Reinforcement Learning for Portfolio Optimization Project Overview

Sumit Nawathe
UMD
sumit.nawathe@gmail.com

James Zhang
UMD
jzhang72@terpmail.umd.edu

Ravi Panguluri
UMD
ravipanguluri02@gmail.com

Sashwat Venkatesh
UMD
sashvenk@terpmail.umd.edu

Abstract

The abstract paragraph should be indented ½ inch (3 picas) on both the left- and right-hand margins. Use 10 point type, with a vertical spacing (leading) of 11 points. The word **Abstract** must be centered, bold, and in point size 12. Two line spaces precede the abstract. The abstract must be limited to one paragraph.

1 Introduction

Our group is seeking to develop a reinforcement learning agent to support portfolio management and optimization. Utilizing both empirical stock pricing data along with alternative data, we look to create a more well-informed portfolio optimization tool.

Our primary motivations for pursuing a reinforcement learning-based approach are as follows:

1. Reinforcement learning lends itself well to learning/opening in an online environment. The agent can interact with its environment, providing real-time feedback/ responsiveness to allow for better results.
2. Our approach involves incorporating alternative data to support the agent's decision making process. Encoding this alt-data into the states matrix of the agent allows for the agent to make better decisions when it comes to adjusting portfolio weights.
3. Given that a reinforcement learning agent's decisions are modeled by a Markov Decision Process, we can easily provide different reward functions to account for a variety of investor preferences or restrictions.

A potential advisor for our project is Yada Zhu, a member of the finance research team at IBM. She was an author on one of the key papers we are referencing for our RL model implementation. Her experience with finance-related machine learning, time-series analysis, and alternative data sources would be very valuable in guiding us throughout our process.

Within our team, Sashwat and Ravi are working on dataset scraping, creation, curation, and cleaning; each is testing different APIs (described in the following section). Sumit and James are working on the RL algorithms; they are taking turns implementing papers described in that section and iterating on the architectures.

2 Dataset Creation

Creating a combined dataset encompassing a textual corpus sufficient for us to build a robust reinforcement learning agent will require pulling data from a wide variety of sources. We aim to use two primary types of data: stock and news data.

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

2.1 Stock Data

For now, since our project will likely be focused on building an agent that can perform well on historical data, we aim to use data from the Wharton Research Data Services (WRDS). WRDS is a pre-eminent source of financial data that we have experience utilizing in the Computational Finance program's other courses that has expansive data on stocks. Specifically, we aim to use data from the Center for Research in Security Prices (CRSP), which has security price, return, and volume data for the NYSE, AMEX and NASDAQ stock markets.

We will begin by creating a universe with just the S&P 100 stocks, but we aim to expand to the S&P 500 and/or stocks on all publicly-listed exchanges available through WRDS.

2.2 News Data

We believe that the inclusion of news data into our reinforcement learning agent's state is important because it could integrate an external understanding of how well a given stock is performing at a given time and how it is specifically exposed to certain market or sector risks. This could provide our agent a better view of the environment and knowledge of our ticker companies in context, which can help it make better decisions to maximize our chosen reward function. To that end, we have a few proposed sources for pulling such data as well as a brief insight into our current progress towards that goal.

2.2.1 Scraping Financial News Sources

We want to focus on a few sites more committed to broadly reporting about a wide variety of companies to limit information asymmetry amongst stocks in our universe. News sources like Morningstar, Benzinga, and YFinance all allow users to filter their news searches by ticker. This will not only make it easier to scrape data, but also ensure that we actually have data for all of the companies in our universe. We plan to collect at least a few years of data on each ticker, but one year of historical news articles for each ticker is a good starting point. Note that it is not essential for every ticker to have a news article for every day in our methodology.

In scraping this data, we will be mindful of news organizations' terms of services and ensure we are scraping ethically. Since the content is publicly available on these sites we will scrape at a reasonable frequency to avoid getting rate limited and/or IP blocked.

2.2.2 Utilizing Paid News APIs

Seeing a gap in real-time and historical data pipelines for business and professional use, many companies have created paid news APIs that allow institutions to query a wide range of news sites for current event information including financial news. Many of these APIs such as Event Registry, newsapi.ai, and Alpha Vantage provide this data. However, in examining most of these APIs there are a few significant issues. First, many of them require a significant payment to get data at a velocity that we would need. And for many, historical data is even more expensive.

However, within the free tier of these services we can make some API calls to query information for some tickers that could be additive to our analysis. It would be difficult to rely on this as a long term solution for data though.

2.2.3 SEC Filings

To enrich our dataset, we are considering utilizing SEC filings data for all NYSE/NASDAQ companies (available through WRDS) that we plan to pull data for. These filings contain essential information regarding a company's financial performance, governance, and compliance that could enhance our measure of company outlook. Specifically we aim to use data from 10-K and 10-Q filings. 10-K's are an annual report on a company's performance, and include information. They are divided into items that show a company's financial statements, stock projections, and pertinent information for shareholders in sections that are denoted as items. For our project, the textual data that is most likely to capture a company's sentiment is Item 1A, Risk Factors. This item is a company issued statement on the risk factors that could affect the operations of the business for the next fiscal year. We will

extract this data from 10-K statements for every company in our trading universe to create sentiment indicators for our RL agent to use.

2.3 Sentiment Analysis

Using the news sources and SEC filings data described above, we wish to generate embeddings from which we can extract sentiment related features to provide to our reinforcement learning agent. Our initial approach, which we have conducted some basic testing on, is to utilize the pre-trained FinBERT model, fine-tuned to recognize the sentiment of financial text to create embeddings for us.

2.3.1 FinBERT Sentiment Scores

For each time period, we will query headlines from articles about the individual stocks. Note that if we use a source like Benzinga, articles are already tagged with relevant stock tickers. We will feed headline data to pre-trained FinBERT. The model then will preprocess the text and generate probabilities of the content being positive, negative, or neutral. From there, we can assign each headline a numerical score based on its maximum probability class. The numerical map could look something like the following: positive: 1, neutral: 0, negative: -1 Over a trading period, we can take some aggregate of these class labels for each stock and feed these class labels to our agent's states matrix; at each time step, this value will be appended to the row corresponding to the stocks' price data. (The state matrix will be defined in greater detail in the Algorithmic and Analytical Challenges section.) We will experiment to find an optimal aggregate function; potential options include providing all logits, taking the mean, or designing a custom function to extract value heuristically. One such function could be

$$\texttt{Value}_{\texttt{Embedding}} = \tanh\left(\frac{\texttt{positive sentiment probability}}{\texttt{negative sentiment probability}}\right)$$

This approach combines the "log likelihood" (ratio of probabilities of positive and negative sentiment) along with a penalty for high neutral sentiment (a measure of uncertainty), using the tanh for normalization. This approach would allow us to adequately detect strong positive/negative sentiment. We will compare this against other aggregate functions in training experiments.

We will test the exact same preprocessing pipeline on the SEC 10-K and 10-Q filings for each company in our universe and integrate them into our states matrix. We will experiment with the same options mentioned previously for the optimal aggregation function. An issue that incorporates SEC filings is that they are recorded on a relatively infrequent basis compared to news and price data. Preliminarily, we will assume that there is no decay in the sentiment between reports. That means, the sentiment embeddings for each company will only be updated on dates where a new filing was reported. On non-reporting dates, the embeddings will be filled forward from the last filing date.

2.3.2 Topic Modeling

In a similar manner to the Financial Statement Analysis assignment from Andy Chakraborty's lecture, we can also utilize FinBERT to categorize news headlines and content into financial-related topics. As demonstrated in that assignment, the correlations between such scores and the performance of companies can be useful to our RL agent by similarly incorporating such embeddings as a tensor. For every news headline and SEC filing related to each stock ticker, we will use the same pre-trained topic classification model as we did in our assignment to give us a numeric mapping of each text's most probable topic. Using this, we will append a column of topic embeddings for each ticker on each day to the states matrix.

3 Algorithmic and Analytical Challenge

Our primary model technique is deep reinforcement learning, which is a branch of machine learning that operates in a game-theoretic-like system. Formally, a reinforcement learning problem is an instance of a Markov Decision Process, which is a 4-tuple (S, A, T, R): S the state space (matrix of selected historical stock price and news data available to our model at a given time; see Methodology section), A the action space (portfolio weights produced by our model, under appropriate constraints),

T the transition function (how the state changes over time, modeled by our dataset), and R (the reward function). The goal is to find a policy (function from S to A) that maximizes future expected rewards. Most reinforcement learning research is spent on providing good information in S to the model, defining a good reward function R, and deciding on a deep learning model training system to optimize rewards.

3.1 Existing Literature

Much of the literature applying RL to portfolio optimization has arisen in the last few years. Some relevant papers are:

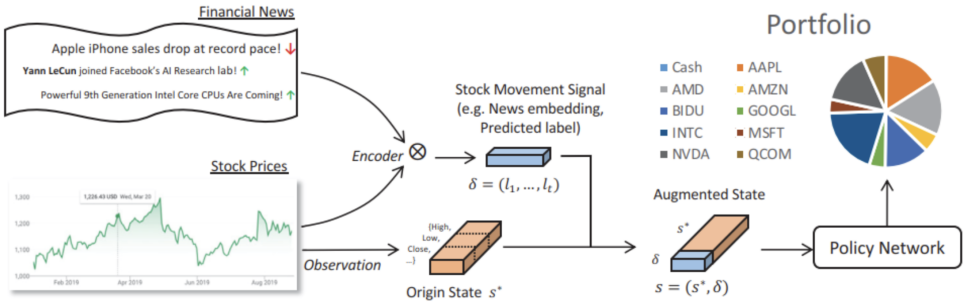
- [2] Deep Reinforcement Learning for Optimal Portfolio Allocation: Using a lookback of recent past returns and a few market indicators (including 20-day volatility and the VIX), this paper implements a simple algorithm for portfolio weight selection to maximize the Differential Sharpe Ratio, a (local stepwise) reward function which approximates (global) Sharpe Ratio of the final strategy. They compare their model with the standard mean-variance optimization across several metrics.
- [3] DRL for Stock Portfolio Optimization Connected with Modern Portfolio Theory: This paper applies reinforcement learning methods to tensors of technical indicators and covariance matrices between stocks. After tensor feature extraction using 3D convolutions and tensor decompositions, the DDPG method is used to train the neural network policy, and the algorithm is backtested and compared against related methods.
- [6] RL-Based Portfolio Management with Augmented Asset Movement Prediction States: The authors propose a method to augment the state space S of historical price data with embeddings of internal information and alternative data. For all assets at all times, the authors use an LSTM to predict the price movement, which is integrated into S. When news article data is available, different NLP methods are used to embed the news; this embedding is fed into an HAN to predict price movement, which is also integrated into S for state augmentation. The paper applies the DPG policy training method and compares against multiple baseline portfolios on multiple asset classes. It also addresses challenges due to environment uncertainty, sparsity, and news correlations.
- [1] A Deep Reinforcement Learning Framework for the Financial Portfolio Management Problem: This paper contains a deep mathematical and algorithmic discussion of how to properly incorporate transaction costs into an RL model. The authors also have a GitHub with implementations of their RL strategy compared with several others.
- [7] Stock Portfolio Selection Using Learning-to-Rank Algorithms with News Sentiment: After developing news sentiment indicators including shock and trends, this paper applies multiple learning-to-rank algorithms and constructs an automated trading system with strong performance.
- [5] MAPS: Multi-agent Reinforcement Learning-based Portfolio Management System: This paper takes advantage of reinforcement learning with multiple agents by defining a reward function to penalize correlations between agents, thereby producing multiple orthogonal (diverse) high-performing portfolios.

3.2 Methodology

We will be implementing, combining, and improving on the methodologies of several of the above papers. Our plan is to develop an RL system that utilizes multiple time periods to achieve strong out-of-sample trading performance. As of this writing, we have partial implementations of papers [1], [2], and [3]. Our final architecture will be most similar to papers [6] and [1].

3.2.1 Markov Decision Process Problem Formulation

Paper [6] includes the following diagram, which is very close to our desired architecture:



An explanation of this diagram: at time t, the origin state S* is a 3D tensor of dimensions $U \times H \times C$ which contains historical price data. U is the size of our universe (for example, for the S&P100, U=100). H is the size of history we are providing (if we are providing 30 day history, then H = 30). C is a categorical value representing the close/high/low price. This format of S* allows us to store, for example, the last 30 days of stock price data for all companies in the S&P100, for any given day. In addition to this, we have news information Delta, obtained from financial news headlines for that day, processed through a pre-trained encoder. This information is added to S* to create the full state S = (S*, Delta)

In our architecture, for S*, we will experiment with the lookback period size and likely reduce it to a 2D array by flattening along the C index, but will otherwise keep S* largely the same. For Delta, we plan to utilize better feature extraction via sentiment scores and topic modeling; we also plan to use different alternative data sources, as described in the Dataset Creation section. In addition, we will extract what company each headline refers to, so our features can be changed over time independently for each company as news articles enter through our environment. The final state S will likely be a 2D matrix, where each row represents a different company (ticker), and along that row we find, concatenated, the following: (1) the past month-or-so of stock price data from S*, and (2) numerical features extracted from recent news data pertinent to that company (as described in the Dataset section). (The straightforward concatenation of price data and news embeddings did not affect the ability of the neural network-based agent to learn.)

Regarding the reward function R, we plan to experiment with both the profit reward function used in [6], as well as the Differential Sharpe Ratio developed in paper [2].

In summary, our project aims to implement and replicate the approach used in [6], with some modifications to S and R as previously described. We will conduct experiments alternative data sources, feature extraction methods, and reward functions (both custom and from other papers listed) to find a good combination that allows this approach to work well on S&P100 stocks; this comprises our novel extension/contribution.

3.2.2 Use of Libraries

We will mainly be using the Gymnasium library to implement the reinforcement learning environments. The Stable Baselines 3 library provides several policy learning techniques that we will experiment with, including Proximal Policy Optimization (PPO) and Deep Deterministic Policy Gradients (DDPG). The papers above discuss the advantages and disadvantages of multiple reward functions and constraints, which we will make improvements upon and provide as options to the user, if applicable.

3.2.3 Strategy Benchmarking

Our final model architecture will be compared against several benchmark financial portfolio selection models. Among these will be the CAPM, an exponential moving average strategy, linear factor models such as the Fama French 3/5-factor models, and the QMJ model. We will compare our returns in-sample and out-of-sample plots, as well as our relative performance on portfolio statistics including cumulative return, Sharpe Ratio, Sortino Ratio, drawdown, etc. The experiment sections in the above papers provide a strong reference for our methodological comparison.

4 User Interface and Visual Analytics

As our project is primarily oriented towards data curation and algorithms, we will not directly have a user interface for a final end-user. However, in order to experiment with and visualize the performance of our various models, we will create a small suite of visualizations in an interactive manner that allows a researcher to choose methods and hyperparameters for our architecture. We will also create a notebook to demonstrate the infrastructure and usage of our deep reinforcement learning model, similar to those on TensorFlow and Huggingface.

5 References

[1] Z. Jiang, D. Xu, and J. Liang, "A Deep Reinforcement Learning Framework for the Financial Portfolio Management Problem." arXiv, Jul. 16, 2017. Accessed: Mar. 29, 2024. [Online]. Available: http://arxiv.org/abs/1706.10059

[2] S. Sood, K. Papasotiriou, M. Vaiciulis, and T. Balch, "Deep Reinforcement Learning for Optimal Portfolio Allocation: A Comparative Study with Mean-Variance Optimization".

[3] J. Jang and N. Seong, "Deep reinforcement learning for stock portfolio optimization by connecting with modern portfolio theory," Expert Systems with Applications, vol. 218, p. 119556, May 2023, doi: 10.1016/j.eswa.2023.119556.

[4] S. Gössi, Z. Chen, W. Kim, B. Bermeitinger, and S. Handschuh, "FinBERT-FOMC: Fine-Tuned FinBERT Model with Sentiment Focus Method for Enhancing Sentiment Analysis of FOMC Minutes," in Proceedings of the Fourth ACM International Conference on AI in Finance, in ICAIF '23. New York, NY, USA: Association for Computing Machinery, Nov. 2023, pp. 357–364. doi: 10.1145/3604237.3626843.

[5] J. Lee, R. Kim, S.-W. Yi, and J. Kang, "MAPS: Multi-agent Reinforcement Learning-based Portfolio Management System," in Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, Jul. 2020, pp. 4520–4526. doi: 10.24963/ijcai.2020/623.

[6] Y. Ye et al., "Reinforcement-Learning based Portfolio Management with Augmented Asset Movement Prediction States." arXiv, Feb. 09, 2020. doi: 10.48550/arXiv.2002.05780.

[7] Q. Song, A. Liu, and S. Y. Yang, "Stock portfolio selection using learning-to-rank algorithms with news sentiment," Neurocomputing, vol. 264, pp. 20–28, Nov. 2017, doi: 10.1016/j.neucom.2017.02.097.