Machine Learning (2024) 113:2795–2839
https://doi.org/10.1007/s10994-023-06511-w

Dynamic datasets and market environments for financial reinforcement learning

Xiao-Yang Liu[1] · Ziyi Xia[1] · Hongyang Yang[1] · Jiechao Gao[2] · Daochen Zha[3] · Ming Zhu[4] · Christina Dan Wang[5] · Zhaoran Wang[6] · Jian Guo[7]

Received: 29 August 2023 / Revised: 28 October 2023 / Accepted: 16 December 2023 /
Published online: 26 February 2024

Abstract

The financial market is a particularly challenging playground for deep reinforcement learning due to its unique feature of dynamic datasets. Building high-quality market environments for training financial reinforcement learning (FinRL) agents is difficult due to major factors such as the low signal-to-noise ratio of financial data, survivorship bias of historical data, and model overfitting. In this paper, we present an updated version of FinRL-Meta, a data-centric and openly accessible library that processes dynamic datasets from real-world markets into gym-style market environments and has been actively maintained by the AI4Finance community. First, following a DataOps paradigm, we provide hundreds of market environments through an automatic data curation pipeline. Second, we provide homegrown examples and reproduce popular research papers as stepping stones for users to design new trading strategies. We also deploy the library on cloud platforms so that users can visualize their own results and assess the relative performance via community-wise competitions. Third, we provide dozens of Jupyter/Python demos organized into a curriculum and a documentation website to serve the rapidly growing community. The codes are available at https://github.com/AI4Finance-Foundation/FinRL-Meta

Keywords Financial reinforcement learning · FinRL · Dynamic dataset · Market environment · AI4Finance · Open finance

✉ Xiao-Yang Liu
xl2427@columbia.edu

[1] Columbia University, New York, USA

[2] University of Virginia, Charlottesville, USA

[3] Rice University, Houston, USA

[4] Institute of Automation, Chinese Academy of Sciences, and University of Chinese Academy of Sciences, Beijing, China

[5] New York University (Shanghai), Shanghai, China

[6] Northwestern University, Evanston, USA

[7] IDEA Research, International Digital Economy Academy, Shenzhen, China

1 Introduction

Financial reinforcement learning (FinRL) (Liu et al., 2021; Hambly et al., 2023) is a promising interdisciplinary field of finance and reinforcement learning, driven by the spirit of "*trade with an (technology) edge*".[1] In the past decade, deep reinforcement learning (DRL) (Sutton & Barto, 2018), as a disruptive technology, has delivered a superhuman performance in Atari games (Mnih et al., 2015), Go (Silver et al., 2016, 2017), StarCraft II (Vinyals et al., 2019), the recent eye-catching ChatGPT (Ouyang et al., 2022), and GPT-4 (OpenAI, 2023). The financial market is a particularly challenging playground for DRL algorithms due to the unique feature of *dynamic datasets*, a sharp contrast to the static ImageNet dataset (Deng et al., 2009).

Researchers designed and applied many deep neural networks to real-world visual applications. However, these applications deal with "static" datasets, such as MNIST, CIFAR-10, Yale Face Database, which are quite different from financial markets, where financial data is dynamic in nature. The market trends, development of companies, and economic situation of different countries are refreshing over time. To handle time-sensitive financial data, models are expected to learn up-to-date information and adapt to real-time market situations.

Existing works (Lussange et al., 2021; Liu et al., 2021; Pricope, 2021) have already applied various DRL algorithms in financial applications, including investigating market fragility (Raberto et al., 2001), designing profitable strategies (Liu et al., 2018; Yang et al., 2020; Zhang et al., 2020), and assessing portfolio risk (Lussange et al., 2021; Bao & Liu, 2019). Exemplar financial trading tasks include

- Fundamentals analysis: e.g., value investment, growth investment
- Technical analysis: commodity trading advisor (CTA), momentum, trend following, etc.
- Macro strategies: e.g., bonds, gold, crude oil, forex.
- Quantitative strategies: statistical arbitrage, merge/event arbitrage, etc.
- High-frequency trading: different sub-strategies, cryptocurrency trading

Some recent works (Lussange et al., 2021; Amrouni et al., 2021; Han et al., 2023) have shown that DRL algorithms can deliver better trading performance than classical strategies and conventional machine learning methods in terms of cumulative return and Sharpe ratio. Several recent works (Lussange et al., 2021; Amrouni et al., 2021; Han et al., 2023) showed the great potential of DRL-based market simulators. However, these works are difficult to reproduce. Several recent efforts have been dedicated to facilitating reproducibility. The FinRL library (Liu et al., 2020, 2021) provided an open-source framework for financial reinforcement learning. Unfortunately, it only focused on the reproducibility of backtesting performance by providing several market environments. A conference version of this work, FinRL-Meta (Liu et al., 2022), provided financial datasets and several benchmark applications, but it did not provide a dynamic dataset, sentiment data, and market simulator.

[1] Find a (technology) edge and position to win.

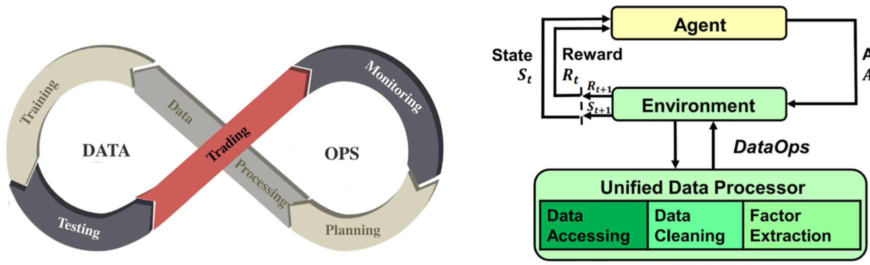


Fig. 1 DataOps paradigm (left) and FinRL-Meta (right)

However, building near-real market environments for financial reinforcement learning (FinRL) is difficult due to major factors such as low signal-to-noise ratio (SNR) of financial data, partial observation, reward delay, survivorship bias of historical data, and model overfitting. Such a *simulation-to-reality gap* (Dulac-Arnold et al., 2021, 2019) degrades the performance of DRL strategies in real markets. A good backtest performance does not necessarily reflect the actual trading performance since the financial dataset could be susceptible to flaws like missing data, noise, and anomalies.

Data-centric AI (Whang et al., 2023; Zha et al., 2023, 2023) is a new trend that appeals to shift the focus from model to data. It focuses on the systematic engineering of data in building AI systems, which can lead to better model behaviors in the real world (Zha et al., 2023). When training DRL agents, if the data is susceptible to flaws, even if our model/strategy can have a good backtest performance, it will be hard to tell whether the model can actually perform well in the real deployment or is just a result of overfitting the training dataset. Therefore, building a standard workflow to ensure data quality is imperative to foster reliable market environments and benchmarks and motivate the research and industrialization of FinRL.

In this paper, we present an updated version of FinRL-Meta, a data-centric and openly accessible library that has been actively maintained by the AI4Finance community, with a particular focus on data quality. The aim of FinRL-Meta is to create an infrastructure to enable real-time paper trading and facilitate the real-world adoption of DRL algorithms. This project also contributes to the broader RL or ML research community since it allows researchers to test DRL agents in real and dynamic environments at a low cost.

To handle highly unstructured financial big data, we follow the DataOps paradigm and implement an automatic data curation pipeline, as shown in Fig. 1 (left). The DataOps paradigm (Atwal, 2019; Ereth, 2018) refers to a set of practices, processes, and technologies that combines automated data engineering and agile development (Ereth, 2018). It helps reduce the cycle time of data engineering and improve data quality. Following the DataOps paradigm, we design an RLOps pipeline tailored for FinRL to continuously produce benchmarks on dynamic market datasets. The RLOps pipeline consists of the following steps:

- The first step is task planning, such as stock trading, portfolio allocation, and cryptocurrency trading.
- Then, we do data processing, including data accessing and cleaning, and feature engineering.
- Next step is where DRL takes part in. In particular, the training–testing-trading process, which will be given in Fig. 4.

- The final step is performance monitoring/analysis.

Figure 1 (right) shows an overview of FinRL-Meta. First, following the DataOps paradigm (Atwal, 2019; Ereth, 2018), we provide hundreds of market environments through an automatic data curation pipeline that collects dynamic datasets from real-world markets and processes them into standard gym-style environments. Second, we reproduce several popular papers as benchmarks, including high-frequency stock trading, cryptocurrency trading and stock portfolio allocation, serving as stepping stones for users to design new strategies. With the help of the data curation pipeline, we hold our benchmarks on cloud platforms so that users can visualize their own results and assess the relative performance via community-wise competitions. Third, we provide dozens of Jupyter/Python demos as educational materials, organized in a curriculum for community newcomers with different levels of proficiency and learning goals. At the same time, we maintain a documentation website to serve the emerging community.

The contribution and novelty of our work is two-fold. Firstly, we have introduced FinRL-Meta, the first unified framework that seamlessly supports a wide array of environments. Additionally, we furnish plug-and-play RL agents to streamline agent evaluation. FinRL-Meta grants access to numerous previously inaccessible research resources in FinRL, thus paving the path for future innovations in FinRL. This contribution significantly enriches the field. Secondly, we have carefully construct a data pipeline for processing financial data, encompassing data access, cleaning, feature engineering, and sentiment analysis. Given the well-known challenge posed by the low signal-to-noise (SNR) ratio in financial data and the dynamic nature of the financial data, our pipeline takes an essential initial step in bridging this gap, thereby facilitating forthcoming research in FinRL. Moreover, our DataOps development paradigm ensures that the data pipeline is easily extensible, allowing researchers to seamlessly integrate their own designs. Our work pioneers a focus on data centrality in FinRL, establishing a framework to support it for the first time, which is a significant and innovative contribution.

The remainder of this paper is organized as follows. Section 2 describes related works. Section 3 presents an overview of FinRL and the FinRL-Meta framework. Section 4 describes the automatic data curation pipeline. In Sect. 5, we present several homegrown examples. Finally, we conclude this paper and discuss future works in Sect. 6.

Disclaimer: The content of this paper is intended for academic research purposes only. Nothing herein is financial advice, and NOT a recommendation to trade real money. Please use common sense and always first consult a professional before trading or investing.

2 Related works

We review the technology landscape and existing works on financial big data, data-centric AI, DataOps practices, data-driven DRL, and FinRL applications.

2.1 Handling financial big data

Financial big data: Financial big data refers to the vast amount of data that is available in the financial industry from various sources. By analyzing this data, traders can make informed decisions about investments. Like big data in other domains, financial big data shares the four key properties, known as the 4V's[2]:

- 1) Volume. Financial big data has a large scale. Now, the market can support high-frequency trading at the level of microseconds. With high-frequency trading, billions of shares can be traded within a day, generating an enormous amount of data that records these transactions.
- 2) Velocity. As the market refreshes at the microsecond level, the velocity of data transmission and processing is very important. Companies are continually seeking ways to speed up data transmission and processing to minimize delays and errors. This includes utilizing closer physical proximity to data vendors, improving cable materials, and developing more efficient algorithms.
- 3) Variety. There are structured and unstructured data in finance. A data vendor may provide structured data for users to access, usually the volume-price data. Besides that, there are a huge amount of alternative data from news, social media, and financial reports that are considered during the process of generating indicators.
- 4) Veracity. Data quality and availability are extremely important. However, big data's larger scale also brings a latent risk of lack of veracity. Due to its close relationship with money and assets, data quality is particularly sensitive in the finance industry.

Data-centric AI: With the 4V's properties, the quality of financial data plays a critical role in enabling strong FinRL models in real deployments. In the past, FinRL research was conducted in a model-centric way, with an emphasis on improving model designs using predetermined datasets. However, solely depending on static datasets does not necessarily result in satisfactory model performance in real-world scenarios, especially when the dataset is flawed (Mazumder et al., 2022). Furthermore, neglecting the importance of data quality can trigger data cascades (Sambasivan et al., 2021), leading to reduced accuracy in real deployments.

Recently, the attention of researchers and practitioners has gradually shifted toward data-centric AI (Whang et al., 2023; Zha et al., 2023, 2023), with a stronger emphasis on improving data quality by the systematic engineering of data. The benefits of data-centric AI have been validated by both researchers and practitioners (Zha et al., 2023; Polyzotis & Zaharia, 2021). In order to drive tangible advancements in FinRL research and deployment, we have made our library data-centric by building upon dynamic datasets and implementing an automated data curation pipeline for quality control.

DataOps practices: From another perspective, a standardized development cycle is necessary for effectively handling highly unstructured financial big data. DataOps (Ereth, 2018; Atwal, 2019) applies the ideas of lean development and DevOps to the data science field. DataOps practices have been developed in companies and organizations to improve the quality and efficiency of data analytics (Atwal, 2019). These implementations consolidate various data sources, and unify and automate the pipeline of data analytics, including data accessing, cleaning, analysis, and visualization.

The DataOps paradigm (Ereth, 2018), or more accurately the methodology, is a way of organizing people, processes and technology to deliver reliable and high-quality data efficiently to all its users. The practice of DataOps focuses on enabling collaboration across

[2] The Four V's of Big Data: https://opensistemas.com/en/the-four-vs-of-big-data/

the organization to drive agility, speed of delivery and new data initiatives. By leveraging the power of automation, DataOps aims to address the challenges associated with inefficiencies in access, preparation, integration and availability of data.

Many researchers studied FinRL applications (Liu et al., 2018; Yang et al., 2020; Zhang et al., 2020; Ardon et al., 2021; Amrouni et al., 2021; Coletta et al., 2021) by building their own market environments. Despite the above-mentioned open-source libraries that provide some valuable settings, there are no established benchmarks yet. On the other hand, the data accessing, cleaning and feature/factor extraction processes are usually limited to data sources like Yahoo Finance and Wharton Research Data Services (WRDS).

However, the DataOps methodology has not been applied to FinRL research. Most researchers access data, clean data, and extract technical indicators (features) in a case-by-case manner, which involves heavy manual work and may not guarantee high data quality.

2.2 Data-driven reinforcement learning

Traditional RL requires a simulated environment to interact with. There are problems such as expensive cost, time consuming and unmanageable stochasticity. While based on the pre-collected datasets, RL will be a powerful method to perform data-driven strategies[3] with no need of interacting with the environment or human intervention. RL has the potential to reduce human labor and therefore improves automation. FinRL-Meta refers to the idea of data-driven RL and takes the character of large scale financial data. We would first introduce several popular topics related to data-driven RL.

Offline RL (Levine et al., 2020): Offline RL is a typical data-driven formulation of reinforcement learning problems. In offline RL, agents learn behaviors from a fixed dataset, without the process of exploration in the environment. It has great potential in tasks where collecting real-time data is inconvenient, either too expensive or risky(e.g., robotic tasks, autonomous driving), or the amount of data is limited (e.g., stock market, clinical surgery).

Curriculum learning: Curriculum learning is a technique that trains the model using multiple stages from simple to complex. With fine-tuning for specific tasks, curriculum learning could have faster convergence and find better minima. (Mahfouz et al., 2023) proposed to use two stages of training for the portfolio management task, first using a neural network to fit the mean-variance optimization, then fine-tuning the model with online reinforcement learning. Well-known products like AlphaGo and ChatGPT both use similar techniques.

RL from human feedback (RLHF): The idea of RLHF was first introduced by OpenAI and DeepMind in 2017 Christiano et al. (2017). RLHF allows human feedback as part of the agent's reward function, making it a better alignment of model performance and human expectations. One of the main reasons that ChatGPT outperforms other large language models is its appropriate usage of RLHF.

Next, we review several popular projects that are related to data-driven reinforcement learning and financial reinforcement learning:

[3] Note that "data-driven" and "data-centric" are two distinct concepts. The former refers to utilizing data to guide policy training, whereas the latter means placing data quality in the central role in FinRL development. The endeavors of "data-driven" and "data-centric" approaches complement each other in their efforts to enhance overall policy performance.

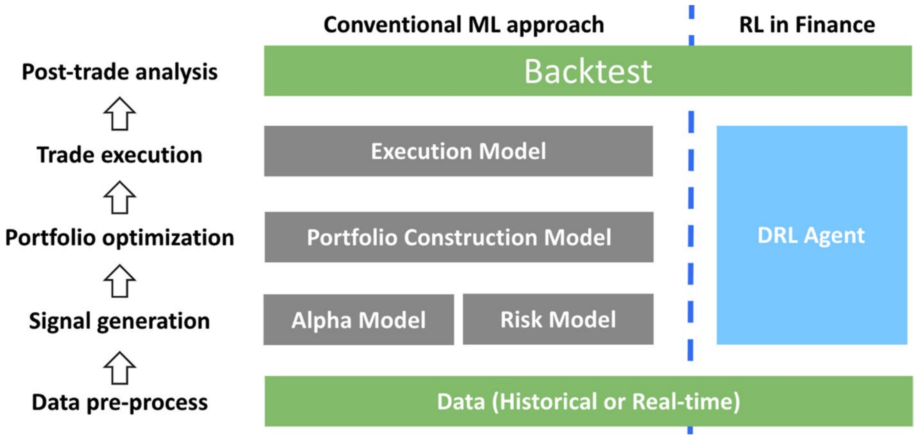


Fig. 2 Comparison between conventional machine learning approach and RLOps in finance for an algorithmic trading process

- **OpenAI gym** (Brockman et al., 2016) Environments are crucial for training DRL agents (Sutton & Barto, 2018). OpenAI gym provides standardized environments for a collection of benchmark problems that expose a common interface, which is widely supported by many libraries (Raffin et al., 2021; Liang et al., 2018; Liu et al., 2021). Three trading environments, TradingEnv, ForexEnv, and StocksEnv, are included to support stock and FOREX markets. However, it has not been updated for years.
- **Game of Go.** AlphaGo (Silver et al., 2016) and AlphaGo Zero (Silver et al., 2017) are programs for games of Go. AlphaGo combines Monte Carlo simulation with value and policy networks, and becomes the first computer program that defeats world champions in Go game. AlphaGo Zero is the updated version, it learns by reinforcement learning by playing against itself, without extra human data or knowledge. These programs provide valuable suggestions for financial reinforcement learning, e.g., how to train the policy network by supervised learning and self-play.
- **D4RL** (Fu et al., 2020) introduces the idea of *datasets for deep data-driven reinforcement learning* (D4RL). It provides benchmarks in offline RL. However, D4RL does not provide financial environments.
- **FinRL** (Liu et al., 2020, 2021) is an open-source library that builds a full pipeline for financial reinforcement learning. It contains three market environments, i.e., stock trading, portfolio allocation, and crypto trading, and two data sources, i.e., Yahoo Finance and WRDS. However, those market environments of FinRL cannot meet the community's growing demands.
- **NeoRL** (Qin et al., 2022) collected offline RL environments for four areas, CityLearn (Vázquez-Canteli et al., 2019), FinRL (Liu et al., 2020, 2021), Industrial Benchmark (Hein et al., 2017), and MuJoCo (Todorov et al., 2012), where each area contains several gym-style environments. Regarding financial aspects, it directly imports market environments from FinRL.

As a new rising technology, DRL is a promising tool for complicated financial tasks. Recently, many Wall Street companies have shown great interest in this rising technology. However, DRL's instability makes it hard to put into practice in the industry. AlphaGo and

ChatGPT's success in data-feed multi-stage learning could be a great approach to adopting RL into their workflow. Hedge funds first feed their private data into supervised learning (imitation learning) models for training, and then use reinforcement learning to fine-tune the model based on interaction with the market environments to achieve super-human performance.

2.3 Financial reinforcement learning (FinRL)

RLOps paradigm in finance. Algorithmic trading (Treleaven et al., 2013; Nuti et al., 2011) has been widely adopted in financial investments. The lifecycle of a conventional machine learning strategy may include five general stages, as shown in Fig. 2 (left), namely data pre-processing, modeling and trading signal generation, portfolio optimization, trade execution, and post-trade analysis. Recently, deep reinforcement learning (DRL) (Silver et al., 2016, 2017; Sutton & Barto, 2018) has been recognized as a powerful approach for quantitative finance, since it has the potential to overcome some important limitations of supervised learning, such as the difficulty in label specification and the gap between modeling, positioning, and order execution.

We would like to extend the principle of *MLOps* (Alla & Adari, 2021)[4] to the *RLOps in finance* paradigm that implements and automates the continuous training (CT), continuous integration (CI), and continuous delivery (CD) for trading strategies. Such a paradigm will have vast profit potential from a broadened horizon and fast speed, which is critical for wider DRL adoption in real-world financial tasks. The *RLOps in finance* paradigm, as shown in Fig. 2 (right), integrates middle stages (i.e., modeling and trading signal generation, portfolio optimization, and trade execution) into a DRL agent. Such a paradigm aims to help quantitative traders develop an end-to-end trading strategy with a high degree of automation, which removes the latency between stages and results in a compact software stack. One of the benefits is that it has the potential to explore and analyze large-scale financial data. Also, it provides chances for traders to continuously update trading strategies. However, the large-scale financial data and fast iteration of trading strategies bring imperative challenges in terms of computing power and time.

FinRL applications: Along with the RLOps paradigm, different FinRL applications can be constructed and deployed. Electronic trading is popular in many countries and is used in stock exchanges, electronic order books, over-the-counter markets, foreign exchange, etc. Electronic trading enhances liquidity since traders can easily buy or sell assets. We describe several FinRL applications inspired by a complete survey (Hambly et al., 2023), including optimal execution, portfolio optimization, option pricing and hedging, market making, smart order routing, and robo-advising.

- 1) Optimal execution is the problem of maximizing the return from buying or selling a given amount of an asset within a given time period. A classical framework is the Almgren-Chriss model, which relies heavily on the assumptions of the dynamics and the permanent and temporary price impact. There are several popular criteria to evaluate the performance of execution strategies, e.g., the profit and loss, implementation shortfall, and the Sharp ratio.

[4] MLOps is an ML engineering culture and practice that aims at unifying ML system development (Dev) and ML system operation (Ops).

- 2) Portfolio optimization aims to maximize some objective function by selecting and trading the best portfolio of assets. One typical model is mean-variance portfolio optimization, which aims to maximize the return for a given risk measured by variance.
- 3) Option pricing and hedging are important in finance. The Black-Scholes model is a typical mathematical model which aims to get the price of a European option given several constraints: stock price, expiration time, and the payoff at expiry.
- 4) Market making aims to earn the bid-ask spread by providing liquidity to the market by placing buy/sell limit orders in the limit order books.
- 5) Robo-advising or automated investment managing provides online financial advice with minimal human intervention. Large language models like ChatGPT (Ouyang et al., 2022) and GPT-4 (OpenAI, 2023) could be powerful assistant if we can integrate them well in the RLOps.

3 FinRL tasks and FinRL-meta framework

We introduce the Markov Decision Process (MDP) as a mathematical model of FinRL tasks, summarize major challenges, and then provide an overview of the proposed FinRL-Meta framework.

3.1 Modeling financial reinforcement learning (FinRL)

FinRL tasks in general take the form of sequential decision-making problems, which can be mathematically formulated as a Markov Decision Process (MDP) with five-tuple $(S, A, R, \mathbb{P}, \gamma)$ as follows

- State space S consists of all possible states;
- Action space A consists of all available actions;
- Reward function $R(s, a, s') : S \times A \times S \rightarrow \mathbb{R}$ assigns a real-valued reward to a transition (s, a, s');
- Transition probability $\mathbb{P}(s'|a, s) : S \times A \times S \rightarrow [0, 1]$ models the dynamics of a system (a.k.a., environment);
- Factor $\gamma \in (0, 1]$ discounts a future reward back to its present value.

MDP is a well-formed mathematical model for sequential decision-making tasks in that a decision-maker can partially or fully influence the outcome. State S and action A define the input and output for decision-making tasks, and reward function R allows a model to learn by goal-seeking optimization. Transition probability $\mathbb{P}$ allows the target problem to be stochastic, which is usually closer to reality. The discount factor γ makes the model consider the future reward when making decisions.

We summarize the state spaces, action spaces, and reward functions of FinRL applications in Table 1. States usually demonstrate the condition of market and the assets, such as the balance, shares of stocks, OHLCV data, technical indicators, sentiment data, etc. Actions are the operations allowed in the market, including buy/sell/hold certain shares of the stock, short or long, change of portfolio weights on stocks, etc. The reward functions indicate what kind of objective we want the agent to achieve, for example, a

Table 1 List of state space, action space, and reward function

Key components	Attributes
State	Balance $b_t \in \mathbb{R}_+$; Shares $\boldsymbol{h}_t \in \mathbb{Z}_+^n$
	Opening/high/low/close price $\boldsymbol{o}_t, \boldsymbol{h}_t, \boldsymbol{l}_t, \boldsymbol{p}_t \in \mathbb{R}_+^n$
	Trading volume $\boldsymbol{v}_t \in \mathbb{R}_+^n$
	Fundamental indicators; Technical indicators
	Social data; sentiment data
	Alpha and beta signals; Smart beta indexes, etc
Action	Buy/sell/hold
	Short/Long
	Portfolio weights
Reward	Change of portfolio value
	Portfolio log-return
	Sharpe ratio
Environments	Dow-30, S &P-500, NASDAQ-100
	Cryptocurrencies
	Foreign currency and exchange
	Futures; options; ETFs; Forex
	CN securities; US securities
	Paper trading; Live Trading

change of portfolio value (a larger positive change leads to a larger positive reward, vice versa) for maximizing excess return, Sharpe ratio for balancing return and risk, etc.

Objective function: Many financial tasks can be written in the form of optimization problems. When using RL, the objective function is to find a policy $\pi(s, a)$ that maximizes the discounted cumulative return $r = \sum_{t=0}^{T} \gamma^t R(s_t, a, s_{t+1})$.

Here we introduce the MDP setup for two of our homegrown examples: stock trading and portfolio optimization. More MDP setups for different tasks can be found in Appendix 3

Example I. For the stock trading task (Liu et al., 2018) on 30 constituent stocks of the Dow Jones Industrial Average (DJIA) index, we specify the "state-action-reward" as follows:

- State $\boldsymbol{s_t} = [b_t, \boldsymbol{p_t}, \boldsymbol{f_t}, \boldsymbol{h_t}] \in \mathbb{R}^{30(I+2)+1}$, where scalar $b_t \in \mathbb{R}_+$ is the remaining balance in the account, $\boldsymbol{p_t} \in \mathbb{R}_+^{30}$ is the prices of 30 stocks, $\boldsymbol{f_t} \in \mathbb{R}^{30 \cdot I}$ is a feature vector and each stock has I technical indicators, and $\boldsymbol{h_t} \in \mathbb{R}_+^{30}$ denotes the share holdings, where $\mathbb{R}_+$ is the set of non-negative real numbers.
- Action $\boldsymbol{a_t} \in \mathbb{R}^{30}$ denotes the trading operations on the 30 stocks, i.e., $\boldsymbol{h_{t+1}} = \boldsymbol{h_t} + \boldsymbol{a_t}$. When an entry $\boldsymbol{a}_t^i > 0, i = 1, ..., 30$, it means a buy-in of $\boldsymbol{a}_t^i$ shares on the i-th stock, negative action $\boldsymbol{a}_t^i < 0$ for selling, and zero action $\boldsymbol{a}_t^i = 0$ keeps $\boldsymbol{h}_t^i$ unchanged.
- Reward function $R(\boldsymbol{s_t}, \boldsymbol{a_t}, \boldsymbol{s_{t+1}}) \in \mathbb{R}$: Reward is an incentive signal to encourage the trading agent to execute action $\boldsymbol{a_t}$ at state $\boldsymbol{s_t}$. In the stock trading task (Liu et al., 2018), the reward function is set to be the change of total asset values, i.e., $R(\boldsymbol{s_t}, \boldsymbol{a_t}, \boldsymbol{s_{t+1}}) = v_{t+1} - v_t$, where v_t and v_{t+1} are the total asset values at state $\boldsymbol{s_t}$ and $\boldsymbol{s_{t+1}}$, respectively, i.e., $v_t = \boldsymbol{p_t}^\intercal \boldsymbol{h_t} + b_t \in \mathbb{R}$.

Example II. Another task is portfolio optimization, also on the 30 constituent stocks of the Dow Johnes Industrial Average (DJIA) index. The "state-action-reward" are specified as follows:

- State $\boldsymbol{s_t} = [v_t, \boldsymbol{p_t}, \boldsymbol{f_t}, \boldsymbol{w_t}] \in \mathbb{R}^{30(I+2)+1}$, where $\boldsymbol{p_t} \in \mathbb{R}^{30}_{+}$ is the current prices of 30 stocks, $\boldsymbol{f_t} \in \mathbb{R}^{30 \cdot I}$ is a feature vector and each stock has I technical indicators, $\boldsymbol{w_t} \in \mathbb{R}^{30}$ is the portfolio weight allocated on last time, and $v_t = v_{t-1}\boldsymbol{w_{t-1}}^{\intercal}\boldsymbol{p_t}/\boldsymbol{p_{t-1}}$ is the total asset value at time t.
- Action $\boldsymbol{a_t} \in \mathbb{R}^{30}$ denotes the new weights assigned to each of the 30 stocks. Note there are always $\sum_{i=1}^{30} \boldsymbol{a}_t^i = 1$. Each $\boldsymbol{a}_t^i > 0$ means allocating a_t^i amount of total asset on the i-th stock. After $\boldsymbol{a_t}$ is applied, $\boldsymbol{w_{t+1}}$ in the state will record the new $\boldsymbol{a_t}$, and v_{t+1} will be recalculated.
- Reward function $R(\boldsymbol{s_t}, \boldsymbol{a_t}, \boldsymbol{s_{t+1}}) \in \mathbb{R}$: Similar to the stock trading task, the reward of portfolio optimization task is also set to be the change of total asset values.

3.2 Major challenges

Training and testing environments based on historical data may not simulate real markets accurately due to the *simulation-to-reality gap* (Dulac-Arnold et al., 2021, 2019), and thus a trained agent cannot be directly deployed in real-world markets. We summarize the main FinRL challenges as follows:

- **Low signal-to-noise ratio (SNR)**: Data from different sources may contain large noises (Wilkman, 2020) such as random noise, outliers, etc. These cause the collected dataset with a low signal-to-noise ratio. It is challenging to identify alpha signals or build smart beta indices using such noisy datasets. FinRL-Meta's data curation pipeline holds data cleaning and feature engineering, which reduce noise and extract useful signals from the raw data.
- **Survivorship bias of historical market data**: Survivorship bias is caused by a tendency to focus on existing stocks and funds without consideration of those that are delisted (Brown et al., 1992). It could lead to an overestimation of stocks and funds, which will mislead the agent.
- **Model overfitting**: Existing research papers mainly report backtesting results. It is highly possible that authors are tempted to tune hyper-parameters of the chosen algorithm and retrain the agent multiple times[5] to obtain better backtesting results, resulting in model overfitting (Gort et al., 2023; De Prado, 2018). This might lead to big trouble during real-time trading. In FinRL-Meta, we use dynamic datasets to control the training–testing-trading window, which discards the outdated data and fetchs latest data. That prevents models from overfitting to the old data that are less influential.
- **Delay**: Financial markets have delays due to data transmission, reward feedback, and actuation. The true reward may be based on the users' interaction with the markets, which may take several days. As the delay increases, the performance of deep reinforcement learning decreases.
- **Partial observation**: The full observability assumption of financial markets can be extended to partial observation (the underlying states cannot be directly observed),

[5] There is information leakage.

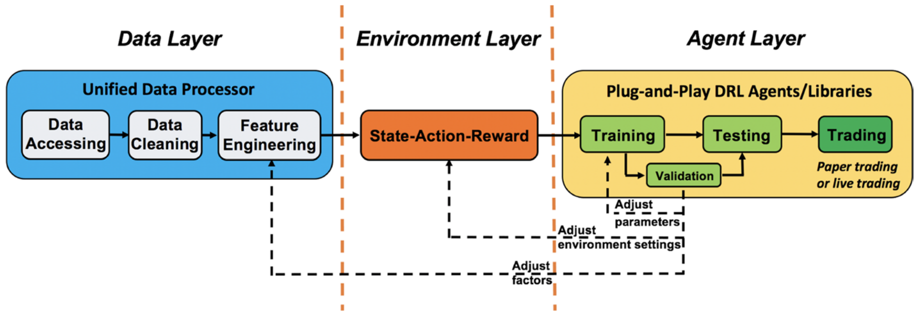


Fig. 3 Overview of FinRL-Meta framework, an automatic data curation pipeline

i.e., partially observable Markov Decision Process (POMDP). A POMDP model utilizes a Hidden Markov Model (HMM) (Mamon & Elliott, 2007) to model a time series that is caused by a sequence of unobservable states. Considering the noisy financial data, it is natural to assume that a trading agent cannot directly observe market states. Studies suggested that the POMDP model can be solved by using recurrent neural networks, e.g., an off-policy Recurrent Deterministic Policy Gradient (RDPG) algorithm (Liu et al., 2020), and a long short-term memory (LSTM) network that encodes partial observations into a state of a reinforcement learning algorithm (Rundo, 2019).

- **Multi-objective reward function**: When we optimize one metric, some other metrics may need to be constrained or improved. Therefore, a trade-off between these metrics may be required when we design the reward function. In addition, the reward formulation involves the weights of metrics, i.e., the weights should be fine-tuned manually to achieve the expected goal. The environment layer in FinRL-Meta allows self-design reward function, in which multiple objectives can be integrated into one formula calculating reward.
- **Low interpretability**: In deep reinforcement learning, neural networks are used to fit the Q-value functions and policies. However, neural networks are black-box; therefore, deep reinforcement learning is of low interpretability.

3.3 Overview of FinRL-meta framework

Following the DataOps paradigm in Fig. 1 and the standard process of data-centric AI (Zha et al., 2023), FinRL-Meta builds a universe of market environments for data-driven financial reinforcement learning. FinRL-Meta follows the *de facto* standard of OpenAI Gym (Brockman et al., 2016) and the *lean principle* of software development. We have an automatic pipeline using the dynamic dataset with the following steps: 1). task planning, 2). data processing, 3). training–testing-trading pipeline, 4). performance monitoring.

3.3.1 DataOps to layer structure

We adopt a layered structure to provide fast and easy process of RL training pipeline following the DataOps. As shown in Fig. 3, FinRL-Meta consists of three layers, data layer, environment layer, and agent layer. Layers interact through end-to-end interfaces, achieving high extensibility. For updates and substitutes inside a layer, this structure minimizes

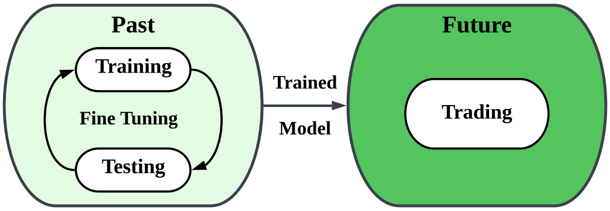


Fig. 4 Overview of the training–testing-trading pipeline in FinRL-meta

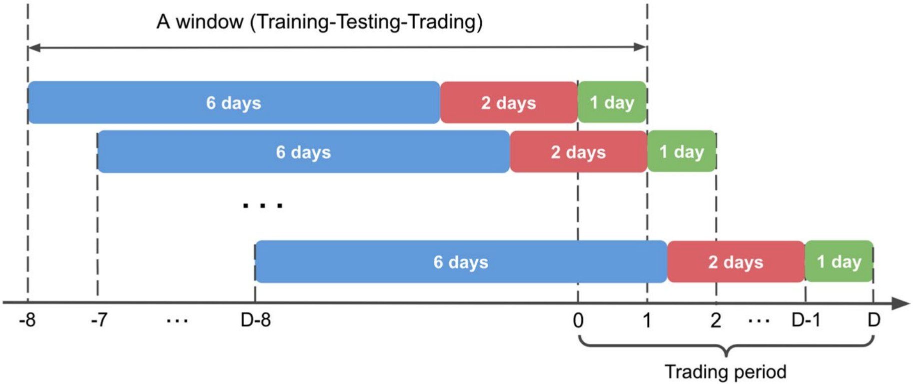


Fig. 5 A rolling window of training–testing-trading pipeline with dynamic dataset

the impact on the whole system. Moreover, the layer structure allows easy extension of user-defined functions and fast updating of algorithms with high performance.

Data Layer: we follow the DataOps paradigm for data curation to reduce the cycle time of data engineering and improve data quality. First, in the data processing step, we provide APIs to collect time series price data and sentiment data from different platforms with a unified interface. Second, the data cleaning step will clean up the raw data, which is usually unstructured and contain different kinds of missing or errors. Third, we will add technical indicators to the data to provide more information on the market in the feature engineering step. More details about the Data Layer will be discussed in Sec. 4.1.

Environment Layer: The well-processed data from the data layer will be made into a gym-style market environment in the environment layer. According to the chosen task, state-action-reward will be set. FinRL-Meta also provides the option of multiprocessing training via vector environment to accelerate the training process.

Agent Layer: we allow a user to plug in a DRL agent and play with a market environment from the environment layer. Currently, three libraries are supported, including Stable-baseline 3, RLlib, and ElegantRL. We call this plug-and-play mode, which will be introduced in detail later.

3.3.2 Dynamic datasets and training–testing-trading pipeline

For dynamic financial data, it is crucial to keep the model learning the latest information from the market. However, it will be very time-consuming to process data and train the model frequently. Thus, FinRL-Meta brings forward the concept of the dynamic dataset.

The dynamic dataset is a standardized workflow of downloading and processing data consistently and periodically following the need for an iteration of "training–testing-trading" pipeline. With the pre-designed data curation pipelines of Data Layer, FinRL-Meta can handle dynamic datasets well. This helps make the time of training controllable.

As shown in Fig. 4, we deploy a training–testing-trading pipeline. The DRL approach follows a standard end-to-end pipeline. The DRL agent is first trained in a training environment and then fined-tuned (adjusting hyperparameters) in a validation environment. Then the validated agent is tested on historical datasets (backtesting). Finally, the tested agent will be deployed in paper trading or live trading markets. As financial data refreshing exceedingly fast, models always need to be retrained on the new data. Figure 5 shows the desing of a rolling window using Training–Testing-Trading Pipeline with dynamic dataset in FinRL-Meta, providing a potential standard workflow of financial reinforcement learning.

3.3.3 Plug-and-play mode for DRL algorithms

A DRL agent can be directly plugged in the above training–testing-trading pipeline. The following DRL libraries are supported:

- **ElegantRL** (Liu et al., 2021): Lightweight, efficient and stable algorithms using PyTorch.
- **Stable-Baselines3** (Raffin et al., 2021): Improved DRL algorithms based on OpenAI Baselines.
- **RLlib** (Liang et al., 2018): An open-source DRL library that offers high scalability and unified APIs.

Take the stock trading task described in Sect. 3.1 as an example. We first download and preprocess the historical data of the 30 constituent stocks of the DJIA index. Then, we construct an environment with the state-action-reward specified in Sect. 4. Given this environment, we choose an algorithm from one of the above three DRL libraries, plug it in with default parameters, and train a trading agent. Users could compare the performances of different algorithms from the same library or different libraries.

4 Automatic data curation pipeline for market environments

Financial big data is usually unstructured, which makes data curation necessary. Following the principles of data-centric AI (Whang et al., 2023; Zha et al., 2023, 2023), we construct an automatic data curation pipeline to process and engineer data. Specifically, we process four types of data (De Prado, 2018), including fundamental data (e.g., earning reports), market data (e.g., OHLCV data), analytics (e.g., news sentiment), and alternative data (e.g., social media data, ESG data). In addition, we incorporate NLP features for financial sentiment analysis. We build market environments using these features, by following OpenAI gym-style APIs (Brockman et al., 2016).

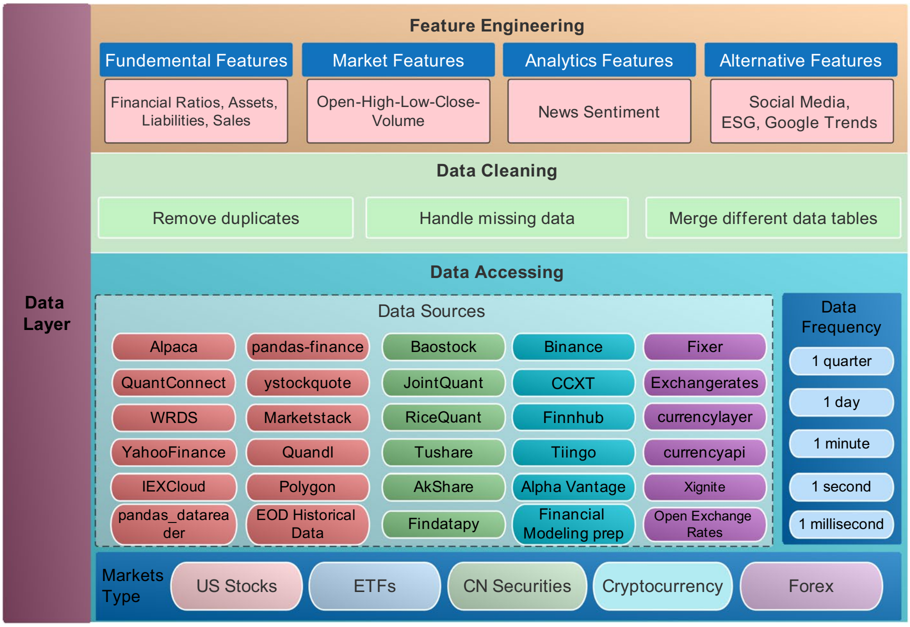


Fig. 6 Data layer of FinRL-Meta

4.1 Data layer for processing and engineering highly unstructured financial big data

Following the training data development pipeline in data-centric AI (Zha et al., 2023), we develop a data layer using DataOps practices (Ereth, 2018), shown in Fig. 6. We establish a standard pipeline for financial data engineering, which processes data from different sources into a unified market environment. The pipeline involves a data accessing module to collect data, as well as data cleaning and feature engineering modules to prepare data and transform it into a form that is appropriate for FinRL model training.

4.1.1 Data accessing

As the first step in the data curation pipeline, we designed a complete process for data accessing. Users can connect data APIs of different market platforms in Table 2 via our common interfaces. A DataProcessor class is prepared for all the supported data sources. All the common functions such as: download_data(), clean_data() can be overwritten according to the specific data sources. In this way, users can access data agilely by specifying the data source name, start date, end date, stock list, time interval, and other parameters. FinRL-Meta has supported more than 30 data sources, covering stocks, cryptocurrencies, ETFs, forex, etc.

Table 2 Supported data sources

Data source	Type	Max frequency	Raw data	Preprocessed data
Alpaca	US Stocks, ETFs	1 min	OHLCV	Prices, indicators
Baostock	CN Securities	5 min	OHLCV	Prices, indicators
Binance	Cryptocurrency	1 s	OHLCV	Prices, indicators
CCXT	Cryptocurrency	1 min	OHLCV	Prices, indicators
IEXCloud	NMS US securities	1 day	OHLCV	Prices, indicators
JoinQuant	CN Securities	1 min	OHLCV	Prices, indicators
QuantConnect	US Securities	1 s	OHLCV	Prices, indicators
RiceQuant	CN Securities	1 ms	OHLCV	Prices, indicators
Tushare	CN Securities	1 min	OHLCV	Prices, indicators
WRDS	US Securities	1 ms	Intraday Trades	Prices, indicators
YahooFinance	US Securities	1 min	OHLCV	Prices, indicators
AkShare	CN Securities	1 day	OHLCV	Prices, indicators
findatapy	CN Securities	1 day	OHLCV	Prices, indicators
pandas_datareader	US Securities	1 day	OHLCV	Prices, indicators
pandas-finance	US Securities	1 day	OHLCV	Prices, indicators
ystockquote	US Securities	1 day	OHLCV	Prices, indicators
Marketstack	50+ countries	1 day	OHLCV	Prices, indicators
finnhub	US Stocks, currencies, crypto	1 day	OHLCV	Prices, indicators
Financial Modeling prep	US stocks, currencies, crypto	1 min	OHLCV	Prices, indicators
EOD Historical Data	US stocks, and ETFs	1 day	OHLCV	Prices, indicators
Alpha Vantage	Stock, ETF, forex, crypto, technical indicators	1 min	OHLCV	Prices, indicators
Tiingo	Stocks, crypto	1 day	OHLCV	Prices, indicators
Quandl	250+ sources	1 day	OHLCV	Prices, indicators
Polygon	US Securities	1 day	OHLCV	Prices, indicators
fixer	Exchange rate	1 day	Exchange rate	Exchange rate, indicators
Exchangerates	Exchange rate	1 day	Exchange rate	Exchange rate, indicators
Fixer	Exchange rate	1 day	Exchange rate	Exchange rate, indicators
currencylayer	Exchange rate	1 day	Exchange rate	Exchange rate, indicators
currencyapi	Exchange rate	1 day	Exchange rate	Exchange rate, indicators
Open Exchange Rates	Exchange rate	1 day	Exchange rate	Exchange rate, indicators
XE	Exchange rate	1 day	Exchange rate	Exchange rate, indicators
Xignite	Exchange rate	1 day	Exchange rate	Exchange rate, indicators

OHLCV means open, high, low, and close prices; volume data

4.1.2 Data cleaning

Raw data retrieved from different data sources are usually of various formats and with erroneous or missing data to different extents. It makes data cleaning highly time-consuming. With a data processor, we automate the data-cleaning process. For example, we include filters to check the alignment of downloaded data and the user's expected frequency and time range. Then the dataframe will be restructured to a uniform layout with the same column order and names. Then missing data will be handled by either back fill or forward fill. In addition, we use stock ticker names and data frequency as unique identifiers to merge all types of data into a unified data table.

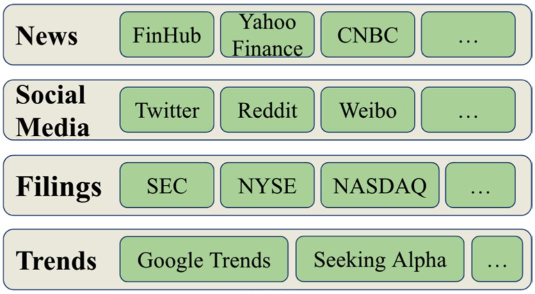
4.1.3 Feature engineering

The idea of data centric AI (Zha et al., 2023) emphasizes that the quality of data is crucial for the AI model's performance. Besides handling missing data and errors, discovering hidden information and signals is equally important. In financial tasks, feature engineering is an important step in the development of training data. Deep learning-based feature engineering has great potential to automate the design of technical indicators (Xiao et al., 2020; Nargesian et al., 2017). FinRL-Meta aggregates effective features to help improve model predictive performance. FinRL-Meta currently supports five types of features:

- **Market features**: Open-high-low-close price and volume data are the typical market data we can directly get from querying the data API. They have various data frequencies, such as daily prices from Yahoo Finance, and TAQ (Millisecond Trade and Quote) from WRDS. In addition, we automate the calculation of technical indicators based on OHLCV data by connecting the Stockstats[6] or TA-lib library[7] in our data processor, such as Moving Average Convergence Divergence (MACD), Average Directional Index (ADX), Commodity Channel Index (CCI), etc.
- **Fundamental features**: Fundamental features are processed based on the earnings data in SEC filings queried from WRDS. The data frequency is low, typically quarterly, e.g., four data points in a year. To avoid information leakage, we use a two-month lag beyond the standard quarter end date, e.g., Apple released its earnings report on 2022/07/28 for the third quarter (2022/06/25) of the year 2022. Thus for the quarter between 04/01 and 06/30, our trade date is adjusted to 09/01 (same method for the other three quarters). We also provide functions in our data processor for calculating financial ratios based on earnings data such as earnings per share (EPS), return on asset (ROA), price to earnings (P/E) ratio, net profit margin, quick ratio, etc.
- **Analytics features**: We provide news sentiment for analytics features. First, we get the news headline and content from WRDS (Li et al., 2019). Next, we use NLTK.Vader[8] to calculate sentiment based on the sentiment compound score of a span of text by normalizing the emotion intensity (positive, negative, neutral) of each word. For the time alignment with market data, we use the exact enter time, i.e., when the news enters the database and becomes available, to match the trade time. For example, if the trade time

[6] Github repo: https://github.com/jealous/stockstats
[7] Github repo: https://github.com/mrjbq7/ta-lib
[8] Github repo: https://github.com/nltk/nltk

Fig. 7 Data sources of NLP features



is every ten minutes, we collect the previous ten minutes' news based on the enter time; if no news is detected, then we fill the sentiment with 0.

- **Alternative features**: Alternative features are useful but hard to obtain from different data sources (De Prado, 2018), such as ESG data, social media data, Google trend searches, etc. ESG (Environmental, social, governance) data are widely used to measure the sustainability and societal impacts of investment. The ESG data we provide is from the Microsoft Academic Graph database, which is an open-resource database with records of scholarly publications. We have functions in our data processor to extract AI publication and patent data, such as paper citations, publication counts, patent counts, etc. We believe these features reflect companies' research and development capacity for AI technologies (Fang et al., 2019; Chen & Liu, 2020). It is a good reflection of ESG research commitment.
- **Natural language processing (NLP) features**: We employ NLP methods to extract patterns from many sources such as Twitter, Weibo, Google Trends, and Sina finance. NLP (Xing et al., 2018) greatly improves the efficiency of data processing, and reduces the labor of reading texts, websites, videos, and so on. NLP in finance can provide meaningful insights, e.g., sentiment analysis and question-answering like ChatGPT (Ouyang et al., 2022), and GPT-4 (OpenAI, 2023) when making decisions. NLP features are extracted from a large amount of raw data, and reflect the states, predictions, and emotions of traders, governments, financial institutions, etc. We list several NLP features here: number of comments, number of replies, number of praise/dispraise, and number of optimism/pessimism.

Apart from the above default features, users can quickly add customized features using open-source libraries or add user-defined features. New features can be added in two ways: 1) Write a user-defined feature extraction function. The returned features are added to a feature array. 2) Store the features in a file, and put it in a default folder. Then, an agent can read these features from the file.

4.2 Financial sentiment analysis

In addition to the features collected directly from the market data, we also perform sentiment analysis on other data sources, and the output sentiment score serves as another feature. FinRL-Meta investigates the potential extension of sentiment analysis to the financial market context and assesses its impact on automated trading. In addition to market data such as price and volume, NLP features can provide complementary information. The use of market data alone is inadequate in capturing unexpected market events, news, and

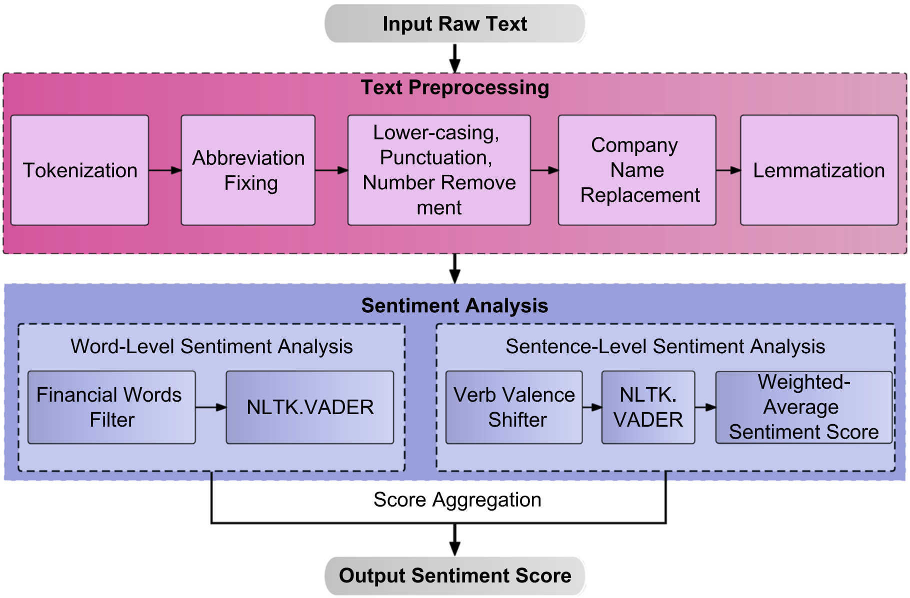


Fig. 8 Lexicon-based sentiment analysis framework

company announcements, leading to a diminished capacity of trading strategies to respond to unpredictable stock price fluctuations. As shown in Fig. 7, the incorporation of NLP features, specifically news, social media, company announcement, and trends sentiments, can help investors analyze market trends and facilitate the examination of the interplay between textual data and stock prices.

Previous studies have extensively investigated the use of NLP techniques and sentiment analysis in stock price prediction and automated trading strategies. In a review study, (Xing et al., 2018) summarized the NLFF (Natural Language Processing for Financial Forecasting) methodologies and their applications in related work. They organized and categorized these techniques into two primary groups: 1) Lexicon-based, which utilizes pre-trained financial sentiment orientation dictionaries (Loughran & McDonald, 2011) to label word segments with sentiment scores; and 2) Automatic Labeling, which employs label propagation frameworks to automatically construct lexicons for the financial domain using seed words (Hamilton et al., 2016; Tai & Kao, 2013). The lexicon-based approach has the advantage of analyzing texts at the word or sentence level by labeling groups of seed word segments with positive, negative, or neutral sentiments and assigning polarity scores to individual words to represent sentiment strengths.

Our observation of financial news revealed that the majority of sentiments are expressed through a small group of signaling words related to trading actions, such as ‘rise’ and ‘drop’. To address this characteristic, we recognized the need for word-level adjustments of sentiment weights for each word segment. As a result, we decided to use the lexicon-based approach in our study. However, we acknowledge that this approach has two major drawbacks: 1) words can have multiple meanings and sentiment strengths in different contexts, and 2) the meaning and sense of a word that is common in one domain, such as e-commerce, may not be common in finance. To overcome these challenges, we explore a specialized framework of lexicon-based sentiment analysis.

In FinRL-Meta, we propose a lexicon-based sentiment analysis framework that is specifically designed for NLP features. To achieve this, we created a customized sentiment dictionary tailored to the characteristics of the financial sector. Our aim is to improve the accuracy of sentiment classification beyond what has been achieved by previous NLP sentiment models by optimizing and extending the existing sentiment analysis framework while adapting it to the textual features of financial news.

4.2.1 Lexicon-based sentiment analysis framework

Figure 8 illustrates the structure of our lexicon-based sentiment analysis framework, which comprises two main stages: text preprocessing and multi-level sentiment analysis. The text pre-processing stage involves five specific procedures aimed at transforming unstructured news text into cleaned, structured text vectors suitable for sentiment analysis.

- 1) In the tokenization stage, the raw texts are tokenized to split them into sentence and word segments, which are then respectively parsed into word-level and sentence-level sentiment analysis functions.
- 2) Abbreviation Fixing is performed to restore abbreviations containing valuable sentiment signaling words to their complete expressions, enabling the sentiment analysis algorithm to capture all sentiment signaling words.
- 3) Lower-casing, Punctuation, and Number Removement are conducted to normalize word segments into their lower-cased format and remove unnecessary punctuation and numbers that contain no sentiment values.
- 4) Company Name Replacement is carried out to remove company names that contain sentiment-sensitive terms, and replace them with a term containing no sentiment sensitivity, such as "Company Target" or "Company Random".
- 5) Lemmatization is employed to group together the inflected forms of a word so they can be analyzed as a single item, identified by the word's lemma. Unlike stemming which reduces all terms to their word stem, lemmatization restores each word to its lemma form based on its part of speech tagging. For instance, the word "developed" is tagged as VERB and restored to the lemma "develop". The word "development" is tagged as NOUN and restored to "development". Since different inflected forms of a word demonstrate distinctive sentiment polarities, the lemmatization method is used instead of stemming.

The sentiment analysis stage of text data involves transforming word and sentence vectors obtained during text preprocessing into corresponding sentiment scores, which are then aggregated to obtain the overall sentiment score. This stage employs distinct methodologies for word and sentence vectors.

- 1) For the word vectors, to filter out irrelevant word segments and concentrate on finance-specific sentiment signaling words, we utilized the LoughranMcDonald MasterDictionary (Xing et al., 2018) with over 80,000 core financial terms as the base financial corpus. We then computed word-level sentiment scores using the VADER (Valence Aware Dictionary for Sentiment Reasoning) model in NLTK (Hutto & Gilbert, 2014).
- 2) For sentence vectors, we customized and adjusted the intensities of signaling words based on sentence logic and semantics using the Adverb Valence Shifter mechanism in

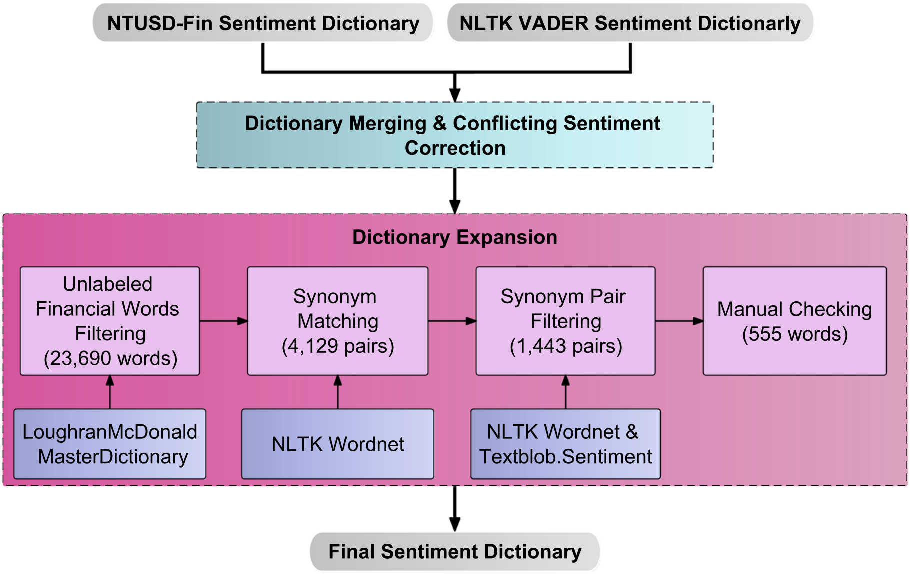


Fig. 9 Financial sentiment dictionary

the VADER package (Hutto & Gilbert, 2014). To incorporate sentiment-strengthening or weakening terms like "hardly" and "significantly", we also implemented the Verb Valence Shifter mechanism which identified the verb and noun words as the sentence separator for each input sentence. After adjusting the internal sentiment weights of each sentence, we devised rules to assign weights to different sentences and aggregate the sentence vectors to obtain sentence-level sentiment scores.

4.2.2 Financial sentiment dictionary construction

The sentiment dictionary is the foundation for lexicon-based sentiment analysis, containing information about the emotions or polarity expressed by words, phrases, or concepts. As noted in the previous section, one of the major obstacles associated with the lexical annotation approach is that the sentiment intensity and polarity of a given word often vary across different domains. For example, in e-commerce, the term "bull" usually refers to an animal species, while in the finance domain, it signifies an inclination for a specific market, security, or industry to rise.

We create a financial sentiment dictionary by customizing and expanding two existing sentiment dictionaries, as shown in Fig. 9. The construction process involved two stages: dictionary merge and dictionary expansion. During the first stage, we merged the NTUSD-Fin Sentiment Dictionary with the built-in dictionary of the NLTK VADER package. The NTUSD-Fin dictionary (Chen et al., 2018) is a sentiment dictionary specifically designed for finance, comprising labels for 8,331 words, 112 hashtags, and 115 emojis, and providing multiple scoring techniques, such as frequency, CFIDF, chi-squared value, market sentiment score, and word vector for tokens. The VADER sentiment dictionary (Hutto & Gilbert, 2014), created based on social media texts, includes 7,502 labeled words. Throughout the merging process, we detected and manually fixed 479 contradictions in the semantic polarities of words between the two dictionaries.

Table 3 Evaluation result of financial sentiment analysis

Approach	Polarity accuracy	Valence correlation
NLTK VADER (Baseline)	0.60	0.66
With Verb Valence Shifter	0.62	0.67
With Financial sentiment dictionary (Our Approach)	0.74	0.70

We also observe that certain financial terms with sentiment values were not accurately captured by the merged sentiment corpus, such as "bullish", "bearish", "outperform", "outgrowth", "high", and "rise". Consequently, in the second stage of the construction process, the dictionary was expanded to label more financial words with sentiment polarity and intensities through a synonym-matching mechanism. This process involves four main steps.

- 1) Unlabeled Financial Words Filtering: the LoughranMcDonald MasterDictionary (Xing et al., 2018), which contains over 80,000 core financial words, was utilized as the base financial corpus to filter out relevant financial words without attached sentiment polarities, resulting in 23,690 words being filtered out.
- 2) Synonym Matching: to assign sentiment polarity and intensity scores to the unlabeled words, the NLTK Wordnet was used to match each word with its labeled synonym. The WordNet (Miller, 1998) is an extensive lexical database of English words that groups nouns, verbs, adjectives, and adverbs into sets of cognitive synonyms called "synsets", each expressing a distinct concept. In this step, the 23,690 unlabeled words were matched with their synonym lists recommended by the WordNet Synset database, and word-synonym pairs with labeled synonyms were selected. This resulted in 4,129 word-synonym pairs being filtered out.
- 3) Synonym Pair Filtering: an algorithm was developed to further filter out words with the highest sentiment values from the remaining 4,129 word-synonym pairs. This algorithm utilized the TextBlob (Loria, 2018) subjectivity indicator and synonym path similarity indicator to identify the most sentiment-rich word-synonym pairs. The algorithm iterated through all words, checking their related synonyms and selecting the synonym with the highest similarity score in the SentiWordNet synonym network. After computing the subjectivity of each word, those with a subjectivity value of less than 0.2 were excluded. This step resulted in 1,443 word-synonym pairs being filtered out.
- 4) Manual Checking: each word-synonym-sentiment pair was manually checked, along with its actual usage in the real news context. If the path similarity between the word and its synonym was less than 0.5, its matched sentiment score was adjusted based on its meaning in the context. After this manual checking step, 555 words were selected as the final extended semantic dictionary.

To evaluate sentiment analysis accuracy involving both sentiment polarities and strengths, we conduct an experiment based on the news headlines corpus used in SemEval-2007 dataset (Strapparava & Mihalcea, 2007), which includes 1000 news headlines manually annotated with sentiment scores on six emotion categories: Anger, Disgust, Fear, Joy, Sadness, and Surprise. The sentiment scores range from -100 (highly negative) to 100 (highly positive), with 0 indicating neutral valence. Of the 1000 headlines, 468 have a positive valence, 526 have a negative valence, and 6 have a neutral valence. For this experiment, only headlines with clear positive emotions (sentiment valence of 50 to 100) and negative emotions (sentiment valence of -100 to -50) were selected, resulting in a dataset of 410 headlines, consisting of 155 positive and 255 negative headlines.

The evaluation results of the experiment are presented in Table 3. Two metrics were used to assess the accuracy of the sentiment analysis approaches Polarity Accuracy, which measures the proportion of predicted polarity (negative, neutral, and positive) that matches the labeled polarity, and Valence Correlation, which measures the correlation between predicted valence scores (ranging from -1 to 1) and labeled valence scores (ranging from -100 to 100), indicating the accuracy of the prediction of news sentiment intensities. As shown in the table, our sentiment analysis approach outperforms the baseline approach in terms of both Polarity Accuracy and Valence Correlation.

4.3 Environment layer for dynamic market environments

FinRL-Meta follows the OpenAI gym-style (Brockman et al., 2016) to create market environments using the cleaned data from the data layer. It provides hundreds of environments with a common interface. Users can build their environments using FinRL-Meta's interfaces, share their results and compare a strategy's trading performance. Following the gym-style (Brockman et al., 2016), each environment has three functions as follows:

- `reset()` function resets the environment back to the initial state s_0
- `step()` function takes an action a_t from the agent and updates state from s_t to s_{t+1}.
- `reward()` function computes the reward value transforming from s_t to s_{t+1} by action a_t.

Detailed descriptions can be found in Yang et al. (2020), Gort et al. (2023).

We plan to add more environments for users' convenience. For example, we are actively building market simulators using limit-order-book data (refer to Appx. 5.2.6), where we simulate the market from the playback of historical limit-order-book-level data and an order matching mechanism. We foresee the flexibility and potential of using a Hidden Markov Model (HMM) (Mamon & Elliott, 2007) or a generative adversarial net (GAN) (Goodfellow et al., 2014) to generate market scenarios (Coletta et al., 2021).

Incorporating trading constraints to model market frictions: To better simulate real-world markets, we incorporate common market frictions (e.g., transaction costs and investor risk aversion) and portfolio restrictions (e.g., non-negative balance).

- **Flexible account settings**: Users can choose whether to allow buying on margin or short-selling.
- **Transaction cost**: We incorporate the transaction cost to reflect market friction, e.g., 0.1% of each buy or sell trade.
- **Risk-control for market crash**: In FinRL (Liu et al., 2020, 2021), a turbulence index (Kritzman & Li, 2010) is used to control risk during market crash situations. However, calculating the turbulence index is time-consuming. It may take minutes, which is not suitable for paper trading and live trading. We replace the financial turbulence index with the volatility index (VIX) (Whaley, 2009) that can be accessed immediately.

Multiprocessing training via vectorized environments: We utilize GPUs for multiprocessing training, namely, the vectorized environments technique of Isaac Gym

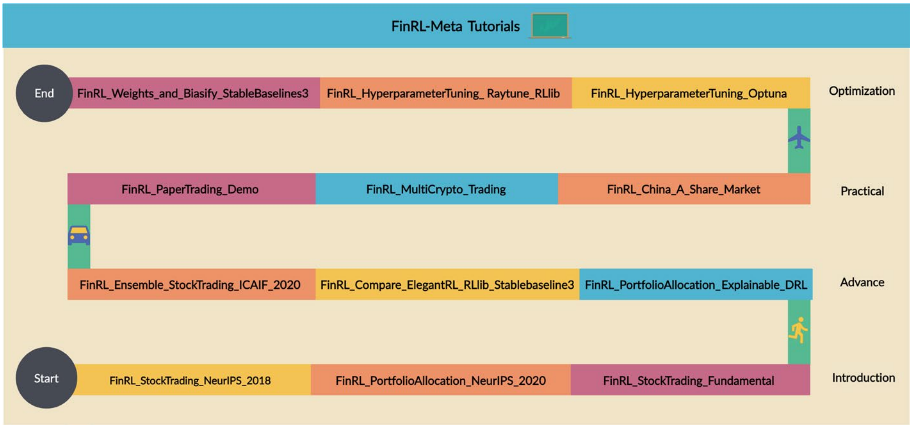


Fig. 10 Demos of FinRL-Meta, organized in a curriculum structure

(Makoviychuk et al., 2021), which significantly accelerates the training process. In each CUDA core, a trading agent interacts with a market environment to produce transitions in the form of {state, action, reward, next state}. Then, all the transitions are stored in a replay buffer and later are used to update a learner. By adopting this technique in our market simulator, we successfully achieve the multiprocessing simulation of hundreds of market environments to improve the performance of DRL trading agents on large datasets.

5 Homegrown examples and tutorials

We provide several homegrown examples and a dozen of tutorials, which serve as stepping stones for newcomers. We reproduce popular papers as benchmarks for follow-up research.

5.1 Performance metrics and baseline methods

We provide the following metrics to measure the trading performance:

- **Cumulative return** $R = \frac{v - v_0}{v_0}$, where v is the final portfolio value, and v_0 is the original capital.
- **Annualized return** $r = (1 + R)^{\frac{365}{t}} - 1$, where t is the number of trading days.
- **Annualized volatility** $\sigma_a = \sqrt{\frac{\sum_{i=1}^{n}(r_i - \bar{r})^2}{n-1}}$, where r_i is the annualized return in year i, $\bar{r}$ is the average annualized return, and n is the number of years.
- **Sharpe ratio** (Sharpe, 1994) $S_T = \frac{\text{mean}(R_t) - r_f}{\text{std}(R_t)}$, where $R_t = \frac{v_t - v_{t-1}}{v_{t-1}}$, r_f is the risk-free rate, and $t = 1, ..., T$.
- **Max. drawdown**: The maximal percentage loss in portfolio value.

The following baseline trading strategies are provided for comparison:

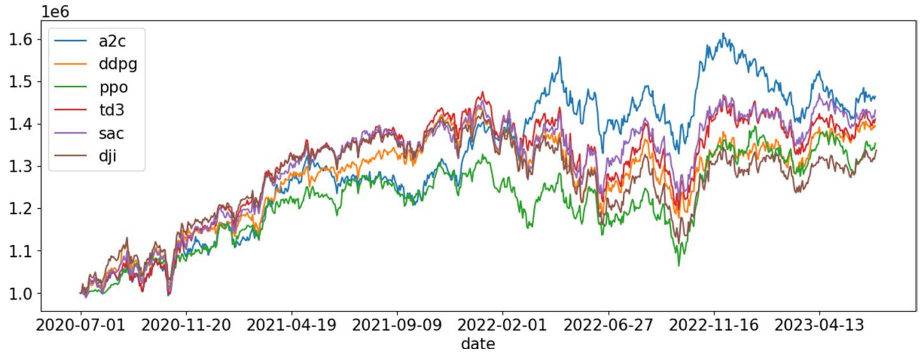


Fig. 11 Reproducing the stock trading of (Liu et al., 2018)

- **Passive trading strategy** (Malkiel, 2003) is a well-known long-term strategy. The investors just buy and hold selected stocks or indexes without further activities.
- **Mean-variance and min-variance strategy** (Ang, 2012) are two widely used strategies that look for a balance between risks and profits. They select a diversified portfolio in order to achieve higher profits at a lower risk.
- **Equally weighted strategy** is a portfolio allocation strategy that gives equal weights to different assets, avoiding allocating overly high weights on particular stocks.

5.2 Homegrown examples

For educational purposes, we provide Jupyter/Python as tutorials[9] as shown in Fig. 10 to help newcomers get familiar with the whole pipeline.

In Sect. 2.3, we have described the RLOps paradigm. Here we will demonstrate how to use RLOps in practice by reproducing several prior papers. We have reproduced experiments in several papers as benchmarks. Users can study our codes for research purposes or use them as stepping stones for deploying trading strategies in live markets. In this subsection, we describe several home-grown examples in detail, in a sequence of simple to advanced. Users could choose the one to learn and run based on their proficiency.

5.2.1 Stock trading task

The stock trading task is one of the most classical problems in FinRL-Meta. The goal is to train a DRL agent to decide the time, ticker, and amount to buy and sell on the stock market.

First, users can use the APIs provided by FinRL-Meta to fetch historical OHLCV (open, high, low, close prices, and volume) data of the stocks from data platforms like Yahoo Finance. After data cleaning and feature engineering, which checks the error and missing data, and then technical indicators are added.

Second, FinRL-Meta will split the processed data into training and testing, and construct a gym-style market environment for each, with the state-action-reward specified in

[9] https://github.com/AI4Finance-Foundation/FinRL-Tutorials.

Table 4 Trading performances of different algorithms in the stock trading task

	A2C	DDPG	PPO	TD3	SAC	DJIA
Initial asset (1e6)	1.0	1.0	1.0	1.0	1.0	1.0
Final asset (1e6)	1.464546	1.395279	1.353576	1.410849	1.431884	1.336998
Cumulative ret	46.45%	39.53%	35.36%	41.08%	43.19%	33.70%
Annual ret	15.48%	13.18%	11.79%	13.69%	14.40%	11.23%
Std. of daily ret	0.970%	0.982%	0.951%	0.996%	0.975%	1.0%
Max. drawdown	−14, 7%	−18.2%	−19.9%	−18.9%	−15.8%	−21.9%
Sharpe ratio	0.72	0.61	0.55	0.62	0.67	0.50

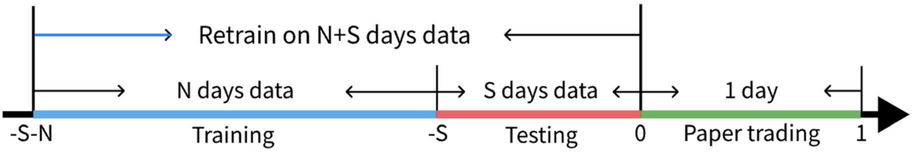


Fig. 12 Data split for a window that consists of training, testing, and trading

Sect. 3.1. Then, we are able to choose algorithms from any one of the DRL libraries and train the agents of these algorithms in the market environment.

Lastly, after we obtain the trained agents, we backtest the agents on the environment with testing data. During backtesting, the performance of the agents will be compared with the performance of several baselines, such as DJIA index, mean-variance, and equally weighted.

In FinRL-Meta's demo, after fetching from Yahoo! Finance, we use the data from 07/01/2010 to 07/01/2020 (10 years) for training and the data from 07/01/2020 to 07/01/2023 (3 years) for trading. The technical indicators in the state space include the following, Moving Average Convergence Divergence (MACD), Relative Strength Index (RSI), Commodity Channel Index (CCI), Average Directional Index (ADX), etc. All these data will be sealed in a gym-style environment.

During the training phase, we utilized dynamic datasets with a rolling window of 22 trading days and trained five different deep reinforcement learning (DRL) algorithms (A2C, DDPG, TD3, PPO, and SAC) in the environment. Figure 11 displays the comparison of their performances with the baseline DJIA index after backtesting all the agents. And performances of the agents measured by different metrics are shown in Table 4. The best agent was found to be A2C, achieving a return of 46.45% compared to DJIA's return of 33.70%. This demo provides a detailed walkthrough of how DRL operates in the stock trading task and serves as a benchmark for subsequent works in the field of financial reinforcement learning (Liu et al., 2018). This benchmark is beneficial for getting into the field of financial reinforcement learning.

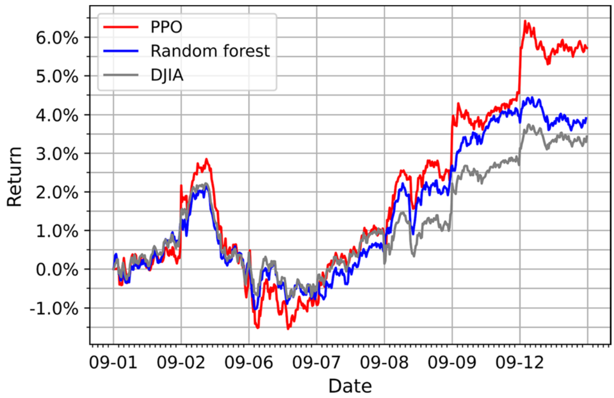


Fig. 13 Stock trading in real-time: cumulative returns for a conventional ML method (random forest) VS. our trained PPO agent

5.2.2 Trading in real time

Practical tasks like stock trading and cryptocurrency trading suffer from the false positive issue due to overfitting, where an agent might perform well on testing data but not on real-world markets. Since backtesting has problems of information leakage and overfitting, its results is not persuasive to show the quality of trained models. Thus, we propose to deploy DRL agent on paper trading.

Figure 12 shows our "training–testing-trading" pipeline. In a window, there are N days' data for training and S days' data for testing. At the end of a window, we perform paper trading for 1 day. Note that we always retrain the agent using $N + S$ days of training and testing data together. Then, we roll the window forward by 1 day ahead and perform the above steps for a new window. Paper trading is always carried out for 1 day. Therefore, D windows correspond to D trading days.

Alg. 1 summarizes the pipeline of paper trading. For D trading days from 0 to $D - 1$, we keep doing the following three steps:

- **Step 1)**. Download and process N-day data, from day $d - S - N$ to day $d - S - 1$. Then build the data into a gym-style environment and train the agent. Then download and process S-day data, from day $d - S$ to day $d - 1$. 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 hyper-parameters.
- **Step 2)**. Build the training and testing data, totally $N + S$ days from day $d - S - N$ to day $d - 1$ (note that there are 390 data points for each day's minute-level data), into a gym-style environment. Update hyper-parameters to the values chosen from **Step 1)**. Then retrain the agent on these $N + S$-day environments.
- **Step 3)**. Deploy the trained agent to the paper trading market and trade from 9:30 am to 4:00 pm.

Table 5 Trading performance of our agent and two baselines

	PPO (ours)	Random forest (Breiman, 2001)	DJIA Index
Initial portfolio value	100, 000	100, 000	100, 000
Final portfolio value	105,720	103, 906	103, 433
Cumulative return	5.72%	3.906%	3.433%
Avg. daily return	0.82%	0.56%	0.49%
Std. of daily return	2.33%	1.72%	1.27%
Max. drawdown	– 4.27%	– 3.10%	– 2.91%
Sharpe ratio	0.82	0.96	1.02

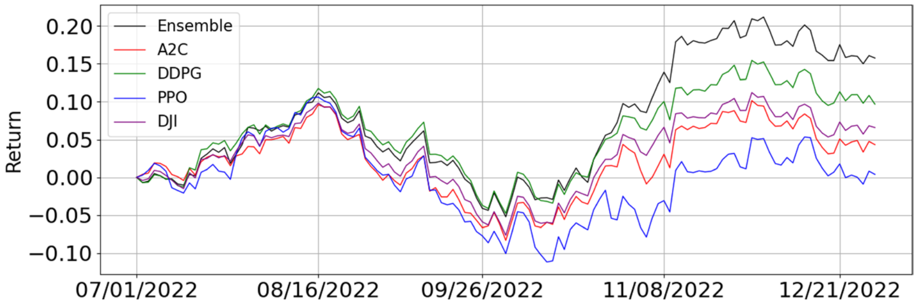


Fig. 14 Reproducing the ensemble strategy of (Yang et al., 2020)

Algorithm 1 Algorithm for stock trading in real time

Initialize a set of hyper-parameters;
for $d = 0$ to $D - 1$ **do**
    % **Step 1)**. Train an agent
    Using data period $[d - S - N, d - S - 1]$ to train an agent,
    Using data period $[d - S, d - 1]$ to test the trained agent and adjust hyper-parameters,
    % **Step 2)**. Retrain the agent
    Using data period $[d - S - N, d - 1]$ to retrain the agent with the adjusted hyper-parameters,
    % **Step 3)**. Perform paper trading for one day
    Using the trained agent to trade in the d-th day.
end for

For the experiment, we select Dow Jones 30 stocks as our trading stocks and use minute-level historical market data from 08/24/2022 to 09/09/2022. The data are downloaded from Alpaca.[10] Then, we use the paper trading APIs provided by Alpaca to do paper

[10] Web page of Alpaca: https://alpaca.markets/

trading from 09/01/2022 to 09/12/2022. The cumulative return of our PPO method and conventional random forest method is shown in Fig. 13. The complete analysis of different metrics is shown in Table 5.

5.2.3 Ensemble strategy

Based on the stock trading task, the ensemble method (Yang et al., 2020) obtains an adaptive agent through different ones, which inherits the best features of agents and performs remarkably well in practice. We consider three component algorithms, Proximal Policy Optimization (PPO), Advantage Actor-Critic (A2C), and Deep Deterministic Policy Gradient (DDPG) to train base agents, which have different strengths and weaknesses. Our ensemble strategy uses a rolling window, and automatically selects the agent with the best performance for each test period. It has three steps: 1) We use a time window of n months to train three agents (PPO, SAC, and DDPG) concurrently. 2) We validate three agents by using a quarterly validation rolling window, and select the agent with the maximum Sharpe ratio. 3) We use the selected agent to trade for the next quarter.

On the 30 constituent stocks of the DJIA index, we use data from 01/01/2010 to 07/01/2022 for training and data from 07/01/2022 to 01/01/2023 for validation and testing through a quarterly rolling window. From Fig. 14, we observe that the ensemble agent outperforms other agents. In the experiment, the return of the ensemble strategy is 0.157, while DJI is 0.068. The ensemble agent has the highest return, which means it performs the best in profits. This benchmark demonstrates that the ensemble strategy is effective in constructing a more reliable agent based on several components of DRL agents.

5.2.4 Podracer on cloud

We reproduce cloud solutions of population-based training, e.g., generational evolution (Li et al., 2021) and tournament-based evolution (Liu et al., 2021). FinRL-Podracer can easily scale out to $\geq$ 1000 GPUs, which features high scalability, elasticity and accessibility by following the cloud-native principle. If GPUs are abundant, users can take advantage of this benchmark to work on high-frequency trading tasks. On an NVIDIA SuperPOD cloud, we conducted extensive experiments on stock trading and found that it substantially outperforms competitors, such as OpenAI and RLlib (Li et al., 2021). Detailed instructions are provided on our website.

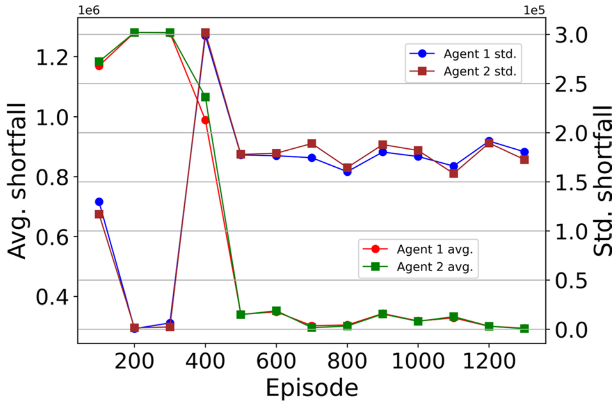
Benchmarks on cloud: We provide demos on a cloud platform, Weights & Biases,[11] to demonstrate the training process. We define the hyperparameter sweep, training function, and initialize an agent to train and tune hyperparameters. On the cloud platform Weights & Biases, users are able to visualize their results and assess the relative performance via community-wise competitions.

5.2.5 Curriculum learning for generalizable agents

Based on FinRL-Meta (a universe of market environments, say $\geq$ 100), one is able to construct an environment by sampling data from multiple market datasets, similar to XLand

[11] Website: https://wandb.ai/site

Fig. 15 Liquidation analysis of (Bao & Liu, 2019)



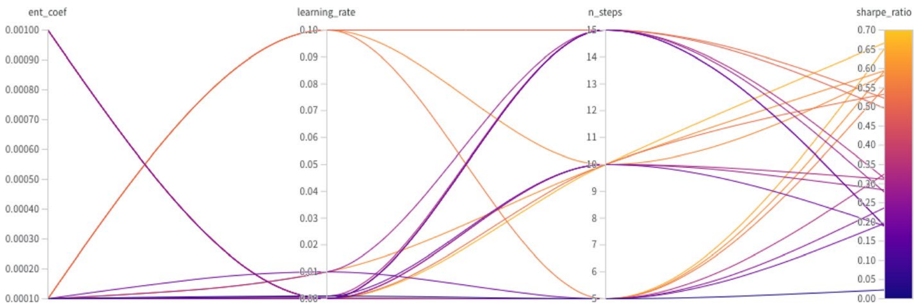


Fig. 16 An example of hyperparameter tuning

(Team et al., 2021). In this way, one can apply the curriculum learning method (Team et al., 2021) to train a generally capable agent for several financial tasks.

5.2.6 Market simulator

Gym market environments: We build all of our environments following OpenAI-gym style (Brockman et al., 2016). The first reason is that this makes it convenient for plugging in any of the three DRL libraries (Stable Baseline3, RLlib, and ElegantRL). Another reason is that it is user-friendly. Newcomers can learn our environments fast and then build their own task-specific environments efficiently.

Synthetic data generation: We create a market simulator (Han et al., 2023) to simulate the markets from the playback of historical limit-order-book-level data and the order matching mechanism. Currently, the simulator is at the minute level (i.e., one time step = one minute), which is changeable. The state is a stack of market indicators and market snapshots from the last few time steps. The action is to place an order. We support market orders and limit orders. We also provide several wrappers to accept typically discrete or continuous actions. Rewards can be configured by the participants with the aim of generating policies that optimize pre-specified indicators. In our simulator, we take into account the following factors: 1) temporary market impact; 2) order delay. We do not consider the following factors in our simulator: 1) permanent market impact of limit orders; 2) non-resiliency limit order book.

Liquidation analysis and trade execution: By reproducing (Bao & Liu, 2019), we build a simulated environment of stock prices according to the Almgren and Chriss model. Then we implement the multi-agent DRL algorithms for both competing and cooperative liquidation strategies. This benchmark demonstrates the trade execution task using deep reinforcement learning algorithms. When trading, traders want to minimize the expected trading cost, which is also called implementation shortfall. In Fig. 15, there are two agents, and we observe that the implementation shortfalls decrease during the training process.

5.3 More FinRL demos and tutorials

- **Portfolio allocation** (Liu et al., 2020): We train a DRL agent to perform a portfolio optimization task on a set of stocks.
- **Cryptocurrency trading** (Liu et al., 2020): We provide a demo (Liu et al., 2020) on 10 popular cryptocurrencies.
- **Paper trading demo**: We provide a demo for paper trading. Users could combine their own strategies or trained agents in paper trading.
- **China A-share demo**: We provide a demo based on the China A-share market data using Tushare.
- **Hyperparameter tuning**: The default hyperparamters may not be the best for our tasks. Reinforcement learning algorithms are sensitive to hyperparamters; therefore, hyperparamter tuning is an important issue. Hyperparamters are tuned based on an objective. We provide several demos for hyperparameter tuning using Optuna (Akiba et al., 2019) and Ray Tune (Liaw et al., 2018). Figure 16 shows an example of hyperparameter tuning in the portfolio allocation task using A2C, which aims to maximize the Sharpe ratio.
- **Robo-advising**: Robo-advising by integrating ChatGPT (Ouyang et al., 2022) and GPT-4 (OpenAI, 2023) will be much more powerful than before. Users are encouraged to develop robo-advisor apps by building upon the demos that we have created using ChatGPT (Ouyang et al., 2022) and GPT-4 (OpenAI, 2023). By engaging in conversation with ChatGPT using a chain of thought prompt, we may be able to obtain good trading advice. Additionally, we have demonstrated that properly guiding ChatGPT can generate new financial factors, which speeds up the process of creating them manually.

6 Conclusion, future works and research directions

Following the data-centric AI principles and the DataOps paradigm, in this paper, we developed FinRL-Meta, a data-centric library that provides openly accessible dynamic financial datasets and reproducible benchmarks. The novelty of FinRL-Meta is two-fold. Firstly, it is the first unified framework that seamlessly supports a wide array of environments. with plug-and-play RL agents to streamline agent evaluation. FinRL-Meta grants access to numerous previously inaccessible research resources in FinRL, thus paving the path for future innovations in FinRL. Secondly, FinRL-Meta has carefully constructed a data pipeline for processing financial data, encompassing data access, cleaning, feature engineering, and sentiment analysis. The DataOps development paradigm ensures that the data pipeline is easily extensible, allowing researchers to seamlessly integrate their own

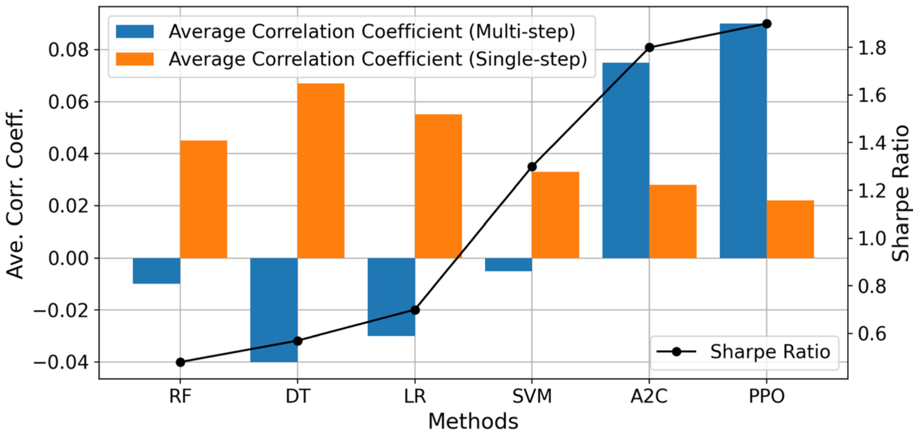


Fig. 17 Reproducing portfolio management of (Guan & Liu, 2021): Comparison of average correlation coefficient and Sharpe ratio among ML and DRL methods

designs. In addition, we have developed a wide range of homegrown examples and tutorials to facilitate the usage of the library. FinRL-Meta will serve as a source of inspiration for both researchers and practitioners who are interested in FinRL.

For future work and research directions, FinRL-Meta aims to build a universe of financial market environments, like the XLand environment (Team et al., 2021). To improve the performance for large-scale markets, we are exploiting GPU-based massive parallel simulation such as (Makoviychuk et al., 2021) and deploying it into projects such as RL for market simulators. Moreover, it will be interesting to explore the evolutionary perspectives (Gupta et al., 2021; Scholl et al., 2021; Li et al., 2021; Liu et al., 2021) to simulate the markets. We are also planning to benchmark the multi-processing performance throughout the developing process. Additionally, we will explore other strategies to perform sentiment analysis, such as ChatGPT (OpenAI, 2023) and FinBERT (Araci, 2019). Furthermore, we plan to extend FinRL-Meta to provide environments for other financial tasks, such as such as stress testing. We believe that FinRL-Meta will provide insights into complex market phenomena and offer guidance for financial regulations.

6.1 Explainability

We reproduce (Guan & Liu, 2021) that compares the performance of DRL algorithms with machine learning (ML) methods on the multi-step prediction in the portfolio allocation task. We use four technical indicators MACD, RSI, CCI, and ADX as features. Random Forest (RF), Decision Tree Regression (DT), Linear Regression (LR), and Support Vector Machine (SVM) are the ML algorithms in comparison. We use data from Dow Jones 30 constituent stocks to construct the environment. We use data from 04/01/2009 to 03/31/2020 as the training set and data from 04/01/2020 to 05/31/2022 for backtesting. In Fig. 17, the results show that DRL methods have a higher Sharpe ratio than ML methods. Also, DRL methods' average correlation coefficients are significantly higher than that of ML methods (multi-step).

We reproduce (Guan & Liu, 2021) that compares the performance of DRL algorithms with machine learning (ML) methods on the multi-step prediction in the portfolio allocation task. We use four technical indicators MACD, RSI, CCI, and ADX as features. Random Forest (RF), Decision Tree Regression (DT), Linear Regression (LR), and Support Vector Machine (SVM) are the ML algorithms in comparison. We use data from Dow Jones 30 constituent stocks to construct the environment. We use data from 04/01/2009 to 03/31/2020 as the training set and data from 04/01/2020 to 05/31/2022 for backtesting.

6.2 Data privacy, strategy privacy and federated learning technology

As our open-source community is continuously developing new features to ensure that FinRL-Meta provides a better user experience, one of the main targets for our next step is to enhance financial data privacy as well as strategy privacy for users. We discuss the potential of integrating the federated learning technology into our open-source FinRL-Meta, in order to achieve data privacy and strategy privacy for our users, say collaboratively training.

Federated learning is a method for training machine learning models from distributed datasets that remains private to data owners. It allows a central machine learning model to overcome the *isolated data island*, i.e., to learn from data sets distributed on multiple devices that do not reveal or share the data with a central server. We understand that in certain scenarios, our users face the problem that they only have a small amount of financial data and are not unable to train a robust model, but they are also hesitant to train their models with others due to privacy reasons. To the best of our knowledge, we would like to name a few representative examples on the use of federated learning to help financial applications. Byrd and Polychroniadou (2020) present a privacy-preserving federated learning protocol on a real-world credit card fraud dataset for the development of federated learning systems. The researchers in WeBank Liu et al. (2021) created FATE, an industrial-grade project that supports enterprises and institutions to build machine learning models collaboratively at large-scale in a distributed manner. FATE has been adopted in real-world applications in finance, health and recommender systems. We also want to mention one research work (Kairouz et al., 2021) which points out the open problems in federated learning. We believe that, as a newly introduced method, federated learning has a lot of undiscovered and exciting applications that we can develop. We would like to encourage our community members to explore the potential of federated learning technologies for the benefit of financial applications.

6.3 Accessibility, accountability, maintenance and rights

FinRL-Meta is an open-source project, held on GitHub. We use the MIT License for research and educational usage. while users can utilize them as stepping stones for customized trading strategies. Codes, market environments, benchmarks and documentations are available on the GitHub repository:

https://github.com/AI4Finance-Foundation/FinRL-Meta.

FinRL-Meta has been actively maintained by the AI4Finance community which has over 12K members at the moment. On GitHub, we keep updating our codes, merging pull requests, and fixing bugs and issues. We welcome contributions from community members, researchers and quant traders.

We have accumulated six competitive advantages over the past five years. The first three are technological innovations:

Table 6 List of key terms for reinforcement learning

Key terms	Description
Agent (Sutton, 2022)	A decision maker
Environment (Sutton, 2022)	A world with which an agent interacts with
Gym-style environment (Brockman et al., 2016)	A standard form of DRL environment by OpenAI
Markov Decision Process (MDP)	A mathematical framework to model decision-making problems
State, Action, Reward	Three main factors in an agent-environment interaction
Policy	A rule that agent follow to make decision
Policy gradient	An approach to solve RL problems by optimizing the policy directly
Deep Q-Learning (DQN) (Mnih et al., 2015)	The first DRL algorithm that uses a neural network to approximate the Q-function
DDPG (Lillicrap et al., 2016)	Deep Deterministic Policy Gradient algorithm
PPO (Schulman et al., 2017)	Proximal Policy Optimization algorithm
Hyperparameter tuning	Change hyperparameter during training to get a converged result faster
Ensemble strategy	An ML technique. Here we combine several DRL agents to a better model
Population-based training (PBT)	Optimise a population of models and hyperparameters, and select the optimal set
Generational evolution (Li et al., 2021)	Employing an evolution strategy over generations
Tournament-based evolution (Liu et al., 2021)	An evolution by asynchronously updating a tournament board of models
Curriculum learning	Training an ML model from easier to harder data, imitating the human curriculum
Simulation-to-reality gap	The difference between simulation environment and real-world task

- FinRL (Liu et al., 2020, 2021) is the first framework to provide an automatic pipeline for financial reinforcement learning.
- For financial big data, the FinRL-Meta project connects with >30 market data sources.
- For cloud solutions, the FinRL-Podracer project (Li et al., 2021) (Liu et al., 2021) scales out to ≥ 1000 GPUs. We have extensive testings on NVIDIA's DGX-2 SuperPod platform.

Based on the above projects and active contributions, an open-source community in the intersection of ML and Finance fields is emerging. Our AI4Finance community is robust with the following three features:

- We have over $12K$ active community members, many of which are actively designing strategies and connecting with paper trading, even live trading. We are collaborating with tens of universities and research institutes, and ≥ 50 software engineers from IT companies.
- Both Columbia University (Department of Electrical Engineering, Department of Statistics) and New York University ((Department of Finance) have opened delicate courses about FinRL, while ≥ 120 students in total have taken it.

Table 7 List of key terms for finance

Key terms	Description
Algorithmic trading	A method of trading using designed algorithm instead of human traders
Backtesting	A method to see how a strategy performs on a certain period of historical data
Signal-to-noise ratio (SNR)	A ratio of desired signal (good data) to undesired signal (noise)
DataOps	A series of principles and practices to improve the quality of data science
Sentiment data	A category in financial big data that contains subjective viewpoints
Historical data	All kinds of data that already existed in the past
Survivorship bias	A bias caused by only seeing existing examples, but not those already died out
Information leakage	When the data contains future information, causing model overfitting
Paper trading	Simulation of buying and selling without using real money
OHLCV	A popular form of market data with: Open, High, Low, Close, Volume
Technical indicators	A statistical calculation based on OHLCV data to indicate future price trends
Market frictions	A financial market friction as anything that interferes with trade
Market crash	A huge drop of market price within a very short time
Volatility index (VIX) (Whaley, 2009)	A market index that shows the market's expectations for volatility
Limit Order Book (LOB)	A list to record the interest of buyers and sellers
Smart beta index	An enhanced indexing strategy to beat a benchmark index
Liquidation, trade execution	An investor closes their position in an asset

- In academia, we have several accepted papers and also delivered several invited talks. Our AI4Finance Foundation (https://github.com/AI4Finance-Foundation) serves as a bridge between machine learning, data science, operation research, and finance communities.

Maintaining these data connectors is a challenge due to the dynamic nature of the financial dataset. Our maintenance plan encompasses two critical aspects: Firstly, we will adhere to the DataOps standard development process, ensuring proper management during iterations on both the data and codebase to uphold data quality. Secondly, given the dynamic nature of the data, we will diligently monitor it and conduct anomaly detection using unit and integration tests. Additionally, we will perform data calibration as needed.

We acknowledge the substantial effort required for maintenance. Therefore, we have open-sourced our codebase and the entire pipeline on GitHub. We hope to encourage and invite the community to actively engage in this initiative, pushing the boundaries in this direction. See Section B.7 for some detailed discussions of maintenance.

Appendix 1: Terminology

We provide a list of key terms for reinforcement learning and finance in Table 6 and Table 7. For terminologies of reinforcement learning, interested users can refer to Sutton (2022) or the classic textbook (Sutton & Barto, 2018). Also, the webpage[12] explains key concepts of RL. For terminologies of finance, interested users can refer to De Prado (2018).

Appendix 2: Dataset documentation and usages

We organize the dataset documentation according to the suggested template of *datasheets for datasets*[13].

2.1: Motivation

- **For what purpose was the dataset created?** As data is refreshing minute-to-millisecond, finance is a particularly difficult playground for deep reinforcement learning. In academia, scholars use financial big data to obtain more complex and precise understanding of markets and economics. While industries use financial big data to refine their analytical strategies and strengthen their prediction models. To serve the rapidly growing AI4Finance community, we create FinRL-Meta that provides data access from different sources, pre-processes the raw data with different features, and builds the data to RL environments. We aim to provide dynamic RL environments that are manageable by users. We aim to build a financial metaverse, a universe of near real-market environments, as a playground for data-driven financial machine learning.
- **Who created the dataset?** FinRL-Meta is an open-source project created by the AI4Finance community. Contents of FinRL-Meta are contributed by the authors of this paper and will be maintained by members of the AI4Finance community.
- **Who funded the creation of the dataset?** AI4Finance Foundation, a non-profit open-source community that shares AI tools for finance, funded our project.

2.2: Composition

- **What do the instances that comprise the dataset represent?** Instances of FinRL-Meta are volume-price data includes: stocks, securities, cryptocurrencies, etc; and sentiment data from social media, ESG, Google Trends, etc. FinRL-Meta provides hundreds of market environments through an automatic pipeline that collects dynamic datasets from real-world markets and processes them into standard gym-style market environments. FinRL-Meta also benchmarks popular papers as stepping stones for users to design new trading strategies.
- **How many instances are there in total?** FinRL-Meta does not store data directly. Instead, we provide codes for a pipeline of data accessing, data cleaning, feature engineering, and building into RL environments. Table 2 provides the supported data

12 OpenAI SpinningUp: https://spinningup.openai.com/en/latest/spinningup/rl_intro.html

13 Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wal- lach, Hal Daumé Iii, and Kate Crawford. Datasheets for datasets. *Communications of the ACM*, 64(12):86-92, 2021.

sources of FinRL-Meta. At the moment, there are hundreds of market environments, dozens of tutorials and demos, and several benchmarks provided.

- **Does the dataset contain all possible instances or is it a sample of instances from a larger set?** With our provided codes, users could fetch data from the data source by properly specifying the starting date, ending date, time granularity, asset set, attributes, etc.
- **What data does each instance consist of?** Now there are several types of financial data, as shown in Table 2:
- **Is there a label or target associated with each instance?** No. There is not label or preset target for each instance. But users can use our benchmarks are baselines.
- **Is any information missing from individual instances?** Yes. In several data sources, there are missing values and we provided standard preprocessing methods.
- **Are relationships between individual instances made explicit?** Yes. An instance is a sample set of the market of interest.
- **Are there recommended data splits?** We recommend users to follow our training–testing-training pipeline, as shown in Fig. 4. Users can flexibly choose their preferred settings, e.g., in stock trading task, our demo access Yahoo! Finance database and use data from 01/01/2009 to 06/30/2020 for training and data from 07/01/2020 to 05/31/2022 for backtesting.
- **Are there any errors, sources of noise, or redundancies in the dataset?** For the raw data fetched from different sources, there are noise and outliers. We provide codes to process the data and built them into standard RL gym environment.
- **Is the dataset self-contained, or does it link to or otherwise rely on external resources?** It is linked to external resources. As shown in Table 2, FinRL-Meta fetch data from data sources to build gym environments.
- **Does the dataset contain data that might be considered confidential?** No. All our data are from publicly available data sources.
- **Does the dataset contain data that, if viewed directly, might be offensive, insulting, threatening, or might otherwise cause anxiety?** No. All our data are numerical.

2.3: Collection process

- **How was the data associated with each instance acquired?** FinRL-Meta fetches data from data sources. as shown in Table 2.
- **What mechanisms or procedures were used to collect the data?** FinRL-Meta provides dynamic market environments that are built according to users' settings. To achieve this, we provide software APIs to fetch data from different data sources. Note that some data sources require accounts and passwords or have limitations on the number or frequency of requests.
- **If the dataset is a sample from a larger set, what was the sampling strategy?** It is dynamic, depending on users' settings, such as the starting date, ending date, time granularity, asset set, attributes, etc.
- **Who was involved in the data collection process and how were they compensated?** Our codes collect publicly available market data, which is free.
- **Over what timeframe was the data collected?** It is not applicable because the environments are created dynamically by running the codes to fetch data in real-time.
- **Were any ethical review processes conducted?** No?

2.4: Preprocessing/cleaning/labeling

- **Was any preprocessing/cleaning/labeling of the data done?** Yes. For the raw data fetched from different sources, there are noise and outliers. We provide codes to process the data and built them into standard RL gym environment.
- **Was the "raw" data saved in addition to the preprocessed/cleaned/labeled data** The raw data are hold by different data sources (data providers).
- **Is the software that was used to preprocess/clean/label the data available?** Yes. We use our own codes to do cleaning and preprocessing.

2.5: Uses

- **Has the dataset been used for any tasks already?** Yes. Thousands of AI4Finance community members use FinRL-Meta for learning and research purpose. There are also courses in colleges using FinRL-Meta as material for teaching financial reinforcement learning. Demos and tutorials are mentioned in Sect. 5.
- **Is there a repository that links to any or all papers or systems that use the dataset?** 1. Research papers that used FinRL-Meta are listed here: https://github.com/AI4Finance-Foundation/FinRL-Tutorials/blob/master/FinRL_papers.md Our conference version of FinRL-Meta (Liu et al., 2022) appeared in NeurIPS 2022 Datasets and Benchmarks Track. Our workshop version of FinRL-Meta (Liu et al., 2021) appeared in NeurIPS 2021 Workshop on Data-Centric AI. 2. The following three repositories have incorporated FinRL-Meta:
  - FinRL-Meta corresponding to the market layer of FinRL (7K stars): https://github.com/AI4Finance-Foundation/FinRL
  - ElegantRL (2.7K stars) supports FinRL-Meta: https://github.com/AI4Finance-Foundation/ElegantRL
  - FinRL-Podracer: https://github.com/AI4Finance-Foundation/FinRL_Podracer
- **What (other) tasks could the dataset be used for?** Besides the current tasks (tutorial, demo and benchmarks), FinRL-Meta will be useful for the following tasks:
  - **Curriculum learning for agents**: Based on FinRL-Meta (a universe of market environments, say ≥ 100), one is able to construct an environment by sampling data samples from multiple market datasets, similar to XLand (Team et al., 2021). In this way, one can apply the curriculum learning method (Team et al., 2021) to train a generally capable agent for several financial tasks.
  - To improve the performance for the large-scale markets, we are exploiting GPU-based massive parallel simulation such as Isaac Gym (Makoviychuk et al., 2021).
  - It will be interesting to explore the evolutionary perspectives (Gupta et al., 2021; Scholl et al., 2021; Li et al., 2021; Liu et al., 2021) to simulate the markets. We believe that FinRL-Meta will provide insights into complex market phenomena and offer guidance for financial regulations.
- **Is there anything about the composition of the dataset or the way it was collected and preprocessed/cleaned/labeled that might impact future uses?** We believe that FinRL-Meta will not encounter usage limits. Our data are fetched from different sources in real-time when running the codes. However, there may be one or two out of

≥ 30 data sources (in Table 2) change data access rules that may impact future use. So please refer to the rules and accessibility of certain data sources when using.

- **Are there tasks for which the dataset should not be used?** No. Since there are no ethical problems for FinRL-Meta, users could use FinRL-Meta in any task as long as it does not violate laws. **Disclaimer: Nothing herein is financial advice, and NOT a recommendation to trade real money. Please use common sense and always first consult a professional before trading or investing.**

2.6: Distribution

- **Will the dataset be distributed to third parties outside of the entity (e.g., company, institution, organization) on behalf of which the dataset was created?** No. It will always be held on GitHub under MIT license, for educational and research purposes.
- **How will the dataset be distributed?** Our codes and existing environments are available on GitHub FinRL-Meta repository https://github.com/AI4Finance-Foundation/FinRL-Meta.
- **When will the dataset be distributed?** FinRL-Meta is publicly available since February 14th, 2021.
- **Will the dataset be distributed under a copyright or other intellectual property (IP) license, and/or under applicable terms of use (ToU)?** FinRL-Meta is distributed under MIT License, for educational and research purposes.
- **Have any third parties imposed IP-based or other restrictions on the data associated with the instances?** No.
- **Do any export controls or other regulatory restrictions apply to the dataset or to individual instances?** No. Our data are fetched from different sources in real time. However, there may be one or two out of ≥ 20 data sources (in Table 2) change data access rules that may impact future use. So please refer to the rules and accessibility of certain data sources when using.

2.7: Maintenance

- **Who will be supporting/hosting/maintaining the dataset?** FinRL-Meta has been actively maintained by AI4Finance Foundation (including the authors of this paper) which has over 12K members at the moment (Mar. 2023). We are actively updating market environments, to serve the rapidly growing open-source AI4Finance community.
- **How can the owner/curator/manager of the dataset be contacted?** We encourage users to join our Slack channel: https://join.slack.com/t/ai4financeworkspace/shared_invite/zt-v670l1jm-dzTgIT9fHZIjjrqprrY0kg or our mailing list: https://groups.google.com/u/1/g/ai4finance,
- **Is there an erratum?** Users can use GitHub to report issues/bugs and use Slack channel, Discord channel, or mailing list (AI4Finance_FinRL at https://groups.google.com/u/2/g/ai4finance) to discuss solutions. AI4Finance community is actively improving the codes, say extracting technical indicators, evaluating feature importance, quantifying the probability of model overfitting, etc.
- **Will the dataset be updated?** Yes, we are actively updating codes and adding more data sources. Users could get information and the newly updated version through our GitHub repository, or join the mailing list: https://groups.google.com/u/1/g/ai4finance.

- **If the dataset relates to people, are there applicable limits on the retention of the data associated with the instances** The data of FinRL-Meta do not relate to people.
- **Will older versions of the dataset continue to be supported/hosted/maintained?** Yes. All versions can be found on our GitHub repository.
- **If others want to extend/augment/build on/contribute to the dataset, is there a mechanism for them to do so?** We maintain FinRL-Meta on GitHub. Users can use GitHub to report issues/bugs and use Slack channel or mailing list to discuss solutions. We welcome community members to submit pull requests through GitHub.
- **How does the platform handle ticker name changes due to corporate actions?** To our knowledge, changing of ticker names is very rare. Therefore, periodically conducting manual checks and adjustments to the data API may be satisfactory. Nevertheless, there exists the potential for implementing automated processes for handling such changes, such as utilizing web crawling techniques to retrieve updated ticker names from sources like https://stockanalysis.com/stocks/. We plan to explore this possibility in future investigations.
- **What about missing data?** We can incorporate a backup data source as a precautionary measure. Oftentimes, data for specific variables, such as stock prices, are available from multiple sources. In the unfortunate event of missing data from the primary source, we can resort to the backup source. We will investigate other possible solutions in our future work.
- **What if data server is down?** Given that we have open-sourced our codebase, users have the option to directly retrieve the data using their own server or PC. Additionally, we plan to progressively introduce supplementary data servers to facilitate data downloading.

Appendix 3: MDP setup for market environments

3.1 Order execution

The order execution task has the following MDP:

- **State** $\mathbf{s_t} = [\mathbf{h_t}, (\mathbf{p_t}, \mathbf{o_t})] \in \mathbb{R}^{1+2*9}$, where $\mathbf{h_t} \in \mathbb{R}_+$ denotes the remaining holds that haven't been executed as orders, $(\mathbf{p_t}, \mathbf{o_t}) \in \mathbb{R}^2 * 9_+$ denotes the current limit order book at time t.
- **Action** $\mathbf{a_t} = [\mathbf{ap_t}, \mathbf{ah_t}] \in \mathbb{R}^2_+$, which denotes the agent would place $\mathbf{ah_t}$ number of shares to the market order with price $\mathbf{ap_t}$.
- **Reward** $r(\mathbf{s_t}, \mathbf{a_t}, \mathbf{s_{t+1}}) \in \mathbb{R}$. In this order execution task, the reward function is set to be the excess return of the agent comparing to the Time-weighted average price (TWAP).

3.2 Paper trading

Paper trading task is the variance of stock trading that trading in real time. It has a similar MDP setup:

- **State** $\boldsymbol{s_t} = [b_t, \boldsymbol{p_t}, \boldsymbol{f_t}, \boldsymbol{h_t}] \in \mathbb{R}^{30(I+2)+1}$, where scalar $b_t \in \mathbb{R}_+$ is the remaining balance in the account, $\boldsymbol{p_t} \in \mathbb{R}^{30}_+$ is the prices of 30 stocks, $\boldsymbol{f_t} \in \mathbb{R}^{30 \cdot I}$ is a feature vector and each stock has I technical indicators, and $\boldsymbol{h_t} \in \mathbb{R}_+^{30}$ denotes the share holdings, where $\mathbb{R}_+$ is the set of non-negative real numbers.

- **Action** $\boldsymbol{a_t} \in \mathbb{R}^{30}$ denotes the trading operations on the 30 stocks, i.e., $\boldsymbol{h_{t+1}} = \boldsymbol{h_t} + \boldsymbol{a_t}$. When an entry $\boldsymbol{a}_t^i > 0, i = 1, ..., 30$, it means a buy-in of $\boldsymbol{a}_t^i$ shares on the i-th stock, negative action $\boldsymbol{a}_t^i < 0$ for selling, and zero action $\boldsymbol{a}_t^i = 0$ keeps $\boldsymbol{h}_t^i$ unchanged.
- **Reward function** $R(\boldsymbol{s_t}, \boldsymbol{a_t}, \boldsymbol{s_{t+1}}) \in \mathbb{R}$: In this paper trading task, the reward function is set to be the change of total asset values, i.e., $R(\boldsymbol{s_t}, \boldsymbol{a_t}, \boldsymbol{s_{t+1}}) = v_{t+1} - v_t$, where v_t and v_{t+1} are the total asset values at state $\boldsymbol{s_t}$ and $\boldsymbol{s_{t+1}}$, respectively, i.e., $v_t = \boldsymbol{p_t}^\intercal \boldsymbol{h_t} + b_t \in \mathbb{R}$.

Acknowledgements We thank Jingyang Rui for participating early design and development of FinRL-Meta and his contribution of our previous conference version in NeurIPS 22. He didn't participate in this version because of employment constraint. We thank Mr. Tao Liu (IDEA Research, International Digital Economy Academy) for technical support of computing platform on this research project. Ming Zhu was supported by National Natural Science Foundations of China (Grant No. 61902387). Christina Dan Wang is Assistant Professor, Shanghai Frontiers Science Center of Artificial Intelligence and Deep Learning, NYU Shanghai; Business Division, NYU Shanghai, Shanghai China 200122; and Supported in part by National Natural ScienceFoundation of China (NNSFC) grant 12271363. Disclaimer: Nothing in this paper and the FinRL-Meta repository is financial advice, and NOT a recommendation to trade real money. Please use common sense and always first consult a professional before trading or investing.

Author contributions X-Y: Designs and leads the development of the whole FinRL-Meta framework. ZX: Contributes to the development of FinRL-Meta, MDP modeling, stock trading, trading in real time, and tutorials. HY: Contributes to related work, stock trading, ensemble strategy, Financial Sentiment Analysis. Jiechao Gao: Contributes to FinRL-Meta's mathematical modeling, revises whole paper. DZ: Contributes to the data curation pipeline, data centric idea. MZ: Contributes to maintaining the open-source repo of FinRL-Meta on GitHub, and proofreading. CDW: Supervises applications in FinRL-Meta and financial sentiment analysis. ZW: Supervises reinforcement learning, MDP modeling, and DRL algorithms. JG: Provides computing resources, and helps proofreading.

Funding None.

Availability of data and materials No data is directly held by us. We release data processing codes in https://github.com/AI4Finance-Foundation/FinRL-Meta

Code availability FinRL-Meta's code is open-sourced at: https://github.com/AI4Finance-Foundation/FinRL-Meta with MIT License.

Declarations

Conflict of interest The authors are from universities and research labs. No competing interests.

Ethics approval We do not contain ethics issues.

Consent to participate FinRL-Meta uses MIT License. We, all authors, welcome any person participate in our project and join our open-source community.

Consent for publication We, all authors, consent publication of everything mentioned in the paper.

References

Akiba, T., Sano, S., Yanase, T., Ohta, T., & Koyama, M. (2019). Optuna: A next-generation hyperparameter optimization framework. In *ACM SIGKDD international conference on knowledge discovery & data mining*.

Alla, S., & Adari, S. K. (2021). What is MLOps? In: Beginning MLOps with MLFlow (pp. 79–124).
Amrouni, S., Moulin, A., Vann, J., Vyetrenko, S., Balch, T., & Veloso, M. (2021). ABIDES-Gym: Gym environments for multi-agent discrete event simulation and application to financial markets. In *ACM International conference on AI in finance (ICAIF)*.
Ang, A. (2012). Mean-variance investing. Columbia Business School Research Paper No. 12/49.
Araci, D. (2019). Finbert: Financial sentiment analysis with pre-trained language models. arXiv preprint arXiv:1908.10063.
Ardon, L., Vadori, N., Spooner, T., Xu, M., Vann, J., & Ganesh, S. (2021). Towards a fully RL-based market simulator. In *ACM international conference on AI in finance (ICAIF)*.
Atwal, H. (2019). Practical DataOps: Delivering agile data science at scale.
Bao, W., & Liu, X.-Y. (2019). Multi-agent deep reinforcement learning for liquidation strategy analysis. In *ICML workshop on applications and infrastructure for multi-agent learning*.
Breiman, L. (2001). Random forests. *Machine Learning, 45*(1), 5–32.
Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., & Zaremba, W. (2016). OpenAI Gym. arXiv preprint arXiv:1606.01540.
Brown, S. J., Goetzmann, W., Ibbotson, R. G., & Ross, S. A. (1992). Survivorship bias in performance studies. *The Review of Financial Studies, 5*(4), 553–580.
Byrd, D., & Polychroniadou, A. (2020). Differentially private secure multi-party computation for federated learning in financial applications. In *Proceedings of the first ACM international conference on AI in finance* (pp. 1–9).
Chen, Q., & Liu, X.-Y. (2020) Quantifying ESG alpha using scholar big data: An automated machine learning approach. In *Proceedings of the first ACM international conference on AI in finance* (pp. 1–8).
Chen, C.-C., Huang, H.-H., & Chen, H.-H. (2018). Ntusd-fin: A market sentiment dictionary for financial social media data applications. In *Proceedings of the 1st financial narrative processing workshop (FNP 2018)* (pp. 37–43).
Christiano, P. F., Leike, J., Brown, T., Martic, M., Legg, S., & Amodei, D. (2017). Deep reinforcement learning from human preferences. *Advances in Neural Information Processing Systems30*.
Coletta, A., Prata, M., Conti, M., Mercanti, E., Bartolini, N., Moulin, A., Vyetrenko, S., & Balch, T. (2021). Towards realistic market simulations: A generative adversarial networks approach. In *ACM international conference on AI in finance (ICAIF)*.
De Prado, M. L. (2018). Advances in financial machine learning.
Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Fei-Fei, L. (2009). ImageNet: A large-scale hierarchical image database. In *IEEE conference on computer vision and pattern recognition* (pp. 248–255).
Dulac-Arnold, G., Mankowitz, D., & Hester, T. (2019). Challenges of real-world reinforcement learning. In *ICML workshop on reinforcement learning for real life*.
Dulac-Arnold, G., Levine, N., Mankowitz, D. J., Li, J., Paduraru, C., Gowal, S., & Hester, T. (2021). Challenges of real-world reinforcement learning: Definitions, benchmarks and analysis. *Machine Learning, 110*(9), 2419–2468.
Ereth, J. (2018). DataOps: Towards a definition. *LWDA, 2191*, 104–112.
Fang, Y., Liu, X.-Y., & Yang, H. (2019). Practical machine learning approach to capture the scholar data driven Alpha in AI industry. In *IEEE international conference on big data (big data)* (pp. 2230–2239). IEEE.
Fu, J., Kumar, A., Nachum, O., Tucker, G., & Levine, S. (2020). D4RL: Datasets for deep data-driven reinforcement learning. arXiv preprint arXiv:2004.07219.
Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. *Advances in Neural Information Processing Systems27*.
Gort, B., Liu, X.-Y., Sun, X., Gao, J., Chen, S., & Wang, C. D. (2023). *Deep reinforcement learning for cryptocurrency trading: Practical approach to address backtest overfitting*. AAAI: AI in Finance Bridge.
Guan, M., & Liu, X.-Y. (2021). Explainable deep reinforcement learning for portfolio management: An empirical approach. In *ACM international conference on AI in finance (ICAIF)*.
Gupta, A., Savarese, S., Ganguli, S., & Fei-Fei, L. (2021). Embodied intelligence via learning and evolution. *Nature Communications*.
Hambly, B., Xu, R., & Yang, H. (2023). Recent advances in reinforcement learning in finance. *Mathematical Finance*.
Hamilton, W. L., Clark, K., Leskovec, J., & Jurafsky, D. (2016). Inducing domain-specific sentiment lexicons from unlabeled corpora. In *Proceedings of the conference on empirical methods in natural language processing. conference on empirical methods in natural language processing*, vol. 2016 (p. 595). NIH Public Access.

Han, J., Xia, Z., Liu, X.-Y., Zhang, C., Wang, Z., & Guo, J. (2023). Massively parallel market simulator for financial reinforcement learning. *AI in Finance Bridge, AAAI*.

Hein, D., Depeweg, S., Tokic, M., Udluft, S., Hentschel, A., Runkler, T.A., & Sterzing, V. (2017). A benchmark environment motivated by industrial control problems. In *IEEE symposium series on computational intelligence (SSCI)* (pp. 1–8). IEEE.

Hutto, C., & Gilbert, E. (2014). Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the international AAAI conference on web and social media*, vol. 8 (pp. 216–225).

Kairouz, P., McMahan, H. B., Avent, B., Bellet, A., Bennis, M., Bhagoji, A. N., Bonawitz, K., Charles, Z., Cormode, G., & Cummings R. (2021). Advances and open problems in federated learning. Foundations and trends® in machine learning 14(1–2), 1–210.

Kritzman, M., & Li, Y. (2010). Skulls, financial turbulence, and risk management. *Financial Analysts Journal, 66*(5), 30–41.

Levine, S., Kumar, A., Tucker, G., & Fu, J. (2020). Offline reinforcement learning: Tutorial, review, and perspectives on open problems. arXiv preprint arXiv:2005.01643.

Li, X., Li, Y., Yang, H., Yang, L., & Liu, X.-Y. (2019). DP-LSTM: Differential privacy-inspired LSTM for stock prediction using financial news. In *33rd conference on neural information processing systems workshop on robust AI in financial services: Data, fairness, explainability, trustworthiness, and privacy, December 2019*.

Li, Z., Liu, X.-Y., Zheng, J., Wang, Z., Walid, A., & Guo, J. (2021). FinRL-Podracer: High-performance and scalable deep reinforcement learning for quantitative finance. In *ACM international conference on AI in finance (ICAIF)*.

Liang, E., Liaw, R., Nishihara, R., Moritz, P., Fox, R., Goldberg, K., Gonzalez, J., Jordan, M., & Stoica, I. (2018). RLlib: Abstractions for distributed reinforcement learning. In *International conference on machine learning* (pp. 3053–3062). PMLR.

Liaw, R., Liang, E., Nishihara, R., Moritz, P., Gonzalez, J. E., & Stoica, I. (2018). Tune: A research platform for distributed model selection and training. In *ICML AutoML workshop*.

Lillicrap, T., Hunt, J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., & Wierstra, D. (2016). Continuous control with deep reinforcement learning. In *International conference on learning representations (ICLR)*.

Liu, X.-Y., Li, Z., Wang, Z., & Zheng, J. (2021). ElegantRL: A lightweight and stable deep reinforcement learning library. GitHub.

Liu, X.-Y., Li, Z., Yang, Z., Zheng, J., Wang, Z., Walid, A., Guo, J., & Jordan, M. (2021). ElegantRL-Podracer: Scalable and elastic library for cloud-native deep reinforcement learning. In *Deep reinforcement learning workshop at NeurIPS*.

Liu, Y., Liu, Q., Zhao, H., Pan, Z., & Liu, C. (2020). Adaptive quantitative trading: An imitative deep reinforcement learning approach. In *Proceedings of the AAAI conference on artificial intelligence*, vol. 34 (pp. 2128–2135).

Liu, X.-Y., Rui, J., Gao, J., Yang, L., Yang, H., Wang, Z., Wang, C. D., & Jian, G. (2021). *FinRL-Meta: Data-driven deep reinforcementlearning in quantitative finance*. NeurIPS: Data-Centric AI Workshop.

Liu, X.-Y., Xia, Z., Rui, J., Gao, J., Yang, H., Zhu, M., Wang, C. D., Wang, Z., & Guo, J. FinRL-Meta: Market environments and benchmarks for data-driven financial reinforcement learning. In *Thirty-sixth conference on neural information processing systems*.

Liu, X.-Y., Xiong, Z., Zhong, S., Yang, H., & Walid, A. (2018). *Practical deep reinforcement learning approach for stock trading*. NeurIPS: Workshop on Challenges and Opportunities for AI in Financial Services.

Liu, X.-Y., Yang, H., Chen, Q., Zhang, R., Yang, L., Xiao, B., & Wang, C. D. (2020). *FinRL: A deep reinforcement learning library for automated stock trading in quantitative finance*. NeurIPS: Deep RL Workshop.

Liu, X.-Y., Yang, H., Gao, J., & Wang, C. D. (2021). FinRL: Deep reinforcement learning framework to automate trading in quantitative finance. In *ACM international conference on AI in finance (ICAIF)*

Liu, Y., Fan, T., Chen, T., Xu, Q., & Yang, Q. (2021). Fate: An industrial grade platform for collaborative learning with data protection. *Journal of Machine Learning Research, 22*(226), 1–6.

Loria, S. (2018). textblob documentation. Release 0.15 *2*(8).

Loughran, T., & McDonald, B. (2011). When is a liability not a liability? Textual analysis, dictionaries, and 10-ks. *The Journal of Finance, 66*(1), 35–65.

Lussange, J., Lazarevich, I., Bourgeois-Gironde, S., Palminteri, S., & Gutkin, B. (2021). Modelling stock markets by multi-agent reinforcement learning. *Computational Economics, 57*(1), 113–147.

Mahfouz, M., Gopalakrishnan, S., Suau, M., Patra, S., Mandic, P. D., Magazzeni, D., & Veloso, M. (2023). Towards asset allocation using behavioural cloning and reinforcement learning. *AAAI AI for Financial Services Bridge*.

Makoviychuk, V., Wawrzyniak, L., Guo, Y., Lu, M., Storey, K., Macklin, M., Hoeller, D., Rudin, N., Allshire, A., Handa, A., & State, G. (2021). *Isaac Gym: High performance GPU-based physics simulation for robot learning*. NeurIPS: Datasets and Benchmarks Track.

Malkiel, B. G. (2003). Passive investment strategies and efficient markets. *European Financial Management, 9*(1), 1–10.

Mamon, R. S., & Elliott, R. J. (2007). Hidden Markov models in finance vol. 4.

Mazumder, M., Banbury, C., Yao, X., Karlaš, B., Rojas, W. G., Diamos, S., Diamos, G., He, L., Kiela, D., & Jurado, D. et al. (2022). Dataperf: Benchmarks for data-centric AI development. arXiv preprint arXiv:2207.10062.

Miller, G. A. (1998). WordNet: An electronic lexical database.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A., Veness, J., Bellemare, M., Graves, A., Riedmiller, M., Fidjeland, A., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature, 518*, 529–33.

Nargesian, F., Samulowitz, H., Khurana, U., Khalil, E. B., & Turaga, D. S. (2017). Learning feature engineering for classification. In *IJCAI*, vol. 17 (pp. 2529–2535).

Nuti, G., Mirghaemi, M., Treleaven, P., & Yingsaeree, C. (2011). Algorithmic trading. *Computer, 44*, 61–69.

OpenAI: GPT-4 technical report. https://arxiv.org/abs/2303.08774 (2023).

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., et al. (2022). Training language models to follow instructions with human feedback. *Advances in Neural Information Processing Systems, 35*, 27730–27744.

Polyzotis, N., & Zaharia, M. (2021). What can data-centric AI learn from data and ML engineering? arXiv preprint arXiv:2112.06439.

Pricope, T.-V. (2021). Deep reinforcement learning in quantitative algorithmic trading: A review. arXiv preprint arXiv:2106.00123.

Qin, R., Gao, S., Zhang, X., Xu, Z., Huang, S., Li, Z., Zhang, W., & Yu, Y. (2022). NeoRL: A near real-world benchmark for offline reinforcement learning. *NeurIPS Datasets and Benchmarks*.

Raberto, M., Cincotti, S., Focardi, S. M., & Marchesi, M. (2001). Agent-based simulation of a financial market. *Physica A: Statistical Mechanics and its Applications, 299*(1–2), 319–327.

Raffin, A., Hill, A., Gleave, A., Kanervisto, A., Ernestus, M., & Dormann, N. (2021). Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*.

Rundo, F. (2019). Deep LSTM with reinforcement learning layer for financial trend prediction in fx high frequency trading systems. *Applied Sciences, 9*(20), 4460.

Sambasivan, N., Kapania, S., Highfill, H., Akrong, D., Paritosh, P., & Aroyo, L. M. (2021). "Everyone wants to do the model work, not the data work": Data cascades in high-stakes AI. In *Proceedings of the 2021 CHI conference on human factors in computing systems* (pp. 1–15).

Scholl, M. P., Calinescu, A., & Farmer, J. D. (2021). How market ecology explains market malfunction. *Proceedings of the National Academy of Sciences118*(26)

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal policy optimization algorithms. arXiv:1707.06347.

Sharpe, W. F. (1994). The sharpe ratio. *Journal of Portfolio Management*.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., & Lanctot, M., et al. (2016). Mastering the game of go with deep neural networks and tree search. *Nature529*(7587), 484–489.

Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., Guez, A., Hubert, T., Baker, L., Lai, M., Bolton, A., et al. (2017). Mastering the game of Go without human knowledge. *Nature, 550*(7676), 354–359.

Strapparava, C., & Mihalcea, R. (2007). Semeval-2007 task 14: Affective text. In *Proceedings of the fourth international workshop on semantic evaluations (SemEval-2007)* (pp. 70–74).

Sutton, R. S. (2022). The quest for a common model of the intelligent decision maker. arXiv preprint arXiv:2202.13252.

Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction.

Tai, Y.-J., & Kao, H.-Y. (2013). Automatic domain-specific sentiment lexicon generation with label propagation. In *Proceedings of international conference on information integration and web-based applications & services* (pp. 53–62).

Team, O. E. L., Stooke, A., Mahajan, A., Barros, C., Deck, C., Bauer, J., Sygnowski, J., Trebacz, M., Jaderberg, M., & Mathieu, M. et al. (2021). Open-ended learning leads to generally capable agents. arXiv preprint arXiv:2107.12808.

Todorov, E., Erez, T., & Tassa, Y. (2012). Mujoco: A physics engine for model-based control. In *IEEE/RSJ international conference on intelligent robots and systems* (pp. 5026–5033). IEEE.

Treleaven, P., Galas, M., & Lalchand, V. (2013). Algorithmic trading review. *Communications of the ACM, 56*, 76–85.

Vázquez-Canteli, J. R., Kämpf, J., Henze, G., & Nagy, Z. (2019). CityLearn v1.0: An OpenAI gym environment for demand response with deep reinforcement learning. In *ACM international conference on systems for energy-efficient buildings, cities, and transportation*.

Vinyals, O., Babuschkin, I., Czarnecki, W. M., Mathieu, M., Dudzik, A., Chung, J., Choi, D. H., Powell, R., Ewalds, T., Georgiev, P., et al. (2019). Grandmaster level in StarCraft II using multi-agent reinforcement learning. *Nature, 575*(7782), 350–354.

Whaley, R. E. (2009). Understanding the VIX. *The Journal of Portfolio Management, 35*(3), 98–105.

Whang, S. E., Roh, Y., Song, H., & Lee, J.-G. (2023). Data collection and quality challenges in deep learning: A data-centric AI perspective. *The VLDB Journal* 1–23.

Wilkman, M. (2020). Feasibility of a reinforcement learning based stock trader. *Aaltodoc*.

Xiao, G., Li, J., Chen, Y., & Li, K. (2020). Malfcs: An effective malware classification framework with automated feature extraction based on deep convolutional neural networks. *Elsevier Journal of Parallel and Distributed Computing, 141*, 49–58.

Xing, F. Z., Cambria, E., & Welsch, R. E. (2018). Natural language based financial forecasting: A survey. *Artificial Intelligence Review, 50*(1), 49–73.

Yang, H., Liu, X.-Y., Zhong, S., & Walid, A. (2020). Deep reinforcement learning for automated stock trading: An ensemble strategy. In *ACM International Conference on AI in Finance*.

Zha, D., Bhat, Z. P., Lai, K.-H., Yang, F., & Hu, X. (2023). Data-centric AI: Perspectives and challenges. arXiv preprint arXiv:2301.04819.

Zha, D., Bhat, Z. P., Lai, K.-H., Yang, F., Jiang, Z., Zhong, S., & Hu, X. (2023). Data-centric artificial intelligence: A survey. arXiv preprint arXiv:2303.10158.

Zhang, Z., Zohren, S., & Roberts, S. (2020). Deep reinforcement learning for trading. *The Journal of Financial Data Science, 2*(2), 25–40.