7CCSMPRJ

Deep Reinforcement Learning for Optimal Portfolio Management

Preliminary Project Report

Author: Arman Mann

Supervisor: Dr. Bart de Keijzer

Student ID: 20009893

June 11, 2021

Contents

1 Introduction

1.1 Aims and Objectives

The primary objective of this project is to investigate the application of Deep Reinforcement Learning (DRL) for the purpose of portfolio allocation in the financial domain. Our research will focus on two widely applied model-free DRL algorithms, namely the Deep Q-Network (DQN) (Mnih et al., 2015) and the Deep Deterministic Policy Gradient (DDPG) (Lillicrap et al., 2016). We will also perform a comparative analysis of the results obtained against various benchmarks and traditional portfolio allocation strategies, while contrasting the differences value based and policy based Reinforcement Learning models in general.

Further, by examining the existing research on the use of Reinforcement Learning for this task and by implementing the above mentioned algorithms, we wish to:

- Develop a deep understanding of the entire domain of Reinforcement Learning, its components, and the differences between various methodologies and concepts such as model-free and model-based, value iteration and policy iteration, discrete and continuous action spaces, off-policy and on-policy learning, etc.
- Gain hands-on experience and understanding of the technical implementations of the DQN and DDPG algorithms, simultaneously becoming adept at using the various tools and frameworks for building RL agents and defining environments.
- Evaluate the performance of the DQN and DDPG algorithms on the task using well defined metrics and thereby demonstrate their usefulness at the task of portfolio management.
- Understand the broader use cases for the application of RL in finance, and the challenges associated with the task of portfolio optimization.

1.2 Background

Reinforcement learning (RL) is a branch of machine learning for experience-driven autonomous learning. (Sutton & Barto, 2018). It may also be considered as a mechanism to direct supervised learning methods towards maximizing some long-term or delayed reward (Kolm & Ritter, 2019). Here an agent learns through positive reinforcement by interacting with its environment (or a model of the environment), and performs actions in order to maximize a numerical reward signal over time. (Sutton & Barto, 2018, p. 3). By evaluating the success and failure of these actions the agent learns to make optimal choices for a given problem.

Rapid advances in the field of Deep Learning in recent years has allowed RL to exploit the ability of Deep Neural Networks to solve dynamic stochastic control problems that have high-dimensional state and action spaces. This has led to the establishment of the field of Deep Reinforcement Learning (DRL) which has seen great successes in self-driving, robotics, and gaming.

Similarly, there is a giant body of work which looks to exploit the decision making ability of Reinforcement Learning algorithms in financial domains to make investment and trade decisions related to the pricing of assets, buying and selling of securities, automated or optimal order execution, dynamic balancing and allocation of portfolios, hedging risks and liability management, etc. This has allowed the field to move forward from the traditional and more popular Supervised Learning techniques that have found widespread applications in finance over the last few decades.

This project deals with problem of optimal portfolio management which is the dynamic and intertemporal process of determining optimal weights for a portfolio of multiple assets in order to maximize expected returns for a given level of risk (Sato, 2019), or conversely to minimize the risk for a desired level of expected returns. However, this can be very complicated in case the portfolio weights are to be revised at each time step, and even more so if transaction costs are considered (X. Li et al., 2019).

The process of sequentially optimizing portfolio weights to maximize returns for a given level of risk may be represented as a Markov Decision Process (MDP), making portfolio optimization a stochastic optimal control problem which can be solved using the Bellman Optimality Equation. In case there exists a perfect model which can represent the environment as an MDP with state-transition probabilities and a reward function which are known, this may be solved with Dynamic Programming (DP). However, in portfolio optimization, a model of the environment is not available as the underlying dynamics of the system are not known. As such, we may apply model-free Reinforcement Learning to find an approximate solution to the Bellman equation and thereby compute the optimal policy entirely from the available sample data.

As mentioned earlier, the two model-free algorithms we will study are the Deep Q-Network (DQN) and the Deep Deterministic Policy Gradient (DDPG).

The standard DQN is an off-policy, value based algorithm, which uses a Deep Convolutional Neural Network (CNN) to approximate the action value function, known as the Q-function (Mnih et al., 2015). This provides a massive advantage over traditional Q-learning which quickly becomes intractable when dealing with problems having large state or action space due to the "curse of dimensionality". However, on using a nonlinear function approximator such as a Deep Neural Network to represent the action value, learning is found to become unstable and even diverge in some cases (Tsitsiklis & Roy, 1997). To counteract this, Mnih et al. (2015) propose two key

ideas. First, the DQN uses Experience Replay (Lin, 1992), wherein the weights of the network are upadted using mini-batches of randomly selected past experiences ($s_t, a_t, r_{t+1}, s_{t+1}$ quadruples) from a "replay buffer". This allows the agent to train on less data and leads to better convergence by eliminating the correlations between consecutive samples. Secondly, a separate target network is proposed which is updated only periodically to further increasing stability while training. Learning occurs through an ϵ-greedy strategy wherein the agent randomly picks an action with a probability of 1–ϵ to balance exploitation with exploration.

The DDPG is an off-policy, actor-critic algorithm, which combines the policy based Deterministic Policy Gradient (DPG) with the value based DQN (Lillicrap et al., 2016). While the original DQN works in a discrete action space, DDPG is the continuous action space variant of the DQN. Here, the critic network uses off-policy data and the Bellman equation to concurrently learn the Q-function while the actor network learns the deterministic policy using feedback from the critic. Similar to the DQN, the DDPG makes use of innovations such as Experience Replay and a separate target network to stabilize the learning of the Q-function.

1.3 Data

After reviewing the related literature, it is found that a majority of the implementations make use of candlestick price data as input features to the DRL algorithms for portfolio optimization. This consists of the Open, High, Low and Close prices for the set of assets selected to make up the portfolio. Exceptions are certain frameworks where data about volumes traded and asset fundamentals is also incorporated. For the purpose of our design, we will stick to the standard input features but may extend our design to include additional features if needed.

Although not yet finalised, the primary options for the dataset to be used (in order of preference) are:

- 5 year stock prices from the S&P500 or Dow Jones Industrial Average
- 5 year NIFTY50 stock prices for the Indian Market
- Popular cryptocurrencies with sufficiently available data

Due to the nature of the data, the process of cleaning and engineering the features themselves should be fundamentally easy and non time-consuming, thereby making this a decision which can be taken after some trial and error once development begins.

1.4 Challenges

Deep Learning algorithms, in general, are extremely sensitive to the chosen hyperparameters. Arriving at the optimal hyperparameter values involves significant amounts of intuition, fine tuning, re-training, and designing of a large number of models for comparison. This naturally takes time and resources, and yet there is no guarantee of finding the hyperparameters which provide the optimal solution. Also, given the large number of trials that are likely to be conducted, utmost care is required to not allow the performance on the test dataset to influence hyperparameters as that would corrupt the results obtained.

Secondly, due to repeatedly experiencing the same samples during training, Reinforcement Learning is particularly susceptible to overfitting and may learn random noise patterns present only in the training data. Moreover, if the neural networks used for approximating the Q-value and policy functions are too complex, the high degree of freedom may cause the network to memorize the samples in the training data, and lead to overfitting. Such agents would be unable to generalise to novel situations in the future.

Finally, given that we are using model-free RL which solves decision problems by iterative updates to the Q-values or policy parameters, the learning process is fundamentally tied to the training data. In the case of market data, the historical asset prices within any time period are just one amongst many paths which can be followed to realise current market prices. This makes model-free RL extremely sample inefficient for problems in the financial domain such as portfolio optimization (Sato, 2019), and techniques such as Experience Replay or Adaptive Learning, which will be discussed in the next section, must be adopted.

2 Literature Review

There has been increasing amount of research which deals with the application of Reinforcement Learning to portfolio optimization. In this section we look at the most relevant work in this setting. Since our study will be analysing the DQN and DDPG algorithms, we have decided to primarily showcase prior and state of the art work making use of these algorithms. However, we also include certain research which we find to be integral to the problem domain.

One of the earliest studies conducted to compare value based and policy based Reinforcement Learning for portfolio optimization was undertaken by Du et al. (2009). They applied Q-Learning (without a neural network), which is the basis of current state of the art value based DRL methods, and policy based Recurrent Reinforcement Learning (RRL) to optimize a stock portfolio. Compared to Q-Learning, the policy based algorithm was found to be more robust for stochastic optimal control

and for dealing with noisy data. This is owing to the fact that policy based methods can produce real valued actions and do not suffer from Bellman's "curse of dimensionality" due to discretised state and action spaces.

With the explosion in availability of computation and therefore in Deep Learning applications, Jin and El-Saawy (2016) made the first attempt to apply Deep Q-Learning to optimize a model portfolio consisting of one high-volatility stock and one low-volatility stock, using a multilayer perceptron (MLP) (instead of a CNN) for approximating the Q-value function. Similar to the standard DQN, ϵ-greedy strategy was adopted to balance exploration and exploitation in combination with Experience Replay. As there was no need for discretisation of state space, the input features consisted of the historical stock prices along with auxillary non-price features such as outstanding shares and portfolio value. The agent's action space was discretised into seven percentages which determined the amount of stocks to be bought or sold. Transaction costs were also considered and the highest average Sharpe Ratio was obtained when volatility was penalised. The model was able to perform on par with and exceed variations of the "do-nothing" equally-weighted and the rebalance benchmarks used by the authors.

Y. Li et al. (2019) conducted a feasibility study for the application of Deep RL technology in financial investment by analyzing the three classical variations of Deep Q-Learning (Deep Q-Network (DQN), Double Deep Q-Network (D-DQN) and Dueling Double Deep Q-Network (DD-DQN)) for stock trading. Their experimental results combine the perceptual and computational abilities of Deep Learning with the decision making ability Reinforcement Learning in order to attain better investment strategies. Further, they find the standard DQN performs better than the more complex variants - implying that the improved algorithms aren't necessarily more effective in all domains.

Lucarelli and Borrotti (2020) treat the binary principles of problem decomposition and financial interaction identification at macro and micro levels as the basis for portfolio optimization. These principles are incorporated into their framework in the form of two elements: a set of local agents that model individual asset behaviors and a global agent that defines the global reward function. This framework is applied to optimize a portfolio comprising of four cryptocurrencies using the 3 variants of Deep Q-Learning seen above: DQN, D-DQN, DD-DQN. Compared on daily returns obtained by agents trained on 2 different reward functions, all produce positive results in most test periods. However, once more, the DQN model by far achieves the best performance, obtaining the highest rate of daily return and performing better on Sharpe Ratio compared to equally-weighted and genetic algorithm based portfolio selection strategies.

Z. Zhang et al. (2020) design LSTM based RL models (DQN, PG, A2C) to directly map market conditions to trade positions, bypassing the need to make predictions based on price forecasts. This

is done with the understanding that an LSTMs will be better able to model price movement than a CNN which is typically used. The models trade 50 liquid futures contracts where each model is trained separately using data from different asset classes. They also apply volatility scaling in order to scale trade positions based on market volatility and compare their results on 9 metrics against baselines such as classical time series and momentum strategies. Amongst the models trained, the DQN achieves the best overall performance largely outperforming the baseline strategies.

Gao et al. (2020) propose a DQN framework for the portfolio selection problem in discretised action space which is specifically designed for multi-asset portfolios where there is no need to predict future market prices. Instead of the traditional Experience Replay with a random replay buffer, they propose using Prioritized Experience Replay (Schaul et al., 2016) which weighs samples in the memory pool proportionally according to the difference between the predicted and real Q-values, allowing for more valuable experience and better adaptation to extreme price movements. Compared against 10 baseline portfolio management strategies, the DQN agent outperforms all benchmarks on Accumulated Rate of Return (ARR), Sharpe Ratio and Maximum Drawdown.

In spite of the breakthrough results obtained in the research conducted above, the issue with finite action space models due to the limitations in the number of action combinations remains, which is that they may converge to local opitmal solutions or no optimums at all. Therefore, we now look at equally compelling work relating to the actor-critic Deep Deterministic Policy Gradient (DDPG) model for optimal portfolio management in continuous action space. Work done by Jiang and Liang (2017) was the first empirical study to employ actor-only RL to manage portfolios of several assets. Citing the instability often caused by training separate actor and critic networks, the authors decided to forgo the critic network and make use of a simple Deterministic Policy Gradient model (leaving out the Q-value estimation) to obtain the optimal trading strategies for a cryptocurrency portfolio. Without assuming any prior knowledge of financial markets, the model-free framework makes use of a Convolutional Neural Network (CNN) to accept historical asset prices as inputs and directly output trading actions (or portfolio weight vectors), eliminating the need to predict asset prices. This overcomes the difficulty in having to accurately predict price movements or designing models to decide "what and how much" to buy/sell based on the prices. The proposed framework when compared against 6 benchmarks was able to outperform all on cumulative returns, other than PAMR.

Jiang et al. (2017) later proposed another model-free Deep Reinforcement Learning trading framework where they apply a "fully exploiting" DDPG-like solution to directly output portfolio weights to optimize cumulative returns from a portfolio of 11 cryptocurrencies and one risk-less asset. The model exploits the Ensemble of Identical Independent Evaluators (EIIE) topology which inspects the history of an asset and evaluates its potential growth in the future. Candlestick features (open,

high, low, close prices) are used as input. The authors propose Portfolio-vector memory (PVM) to allow the network to be trained parallelly against samples in a mini-batch as well as prevent the gradient vanishing when using other forms of memory. They also propose an Online Stochastic Batch Learning (OSBL) scheme to consecutively analyze inbound market information more efficiently. The framework is tested with three different neural networks, a CNN, an RNN and an LSTM. The results show that the designed models easily outperform 3 benchmarks and 12 traditional portfolio management strategies, reporting *at least* 4-fold returns over 20 day periods.

Liang et al. (2018) adapt 3 different RL algorithms, the DDPG, Proximal Policy Optimization (PPO) and Policy Gradient (PG) to optimize stock portfolios in continuous action-space. The DDPG here too makes use of the EIIE topology described above, however the CNN is replaced by a Deep Residual Network (ResNet) (He et al., 2016) to better cope with vanishing and exploding gradients. Portfolios are constructed randomly consisting of Chinese stocks and are periodically optimized after penalizing the agent in case of volatility in the prices for the underlying assets. The study also proposes a novel Adversarial Learning approach which adds random noise to the input features for greater training robustness and risk sensitivity. This improves performance however also leads to greater downside risk. Only results for the PG model are reported as the other two algorithms fail to learn a useful policy. Compared with the Uniform Constant Rebalanced Portfolios (UCRP) trading strategy and backtested on the Chinese asset market, the PG is seen to perform better.

Xiong et al. (2018) also apply the DDPG algorithm to find an optimal stock trading strategy to maximize investment return. DDPG is chosen, once again, due to it's ability to handle continuous action space in contrast with the DQN algorithm. Results demonstrate that the model is robust in terms of calibrating risk and significantly outperforms the Dow Jones Industrial Average (DJIA) and min-variance portfolio strategy.

X. Li et al. (2019) propose a novel Adaptive Deep Deterministic Reinforcement Learning scheme (Adaptive DDPG) for finding the optimal strategy for stock portfolio allocation. The model is an application of optimistic and pessimistic Deep RL which allows dynamic updates to the reward signals by distinguishing between positive and negative environment feedback. As such, the model is able to learn differently based on positive and negative forecasting errors and achieves higher returns when compared to the vanilla DDPG, the DJIA, and other traditional portfolio allocation strategies such as min and mean variance.

Huang et al. (2020) present a DDPG framework which enables shorting in continuous action space, allowing them to execute arbitrage trades against the market. Additionally, they showcase further optimizations such as redesigning the activation function and the neural network used by the agent.

In addition to the candlestick data (open, high, low, close), their model also incorporates 3 non-price features: P/E, P/B and Turnover factors, as well as the CSI300 index price. Working with a portfolio consisting of the CSI300 price to represent the market's rate of return and stocks from the CSI500 index, their design is able to consistently achieve higher returns than the market even when stocks are selected randomly.

More recently, H. Zhang et al. (2021) developed a novel DDPG framework for multi-asset trading in continuous action space using LSTMs and Fully Connected neural networks to model their critic and actors respectively. Here too ϵ-greedy strategy is implemented to provide a balance between exploration and exploitation. The framework is used to compare performance with and without transaction costs, however arrives at a counter-intuitive result with findings showing the model with transaction costs to be more robust and profitable. This is explained by interpreting the transaction cost as a penalty which trains the agent to be more cautious and thereby choose better actions in order to be profitable. Both proposed strategies, however, outperform the 7 benchmarks used for comparison in terms of Compound Annual Return Rate (CARR), Maximum Drawdown, and stability measured using the Sharpe Ratio.

3 Schedule

This section presents a timeline for the completion of project related tasks and deliverables. The schedule is created to work alongside and meet the following project deadlines:

- 11^{th} June: Preliminary Report Due
- 5^{th} July: Individual Meeting 1
- 26^{th} July: Group Meeting 4
- 16^{th} August: Individual Meeting 2
- 3^{rd} September: Project Report Due
- 6^{th} September: Video Presentation Due

The identified tasks and deliverables have been put together in the Gantt chart in Figure 1 below where we also set apart sufficient buffer time in case certain tasks take longer than expected.

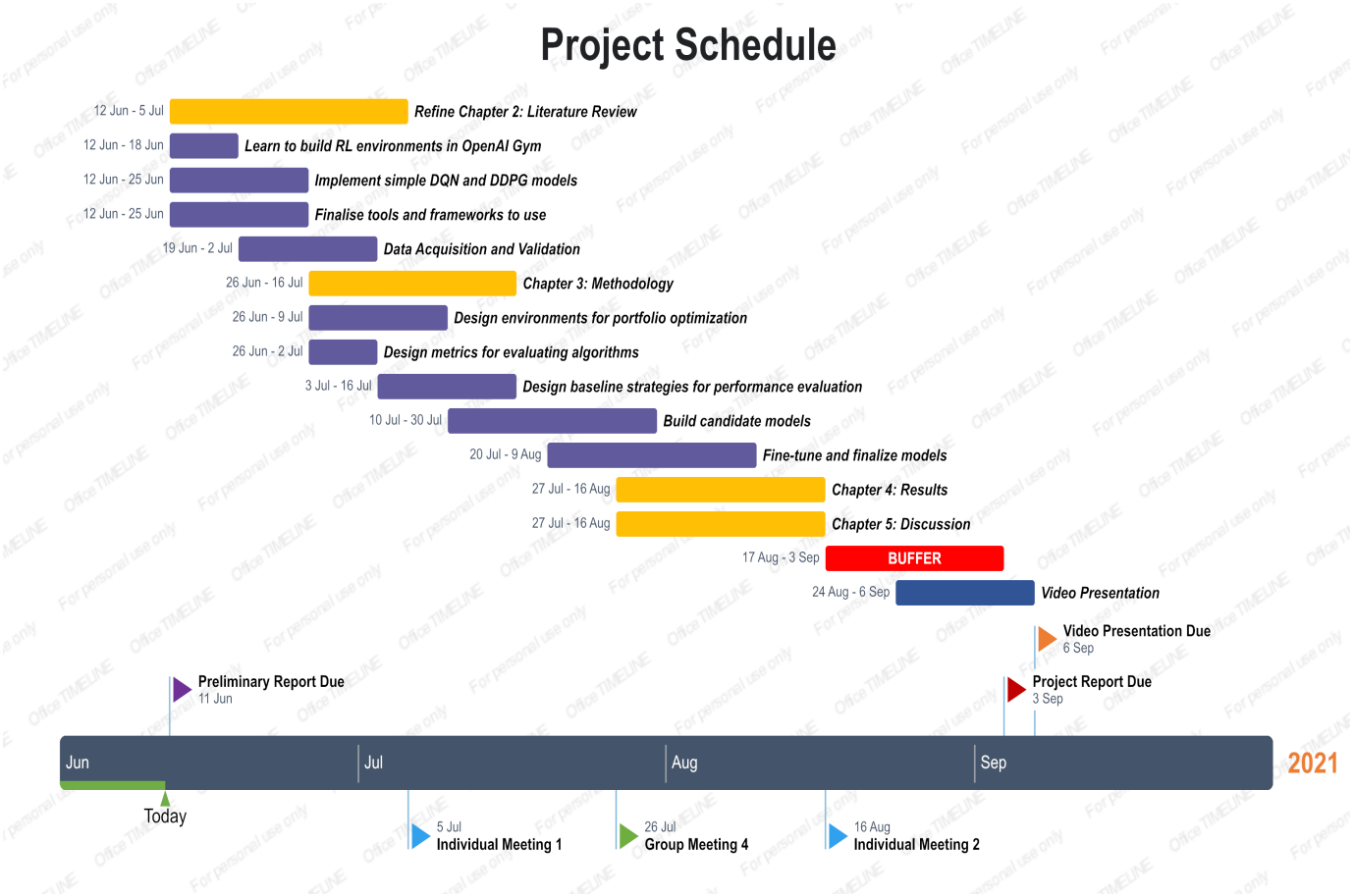

Figure 1: Gantt chart depicting project timeline

References

Du, X., Zhai, J., & Lv, K. (2009). Algorithm trading using q-learning and recurrent reinforcement learning. *Positions*, *1*.

Gao, Z., Gao, Y., Hu, Y., Jiang, Z., & Su, J. (2020). Application of deep q-network in portfolio management. *2020 5th IEEE International Conference on Big Data Analytics (ICBDA)*, 268–275.

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.

Huang, G., Zhou, X., & Song, Q. (2020). Deep reinforcement learning for portfolio management based on the empirical study of chinese stock market. *arXiv:2012.13773*.

Jiang, Z., & Liang, J. (2017). Cryptocurrency portfolio management with deep reinforcement learning. *2017 Intelligent Systems Conference (IntelliSys)*, 905–913.

Jiang, Z., Xu, D., & Liang, J. (2017). A deep reinforcement learning framework for the financial portfolio management problem. *arXiv:1706.10059*.

Jin, O., & El-Saawy, H. (2016). Portfolio management usig reinforcement learning.

Kolm, P. N., & Ritter, G. (2019). Modern perspectives on reinforcement learning in finance. *The Journal of Machine Learning in Finance*, *1*(1).

Li, X., Li, Y., Zhan, Y., & Liu, X.-Y. (2019). Optimistic bull or pessimistic bear: Adaptive deep reinforcement learning for stock portfolio allocation. *arXiv:1907.01503*.

Li, Y., Ni, P., & Chang, V. An empirical research on the investment strategy of stock market based on deep reinforcement learning model. In: *Proceedings of the 4th international conference on complexity, future information systems and risk*. 2019, 52–58.

Liang, Z., Chen, H., Zhu, J., Jiang, K., & Li, Y. (2018). Adversarial deep reinforcement learning in portfolio management. *arXiv:1808.09940*.

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

Lin, L.-J. (1992). *Reinforcement learning for robots using neural networks* (Doctoral dissertation). Carnegie Mellon University.

Lucarelli, G., & Borrotti, M. (2020). A deep q-learning portfolio management framework for the cryptocurrency market. *Neural Computing and Applications*, *32*, 17229–17244.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M. A., 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*(7540), 529–533.

Sato, Y. (2019). Model-free reinforcement learning for financial portfolios: A brief survey. *arXiv:1904.04973*.

Schaul, T., Quan, J., Antonoglou, I., & Silver, D. (2016). Prioritized experience replay. *International Conference on Learning Representations (ICLR*.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction* (2nd). The MIT Press.

Tsitsiklis, J. N., & Roy, B. V. (1997). An analysis of temporal-difference learning with function approximation. *IEEE Transactions on Automatic Control*, *42*(5), 674–690.

Xiong, Z., Liu, X.-Y., Zhong, S., Yang, H., & Walid, A. (2018). Practical deep reinforcement learning approach for stock trading. *arXiv:1811.07522*.

Zhang, H., Jiang, Z., & Su, J. (2021). A deep deterministic policy gradient-based strategy for stocks portfolio management. *2021 IEEE 6th International Conference on Big Data Analytics (ICBDA)*, 230–238.

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