simulation environment are two critical components of FinRL that influence the definition of tasks. In this section, we provide detailed information on market data and GPU-optimized parallel market environments.

4.1 | Automated Data Curation Pipeline for Dynamic Market Data

We first describe different sources of dynamic market data used in FinRL Contests, along with feature engineering, market data split strategies, and the automated data curation pipeline.

Data Sources. We leverage both structured market data and unstructured financial documents. Structured market data captures quantitative market behavior, such as historical price information and market indicator data. Unstructured financial documents provide information about economic trends, regulations, and events that drive market movements. The structured market data is processed into price features \mathbf{p}_t , which are included in state \mathbf{s}_t . We outline the financial data utilized in FinRL as follows:

Indicator	Name	Description
macd	Moving Average Convergence Divergence	A trend-following momentum indicator that shows the relationship between two exponential moving averages (EMAs) of a stock's price. Traders use the MACD to identify potential trend changes, divergence between price and momentum, and overbought or oversold conditions.
boll_ub	Bollinger Bands Upper Band	Bollinger Bands are used to visualize the volatility and potential price levels of a stock. The upper band represents the upper volatility boundary, showing where the price might find resistance.
boll_lb	Bollinger Bands Lower Band	Similarly, the lower band represents the lower volatility boundary and shows where the price might find support.
rsi_30	Relative Strength Index for 30 periods	A momentum oscillator that measures the speed and change of price movements. RSI oscillates between zero and 100.
cci_30	Commodity Channel Index for 30 periods	A versatile indicator that can be used to identify a new trend or warn of extreme conditions. It measures the current price level relative to an average price level over a given period of time.
dx_30	Directional Movement Index for 30 periods	An indicator that assesses the strength and direction of a trend of a stock. It does this by comparing highs and lows over time.
close_30	30-Period Simple Moving Average of Closing Prices	Represents the average closing price over the last 30 periods. This moving average provides a smoothed representation of the asset's price over the 30 periods, making it easier to identify trends and potential support/resistance levels.
close_60	60-Period Simple Moving Average of Closing Prices	Represents the average closing price over the last 60 periods. This moving average provides a smoothed representation of the asset's price over the 60 periods, making it easier to identify trends and potential support/resistance levels.
vix	Volatility Index	Often referred to as the "fear index", it represents the market's expectation of 30-day forward-looking volatility. It is calculated from the prices of selected stock option contracts on the S&P 500 Index.
turbulence	Turbulence	To control the risk in a worst-case scenario, such as the financial crisis of 2007–2008, FinRL employs the financial turbulence index that measures extreme asset price fluctuation.

TABLE 2 Market indicators and their descriptions

- **OHLCV data.** OHLCV (open, high, low, close, volume) data is typical historical volume-price data in finance. We provide daily OHLCV data for stocks from 1999 to 2023. This dataset is prepared through `yfinance`[‡] (an open-source Python library) and FinRL-Meta^{5,6}, released under the CDLA-permissive-2.0 license. The missing values are removed. We also provide data APIs to download market data, such as Alpaca[§], which are held on the open-source repository FinRL-Meta^{¶5,6}.
- **Limit order book (LOB) data** at second level for cryptocurrency trading[#]. LOB data offers a detailed view of market depth and liquidity, capturing the behavior of market participants and providing valuable insights into market trends. Given its high granularity and extensive size, this data is ideally suited for participants to leverage massively parallel simulations, enabling more effective development of cryptocurrency trading strategies.
- **Financial news.** The Financial News and Stock Price Integration Dataset (FNSPID)²⁸ contains 15 million time-aligned financial news articles from 1999 to 2023 for constituent companies in S&P 500. It is released under CC-BY-NA 4.0 for academic purposes. From the FNSPID dataset, we randomly select one news article per day per stock. The news is aggregated with OHLCV data to ensure temporal alignment and create a multimodal dataset. For cryptocurrency news, we aggregate BTC news collected from multiple sources^{||} and ensure the temporal alignment with the LOB data. We also provide data APIs to access financial news, such as Yahoo Finance and Finnhub. These data APIs are held on the open-source repository FinGPT^{**22}.

Feature Engineering. We conduct feature engineering before making the dataset available for contests.

- **Market indicators.** Ten market indicators listed in Table 2 are computed based on OHLCV data. These indicators are used to enrich the feature vector \mathbf{f}_t in state \mathbf{s}_t .

[‡] <https://yfinance-python.org/>

[§] <https://alpaca.markets/>

[¶] <https://github.com/AI4Finance-Foundation/FinRL-Meta>

[#] <https://www.kaggle.com/datasets/martinsn/high-frequency-crypto-limit-order-book-data/data>

^{||} The aggregated BTC news dataset: https://huggingface.co/datasets/SecureFinAI-Lab/FinRL_BTC_news_signals. Data sources: https://huggingface.co/datasets/edaschau/bitcoin_news; <https://github.com/soheilrahsaz/cryptoNewsDataset>

^{**} <https://github.com/AI4Finance-Foundation/FinGPT>

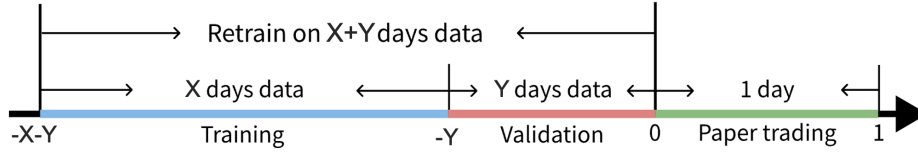


FIGURE 2 Data split for a window with training, validation, and trading.

- **ML-learned factors.** For the LOB data, bid and ask prices at different market depths are extracted, and 101 adapted alpha factors²⁹ are constructed. An RNN is trained to distill 8 strong factors from the 101 weak alphas, modeling and predicting the prices. These factors are used to enrich the feature vector \mathbf{f}_t in state \mathbf{s}_t .

The feature selection process involves two steps: 1) compute the Pearson correlation matrix of the features, and 2) select one representative feature from each group of highly correlated features, following the method proposed by Gort et al³⁰.

LLM-Engineered Signals. LLMs have been envisioned to be very promising for generating alpha signals from multiple financial data sources^{23,31,32}. We encourage participants to leverage LLMs and train a trading agent for making informed decisions. Here we show the sentiment and risk scores generated by DeepSeek models from financial news²⁵:

- **Sentiment score.** An LLM assigns a sentiment score u of 1 to 5 according to the news, with 1 for strongly negative and 5 for highly positive. For example, we can use DeepSeek-V3³³ to generate signals. It is included in the feature vector \mathbf{f}_t and is used to adjust actions via the sentiment factor $l_t^i = 1 + 0.05(u - 3)\text{sign}(a_t^i)$. The factor is close to 1 for the stability of the algorithm. The adjusted action is $a_t^{i'} = l_t^i a_t^i$, which is amplified under positive sentiment and dampened under negative sentiment.
- **Risk level.** An LLM assigns a risk level q of 1 to 5 from the news, with 1 for low risk and 5 for high risk. It is included in the feature vector \mathbf{f}_t and is used to penalize rewards through the risk factor $m_t^i = 1 + 0.05(q_t^i - 3)$. The aggregated risk factor is $M_t = \sum_i^K w_t^i m_t^i$, where w_t^i is the portfolio weight of the stock i and $\sum w^i = 1$. $M_t > 1$ penalizes the reward for high risk; $M_t < 1$ increases the reward for low risk.

Automated Data Curation Pipeline. After preparing the datasets, we split them into two parts, one released to participants for training and the other held by organizers for evaluation purposes. We use the temporal partitioning approach to maintain the temporal integrity of financial data and prevent future information leakage.

- **Dataset released to participants.** We provide long-span historical datasets for participants to develop their models. It covers different market patterns, such as financial crises, bull markets, and bear markets. Participants are not restricted to the provided datasets and APIs. They are encouraged to use external data sources, such as alternative market indicators, financial news feeds, and tweets.
- **Dataset withheld for evaluation.** Depending on the task and data availability, we use two methods to obtain the evaluation dataset: 1) we extract the most recent $\sim 15\%$ of the original dataset, withhold it as the evaluation dataset, and encrypt all timestamps; 2) we collect out-of-sample data for the period after the model submission deadline. It involves downloading market data and financial news via `yfinance` or other APIs. These two methods will avoid future data leakage and ensure a fair assessment. This setup ensures that the evaluation simulates real-world, time-forward deployment, where models should generalize to new, unseen market data.

To better reflect real-world forward-moving financial markets, we present an automated data curation pipeline. We illustrate this pipeline using the example of the paper trading task. The conventional backtesting approach splits historical market data into in-sample and out-of-sample time periods. Backtesting may be performed multiple times on the out-of-sample data. However, stock paper trading is strictly required to be carried out **once** in real-time on the real-world market.

Fig. 2 shows our “training-validation-trading” pipeline. In a window, there are X days’ data for training and Y days’ data for validation. At the end of a window, we perform paper trading for 1 day. Note that we always retrain the agent using $X + Y$ days of training and validation data together. Then, we roll the window forward by 1 day ahead and perform the above steps for a new window. A paper trading is always carried out for 1 day. Therefore, Z windows correspond to Z trading days.

Alg. 1 summarizes the pipeline of paper trading. For Z trading days ($z = 0, 1, \dots, Z-1$), we keep doing the following three steps:

Algorithm 1 Pseudo code for stock paper trading

```

1: Initialize a set of hyperparameters;
2: for  $z=0$  to  $Z-1$  do
3:   # Step 1). Train an agent
4:   Train agent on data period  $[z-Y-X, z-Y-1]$ 
5:   Validate trained agent and tune hyperparameters on data period  $[z-Y, z-1]$ 
6:   # Step 2). Retrain the agent
7:   Retrain agent on data period  $[z-Y-X, z-1]$  using tuned hyperparameters
8:   # Step 3). Perform paper trading for one day
9:   Trade on day  $z$  with trained agent
10: end for

```

- **Step 1).** Download and process X -day data, from day $z - Y - X$ to day $z - Y - 1$. Then build the data into a gym-style environment and train the agent. Then download and process Y -day data, from day $z - Y$ to day $z - 1$. Then build the data into a gym-style environment and validate how the agent performs. According to the agent's performance on the validation environment, adjust the hyperparameters.
- **Step 2).** Build the training and validation data, totally $X + Y$ days from day $z - Y - X$ to day $z - 1$, into a gym-style environment. Update hyperparameters to the values chosen from **Step 1)**. Then retrain the agent on this $X + Y$ -day environment.
- **Step 3).** Deploy the trained agent to the paper trading market.

Before each trading day z , we use the historical data period from $z - Y - X$ to $z - Y - 1$ to train the FinRL agent. Then, data periods from $z - Y$ to $z - 1$ are used to validate the trained agent, adjusting hyperparameters accordingly. We retrain the agent by combining all the data from the training and validation period, that is, the data period from $z - Y - X$ to $z - 1$. Finally, we perform 1 day paper trading on the z -th day. We continue this process for $z = 0, 1, \dots, Z - 1$.

4.2 | Market Environments

The financial data is processed into the standard gym-style market environments to build a standard practical environment. This part describes the market environments with near-real market constraints. To address the sampling bottleneck of the training stage, we develop massively parallel market environments on GPUs.

4.2.1 | Market Environment with Trading Constraints

After processing the financial data into standard gym-style market environments, we define the key operations in the FinRL market environment, including:

- **reset:** $\mathbf{s}_t \rightarrow \mathbf{s}_0$, resets the environment to its initial state.
- **step:** $(\mathbf{s}_t, \mathbf{a}_t) \rightarrow \mathbf{s}_{t+1}$, executes \mathbf{a}_t and updates \mathbf{s}_t to \mathbf{s}_{t+1} .
- **reward:** $(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1}) \rightarrow r_{t+1}$, computes the reward.

To reflect the real-world trading scenario, we incorporate near-real trading constraints:

- **Transaction costs.** We set a cost of 0.1% for each action {buy, sell}, accounting for commission and slippage.
- **Market volatility.** The turbulence index and VIX index are risk indicators. A large value signals heightened volatility from factors like investor fear and increased uncertainty, while a small value signals increased stability in markets.

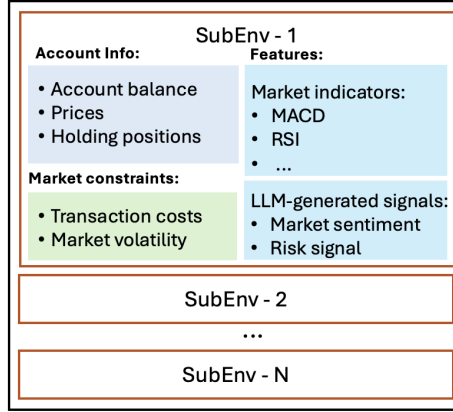


FIGURE 3 Vectorized environment.

4.2.2 | Massively Parallel Simulations on GPUs

Stable policy updates require a low-variance gradient estimate $\nabla J(\theta)$ from a large n independent trading trajectories τ , where there is high parallelism, as shown in (4). PyTorch's `vmap` can map operations over some dimension onto parallel GPU cores, exploiting the parallelism of (3). Therefore, we construct vectorized environments and provide GPU optimization via `vmap`. As shown in Fig. 3, a vectorized environment manages parallel sub-environments (SubEnvs).

Using `vmap`, the `step` and `reward` functions are vectorized to operate in massively parallel environments. For example, the `reward` function, vectorized by `vmap`, computes the reward on $(\mathbf{s}_t, \mathbf{a}_t, \mathbf{s}_{t+1})$ for each SubEnv simultaneously. This computation is dispatched to available GPU cores, each responsible for calculating its assigned data.

Data samples are stored as tensors in GPU memory. They have the shape $N \times T \times D$:

- N is the number of parallel SubEnvs.
- T is the number of steps in a trajectory.
- D is the dimension as in Section 3.2, where $D = K(I + 2) + 1$ for state, $D = K$ for action, and $D = 1$ for reward.

The tensors for states ($\mathbf{s} \in \mathbb{R}^{K(I+2)+1}$), actions ($\mathbf{a} \in \mathbb{R}^K$), and rewards ($r \in \mathbb{R}$) are as follows:

$$\begin{bmatrix} \mathbf{s}_0^1 & \mathbf{s}_1^1 & \cdots & \mathbf{s}_{T-1}^1 \\ \mathbf{s}_0^2 & \mathbf{s}_1^2 & \cdots & \mathbf{s}_{T-1}^2 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{s}_0^N & \mathbf{s}_1^N & \cdots & \mathbf{s}_{T-1}^N \end{bmatrix}, \begin{bmatrix} \mathbf{a}_0^1 & \mathbf{a}_1^1 & \cdots & \mathbf{a}_{T-1}^1 \\ \mathbf{a}_0^2 & \mathbf{a}_1^2 & \cdots & \mathbf{a}_{T-1}^2 \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{a}_0^N & \mathbf{a}_1^N & \cdots & \mathbf{a}_{T-1}^N \end{bmatrix}, \begin{bmatrix} r_1^1 & r_2^1 & \cdots & r_T^1 \\ r_1^2 & r_2^2 & \cdots & r_T^2 \\ \vdots & \vdots & \ddots & \vdots \\ r_1^N & r_2^N & \cdots & r_T^N \end{bmatrix}.$$

Storing data samples as tensors in GPU memory bypasses the CPU-GPU bandwidth bottleneck.

Improved Sampling Speed with Massively Parallel Environments on GPU. We evaluated the sampling speed measured in samples per second using vectorized environments for stock trading. We used the PPO agent and the OHLCV data of 30 constituent stocks in the Dow Jones index, from 2020-01-01 to 2022-01-01. The NVIDIA A100 GPU is used. The numbers of parallel environments vary from 1, 2, 4, ..., to 2,048. As shown in Fig. 4, the average sampling speed with 2,048 parallel environments is 227,212.54 samples per second. The sampling speed is improved by $1,649.93\times$ compared with a single environment. The sampling speed scales approximately linearly with the number of parallel environments. The results show the effectiveness of massively parallel simulation in improving sampling speed in FinRL Contest tasks.

5 | BENCHMARKING PROTOCOL

To assist participants in becoming familiar with FinRL contests and to ensure fair comparisons, we establish a benchmarking protocol and provide several baseline methods.

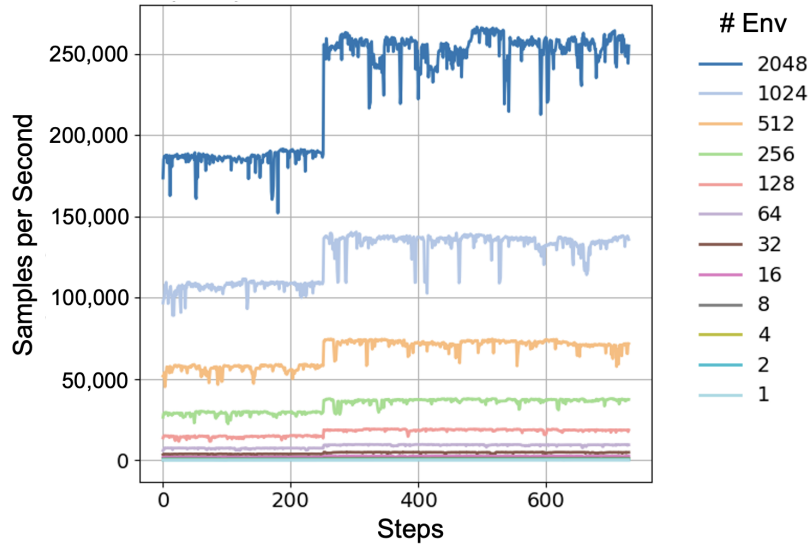


FIGURE 4 Samples per second for the stock trading task (using NVIDIA A100 GPU).

5.1 | Baseline Methods

5.1.1 | Market Index and Mean-Variance Method

We use market indexes and the mean-variance optimization strategy as baselines. They are widely adopted in the financial industry.

- **DJIA.** The Dow Jones Industrial Average index is a price-weighted index for 30 blue-chip U.S. companies. It is calculated as the sum of constituent stock prices divided by the Dow Divisor. The divisor is a constant adjusted for stock splits and structural changes. The data can be downloaded by `yfinance`. DJIA is one of the oldest and most recognized market indexes. It provides a real-world baseline to evaluate whether a FinRL agent outperforms a passive investment strategy.
- **S&P 500.** The Standard and Poor's 500 is a capitalization-weighted index tracking the stock performance of 500 leading companies. The data can be downloaded by `yfinance`. Covering approximately 80% of the total market capitalization, it offers a broader and more diversified baseline.
- **Mean-variance optimization.** Mean-variance optimization strategy, as part of Modern Portfolio Theory (MPT)³⁴, constructs portfolios that maximize the expected return for a given level of risk. It uses expected asset returns and covariances to solve an optimization problem. We typically use the past one year's daily price data to calculate expected returns and the covariance matrix. We limit individual stock weights to a maximum of 5%. Mean-variance optimization is a foundational technique in portfolio management and can serve as a classical financial optimization strategy baseline.

5.1.2 | Ensemble Agent

Ensemble methods have shown effectiveness in enhancing overall performance by combining selected actions or action probabilities from component RL algorithms³⁵. We train ensemble trading agents to mitigate policy instability.

5.1.2.1 | Agent Diversity

The diversity of component agents is essential for risk mitigation by leveraging various trading agents. Achieving high diversity requires training multiple agents across environments that simulate different market scenarios.

Using KL divergence in objective functions. To enforce diversity among the component agents, we introduce a Kullback-Leibler (KL) divergence term into the agent's objective function. The KL divergence measures how one probability distribution diverges from another³⁶. The KL divergence term penalizes similarities in policies between different agents, encouraging them

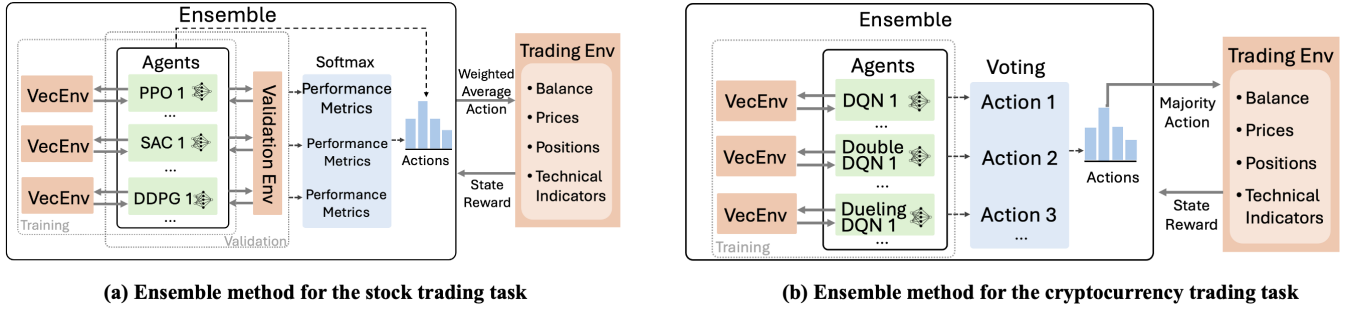


FIGURE 5 Ensemble methods.

to converge on different trading strategies. The new objective function for a component agent is as follows:

$$\max L_{\text{new}}(\theta_A) = L_{\text{original}}(\theta_A) + \lambda \sum_{A \neq B} \text{KL}(\pi_{\theta_B} \parallel \pi_{\theta_A}), \quad (5)$$

where θ_A are the policy parameters for agent A , $L(\theta_A)$ is the objective function, $\text{KL}(\pi_{\theta_B} \parallel \pi_{\theta_A})$ is the KL divergence between agent A 's and agent B 's policies, and $\lambda > 0$ is a regularization constant. After obtaining the trained agents, one can ensemble them using a majority voting method or the weighted sum method¹⁷.

Using various datasets. The financial datasets used for training component agents are varied. For each stock or crypto, a random percentage change ranging from -1% to 1% is generated and applied to its prices, which shifts the price scale while preserving the original price trends. Agents are also trained on different stocks from the test set for the stock trading task. It enables agents to learn various strategies for a broader range of stocks rather than reacting to a limited number of stocks.

5.1.2.2 | Stock Trading Task

As shown in Fig. 5 (a), the ensemble includes PPO, SAC, and DDPG agents. The ensemble's final trading action is determined by weighted averaging over the agents' action probabilities. The process is as follows:

- **Traning.** Agents are trained independently with a VecEnv on a 30-day training rolling window, using massively parallel simulation in Section 4.2.2 and agent diversity methods in Section 5.1.2.1.
- **Validation.** After training, agents are validated on a 5-day rolling window. Sharpe ratios are calculated to evaluate their ability to balance returns with associated risks.

$$\text{Sharpe Ratio} = \frac{\bar{r}_p - r_f}{\sigma_p}, \quad (6)$$

where \bar{r}_p is the portfolio return, r_f is a chosen risk-free rate, and σ_p is the standard deviation of the portfolio return.

- **Weights calculation.** Agents with very low Sharpe ratios are discarded. Weights for the remaining agents are calculated using a softmax function applied to their Sharpe ratios.
- **Trading.** The ensemble acts based on a weighted average of agent action probabilities during a 5-day trading window.

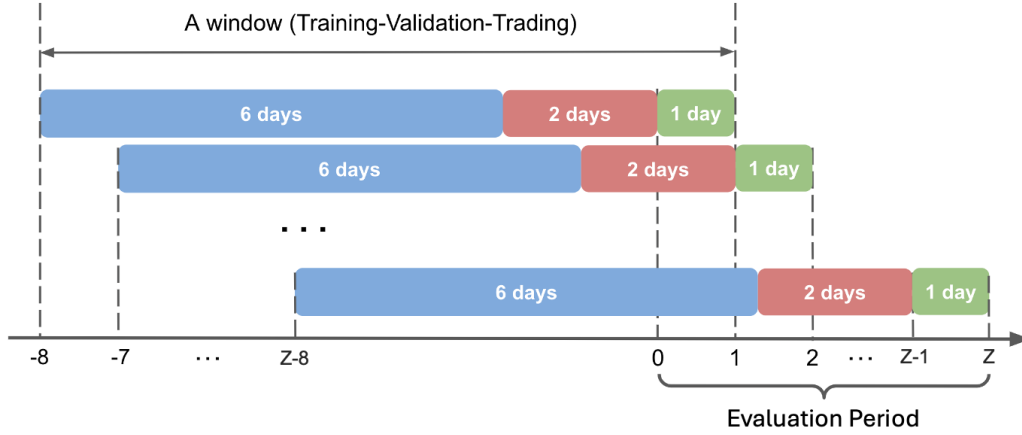
This rolling window approach ensures that the ensemble method remains adaptive to the continuously changing market.

5.1.2.3 | Crypto Trading Task

For crypto trading at a relatively high frequency, market movements can be modeled as discrete events, which require a discrete action space. As shown in Fig. 5 (b), DQN³⁷, Double DQN³⁸, and Dueling DQN³⁹ are used to handle this discrete action space. In addition, the dataset for a single crypto is relatively small. DQN and its variants, with fewer parameters and simpler architectures, can be trained faster to avoid overfitting. Moreover, trading at a high frequency requires fast responses, and DQN agents can offer lower latency in decision-making compared to more complex models. The ensemble model uses majority voting to combine the actions of component agents. Majority voting ensures the chosen action reflects consensus among agents, mitigating biases from any single agent's actions⁴⁰. The process is as follows:



(a) Evaluation framework for FinRL Contest 2024 and 2025 tasks.



(b) Rolling windows for FinRL Contest 2023 data-centric stock trading task and paper trading task.

FIGURE 6 The rolling windows for the evaluation of the stock trading task.

- **Training.** Each component agent is independently trained with a VecEnv, using the massively parallel simulation in Section 4.2.2 and the agent diversity methods in Section 5.1.2.1.
- **Action ensemble and trading.** During the trading phase, each agent processes the same market state and determines an action based on its policy. The majority action is selected as the final ensemble action.

5.2 | Evaluation Protocol

To evaluate the models, we adopt two frameworks, backtesting and rolling windows, as shown in Fig. 6:

- **Backtesting.** We use backtesting to evaluate models for trading tasks in FinRL Contests 2024 and 2025, as shown in Fig. 6 (a). The datasets released to participants are used for their training and validation. The withheld out-of-sample dataset is used for evaluation. We did not use the rolling window framework because the evaluation datasets for stock trading have long time spans, and the crypto data is high-frequency. In both cases, retraining models frequently would be too time-consuming and difficult to manage.
- **Rolling windows.** We use the rolling window framework to evaluate models for FinRL Contest 2023 data-centric stock trading and paper trading, as discussed in Section 4.1. Fig. 6 (b) presents the mechanism of each rolling window for the stock trading task. There are in total 9 days in a rolling window. We divide it into three stages, namely, 6 days of training, 2 days of validation, and 1 day of trading using the trained DRL agent. The models are retrained for 8 days before each trading day. The validation stage allows adjusting the hyperparameters or selecting an agent from the ensemble. Finally, the well-trained agent performs stock trading on the last day of a rolling window.

To ensure fair and reproducible benchmarking for all models, the evaluation process follows the principles below:

Metric	Definition	Range	Direction
Cumulative return	The total return generated by the trading strategy over a trading period.	$(-\infty, +\infty)$	Higher is better
Annualized return	The geometric average amount of money earned by the agent each year over a given time period.	$(-\infty, +\infty)$	Higher is better
Annualized volatility	The annualized standard deviation of daily returns.	$[0, +\infty)$	Lower is better
Sharpe ratio	The excess return per unit of volatility.	$(-\infty, +\infty)$	Higher is better
Maximum drawdown	The largest single drop in the portfolio value from peak to trough.	$[-1, 0]$	Higher is better
Rachev Ratio	A risk-adjusted return that measures the potential upside gain compared to potential downside risk, using the tails of the return distribution.	$(-\infty, +\infty)$	Higher is better
Return over maximum drawdown (RoMaD)	It is calculated as the cumulative return divided by the absolute value of maximum drawdown.	$(-\infty, +\infty)$	Higher is better
Sortino ratio	The excess return divided by the downside deviation.	$(-\infty, +\infty)$	Higher is better
Calmar ratio	The annualized excess return divided by the maximum drawdown.	$(-\infty, +\infty)$	Higher is better
Omega ratio	It compares the probability of achieving returns above a threshold to the probability of falling below it.	$[0, +\infty)$	Higher is better
Win/Loss ratio	It is calculated by dividing the number of winning trades by the number of losing trades.	$[0, +\infty)$	Higher is better

TABLE 3 Evaluation metrics.

- **Uniform evaluation platform.** All models are evaluated on the same infrastructure to eliminate differences caused by hardware or software variations. Specifically, we conduct evaluations using Google Cloud Platform (GCP) virtual machines with identical hardware configurations. The software environment is Ubuntu 22.04.2 LTS and Python 3.10.12.
- **Controlled evaluation setting.** All models are evaluated in the same environment settings, including testing data, initial investment amount, and transaction costs. No additional training or model adjustment is allowed during the evaluation phase.

6 | PERFORMANCE

6.1 | Performance Metrics

We use the well-established quantitative metrics² in finance to evaluate models, as shown in Table 3. Specifically, we include cumulative and annualized returns to measure overall profitability and incorporate risk-adjusted performance metrics such as the Sharpe, Sortino, and Calmar ratios. Additionally, we consider downside risk indicators, including annualized volatility and maximum drawdown. For each metric, we report its definition, numerical range, and optimization direction (i.e., whether higher or lower values indicate better performance).

6.2 | Stock Trading Tasks

6.2.1 | Data-Centric Stock Trading

The data-centric stock trading task in FinRL Contest 2023 is to train a trading agent by using novel data and feature engineering strategies. The training dataset is the OHLCV dataset with 10 market indicators for all 30 constituent stocks in the Dow Jones Index from 07/01/2010 to 10/24/2023 (3,352 trading days). The market indicators are shown in Table 2. The models are evaluated for two periods: 10/25/2023-11/14/2023 (15 trading days, pre-submission deadline) and 11/15/2023-11/12-2023 (6

Team	Testing Period	Cumulative Return	Sharpe Ratio*	Maximum Drawdown
SZU-FIN-621	10/25/2023-11/14/2023	1.05%	2.26	-1.36%
	11/15/2023-11/22/2023	-0.03%	-10.25	-0.03%
Nik-Elena	10/25/2023-11/14/2023	3.50%	9.56	-0.40%
	11/15/2023-11/22/2023	0.04%	0.95	-0.15%
WeCan	10/25/2023-11/14/2023	2.35%	8.07	-0.55%
	11/15/2023-11/22/2023	-0.05%	-1.74	-0.11%
DJIA	10/25/2023-11/14/2023	5.42%	0.45	-1.87%
	11/15/2023-11/22/2023	0.80%	0.49	-0.18%

TABLE 4 The performance of top winner teams for FinRL Contest 2023 Task 1 Data-Centric Stock Trading. The stocks are constituents of the Dow Jones Index ($K = 30$). *The results of the Sharpe ratio were reported wrongly.

Model	Ensemble-1	Ensemble-2	Ensemble-3	PPO	SAC	DDPG	Min-Variance	DJIA
Cumulative Return	62.60%	58.77%	46.89%	63.37%	50.62%	63.19%	13.9%	18.95%
Annual Return	18.22%	17.25%	14.15%	18.41%	15.14%	18.36%	7.34%	6.15%
Annual Volatility	11.76%	12.61%	12.70%	11.35%	11.67%	11.93%	18.16%	15.14%
Sharpe Ratio	1.48	1.33	1.11	1.55	1.27	1.47	0.48	0.47
Sortino Ratio	2.34	2.14	1.74	2.44	2.05	2.37	0.73	0.67
Max Drawdown	-8.98%	-11.27%	-12.27%	-9.96%	-12.02%	-13.15%	-14.9%	-21.94%
RoMaD	6.97	5.22	3.82	6.36	4.21	4.81	1.10	0.86
Calmar Ratio	2.03	1.53	1.15	1.85	1.26	1.40	0.49	0.28
Omega Ratio	1.31	1.28	1.23	1.33	1.27	1.32	1.09	1.08

TABLE 5 Stock trading task performance. Models are trained, validated, and tested on a rolling window basis on OHLCV datasets for 30 Dow Jones stocks. Ensemble models use weighted averages on agent action probabilities.

trading days, post-submission deadline). Table 4 shows the cumulative return, Sharpe ratio, and maximum drawdown of the top three winners and the Dow Jones Index. The teams used different approaches ^{††}:

- **SZU-Fin-621** employed PPO-Switch, which is an ensemble method combining multiple PPO agents. The agent pool was created by training different PPO agents with diverse online portfolio selection (OPS)⁴¹ price features. The agent was selected based on its short-term and long-term returns, and the action space was sparsified to enhance capital efficiency⁴².
- **Nik-Elena** added new technical indicators in the state, including the Relative Strength Index for different periods, On-Balance Volume, and Moving Average for different periods. The team also enlarged the negative return when the turbulence exceeded a threshold.
- **WeCan** added 101 formulaic alpha factors²⁹ as new features in the state, in addition to the provided market indicators.

The results show that the teams’ strategies achieved better risk-adjusted returns and risk management, but showed poorer profitability compared with the Dow Jones Index. In the first testing period, from 10/25/2023 to 11/14/2023, all teams outperformed the Dow Jones Index in the Sharpe ratio and maximum drawdown. Nik-Elena achieved the highest Sharpe ratio of 9.56 and the best maximum drawdown of -0.40%, indicating excellent risk-adjusted performance and effective downside protection. In the second testing period from 11/15/2023 to 11/22/2023, SZU-FIN-621 and WeCan had negative cumulative returns and Sharpe ratios. Nik-Elena had a small positive return of 0.04% but still underperformed the market index. However, Nik-Elena still achieves a higher Sharpe ratio and maximum drawdown than the Dow Jones Index, indicating better risk control even in challenging market environments. Participants’ models showed strong risk management compared to the DJIA in the first testing period, but their generalization to new, unseen market conditions remains a challenge.

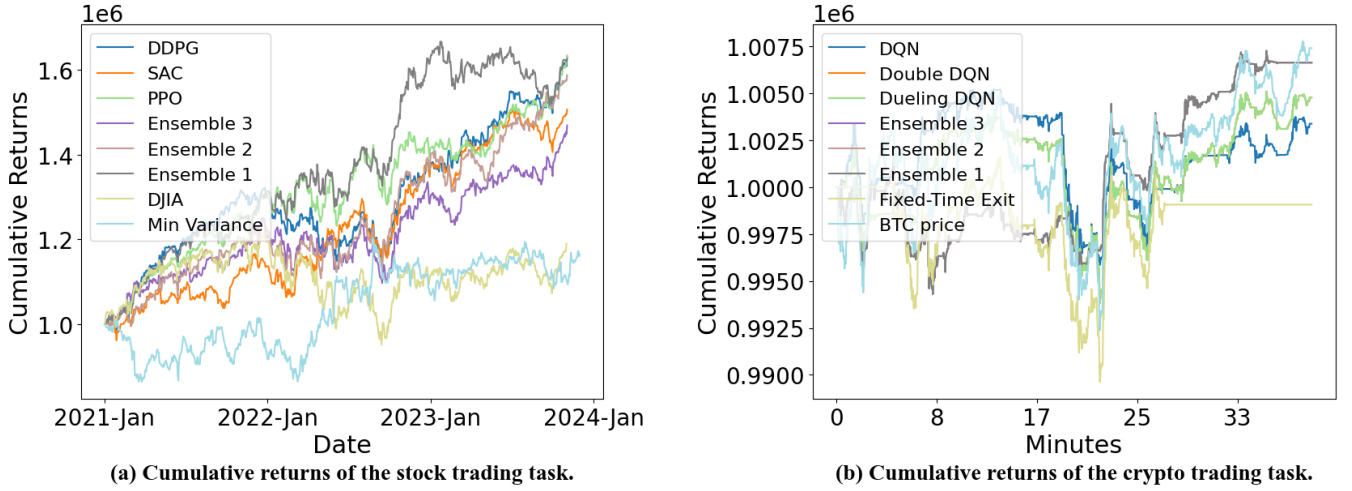


FIGURE 7 Cumulative returns of different strategies for the stock trading task and crypto trading task.

6.2.2 | Stock Trading with Ensemble Methods

We performed the stock trading task for 30 stocks in the Dow Jones index by using three ensemble models, and individual PPO, SAC, and DDPG agents.

Stock data: We use historical daily OHLCV data for all 30 stocks in the Dow Jones index from 01/01/2021 to 12/01/2023 (734 trading days). OHLCV data is a rich source for learning financial market behaviors and trends. We use technical indicators listed in Table 2. These indicators enrich the data with more insights into market behaviors and trends. Therefore, $K = 30$ and $I = 10$ in the setting of Section 3.2.

Agents for stock trading task. We use PPO, SAC, and DDPG agents. The policy network for each agent consists of a feed-forward network with two hidden layers, having 64 units and 32 units, respectively. We set a learning rate of $3 \cdot 10^{-4}$ and a batch size of 64. All ensemble models and individual agents are trained, validated, and tested on a **rolling-window basis** with 30-day training, 5-day validation, and 5-day testing windows.

Ensemble methods. The first method (Ensemble 1) consists of 1 PPO, 1 SAC, and 1 DDPG agents; the second method (Ensemble 2) consists of 5 PPO, 5 SAC, and 5 DDPG agents; and the third method (Ensemble 3) consists of 10 agents for each type. As in Section 5.1.2.2, all three ensemble models use the weighted average approach to combine component agent action probabilities.

Results. As seen in Table 5, the PPO agent achieves the highest cumulative returns of 63.37%, Sharpe ratio of 1.55, and Sortino ratio of 2.44, showing an ability to maintain high returns with controlled volatility and downside risk. Although DDPG's cumulative returns are comparable to PPO's, its higher maximum drawdown of -13.15% signals a greater risk of large value drops, which is a concern for risk management. SAC has a lower maximum drawdown than DDPG but underperforms in other metrics. All individual agents significantly outperform two traditional baselines across all metrics. The ensemble models also maintain profitability and risk management advantages over the baselines. Ensemble 1 has a high cumulative return of 62.60%, and as shown in Fig. 7 (a), it shows superior performance from Sep 2022 to Oct 2023. Ensemble 1 also achieves the smallest maximum drawdown and a higher Sharpe ratio than SAC and DDPG. Ensembles 1 and 2 have high RoMaD and Calmar ratios, showing an ability to quickly recover from peak-to-trough losses and a potential for steady growth in market adversities.

6.2.3 | LLM-engineered Signals for Stock Trading

We designed two tasks around LLM-engineered signals from news, including FinRL Contest 2024 LLM-Engineered Signals with RLME and FinRL Contest 2025 FinRL-DeepSeek for Stock Trading.

†† GitHub repo for models and reports: https://github.com/Open-Finance-Lab/FinRL_Contest_2023/tree/main/Task_1

Team	Algorithm	Cumulative Return	Sharpe Ratio	Max Drawdown	Rachev Ratio
Otago Alpha	PPO	191.14%	1.0800	-28.16%	1.0557
Ruijian & Sally	Refined GRPO	335.57%	0.9500	-50.24%	1.0300
Kuxlnw	Adaptive Portfolio	163.88%	0.0574	-33.10%	0.9389
	PPO 100	204.51%	0.0671	-36.56%	1.0027
	CPPO 100	90.52%	0.0359	-34.78%	1.0194
	PPO-DeepSeek 100	94.43%	0.0433	-34.09%	0.9441
	CPPO-DeepSeek 100	91.26%	0.0383	-43.03%	0.8684
Queen's Gambit	PPO with MIST 1	238.70%	0.2859	-92.20%	6.7999*
	PPO with MIST 2	342.65%	0.2897	-92.47%	6.6723*
S&P 500	/	90.03%	0.0448	-33.93%	0.9047
Nasdaq-100	/	164.52%	0.0558	-35.56%	0.9434

TABLE 6 The performance of top winner teams for FinRL Contest 2025 Task 1 FinRL-DeepSeek for Stock Trading. * The Rachev ratio of Queen's Gambit is reported wrongly and needs further scrutiny.

FinRL Contest 2024 LLM-Engineered Signals with RLHF. Participants fine-tuned LLMs to generate signals through reinforcement learning from market feedback. The training dataset was daily OHLCV data and financial news for $K = 7$ large-cap stocks from 01/01/2020 to 10/08/2022 (447 trading days). The effectiveness of LLM-engineered signals is evaluated on a dataset from 10/09/2022 to 12/15/2023 (229 trading days), using a long-short trading strategy, i.e., long (buy) the three stocks with the highest sentiment scores and short (sell) the bottom three, closing all positions three days later. **Aethernet**^{†† 43} fine-tuned the LLaMA-3.2-3B-Instruct model through RLHF to generate sentiment scores from news. The team designed a reward function to assess the sentiment score based on market data. The model had a cumulative return of 134.05%, while a simple buy-and-hold strategy yields a 72.71% return. It indicates that the LLM-engineered signals were effective even using a simple long-short execution strategy. The win/loss ratio of 1.5 suggests that the model generated more profitable trades than losing ones.

FinRL Contest 2025 FinRL-DeepSeek for Stock Trading. This task uses a different approach to combine FinRL and LLM-engineered signals, where the signals are integrated into the environment to train FinRL agents. The training dataset is daily OHLCV data with 10 market indicators (listed in Table 2) and financial news for Nasdaq 100 constituent stocks from 01/01/2013 to 12/31/2018 (1510 trading days). The models are evaluated on the testing dataset from 01/01/2019 to 12/31/2023 (1258 trading days). The winning teams used different approaches:

- **Otago Alpha**⁴⁴ added the put-call ratio to the feature vector, which is an assessment of put and call option transaction volumes. The put-call ratio was sourced from the OptionMetrics via the Wharton Research Data Services (WRDS) database and Bloomberg Terminal.
- **Ruijian & Sally**^{§§ 45} used the Group Relative Policy Optimization (GRPO)⁴⁶ algorithm, which is refined by Decoupled Clipping and Dynamic sAmpling Policy Optimization (DAPO). The team also designed a reward function adjusted by sentiment and risk.
- **Kuxlnw**^{¶¶ 47} adopted a dynamic model selection mechanism that adapts to market conditions and macroeconomic indicators. The team added new features to the environment, including interest rates, gold prices, and oil prices. They also used DeepSeek-generated sentiment scores and risk levels in the features. They trained specialized FinRL agents for each stock. Then they employed a mechanism to dynamically allocate the most suitable trading agents based on bull or bear markets. The algorithms they use include PPO with 100 training epochs (PPO 100), PPO with DeepSeek-generated signals and 100 training epochs (PPO-DeepSeek 100), CPPO with 100 training epochs (CPPO 100), CPPO with DeepSeek-generated signals and 100 training epochs (CPPO-DeepSeek 100), and Adaptive Portfolio applying the dynamic model selection mechanism.
- **Queen's Gambit**^{## 48} proposed a two-phase framework, Market-Informed Sentiment for Trading (MIST), to extract signals from news. In the first phase, it used a structured prompt with few-shot examples for DeepSeek-R1 7B to evaluate financial news and assign a sentiment score (1–5). In the second stage, the team designed an advanced prompt incorporating market behavior (i.e., short-term stock price change direction) to either reinforce or contradict the sentiment extracted in the first

^{††} https://github.com/Arnav-GrOver/ICAFI_FinRL-2024

^{§§} <https://github.com/Ruijian-Zha/FinRL-DAPO-SR>

^{¶¶} <https://github.com/Vorakorn1001/FinRL-DeepSeek>

^{##} <https://github.com/sahar-arshad/QueensGambit-FinRL2025>

Model	Ensemble-1	Ensemble-2	Ensemble-3	DQN	Double DQN	Dueling DQN	Fixed-Time Exit	BTC Price
Cumulative Return	0.66%	0.66%	0.66%	0.34%	0.48%	0.48%	-0.1%	0.74%
Sharpe Ratio	0.28	0.28	0.28	0.15	0.21	0.21	-0.03	0.20
Maximum Drawdown	-0.73%	-0.73%	-0.73%	-0.93%	-0.98%	-0.98%	-1.00%	-1.3%
RoMaD	0.90	0.90	0.90	0.37	0.49	0.49	0.10	0.59
Sortino Ratio	0.39	0.39	0.39	0.20	0.29	0.29	-0.04	0.28
Omega Ratio	1.08	1.08	1.08	1.04	1.05	1.05	0.99	1.05
Win/Loss Ratio	1.622	1.622	1.622	1.309	1.617	1.617	0.5	-

TABLE 7 crypto trading task performance. The second-level LOB data for Bitcoin is split into out-of-sample data for training and in-sample data for testing. Ensemble models use majority voting on agent actions.

stage. The final trading signal is integrated into the environment to train FinRL agents. The team trained two PPO agents, one with only the first stage of MIST (PPO with MIST 1) and one with two stages of MIST (PPO with MIST 2).

The performance of the winning teams are shown in Table 6. Team Otago Alpha, Ruijian & Sally, and Queen’s Gambit showed a superior performance over S&P 500 and Nasdaq-100 in cumulative return and Sharpe ratio. Otago Alpha achieved the highest Sharpe ratio (1.08) and maximum drawdown ($-28/16\%$), and the cumulative return outperformed two market indices, showing a good balance between profitability and risk management. Although Ruijian & Sally and Queen’s Gambit showed high cumulative returns, they had a worse maximum drawdown of -50.24% and -92.47% respectively compared with two market indices, indicating a larger downside risk. A Rachev ratio larger than 1 indicates that their strategies’ upside tail rewards outweighed their downside tail risks during extreme market movements. Kuxlnw’s adaptive portfolio underperformed its PPO 100 model in cumulative return, Sharpe ratio, and Rachev ratio, suggesting that a multi-asset trading agent may have more stable performance in this setting.

6.3 | Crypto Trading Tasks

The FinRL Contest 2024 crypto Trading with Ensemble Learning and the FinRL Contest 2025 FinRL-AlphaSeek for Crypto Trading are tasks designed for Bitcoin trading.

6.3.1 | Crypto Trading with Ensemble Methods

In our experiment, the crypto trading task for Bitcoin (BTC) ($K = 1$) is performed using three ensemble models, and individual DQN, Double DQN, and Dueling DQN agents.

crypto data: The dataset comprises second-level LOB data for BTC from 04/07/2021 11:32:42 to 04/19/2021 09:54:22. Adaptations from 101 formulaic alphas²⁹ are calculated based on LOB data to extract insights into market behaviors, such as momentum, mean-reversion, and market anomalies. A recurrent neural network (RNN) further processes the 101 alphas into $I = 8$ technical indicators. It reduces the complexity of the input data and enhances the ability to predict market trends, thus improving generalization and avoiding overfitting. The RNN is trained on data from 04/07/2021 11:32:42 to 04/17/2021 00:38:02 without future information leaks. The agents are trained on the in-sample data from 04/17 00:38:03 to 04/19 09:09:21 and tested on the out-of-sample data from 04/19 09:09:22 to 04/19 09:54:22.

Agents for crypto trading task. We use DQN, Double DQN, and Dueling DQN agents. The policy network for each agent consists of a feed-forward neural network with three 128-unit hidden layers. We set an exploration rate of 0.005, a learning rate of $2 \cdot 10^{-6}$, and a batch size of 512.

Ensemble methods The first method (Ensemble 1) consists of 1 DQN, 1 Double DQN, and 1 Dueling DQN agents; the second method (Ensemble 2) consists of 3 DQN, 3 Double DQN, and 3 Dueling DQN agents; and the third method (Ensemble 3) consists of 10 agents for each type. As in Section 5.1.2.3, three ensemble models use a majority voting approach to aggregate the agents’ actions. All ensemble models and individual agents are trained on the in-sample data and tested on the out-of-sample data.

Results. As seen in Table 7 and Fig. 7 (b), Double DQN and Dueling DQN agents have similar performance, with cumulative returns of 0.48%. This is lower than the BTC price baseline. Despite this, they achieve higher Sharpe ratios of 0.21 and lower maximum drawdowns of -0.98% than the fixed-time exit strategy and BTC price baseline, suggesting effective risk management.

Contest	Team	Cumulative Return	Sharpe Ratio	Maximum Drawdown
FinRL Contest 2024	Fermion	0.23%	0.0037	-0.28%
FinRL Contest 2025	Mt.Everest	-0.05%	-0.0008	-1.23%
Baseline	BTC price	-7.35%	-0.0022	-17.02%

TABLE 8 The performance of winners for FinRL Contest 2024 Task 1 “crypto Trading with Ensemble Learning” and FinRL Contest 2025 Task 2 “FinRL-AlphaSeek for Crypto Trading”.

The three different ensemble models have similar performances, which may be due to the limited action space at each timestep, causing agents to output identical actions. Their cumulative returns are close to the BTC price baseline. Moreover, the ensemble models outperform all individual agents in all metrics, achieving the highest Sharpe ratio of 0.28 and the lowest maximum drawdown of -0.73% . They also have the highest win/loss ratio of 1.62. This shows that ensemble methods can mitigate the risks associated with the decision-making failures of single agents.

We observe that individual agents and majority voting ensembles have near identical performances; in the restricted action space, agent policies may be converging, indicating a greater risk of significant losses due to less diversified trading actions. Compared to the spot price and fixed-time exit, the ensemble consistently achieved a higher return over the maximum drawdown ratio, highlighting superior risk-adjusted returns. Due to a lack of diversity in agents, we observed near-identical results between ensembles and individual agents. Nonetheless, over a relatively short timespan of 30 minutes, we observe that the vectorized agents can outperform other strategies.

This was crucial in maintaining portfolio stability amid high market volatility. The ensemble generally showed a higher win rate and a lower loss rate than the individual agents, reflecting its enhanced decision-making accuracy and consistency. The comparative analysis between the ensemble model and individual trading agents underscores the ensemble’s superior capability in managing risks and capitalizing on market opportunities. While individual agents provide valuable insights and are crucial components of the ensemble, the aggregated approach of the ensemble offers a more robust and effective solution for trading in the volatile crypto markets.

6.3.2 | Crypto Trading in FinRL Contests

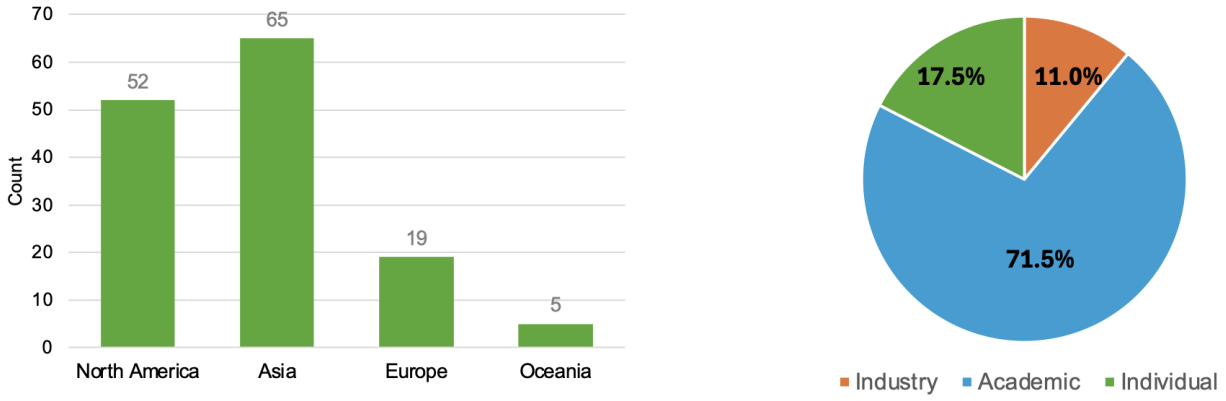
Based on the previous experiments, we designed tasks allowing participants to explore novel factor mining strategies or ensemble methods. The training dataset is a second-level LOB dataset for Bitcoin from 04/07/2021 11:32:42 to 04/17/2021 00:38:02. The testing dataset is from 04/17/2021 00:38:02 to 04/19/2021 09:54:22. We re-partitioned the dataset to allow for a longer testing period. The performance of winners is shown in Table 8.

- **Fermion** used ensemble methods to combine the strengths of different on-policy and off-policy algorithms. The team applied majority voting based on confidence scores to determine the trading action.
- **Mt.Everest** added new technique indicators in the state of the market environment, including order book imbalance, spread-based liquidity, order flow imbalance, price impact of trades, market-to-limit order ratio, intraday volatility, momentum scores, and noise-to-signal ratio. In addition, the team used principal component analysis (PCA) to extract the top 10 strongest factors from 101 alpha factors²⁹. To construct the agent pool, they applied different policy networks, including LSTM, transformer, and dilated CNNs. The action was determined through majority voting.

Both teams outperformed the baseline BTC price in cumulative returns, Sharpe ratio, and maximum drawdown. Fermion achieved a positive cumulative return of 0.23% and a Sharpe ratio of 0.0037. Both teams’ higher maximum drawdown shows better downside risk control and loss mitigation, which is a notable achievement given BTC’s inherent volatility.

In addition, FinRL-Crypto^{||| 30} introduces a multi-crypto trading strategy. The agents were trained using five-minute-level OHLCV data for $K = 10$ cryptocurrencies from 02/02/2022 to 04/30/2022. The agents were trained with multiple hyperparameter settings across 10 training-validation splits. To address backtest overfitting, agents with a high probability of overfitting were rejected at a 10% significance level. The model was evaluated on the dataset from 05/01/2022 to 06/27/2022, which includes two market crashes. The best-performing PPO agent achieved a cumulative return of -34.96% , outperforming the S&P crypto

||| https://github.com/AI4Finance-Foundation/FinRL_Crypto



(a) Geographical distribution of 141 participants who reported institutions. (b) Distribution of affiliation types of 200 participants in total.

FIGURE 8 Distribution of participation by geography and affiliation type.

Broad Digital Market Index (S&P BDM Index, -50.78%) and the equal-weight strategy (-47.78%). It also had lower volatility (0.0020), compared with S&P BDM Index (0.0581) and the equal-weight strategy (0.0042).

7 | ORGANIZATIONAL ASPECTS

We held FinRL contests at several academic conferences, including ACM International Conference on AI in Finance (ACM ICAIF 2023, 2024), IEEE International Conference on Intelligent Data and Security (IEEE IDS 2025), and IEEE International Conference on Cyber Security and Cloud Computing (IEEE CSCloud 2025). In this section, we summarize the organizational experience of the FinRL Contests, including encouraging participation, contest process, platform setup, event promotion, registration and submission approaches, and communication.

Participation. From 2023 to 2025, a total of 200+ students, researchers, and practitioners participated. Fig. 8 shows the distribution of participation by geography and type of affiliation. Among the 141 participants who reported their institutions, 37% of participants are from North America, including the United States and Canada. 46% of participants come from Asia, mainly from China, South Korea, and India. Among the 200+ participants, 71.5% of them come from 62 academic institutions, such as Columbia University, the National University of Singapore, and Shenzhen University. 17.5% of them are from 21 industrial institutions. (The statistics exclude the ongoing FinAI Contest 2025 @ IEEE CSCloud).

Contest Process. We first set important dates, including team registration, starter-kit release, model submission deadline, report submission deadline, and leaderboard release. The full contest typically lasts 2–3 months. It aims to provide participants with sufficient time for solution development while ensuring timely evaluation and feedback.

- **Team registration.** Each team consists of 1-4 people. Team registration starts as early as possible to leave enough time for promotion. The starter kit is nearly ready when team registration begins so that registered teams can be engaged and start early. Registration remains open until the model submission deadline to continuously encourage new teams to join.
- **Starter kit release.** The starter kit is released within a week after team registration begins. This short buffer period is used to organize and clean the prepared datasets, code, and instructions. The early release ensures that teams can be engaged early and have ample time to develop their solutions.
- **Model submission.** The model submission deadline is set more than a month after the starter kit release. This period gives teams sufficient time to get familiar with the tasks, study the tutorials, explore the dataset, use the code, and develop their own models.
- **Report submission.** In addition to models, participant teams are encouraged to submit a 2-3 page short paper describing the methodologies and performance. The report submission deadline is set one week after the model submission deadline. Participant teams can finalize documentation after completing model development. We invited a panel of 11 expert reviewers, including Ph.D. researchers, postdoctoral fellows, and professors from Columbia University, New York University Shanghai,

Stevens Institute of Technology, and Princeton University. They review the submitted reports and give valuable revision advice.

- **Result announcement.** Final results are announced 15 days after the report submission deadline. During this period, evaluators assess the submitted models, and experts review the reports. Final rankings are determined based on a weighted score, 60% of model performance and 40% of the report assessment.

Platform Setup. Multiple platforms are set up to host different resources for the contest, as shown in Table 1:

- **Contest website.** The website includes the contest overview, task description, contest timeline, registration and submission guidelines, contact information, and links to related resources. It is set up by using **GitHub Pages** for wide accessibility. It also serves as the main portal for announcements and updates.
- **GitHub repository.** A GitHub repo is set up to host the starter kit. The datasets, code, and detailed descriptions are organized in a folder for each task. It also contains around 8 tutorial notebooks for participant teams to get started.
- **Documentation website.** The documentation serves as a detailed guide and resource hub of FinRL Contests for participants. It includes FinRL introduction, task descriptions, and detailed explanations of the starter kit and baseline solutions.

Event Promotion. The FinRL Contests welcome students, researchers, and practitioners who are passionate about finance and machine learning. The contest is promoted widely through mailing lists of Google Groups and universities, and social media platforms, such as LinkedIn, Twitter, and Facebook, highlighting its inclusive and challenging nature. We also collaborate with some media partners, including Wilmott, PyQuant News, and Paris Machine Learning Group.

Registration and Submission. Team registration and model submission are through **Google Forms** for easy management. Registration requires a team name and members' information, including name, email, and affiliation. For model submission, teams provide a link to the GitHub and/or Hugging Face repository. The repository should be well organized, including code, model weights, scripts, and detailed instructions for evaluation. The report submission is through **OpenReview** or the channel required by the conference.

Communication Channel. We have multiple channels to communicate with participants for their questions, announcements, and tutorials.

- **Contact email.** We use an official contact email, `finrlcontest@gmail.com`, to send important announcements and respond to inquiries.
- **Group chat.** We set up a Discord channel of 500+ people and 2 WeChat groups of 270+ people to facilitate participant networking, discussion, and direct Q&A.
- **GitHub issue.** Participants are encouraged to create GitHub issues in the starter kit repository to seek technical support.
- **Online sessions.** 3 online Q&A sessions were organized via Zoom to address common concerns and questions. 1 online tutorial session was organized to introduce the FinRL framework, the contest, and the usage of the starter kit.

8 | CONCLUSIONS AND FUTURE WORKS

In this paper, we present the financial reinforcement learning benchmark through FinRL Contests, held from 2023 to 2025. The contests provided standardized task definitions, curated market datasets, GPU-optimized parallel market environments, and defined evaluation metrics to ensure reproducibility and comparability across various financial applications such as stock trading, cryptocurrency trading, and the integration of signals from LLMs. Participant teams employed diverse strategies, including feature engineering, ensemble learning methods, and LLM-generated market signals, demonstrating significant progress in both performance and risk management compared to conventional market benchmarks.

For future work, we will continue exploring and integrating LLM-generated signals from multimodal financial data in FinRL, such as SEC filings, earnings conference calls, alternative data, etc. The current breakthrough of LLMs in strong reasoning ability during inference time could also bring great potential in financial tasks. We will further develop FinRL trading agents that can be integrated into real-time intelligent trading systems, capable of processing and reasoning over multimodal data. We will utilize LLMs to process real-time financial data (e.g. financial dataset and FintHub) into trading signals, which will be put into the state of the trading agent. We will continue to actively integrate the most cutting-edge techniques with financial applications.

AUTHOR CONTRIBUTIONS

This is an author contribution text.

ACKNOWLEDGMENTS

We thank Li Deng, Zihan Ding, Jian Guo, Zhouchi Lin, Christina Dan Wang, Zhaoran Wang, Matt White, Bo Wu, Xiaojun Wu, Zhuoran Yang, Daochen Zha, and Chuheng Zhang for serving as advisors of FinRL Contest 2023-2025. We thank Mostapha Benhenda, Jin Bo, Jiale Chen, Qian Chen, Arnav Grover, Ethan Havemann, Sarah Huang, Colin Lin, Shivan Mukherjee, Jaisal Patel, Charlie Shen, Kent Wu, Yangyang Yu, and Andy Zhu for serving as organizers, and Steve Ewald, Andrew Li, Nikola Maruszewski, Christopher Minn, and Gavin Wang for creating the evaluation platform.

We thank Zihan Dong, Allan Feng, Astarag Mohapatra, Louis Owen, Guoxuan Wang, Zhiyuan Wang, Bruce Yang, Luna Zhang, Jiahao Zheng, and Ming Zhu from the open-source AI4Finance community, who contributed to the open-source FinRL project. We also thank Jimin Huang from the FinAI community. We thank Renyuan Xu, Huining Yang, and Bo An for academic feedback on FinRL papers.

We thank all participant teams!

FINANCIAL DISCLOSURE

The authors thank Vatic Investment and the Shanghai Frontiers Science Center of Artificial Intelligence and Deep Learning at NYU Shanghai for covering registration fees for participant teams.

Keyi Wang, Nikolaus Holzer, Jiechao Gao, Anwar Walid, and Xiao-Yang Liu Yanglet acknowledge the support from Columbia's SIRS and STAR Program, as well as The Tang Family Fund for Research Innovations in FinTech, Engineering, and Business Operations. Xiao-Yang Liu Yanglet also acknowledges the support from a NSF IUCRC CRAFT center research grant (CRAFT Grant 22017) for this research. The opinions expressed in this publication do not necessarily represent the views of NSF IUCRC CRAFT.

CONFLICT OF INTEREST

The authors declare no potential conflict of interest.

References

1. Liu XY, Yang H, Chen Q, et al. FinRL: A deep reinforcement learning library for automated stock trading in quantitative finance. *Deep Reinforcement Learning Workshop, NeurIPS*. 2020.
2. Liu XY, Yang H, Gao J, Wang CD. FinRL: deep reinforcement learning framework to automate trading in quantitative finance. *ACM International Conference on AI in Finance*. 2022.
3. Liu XY, Xiong Z, Zhong S, Yang H, Walid A. Practical deep reinforcement learning approach for stock trading. *Workshop on Challenges and Opportunities for AI in Financial Services, NeurIPS*. 2018.
4. Li Z, Liu XY, Zheng J, Wang Z, Walid A, Guo J. FinRL-Podracr: high performance and scalable deep reinforcement learning for quantitative finance. *Proceedings of the second ACM international conference on AI in finance*. 2021:1–9.
5. Liu XY, Xia Z, Rui J, et al. FinRL-Meta: Market environments and benchmarks for data-driven financial reinforcement learning. *Advances in Neural Information Processing Systems*. 2022;35:1835–1849.
6. Liu XY, Xia Z, Yang H, et al. Dynamic datasets and market environments for financial reinforcement learning. *Machine Learning - Nature*. 2024.
7. Charpentier A, Elie R, Remlinger C. Reinforcement learning in economics and finance. *Computational Economics*. 2021:1–38.
8. Fischer TG. Reinforcement learning in financial markets-a survey. *FAU discussion papers in economics*. 2018.
9. Hambly B, Xu R, Yang H. Recent advances in reinforcement learning in finance. *Mathematical Finance*. 2023;33(3):437–503.
10. Sun S, Wang R, An B. Reinforcement learning for quantitative trading. *ACM Transactions on Intelligent Systems and Technology*. 2023;14(3):1–29.
11. Bai Y, Gao Y, Wan R, Zhang S, Song R. A review of reinforcement learning in financial applications. *Annual Review of Statistics and Its Application*. 2025;12(1):209–232.
12. Rafailov R, Sharma A, Mitchell E, Manning CD, Ermon S, Finn C. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*. 2023;36:53728–53741.
13. Achiam J. Spinning Up in Deep Reinforcement Learning. 2018.

14. Ying C, Zhou X, Yan D, Zhu J. Towards Safe Reinforcement Learning via Constraining Conditional Value at Risk. *ICML 2021 Workshop on Adversarial Machine Learning*. 2022.
15. Deng Y, Bao F, Kong Y, Ren Z, Dai Q. Deep direct reinforcement learning for financial signal representation and trading. *IEEE Transactions on Neural Networks and Learning Systems*. 2016;28(3):653–664.
16. Avramelou L, Nousi P, Passalis N, Tefas A. Deep reinforcement learning for financial trading using multi-modal features. *Expert Systems with Applications*. 2024;238:121849.
17. Yang H, Liu XY, Zhong S, Walid A. Deep reinforcement learning for automated stock trading: An ensemble strategy. *ACM International Conference on AI in Finance*. 2020.
18. Zhang Z, Zohren S, Roberts S. Deep reinforcement learning for trading. *The Journal of Financial Data Science*. 2020;2(2):25–40.
19. Ardon L, Vadori N, Spooner T, Xu M, Vann J, Ganesh S. Towards a fully RL-based Market Simulator. *ACM International Conference on AI in Finance (ICAIF)*. 2021.
20. Amrouni S, Moulin A, Vann J, Vyetenko S, Balch T, Veloso M. ABIDES-Gym: Gym Environments for Multi-Agent Discrete Event Simulation and Application to Financial Markets. *ACM International Conference on AI in Finance (ICAIF)*. 2021.
21. Coletta A, Prata M, Conti M, et al. Towards Realistic Market Simulations: a Generative Adversarial Networks Approach. *ACM International Conference on AI in Finance (ICAIF)*. 2021.
22. Liu XY, Wang G, Yang H, Zha D. FinGPT: Democratizing internet-scale data for financial large language models. *Workshop on Instruction Tuning and Instruction Following, NeurIPS*. 2023.
23. Nie Y, Kong Y, Dong X, et al. A Survey of Large Language Models for Financial Applications: Progress, Prospects and Challenges. *arXiv:2406.11903*. 2024.
24. Christiano PF, Leike J, Brown T, Martic M, Legg S, Amodei D. Deep reinforcement learning from human preferences. *Advances in neural information processing systems*. 2017;30.
25. Benhenda M. FinRL-DeepSeek: LLM-Infused Risk-Sensitive Reinforcement Learning for Trading Agents. *arXiv:2502.07393*. 2025.
26. Peters J, Bagnell JA. *Policy gradient methods*:774–776; Boston, MA: Springer US . 2010.
27. Mohamed S, Rosca M, Figurnov M, Mnih A. Monte Carlo gradient estimation in machine learning. *J. Mach. Learn. Res.*. 2020;21(1).
28. Dong Z, Fan X, Peng Z. Fnsfid: A comprehensive financial news dataset in time series. *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2024.
29. Kakushadze Z. 101 formulaic alphas. *arXiv preprint arXiv:1601.00991*. 2016.
30. Gort BJD, Liu XY, Gao J, Chen S, Wang CD. Deep reinforcement learning for cryptocurrency trading: Practical approach to address backtest overfitting. *AAAI Bridge on AI for Financial Services*. 2023.
31. Guo J, Wang S, Ni LM, Shum HY. Quant 4.0: engineering quantitative investment with automated, explainable, and knowledge-driven artificial intelligence. *Frontiers of Information Technology & Electronic Engineering*. 2024;25(11):1421–1445.
32. Cao B, Wang S, Lin X, et al. From Deep Learning to LLMs: A survey of AI in Quantitative Investment. *arXiv preprint arXiv:2503.21422*. 2025.
33. Liu A, Feng B, Xue B, et al. DeepSeek-v3 technical report. *arXiv preprint arXiv:2412.19437*. 2024.
34. Markowitz H. Portfolio Selection. *The Journal of Finance*. 1952;7(1):77–91.
35. Wiering MA, Hasselt vH. Ensemble algorithms in reinforcement learning. *Trans. Sys. Man Cyber. Part B*. 2008;38(4):930–936.
36. Pérez-Cruz F. Kullback-Leibler divergence estimation of continuous distributions. 2008:1666–1670.
37. Mnih V, Kavukcuoglu K, Silver D, others . Human-level control through deep reinforcement learning. *Nature*. 2015;518:529–533.
38. Hasselt Hv, Guez A, Silver D. Deep reinforcement learning with double Q-Learning. 2016:2094–2100.
39. Wang Z, Schaul T, Hessel M, Van Hasselt H, Lanctot M, De Freitas N. Dueling network architectures for deep reinforcement learning. 2016;48:1995–2003.
40. Ganaie M, Hu M, Malik A, Tanveer M, Suganthan P. Ensemble deep learning: a review. *Eng. Appl. Artif. Intell.*. 2022;115(C).
41. Yin J, Wang R, Guo Y, et al. Wealth Flow Model: Online Portfolio Selection Based on Learning Wealth Flow Matrices. *ACM Trans. Knowl. Discov. Data*. 2021;16(2).
42. Kremer PJ, Lee S, Bogdan M, Paterlini S. Sparse portfolio selection via the sorted L1-Norm. *Journal of Banking & Finance*. 2020;110:105687.
43. Grover A. FinRLlama: A Solution to LLM-Engineered Signals Challenge at FinRL Contest 2024. *arXiv:2502.01992*. 2025.

44. Li S, Yu M, Dossor F. Option-Driven Sentiment in FinRL: a PPO Approach to Trading . *2025 IEEE 11th International Conference on Intelligent Data and Security (IDS)*. 2025:62-64.
45. Zha R, Liu B. A New DAPO Algorithm for Stock Trading . *2025 IEEE 11th International Conference on Intelligent Data and Security (IDS)*. 2025:46-48.
46. Yu Q, Zhang Z, Zhu R, et al. DAPO: An Open-Source LLM Reinforcement Learning System at Scale. *arXiv preprint arXiv:503.14476*. 2025.
47. Kosidphokin V, Loedtrakunchai P, Sinamnuaiphon N, Kuptanon S. FinRL: Adaptive Model Selection for Reinforcement Learning in Stock Trading . *2025 IEEE 11th International Conference on Intelligent Data and Security (IDS)*. 2025:71-72.
48. Arshad S, Ameer H, Azhar N, Latif S. FinRL Contest 2025 Task 1:Market-Aware In-Context Learning Framework for Proximal Policy Optimization in Stock Trading Using DeepSeek . *2025 IEEE 11th International Conference on Intelligent Data and Security (IDS)*. 2025:76-78.