Portfolio Optimization Based on GH Distributions with Mean-CVaR-Skewness Analysis Using Deep Reinforcement Learning

Authors: Jingyuan Sun, Luyang Han, Feiyang Peng, Yuchen Xiang, Yang Luo

Supervisor: Prof. Arodh Lal Karn

SURF-2024-0035

Abstract

This research combines the Generalized Hyperbolic (GH) distribution and Deep Reinforcement Learning (DRL) for portfolio optimization. Using 30 stocks from Yahoo Finance, the model fits the GH distribution and employs a DRL framework with a policy network containing self-attention and CNN-LSTM layers, an EM algorithm for parameter estimation, and a custom reward function incorporating commissions. Simultaneously, the DRL agent optimizes portfolio allocation by balancing CVaR and skewness. Hence, the environment provides iterative reward signals to the agent based on actions that control risk while maximizing return asymmetry. Ultimately, the findings indicate the approach yields superior portfolio performance under complex market conditions.

Mean-CVaR-Skewness Optimization in Portfolio Management

Introduction

In portfolio optimization, the traditional mean-variance model assumes that asset returns follow a normal distribution. However, this assumption often fails in real-world markets. Asset returns typically exhibit fat tails and asymmetry, necessitating a more flexible distribution model to describe these characteristics. To address this, we propose a new optimization framework combining the Generalized Hyperbolic (GH) distribution with deep reinforcement learning, aiming to minimize Conditional Value at Risk (CVaR) while maximizing skewness, thus optimizing portfolio returns under controlled risk.

Generalized Hyperbolic Distribution

The random vector $\mathbf{X}$ is given by:

$$\mathbf{X} = \boldsymbol{\mu} + Z\boldsymbol{\gamma} + \sqrt{Z}A\mathbf{N}, \quad \mathbf{N} \sim N_k(\mathbf{0}, I_k), \quad A^{-1}A = \Sigma, \quad Z \sim \mathrm{GIG}(\lambda, a, b).$$

The PDF of Z under GIG is:

$$h_{GIG}(Z;\lambda,a,b) = \begin{cases} \frac{a^{-\lambda}(ab)^{\lambda/2}}{2K_\lambda(\sqrt{ab})} Z^{\lambda-1} e^{-\frac{aZ+b/Z}{2}}, & \omega > 0, \\ 0, & Z \le 0. \end{cases}$$

Let $\chi = (\mathbf{x}-\boldsymbol{\mu})^\top \Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu}) + b$ and $\psi = \boldsymbol{\gamma}^\top \Sigma^{-1}\boldsymbol{\gamma} + a$, the PDF of X is:

$$f(\mathbf{x}) = \frac{e^{(\mathbf{x}-\boldsymbol{\mu})^\top \Sigma^{-1}\boldsymbol{\gamma}}}{(2\pi)^{d/2}|\Sigma|^{1/2}} \frac{\left(\frac{a}{b}\right)^{\lambda/2}}{2K_\lambda(\sqrt{ab})} \left(\frac{\chi}{\psi}\right)^{\frac{\lambda-d/2}{2}} K_{\lambda-d/2}\left(\sqrt{\chi\psi}\right)$$

CVaR under GH Distribution

$$\mathrm{CVaR}_\beta(x) = -\frac{1}{\beta}\int_{-\infty}^{\mathrm{VaR}} x f(x)\, dx$$

where $f(x)$ is the PDF of return x, β is the cut-off point, and VaR is the Value at Risk level.

Let $\boldsymbol{\omega} = (\omega_1, \omega_2, \ldots, \omega_n)$ with $\omega_i > 0$ and $\sum_{i=1}^n \omega_i = 1$. Then,

$$w^\top X = w^\top \mu + Zw^\top \gamma + \sqrt{Z}w^\top AN \sim \mathrm{GH}_1(\lambda, a, b, w^\top\mu, w^\top\gamma, w^\top\Sigma w).$$

Set $m = w^\top\mu$, $s = w^\top\Sigma w$, and $g = w^\top\gamma$. Define $Q(x) = \frac{(x-m)^2}{s}$ and $W(x) = \sqrt{(a+Q(x))\left(b+\frac{g^2}{s}\right)}$, the Conditional Value at Risk (CVaR) at level β is calculated by:

$$\mathrm{CVaR}_\beta(x) = -\frac{1}{\beta}\int_{-\infty}^{\mathrm{VaR}_\beta(x)} x\frac{\left(\sqrt{\frac{b}{a}}\right)^\lambda \left(b+\frac{g^2}{s}\right)^{\frac{d}{2}-\lambda}}{(2\pi)^{\frac{d}{2}}\sqrt{s}K_\lambda(\sqrt{ab})} \times \frac{K_{\lambda-\frac{d}{2}}(W(x)) \cdot e^{\frac{(x-m)g}{s}}}{(W(x))^{\frac{d}{2}-\lambda}} dx$$

Approximation of CVaR

Direct computation of CVaR is complex; a simpler approach is introduced[1]:
Given $X = \mu + Z\gamma + \sqrt{Z}AN$, transform X using A^{-1}:

$$A^{-1}X = A^{-1}\mu + A^{-1}Z\gamma + \sqrt{Z}N, \quad \text{let } Y = A^{-1}X = \mu_0 + \gamma_0 Z + \sqrt{Z}N,$$

where $\mu_0 = A^{-1}\mu$, $\gamma_0 = A^{-1}\gamma$. Let $x^\top = \omega^\top A$ so: Approximate $\mathrm{CVaR}_\beta(\omega^\top X)$ as:

$$\mathrm{CVaR}_\beta(x^\top Y) = -x^\top\mu_0 + \sqrt{x^\top x}\left[v_+ + v_- \cos(\theta(x,\gamma_0))\right],$$

where:

$$v_+ = \frac{\mathrm{CVaR}_\beta(Y_b) + \mathrm{CVaR}_\beta(Y_{-b})}{2}, \quad v_- = \frac{\mathrm{CVaR}_\beta(Y_b) - \mathrm{CVaR}_\beta(Y_{-b})}{2}, \quad b = \|\gamma_0\|,$$

with $Y_b = \|\gamma_0\|Z + \sqrt{Z}N$ and $Y_{-b} = -\|\gamma_0\|Z + \sqrt{Z}N$.

The probability density function of Y_a is:

$$f_a(y) = \frac{(\sqrt{\psi/\chi})^\lambda (\psi+a^2)^{\frac{1}{2}-\lambda}}{\sqrt{2\pi}K_\lambda(\sqrt{\chi\psi})} \times \frac{K_{\lambda-\frac{1}{2}}\left(\sqrt{(\chi+y^2)(\psi+a^2)}\right)e^{ay}}{\sqrt{(\chi+y^2)(\psi+a^2)}^{\frac{1}{2}-\lambda}}$$

$$\text{Gradient: } \frac{\partial\,\mathrm{CVaR}_\beta\left(x^\top Y\right)}{\partial x^\top} \cdot \frac{\partial X^\top}{\partial \omega^\top} = \left(\left[-\mu_0 + v_+\frac{x}{\|x\|} + v_-\frac{\gamma_0}{\|\gamma_0\|}\right]A\right)^\top$$

EM Algorithm and Skewness

To efficiently estimate these parameters, we employ the Expectation-Maximization (EM) algorithm[2]. The EM algorithm iterates between the E-step, which calculates the expectation of the latent variables w_i given the current parameter estimates, and the M-step, which maximizes the expected log-likelihood to update the parameters. This process is repeated until the parameters converge, leading to stable estimates.

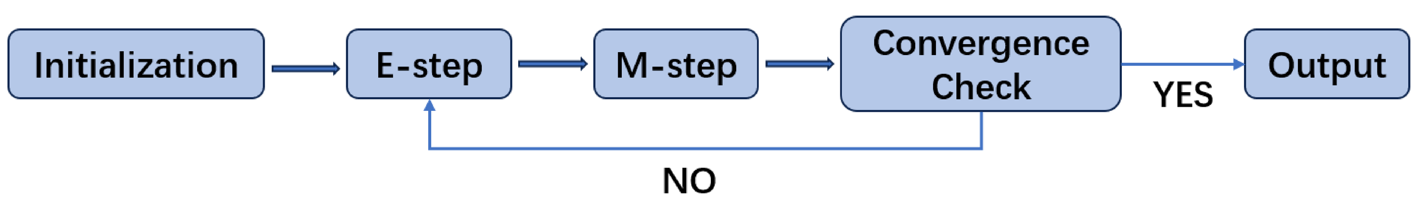

Figure 1. EM Algorithm flow chart

After estimating these parameters, the portfolio return $\omega^T X$ can be expressed as:

$$\omega^T X = \omega_1 X_1 + \omega_2 X_2 + \cdots + \omega_n X_n.$$

This expression preserves the GH distribution properties, allowing us further to calculate the skewness and CVaR of the portfolio returns.

Skewness, as a measure of distribution asymmetry, is defined as the normalized third moment. In the derivation process, we calculated the third moment of the portfolio return. By expanding it and incorporating the higher moment formulas of the GIG distribution, we ultimately obtained the closed-form expression for skewness:

$$\kappa(\omega^T X) = \frac{(\omega^T\gamma)^3 Q - 3(\omega^T\gamma)(\omega^T\Sigma\omega)Var(Z)}{E[Z]\sqrt{\omega^T\Sigma\omega} + (\omega^T\gamma)^2 Var(Z)},$$

where $Q = E[Z^3] + 2E[Z]^3 - 3E[Z^2]E[Z]$. This formula shows how skewness is influenced by the portfolio weights ω, the distribution parameter γ, and the covariance matrix Σ within the GH distribution framework.

Optimization

We aim to minimize the CVaR in our stock portfolio while maximizing the skewness, as this approach allows us to minimize risk and maximize return.

$$\begin{cases} \min\limits_{\omega\in\mathbb{R}^n} CVaR_\beta(\omega^T X), \\ \max\limits_{\omega\in\mathbb{R}^n} \mathrm{Skew}(\omega^T X), \end{cases} \quad \text{s.t.} \quad \begin{cases} \omega^T e = 1, \\ \mathbb{E}\left[\omega^T \mathbf{X}\right] = \omega^T\mu + \omega^T\gamma\mathbb{E}[T] = m. \end{cases}$$

If $\omega^T\mu$ and $\omega^T\gamma$ are fixed, with Theorem 1[3], then $CVaR(\omega^T\mathbf{X})$ is non-decreasing with respect to $\omega^T\Sigma\omega$.

Using the formula of CVaR and Skewness above, we can draw the relationship between CVaR, Skewness, and $\omega^T\Sigma\omega$, ω in Python:

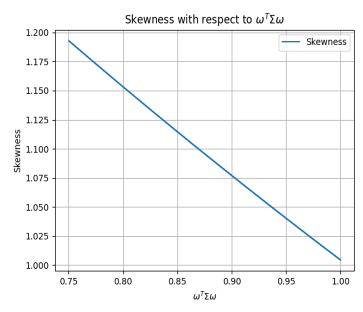

Figure 2. Skewness with respect to $\omega^T\Sigma\omega$

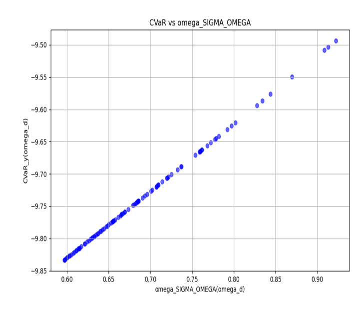

Figure 3. CVaR with respect to $\omega^T\Sigma\omega$

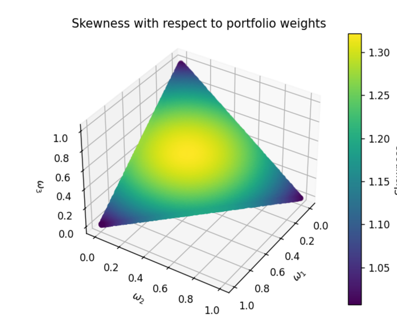

Figure 4. Skewness with respect to ω

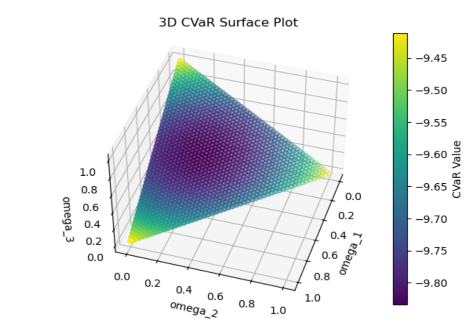

Figure 5. CVaR with respect to ω

It is evident from the figure above that the smaller the $\omega^T\Sigma\omega$, the smaller the CVaR and the larger the skewness. We draw the graph of the relationship of Skewness and CVaR with ω. The relationship between ω and skewness is opposite to the relationship between ω and CVaR, indicating that a larger ω leads to higher risk and lower asymmetry in the portfolio returns.

Deep Reinforcement Learning for Optimization

Deep Reinforcement Learning

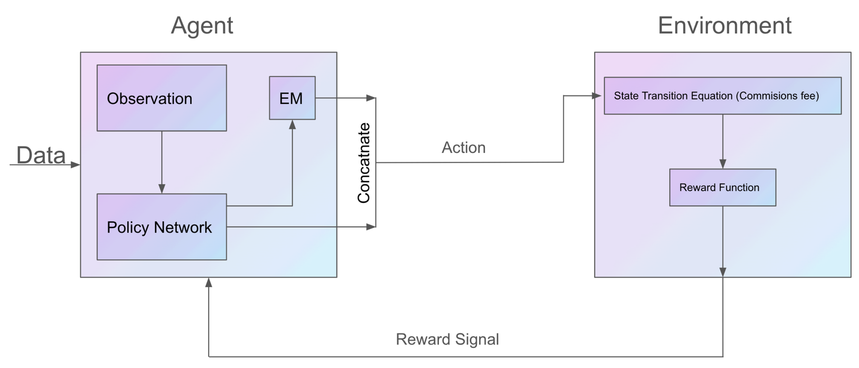

Figure 6. The Process of DRL

Here's how we'll process our data using DRL in general: The DRL includes the environment, the agent, and the steps required for training. Its training process and the structure of each part are shown in the figure below. We constructed an asset management DRL model for the stock market based on the framework proposed by Jiang[4]

Data Preprocess

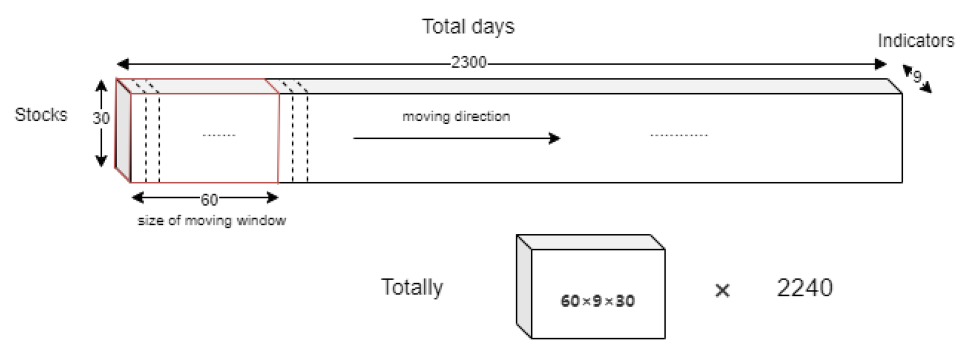

Figure 7. The figure for input data

Firstly, we obtain the data of 30 U.S. stocks through Yahoo Finance and use feature engineering[5] to clean and supplement the data, add log return and other indicators. Using the moving window method, the original data with a shape of (2300, 60, 30) was processed along the temporal dimension with a step size of 1 day, resulting in an increased data volume and more training iterations[6].

The final data shape is (2240,60,9,30), which represents a 4-dim feature tensor consisting of 2240 matrices with a time of 60 days, 9 indicators, and all stock names, as shown in the figure. Its sum is subsequently saved in the DRL environment.

Reward Function

At the same time, the environment also sets the reward function as shown below, the parameters of the policy network, saving the actions of the agent and the state space containing the market situation returned to the agent[4].

$$Rewards = \sum(cash + asset\ value) + additional\ reward - total\ penalty - initial\ cash$$

Policy Network

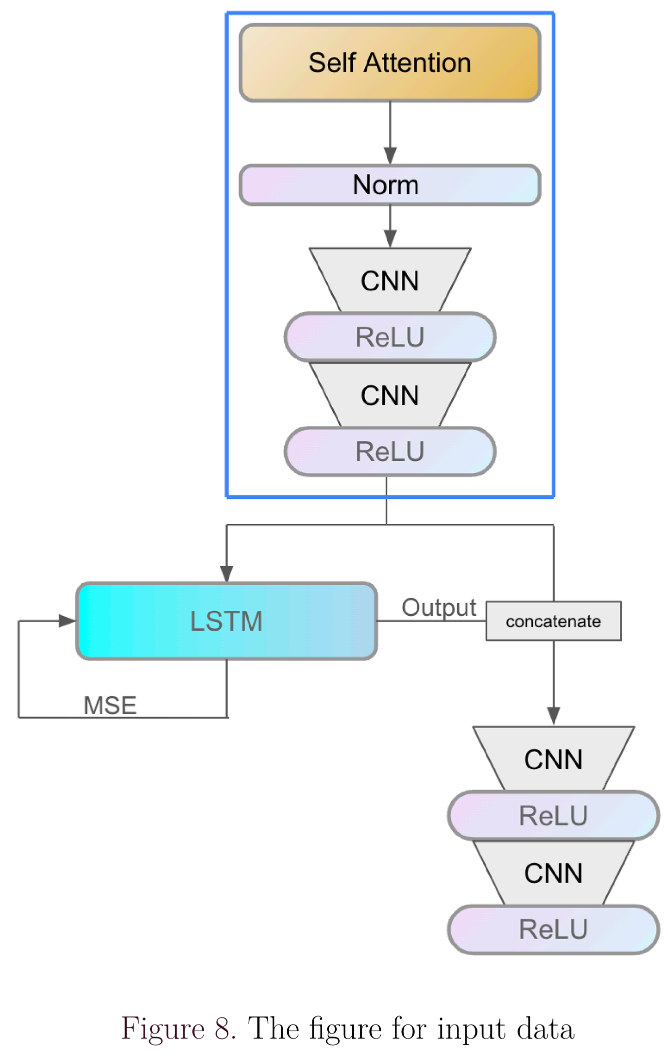

Figure 8. The figure for input data

The topology of the policy network consists of three components: feature extraction, return rate prediction, and classification of 30 stocks in each iteration. At the beginning of each episode, the agent inputs the processed data. First, a self-attention layer is employed to enhance the temporal features of the time-stock subspace by matrix multiplication while preserving the spatial characteristics of the data[7]. Next, a CNN is used to extract spatial features from the daily stock indicators data. The output of the feature extraction network serves as a shared layer, which is fed into both the LSTM and CNN-Classification networks. Due to the excellent performance of LSTM on long time series data[8], it is employed to complete the prediction task. The LSTM network is iteratively optimized first, and then its results are concatenated with the shared layer output and fed into the CNN-Classification network. The network is trained using the CVaR-Skewness joint loss and Adam optimizer[9], ultimately producing a (30, 1) weight result as the strategy to be adopted by the agent in this iteration, this represents the weight of each stock in the investment portfolio.

The figure illustrates the flow of input data through the different layers of the network, starting from the initial feature extraction to the final output tensor used for decision-making in the DRL environment.

Training Results

The two optimization tasks in the policy network regression and classification are merged into a joint loss for optimization, and we can meet the needs of different types of investments by adjusting the coefficient size before each loss. The agent continuously improves the value of reward according to various optimization strategies such as PPO, DDPG, and other algorithms, and obtains the optimal weight to achieve the highest stock return rate finally.

References

[1] N. Abudurexiti, K. He, D. Hu, S. T. Rachev, H. Sayit, and R. Sun, "Portfolio analysis with mean-CVaR and mean-CVaR-skewness criteria based on mean-variance mixture models," Annals of Operations Research, vol. 336, no. 1, pp. 945–966, 2023.

[2] W. Hu, "Calibration of multivariate generalized hyperbolic distributions using the EM algorithm, with applications in risk management, portfolio optimization, and portfolio credit risk," Ph.D. dissertation, Florida State Univ., Tallahassee, FL, 2005. [Online]. Available: http://fsu.digital.flvc.org/

[3] X. Shi and Y. S. Kim, "Coherent risk measures and normal mixture distributions with applications in portfolio optimization," International Journal of Theoretical and Applied Finance, vol. 24, no. 04, pp. 2150019-4–2150019-5, 2021.

[4] Z. Jiang, D. Xu, and J. Liang, "A deep reinforcement learning framework for the financial portfolio management problem," arXiv preprint arXiv:1706.10059, 2017.

[5] X. Y. Liu, H. Yang, Q. Chen, R. Zhang, L. Yang, B. Xiao, and C. D. Wang, "FinRL: A deep reinforcement learning library for automated stock trading in quantitative finance," arXiv preprint arXiv:2011.09607, 2020.

[6] B. Babcock, M. Datar, and R. Motwani, "Sampling from a moving window over streaming data," in *Proc. Annu. ACM-SIAM Symp. Discrete Algorithms (SODA)*, Stanford InfoLab, 2002.

[7] A. Vaswani et al., "Attention is all you need," Advances in Neural Information Processing Systems, vol. 30, 2017.

[8] S. Hochreiter and J. Schmidhuber, "Long short-term memory," Neural Computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[9] D. P. Kingma, "Adam: A method for stochastic optimization," arXiv preprint arXiv:1412.6980, 2014.