Augmented Lagrangian Risk-constrained Reinforcement Learning for Portfolio Optimization (Student Abstract)

Bayaraa Enkhsaikhan and Ohyun Jo*

Department of Computer Science, Chungbuk National University, Cheongju, 28644, South Korea
bayaraa@chungbuk.ac.kr, ohyunjo@chungbuk.ac.kr

Abstract

We applied Risk-averse Reinforcement Learning (RL) to optimize investment portfolios while incorporating risk constraints. Given that portfolios must adhere to risk constraints set by investors and regulators, enforcing hard constraints is essential for practical portfolio optimization. Traditional techniques often lack the flexibility to model the complexities of dynamic financial markets. To address this, we used the Augmented Lagrangian Multiplier (ALM) to impose constraints on the agent, reducing risk during decision-making. Our risk-constrained RL algorithm demonstrated no constraint violations during testing and outperformed other Risk-averse RL methods, indicating its potential for optimizing portfolios for risk-averse investors.

Introduction

Portfolio optimization in finance involves constructing an investment portfolio to achieve the highest possible return for a given level of risk or minimizing risk for a given return. Traditional methods often rely on fixed inputs and assume constant conditions, which are unrealistic in the complex and ever-changing financial markets (Heaton, Polson, and Witte 2017). Recent advancements in RL have proven effective for dynamic portfolio optimization, as they do not require supervised training and are well-suited to the task. Existing RL models, however, have focused on maximizing returns or risk-adjusted returns, such as the Sharpe ratio, without fully addressing investors' varying risk preferences.

Our proposed solution is a constrained RL model that incorporates risk and other practical constraints for portfolio optimization. Risk is assessed using advanced techniques like Value at Risk (VaR), making the model more practical for investors with diverse risk perspectives.

Methodology

The architecture of the proposed model is illustrated in Figure 1. It follows the Deep Deterministic Policy Gradient (DDPG) optimization algorithm (Lillicrap et al. 2016). There are three deep learning networks: the actor, critic, and cost networks. The actor network takes action after observing state information as input. The critic network predicts the expected reward value at that state given the action from the actor network. Finally, the cost network predicts the expected risk given the action and state information. The trading decisions are made based on market and price information, with rewards contingent on the state and decision. The agent aims to optimize cumulative rewards whereas ensuring compliance with the portfolio risk constraint.

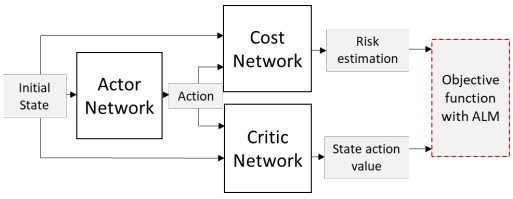

Figure 1: Architecture of the forward propagation process.

Action: Our set of actions involves assigning weights to each asset in the portfolio, where the weight for the j-th asset at time t is denoted as $a_{j,t}$. These weights are restricted within the range [0, 1]. The overall value of the portfolio at time t is u_t (with an episode length of 1000).

Reward: The reward is determined based on the changes in values over each time step. The immediate reward at time step t is calculated as follows (the fee is 0.25% of the transaction amount) :

$$r_t = u_t - u_{t-1} - fee_t \tag{1}$$

State observations: We have selected current and historical price and volume data with different financial indicators over the preceeding month, three months, half-year, and yearly periods as observation values. Therefore, our problem definition is (c is constraint function):

$$\max_a \sum_t E[r_t], \text{ subject to } c(s_t, a_t) < \zeta \tag{2}$$

Critic network: DDPG algorithm is well-suited for our Actor-Critic networks, as indicated by (Lillicrap et al. 2016). The Q-value is assessed using the $Q(a_t, s_t)$ function. Derived from the Bellman equation, we obtain following critic network, $Q(s, a) = r_t + \gamma Q'(s', \pi'(s'))$, where s' is the next state, a' is the next action, γ is the discount factor for rewards, and r is the reward defined by Eq 1, Q' and π' are the target networks. The loss function needs to be minimized

*Corresponding auhtor.

to train the critic is (here, $d \in \{0; 1\}$ is to indicate whether s' is final state of episode or not):

$$L = \frac{1}{N}\sum_{i=1}^{N}(Q(s,a) - (r + \gamma(1-d)Q'(s', \pi'(s'))))^2 \quad (3)$$

Cost network: Let y be the return, and F_y be cumulative distribution function of the return. The VaR at $\alpha \in (0, 1)$ confidence level defined as (we used historical simulation method for VaR):

$$VaR_\alpha(y) = -inf\{y \in R : F_y(y) > \alpha\} \quad (4)$$

The cost function. $c(s, a)$ is to predict Eq 4. Hence constraint (ζ is limitation given by the investor in Eq 2.):

$$C(s_i, a_i) = \begin{cases} 0, & \text{if } c(s_i, a_i) < \zeta \\ c(s_i, a_i) - \zeta, & \text{else} \end{cases} \quad (5)$$

Actor network: Noise is used to satisfy exploration of actor parameterized by w_π, $\pi^{w_\pi}(s_i) = a_i$. We use the following Dirichlet distribution instead of Ornstein-Uhlenbeck process for action sampling due to action limitation (Tian et al. 2022).

$$Dir(\upsilon) = \frac{1}{B(\upsilon)}\prod_{i=1}^{K} x_i^{\upsilon_i - 1}; B(\upsilon) = \frac{\prod_{i=1}^{K}\Gamma(\upsilon_i)}{\Gamma(\sum_{i=1}^{K}\upsilon_i)} \quad (6)$$

here, υ is a vector, $B(\upsilon)$ is multivariate beta function and K is the number of assets. The shape of the distribution is determined by a vector υ. $\Gamma(\upsilon)$ is gamma function. Dirichlet policy can solve some issues of DDPG since our action is the asset weights and summed up to 1. The shape determining vector υ is calculated by: $\upsilon = \kappa a$. κ ($\kappa > 1$) is hyperparameter that controls exploration because it is similar to kurtosis coefficient, a is the initial action before exploration, and the final action is sampled from Eq 6. Insufficient κ can harm model convergence. Augmented Lagrangian method (Hestenes 1969) can be constructed using following objective function to minimize:

$$L^{w_\pi} = -Q(s, a^{w_\pi}) + \lambda C(s, a^{w_\pi}) + \rho/2(C(s, a^{w_\pi}))^2 \quad (7)$$

where $\lambda \geq 0$ and $\rho > 0$ are parameters that are learnt. In order to optimize actor network, Eq 7 is minimized by the following steps:

- Initialize parameters: ρ is suggested to be high.
- Minimize the objective: $w_\pi \leftarrow argmin_w L(w_\pi, \lambda)$
- Update Lagrange multipliers: $\lambda \leftarrow \lambda + \rho J_C^{w_{k+1}}$
- Update Penalty: Adjust ρ to increase it gradually, till $t = 1000$, then capped the penalty parameter to prevent ill-conditioning as suggested by (Jorge Nocedal 2006)
- Convergence Check: Repeat these steps till convergence.

The critic network's parameters are optimized minimizing Eq 3. The cost network's parameters are learned by minimizing its' error on prediction Eq 4. The actor network's parameters are optimized minimizing Eq 7, the critic and cost networks are fixed during the actor optimization.

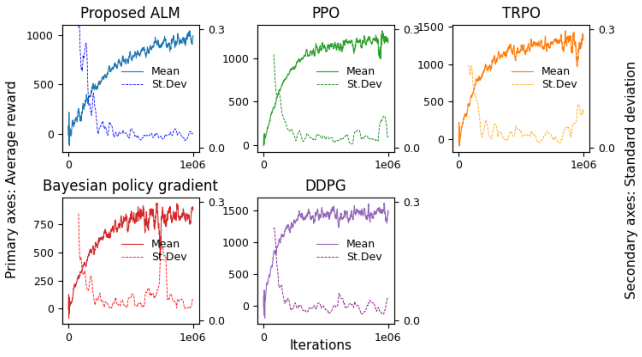

Figure 2: Training rewards based on specific objective function.

Experimental Result and Conclusion

In Figure 2, the learning progress of various RL algorithms—namely, Proximal Policy Optimization (PPO) (Schulman et al. 2017b), Trust Region Policy Optimization (TRPO) (Schulman et al. 2017a), Bayesian Policy Gradient (BPG) (Eriksson and Dimitrakakis 2020), and DDPG (Lillicrap et al. 2016)—is shown. The rewards for each algorithm are calculated based on their respective objective functions. Among these, DDPG demonstrates faster learning due to its ability to make riskier decisions and explore more extensively. Conversely, both the proposed model and BPG exhibit slower convergence as they employ safer and more constrained exploration strategies. Nevertheless, the proposed model performed relatively well, adhering to constraints in the test dataset as shown in Table 1. Although DDPG achieved the highest returns, it significantly violated constraints, unlike the proposed model, which maintained compliance throughout.

Selected models and objectives	Annual average return	Constraint Violation Number	Sharpe ratio
DDPG-Expected return	40.3%	154	0.2847
PPO-Sharpe ratio	26.6%	187	0.2234
TRPO-Sharpe ratio	26.8%	93	0.3890
BPG	33.0%	28	0.3377
Proposed ALM	35.7%	0	0.3369

Table 1: Performance comparison of selected trading strategies

	i) Lagrangian multiplier	ii) Quadratic penalty	iii) Performance
Full model	Included	Included	100%
Model 1	Not included	Included	96%
Model 2	Included	Not included	94%
Model 3	Not included	Not included	89%

Table 2: Ablation performance. The Full model-Model's performance is regarded as 100% and other models are compared to the full model

Acknowledgments

This work was supported in part by the Institute of Information & Communications Technology Planning & Evaluation (IITP) funded by the Korean Government [Ministry of Science and ICT (MSIT)] through the Development of 5G+ Intelligent Basestation Software Modem under Grant 2021-0-00165, and in part by the National Research Foundation of Korea (NRF) funded by the Korean Government (MSIT) under Grant 2021R1A2C2095289.

References

Eriksson, H.; and Dimitrakakis, C. 2020. Epistemic Risk-Sensitive Reinforcement Learning. In *28th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning ESANN*.

Heaton, J. B.; Polson, N. G.; and Witte, J. H. 2017. Deep learning for finance: deep portfolios. *Applied Stochastic Models in Business and Industry*, 33(1): 3–12.

Hestenes, M. R. 1969. Multiplier and gradient methods. *Journal of Optimization Theory and Applications*, 4(5).

Jorge Nocedal, S. J. W. 2006. *Numerical Optimization*. Springer New York, NY, 2 edition. ISBN 9780387303031.

Lillicrap, T. P.; Hunt, J. J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; and Wierstra, D. 2016. Continuous control with deep reinforcement learning. In *Proceedings of the 4th International Conference on Learning Representations, ICLR*.

Schulman, J.; Levine, S.; Moritz, P.; Jordan, M. I.; and Abbeel, P. 2017a. Trust Region Policy Optimization. In *Proceedings of the 31st International Conference on Machine Learning, ICML*.

Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; and Klimov, O. 2017b. Proximal Policy Optimization Algorithms. *arXiv*.

Tian, Y.; Han, M.; Kulkarni, C.; and Fink, O. 2022. A prescriptive Dirichlet power allocation policy with deep reinforcement learning. *Reliability Engineering & System Safety*, 224: 108529.