Deep Reinforcement Learning (DRL) for portfolio allocation

Eric Benhamou, David Saltiel, Jean Jacques Ohana, Jamal Atif, and Rida Laraki

AI Square Connect, HOMA Capital, LAMSADE Dauphine, France

To Appear in ECML PKDD 2020 Demo track

Abstract. Deep reinforcement learning (DRL) has reached an unprecedent level on complex tasks like game solving (Go [6], StarCraft II [7]), and autonomous driving. However, applications to real financial assets are still largely unexplored and it remains an open question whether DRL can reach super human level. In this demo, we showcase state-of-the-art DRL methods for selecting portfolios according to financial environment, with a final network concatenating three individual networks using layers of convolutions to reduce network's complexity. The multi entries of our network enables capturing dependencies from common financial indicators features like risk aversion, citigroup index surprise, portfolio specific features and previous portfolio allocations. Results on test set show this approach can overperform traditional portfolio optimization methods with results available at our demo website.

Keywords: Deep reinforcement learning, portfolio selection, convolutional networks, index surprise, risk aversion

1 Introduction

Markovitz approach: Portfolio optimization and its implied diversification have been instrumental in the development of the asset management industry. The seminal work of [5] paved the way for a rational approach to allocate funds among the possible investment choices, quantifying returns and risk statistically. However, a straight out of the box Markovitz portfolio optimization tends to be unreliable in practice as risk estimations are very sensitive to input selections and leads to very unstable conclusions. In particular, it is well-known that naive equally weighted portfolios often outperform mean-variance optimized portfolios [2], that the latter can produce extreme or non-intuitive weights [1]. These well documented problems do not mean a quantitative approach is necessarily flawed but rather that it has to adapt to its environment and motivates for DRL.

Hence, we cast the portfolio optimization problem as a continuous control program with delayed rewards, using state of the art deep reinforcement learning methods. These methods do not aim at making predictions (like in supervised learning) but rather at learning the optimal policy for portfolio weights within

a dynamic and continuously changing market. This approach has the major advantage to adapt to changing market conditions.

Related Work: The idea of applying DRL to portfolio allocation has recently taking off in the machine learning community with some recent works on crypto currencies [8], [4]. Compared to traditional approaches on financial time series, both [8] and [4] found that novel architectures such as Convolutional Neural Network (CNN) tend to perform better for crypto currencies and Chinese stock markets. In this work, we extend their work by integrating common financial states to incorporate in our deep network some common characteristics of financial markets. We also show that it increases the network efficiency to use previous allocations. Although [4] claimed that adding noise to the learning problem should ease model training we found the opposite conclusion. This may be explained by the larger number of features used in our work that should better capture the dynamic nature of financial markets. In this demo, we showcase how state-of-the-art DRL is able to pick the best portfolio allocation out of sample using financial features used by asset managers: risk aversion index, correlation between equities and bonds, Citi economic surprise index. We provide performance out of sample and test various configurations, that are summarized in a demo website.

Contributions: Our contributions are to provide critical insights for the design of this Deep RL problem validated empirically. In particular, we found that

- reward function is criticial. Sharpe ratio reward leads to worse results.
- CNN performs better than LSTM and captures implicit features.
- dependency to previous allocations enhances the model.
- adding noise does not improve the model.

2 Method used

Network Architecture: Our network (as described in figure 1) uses three types of inputs: portfolio returns observed over the last two weeks, one month and one year (network 1); common asset features like correlation between equities and bonds, Citigroup economic surprise and risk aversion indexes (network 2) observed over the last week, as well as the previous portfolio allocation (network 3). We concatenate these 3 networks into a final one using two dense layers and a final softmax one to infer the portfolio weights. Our reward is either the sharpe ratio or the net value of the final portfolio. We also use either convolution layers for the network 1 and 2 (convolution 2D and 1D respectively) or LSTM units. We introduce noise in the training to challenge it but found that this does not improve contrarily to what [4] found. Train dataset is from 01-Jan-2010 to 31-Dec-2017, while test data set ranges from 01-Jan-2018 to 31-Dec-2019. Results of the various trained networks can be visualized in http://www.aisquareconnect.com/deeprl/models.html.

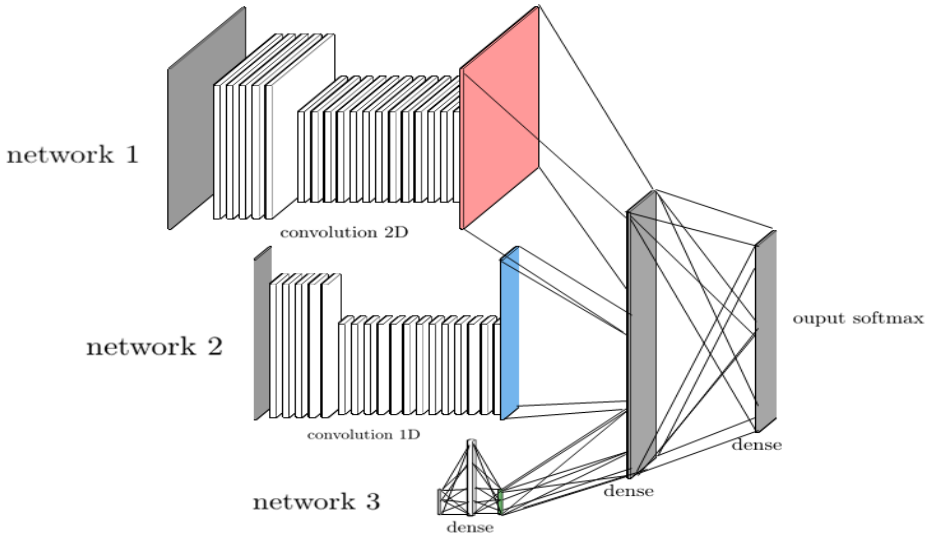

Fig. 1. Best DRL network architecture

DRL algorithm: To find the optimal action $\pi^*(S_t)$ (the portfolio allocation) according to states S_t (financial information) given a reward R_t (the best net portfolio final performance), we represent the policy as our 3 entries network and use deep policy gradient method with non linear activation (Relu). We use buffer replay to memorize all marginal rewards, so that we can start batch gradient descent once we reached the final time step. We use traditional Adam optimization so that we have the benefit of adaptive gradient descent with root mean square propagation [3].

Results Performance results are given below in table 1. Overall, DRL is able to substantially overperform not only traditional methods like static Markovitz or even dynamic Markovitz but also the best portfolio (Naive winner) in terms of net performance and Sharpe ratio, with a final annual net return of 9.49% compared to 4.40 % and 5.27% for static and dynamic Markovitz methods. Dynamic Markovitz method consists in computing the Markovitz optimal allocation every 3 months while the static Markovitz method simply uses the Markovitz optimal portfolio computed on train data set. The Naive winner method consists in just selecting the best portfolio over the train data set. It turns out this is also the best portfolio on the test data set.

3 Target Users and future extension

Our system aims at asset managers to present this new approach. We differentiate from current offerings in portfolio optimization as they mostly rely on Markovitz optimization.
In conclusion, we found that DRL overperforms very substantially not only the

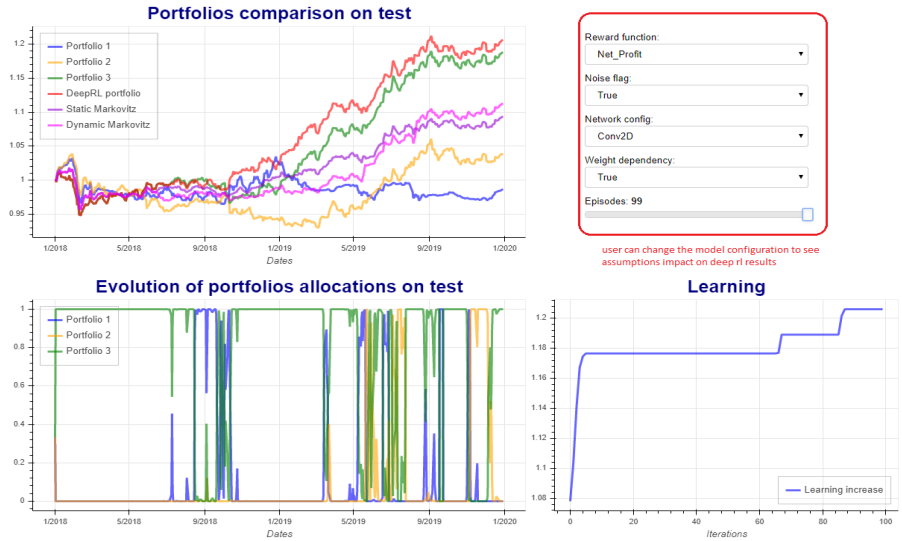

Fig. 2. Deep RL Portfolio Optimisation website

Table 1. Performance results

	Portfolio 1	Portfolio 2	Portfolio 3	Static Markovitz	Dynamic Markovitz	Deep RL	Naive winner
Performance	-0.55%	1.94%	8.47%	4.40%	5.27%	**9.49%**	8.47%
Std dev	4.57%	5.25%	4.91%	**3.55%**	4.22%	4.78%	4.91%
Sharpe ratio	na	0.37	1.72	1.24	1.25	**1.99**	1.72

static but also the dynamic Markovitz method suggesting that DRL captures dependency between the environment states and the reward. Future extensions to this work are to analyze more features and check whether we can enrich our RL problem with more predictive features, like news and market sentiments.

References

1. Black, F., Litterman, R.: Global portfolio optimization. Financial Analysts (1992)
2. DeMiguel et al.: Optimal versus naive diversification: How inefficient is the 1/n portfolio strategy? Review of Financial Studies **22**, 1915–1953 (2009)
3. Kingma, D., Ba, J.: Adam: A method for stochastic optimization (2014)
4. Liang et al.: Adversarial deep reinforcement learning in portfolio management (2018)
5. Markowitz, H.: Portfolio selection. Journal of Finance **7**, 77–91 (1952)
6. Silver et al.: Mastering the game of go without human knowledge. Nature **550**, 354– (Oct 2017)

7. Vinyals et al.: Grandmaster level in starcraft ii using multi-agent reinforcement learning. Nature **575** (11 2019)
8. Zhengyao et al.: Reinforcement learning framework for the financial portfolio management problem. arXiv (2017)