

Portfolio Optimization Based on Deep Learning and Factor Constraints

Jinze Piao^{1,a,*}

¹*School of Humanities and Social Science, Beihang University, Beijing, China*
a. pjz19379076@buaa.edu.cn

**corresponding author*

Abstract: Portfolio construction can help investors achieve a balance between risks and returns, and rationally allocate funds to maximize returns. This paper combines four deep learning models LSTM, GRU, CRNN, TCN with investment portfolio strategies Mean-Variance, Mean-CVaR, selects stock data from different industries in the US stock market to construct portfolios, and these strategies are tested in both bull and bear market environment. Comparative analysis of cumulative returns reveals that in the bull market, the cumulative return of TCN+MV and GRU+MV is the highest. In the bear market, the cumulative returns of portfolios using the four deep learning algorithms combined with MV are similar and overall outperform those combined with MC. Furthermore, based on the deep learning algorithm and MV model, this paper selects multiple factors for scoring, and uses factor scores as constraints to the process of portfolio optimization, and the cumulative return has been significantly improved. The method in this paper can provide a theoretical reference for investors to construct investment portfolios and weigh risks and benefits according to individual needs.

Keywords: portfolio optimization, deep learning, price prediction, factor constraint model

1. Introduction

In the decades following Markowitz's introduction of the Mean-Variance model (MV), portfolio theory has been extensively researched and utilized [1]. The concept of investment portfolio is not simply to allocate funds to different assets, but to maximize the expected return under a given risk level through ingenious trade-offs and optimization. The importance of investment portfolio lies in its ability to help investors achieve effective allocation of funds and diversification of risks and improve the overall performance of the investment portfolio.

However, some traditional portfolio methods have certain limitations, such as the assumption of asset returns, simplification of investors' risk attitudes, and the handling of correlations between assets. As time has passed, novel portfolio approaches have surfaced. Certain techniques take into account the nonlinear association between assets and the features of non-normal distribution. Examples of such techniques include the Value at Risk (VaR) model, Conditional Value at Risk (CVaR) model, and Omega model [2-4]. These models provide more choice and flexibility in taking into account investors' different attitudes and preferences for risk [5].

In some portfolio-based studies, many scholars focus on selected stocks. Selecting high-quality stocks for investment portfolios is an important factor in improving portfolio performance [6]. Some

studies have screened high-quality stocks through multi-factor performance, that is, comprehensive consideration of stock fundamentals, market value, and technical indicators. Based on this, investment portfolios can effectively increase the rate of return [7-8]. Some studies combine classic portfolio strategies, such as pair trading, by selecting stocks with different trends in the same industry, setting buying and shorting criteria to select stocks to build a portfolio [9-10]. The advantage of this method is that it can use the relative value relationship between stocks to achieve portfolio optimization in different market environments. There are also some studies that combine machine learning algorithms with investment portfolios, first use machine learning algorithms to predict future stock prices and build investment portfolios based on the predicted prices. By analyzing and learning from a vast amount of historical data, these algorithms can improve the accuracy of predicting future market trends. Incorporating these predictions into portfolio optimization can lead to more effective investment decision-making [11-12]. At the same time, the reinforcement learning method also shows potential in the construction of portfolio strategies. Through interactive learning with the environment, the agent can make corresponding trading decisions according to market changes [13].

The research in this paper is to combine the deep learning algorithm (LSTM, GRU, CRNN, TCN) with portfolio construction methods (Mean-Variance, Mean-CVaR) to build the portfolio strategies, then select stocks from different industries in the US stock market, and evaluate the effectiveness of the strategies based on cumulative returns. The innovation of this paper mainly lies in two aspects. One is that CRNN and TCN are rarely used in existing research. This paper combines the prediction advantages of deep learning with portfolio models to provide a variety of ideas for portfolio construction; secondly, existing literature mostly uses Mean-Variance in portfolio construction. This paper also introduces Mean-CVaR model and optimizes the MV model through factor score constraints.

2. Data

In order to better diversify risks and respond to market changes, this paper selects six stocks from different industries such as technology industry, financial industry, medical industry, energy industry, and catering industry in the U.S. stock market: 'AAPL', 'GS', 'JNJ', 'KO', 'XOM', 'MCD'. The stock data comes from Yahoo Finance and is imported through `yfinance` in python. To validate the strategies' effectiveness, this research examines stock transactions in two distinct market environments: bull market (2011/01/01—2013/06/30) and bear market (2020/01/01—2022/06/30). The training set comprises the first 80% of the data, while the test set consists of the remaining 20% of the data. This research employs four deep learning algorithms to predict the adjusted close of each stock on the last 20% of trading days. Subsequently, Mean-Variance and Mean-CVaR methods are utilized to construct portfolios. The performance of these strategies is then compared across different real market environments.

The Table 1 below shows that stocks exhibit higher standard deviation during the bear market, indicating that the stock price has higher volatility and may face higher risks. Skewness analysis reveals a leftward skew in the distribution of stock prices during the bear market, indicating a higher proportion of lower values compared to a normal distribution.

Table 1: The summary of data.

2011/01/01—2013-06-30						
stock	mean	std	min	max	skewness	kurtosis
AAPL	14.18	3.25	9.57	21.4	0.44	-1.09
GS	104.11	19.66	71.71	141.68	0.18	-1.26
JNJ	49.92	6.32	40.63	67.03	1.17	0.54
KO	25.42	2.54	21.09	31.21	0.3	-0.89
XOM	53.44	4.09	42.49	60.58	-0.39	-0.65
MCD	66.21	6.88	51.39	78.52	-0.46	-0.63
2020/01/01—2022/06/30						
stock	mean	std	min	max	skewness	kurtosis
AAPL	124.77	31.82	54.92	180.43	-0.39	-0.77
GS	279.5	78.37	125.11	404.78	-0.16	-1.43
JNJ	150	14.59	101.98	179.75	-0.2	-0.69
KO	50.43	6.13	33.97	63.82	0.14	-0.6
XOM	52.5	16.9	26.34	101.09	0.71	-0.18
MCD	212.8	26.63	127.29	260.88	-0.45	-0.35

3. Models

3.1. Four Deep Learning Algorithms

3.1.1.LSTM

Long Short-Term Memory is a specialized type of Recurrent Neural Network (RNN) that incorporates three distinct gates: forget gate (selectively discards unnecessary historical information), input gate (selectively adds certain historical information to the memory cell), and output gate (selectively passes certain historical information from the memory cell to the output). These gates allow LSTM to choose which historical stock price information to retain or discard over time and update new information. In stock price prediction, LSTM can predict future stock price trends by learning long-term dependencies and important features of historical stock price data.

$$\text{Forget Gate: } f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \quad (1)$$

$$\text{Input Gate: } i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad (2)$$

$$\text{Cell State Update: } \tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \quad (3)$$

$$\text{New Cell State: } C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \quad (4)$$

$$\text{Output Gate: } o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \quad (5)$$

$$\text{Hidden state: } h_t = o_t \odot \tanh(C_t) \quad (6)$$

$$\text{Output: } y_t = \text{Activation}(W_y \cdot h_t + b_y) \quad (7)$$

In the equations, the Sigmoid function σ is applied to a linear combination of the previous hidden state (h_{t-1}) and the current input (x_t), with corresponding weight and bias terms (W and b). The resulting output is then element-wise multiplied (represented by the symbol \odot) by the output of the hyperbolic tangent function (\tanh) applied to another linear combination of the same inputs, with

different weight and bias terms. The notation $[h_{t-1}, x_t]$ indicates that the previous hidden state and current input are concatenated together and treated as a single input.

3.1.2. GRU

Gated Recurrent Unit is a type of neural network that is designed to improve upon traditional recurrent neural networks. It achieves this by introducing two gating mechanisms - the reset gate (selectively forgets outdated and irrelevant stock price information) and the update gate (selectively merges stock price inputs into the current state) - which help the network selectively remember or forget information from previous time steps. In the context of stock price forecasting, GRU has shown promise in accurately predicting future trends by leveraging the long-term dependencies in historical stock price data.

$$\text{Reset Gate: } r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \quad (8)$$

$$\text{Update Gate: } z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \quad (9)$$

$$\text{New State: } \tilde{h}_t = \tanh(W_h \cdot [r_t \odot h_{t-1}, x_t]) \quad (10)$$

$$\text{Hidden state: } h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \quad (11)$$

$$\text{Output: } y_t = \text{Activation}(W_y \cdot h_t + b_y) \quad (12)$$

where σ represents the Sigmoid function, The element-wise multiplication operation is represented by the symbol \odot , the weight matrix (W) is used to transform the input at each time step, then $[h_{t-1}, x_t]$ includes the previous hidden state (h_{t-1}) and the current input (x_t)

3.1.3. CRNN

The core idea of Convolutional Recurrent Neural Network model is to establish a seamless combination between CNN and RNN, so as to simultaneously use CNN to extract local spatial features and RNN to model sequence relationships. This combination enables the CRNN model to efficiently handle sequence data with temporal and spatial correlations.

CNN part:

$$\text{convolutional layer: } c_i = f(W_i * x + b_i) \quad (13)$$

$$\text{pooling layer(optional): } p_i = \text{Pooling}(c_i) \quad (14)$$

RNN part:

$$\text{RNN layer: } h_t = \text{RNN}(h_{t-1}, p_t) \quad (15)$$

$$\text{bidirectional RNN(optional): } h_t = \text{BiRNN}(h_{t-1}, p_t) \quad (16)$$

Fully connected layer and output:

$$\text{fully connected layer: } z = W_o * h_T + b_o \quad (17)$$

$$\text{output layer: } y = \text{Activation}(z) \quad (18)$$

This neural network architecture utilizes convolutional layers to generate output maps (c_i) using activation functions (f) and corresponding weight and bias items (W_i and b_i). These output maps are then processed by pooling layers to produce output features (p_i). The output from the pooling layer

serves as the input for a recurrent neural network (RNN), such as LSTM or GRU, which employs a hidden state (h_t) to capture temporal dependencies in the input data. The RNN layer's output is further processed by a fully connected layer with weights and biases (W_o and b_o), along with an appropriate activation function (Activation). The final network output is denoted as y .

3.1.4.TCN

Temporal Convolutional Network draws inspiration from convolutional neural networks, and the use of convolutional layers can effectively capture the temporal relationship and local patterns in sequence data, so as to realize the modeling and prediction of sequence data. In stock price forecasting, the TCN model can capture long-term dependencies and local patterns in time series data to help analyze stock price trends and fluctuations.

$$h_i = g(W_j \cdot x[i:i+k-1] + b_j) \quad (19)$$

$$h^{(0)} = x \quad (20)$$

$$h^{(l)} = g(W^{(l)} \cdot h^{(l-1)} + b^{(l)}) + h^{(l-1)} \quad (21)$$

$$h^{(l)} = g(W^{(l)} \cdot h^{(l-1)} + b^{(l)}) + h^{(l-1)} + h^{(0)} \quad (22)$$

This neural network architecture involves convolutional layers that produce output feature maps (h_i) using convolutional kernels (W_j represents the weight) applied to subsequences of the input sequence ($x[i:i+k-1]$). g is nonlinear activation function (such as ReLU), b_j is a bias term, the index of the convolutional layer is denoted by l , and $h^{(0)}$ represents the input sequence, ($h^{(l)}$) is used as the output to the next layer.

3.2. Two Portfolio Models

Mean-Variance (MV) and Mean-CVaR (MC) methods share a common objective of balancing risks and returns in investment decision-making, and construct investment portfolios with good risk-return characteristics. For an investment portfolio, its expected return is:

$$E(R_p) = \sum_{i=1}^n w_i \cdot \mu_i \quad (23)$$

Subject to

$$\sum_{i=1}^n w_i = 1 \quad (24)$$

$$0 \leq w_i \leq 1 \quad \text{for } i = 1, 2, \dots, n \quad (25)$$

where the weight assigned to each asset in the portfolio is denoted by w_i , and the expected return of each asset is represented by μ_i

Mean-Variance model, introduced by Harry Markowitz in the 1950s, aims to identify the optimal portfolio allocation by minimizing the portfolio's variance while maximizing the Sharpe ratio [1]. This approach considers both the expected returns (mean) and the level of risk (variance) associated with each asset in the portfolio. The portfolio variance σ_p^2 and Sharpe Ratio is:

$$\sigma_p^2 = \sum_{i=1}^n \sum_{j=1}^n W_i \cdot W_j \cdot \sigma_i \cdot \sigma_j \cdot \rho_{ij} \quad (26)$$

$$\text{Sharpe Ratio} = \frac{E(R_p) - R_f}{\sigma_p} \quad (27)$$

where ρ_{ij} is the covariance of i and j , and R_f is risk free rate.

Mean-CVaR model focuses on the loss exceeding the confidence level and controls the conditional value at risk within a certain level [3].

$$\text{CVaR}_p = \frac{1}{1-\alpha} \int_0^\alpha \text{VaR}_p(q) dq \quad (28)$$

where CVaR_p represents the CVaR of the portfolio, α is the confidence level, and $\text{VaR}_p(q)$ represents the VaR of the portfolio at the quantile q .

4. Results

4.1. Prediction Evaluation

This study evaluates the performance of different models in predicting stock adjusted close by using two commonly metrics - mean squared error (MSE) and mean absolute error (MAE). This part is able to identify which models perform the best in predicting stock adjusted close and provide valuable insights for future research and investment decision-making.

$$\text{MSE} = \frac{1}{n} \sum_{t=1}^n (y_t - \hat{y}_t)^2 \quad (29)$$

$$\text{MAE} = \frac{1}{n} \sum_{t=1}^n |y_t - \hat{y}_t| \quad (30)$$

The data in Table 2 indicates that CRNN and TCN have lower MSE and MAE compared to LSTM and GRU. Furthermore, the error gap between the models is larger in the bull market forecast, while it is smaller in the bear market forecast. Overall, TCN outperforms other models in stock price prediction, while LSTM exhibits larger prediction errors.

Table 2: The Performance of different models.

		LSTM	GRU	CRNN	TCN
2013/02/14—2013-06-30					
MSE	mean	1.11×10^{-3}	1.04×10^{-3}	6.65×10^{-4}	4.03×10^{-4}
	std	5.04×10^{-4}	4.05×10^{-4}	2.54×10^{-4}	2.02×10^{-4}
MAE	mean	2.46×10^{-2}	2.41×10^{-2}	1.90×10^{-2}	1.47×10^{-2}
	std	6.35×10^{-3}	5.43×10^{-3}	4.19×10^{-3}	4.31×10^{-3}
2022/02/14—2022/06/30					
MSE	mean	9.42×10^{-4}	8.83×10^{-4}	4.82×10^{-4}	3.59×10^{-4}
	std	2.69×10^{-4}	2.44×10^{-4}	1.34×10^{-4}	2.83×10^{-4}
MAE	mean	2.23×10^{-2}	2.16×10^{-2}	1.53×10^{-2}	1.39×10^{-2}
	std	3.57×10^{-3}	3.58×10^{-3}	2.05×10^{-3}	5.25×10^{-3}

4.2. Portfolio Construction

The data predicted by different deep learning models is used to construct the investment portfolio, and the weight distribution of each asset is obtained. After that, the effect of the portfolio is evaluated in the real market environment within the transaction date. This part implements the buy-and-hold strategy, that is, after determining the weight of each stock, hold it until the end of the maturity date, without considering transaction costs, leverage, and short selling.

4.2.1. Bull Market

The cumulative return of the S&P 500 during the specified time period is 1.057. The first sub-graph demonstrates that the cumulative returns of the four deep learning algorithms combined with mean-variance (MV) strategies surpass that of the S&P 500, indicating their ability to outperform the market. Among these strategies, GRU+MV and TCN+MV exhibit higher cumulative returns of 1.1525 and 1.1517, respectively, surpassing the other two strategies. In the second sub-graph, the cumulative return of GRU+MC is comparable to that of the S&P 500, while the TCN+MC strategy performs the best, exhibiting a higher cumulative return of 1.091 compared to the S&P 500. Overall, the sub-graphs illustrate that given a consistent prediction result, the MV strategies demonstrates superior performance compared to the MC strategies (See Figure 1).

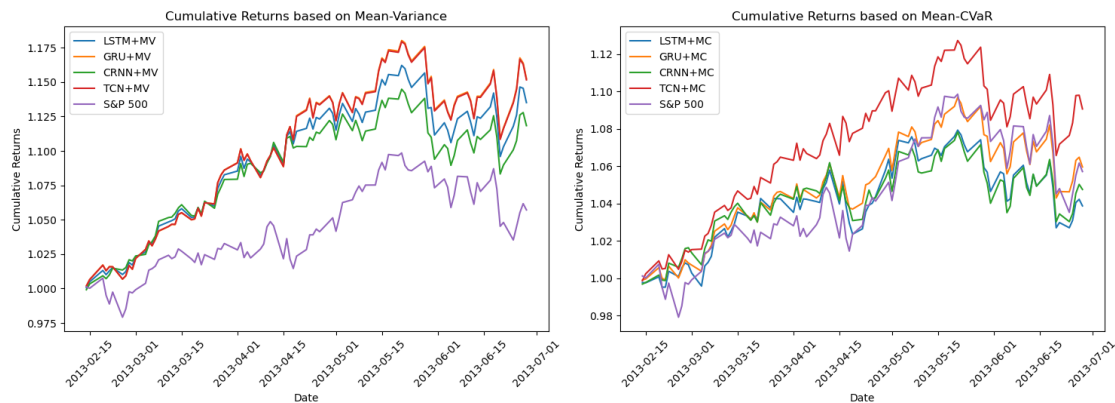


Figure 1: Portfolio performance during bull market.

4.2.2. Bear Market

The cumulative return of S&P 500 over the time period is 0.848. As can be seen from the first sub-graph, the cumulative returns of the four deep learning algorithms combined with MV strategies surpass 1 and significantly outperform the S&P 500. This indicates that these four strategies exhibit market-beating performance and yield higher returns. Numerically, the cumulative returns of the four strategies are relatively close, all around 1.1. In the second sub-graph, the cumulative returns of the four deep learning algorithms combined with MC strategies exceed that of the S&P 500. Among them, the CRNN+MC strategy achieves the highest cumulative return of 1.06. In general, while all the investment strategies produce better cumulative returns than the benchmark index, the strategies based on the MV model achieved the highest returns (See Figure 2).

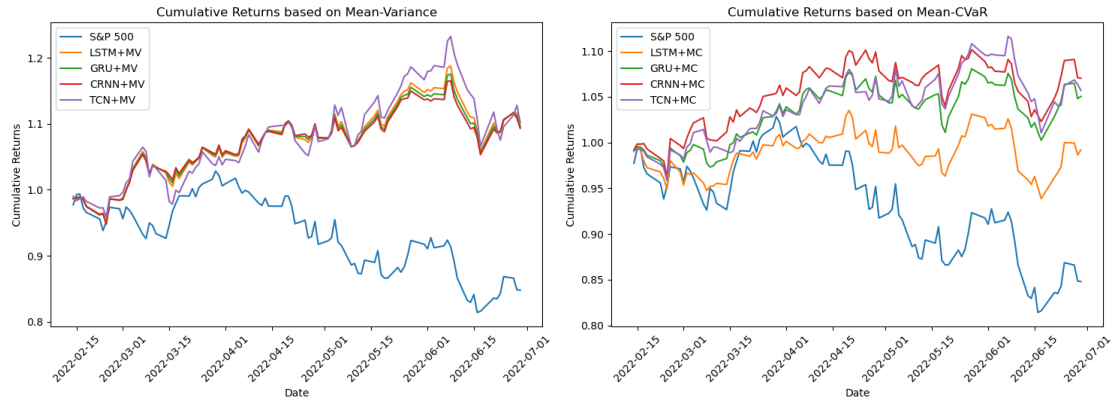


Figure 2: Portfolio performance during bear market.

4.3. Factor Scores

Based on the results, the use of MV in portfolio construction yields better results than MC when prediction outcomes are fixed. This paper extends the combination of four deep learning algorithms with the MV model and introduces score constraints for portfolio optimization. The selected factors include size factor (market capitalization), value factors (PE, PS), volatility factor (stock price volatility), momentum factors (RSI, daily trading volume), and profitability factor (daily returns) [14]. Based on the data collected during the training period, factors that have the highest positive impact on stock returns are assigned a score of 6, while the scores decrease sequentially in descending order. The lowest score assigned is 1. The final score for each stock is calculated as the average score, representing its overall performance.

Let vector s represent the scores of all stocks, and vector w represent the weights assigned to the stocks. The constraint T is introduced to ensure that the portfolio score exceeds the target constraint. The following condition is incorporated into the optimization of the MV model:

$$w^T \cdot s \geq T \quad (31)$$

Different T values will bring different investment returns. In a bull market, lower T score constraints result in higher cumulative returns. When the minimum T is set at 2.5, the cumulative return of the portfolio for the four strategies is 1.1525, higher than that of the S&P 500, 1.0572. In the bear market, all five T value constraints lead to cumulative returns for the investment portfolio that exceed the S&P 500's performance. The higher the constraint on the T value, the greater the cumulative return. Specifically, with a minimum T of 4.5, the portfolio achieves a cumulative return close to 1.13.

In conclusion, incorporating factor score constraints into the deep learning algorithm + MV framework allows for the identification of stocks with higher potential returns, leading to optimized investment portfolios and superior overall returns (See Table 3 and Figure 3).

Table 3: Optimization with integrated factor scores.

	LSTM+MV	GRU+MV	CRNN+MV	TCN+MV
2013/02/14—2013-06-30				
min T=2.5	1.1525	1.1525	1.1525	1.1525
min T=3	1.1187	1.1293	1.1081	1.1246
min T=3.5	1.098	1.097	1.0881	1.0918
min T=4	1.0639	1.0639	1.0637	1.0563
min T=4.5	1.0306	1.0306	1.0302	1.0208
2022/02/14—2022/06/30				
min T=2.5	1.0492	1.0492	1.0492	1.0492
min T=3	1.0722	1.0728	1.0727	1.0733
min T=3.5	1.0989	1.0992	1.0998	1.097
min T=4	1.1182	1.1182	1.1182	1.114
min T=4.5	1.1267	1.1267	1.1267	1.1257

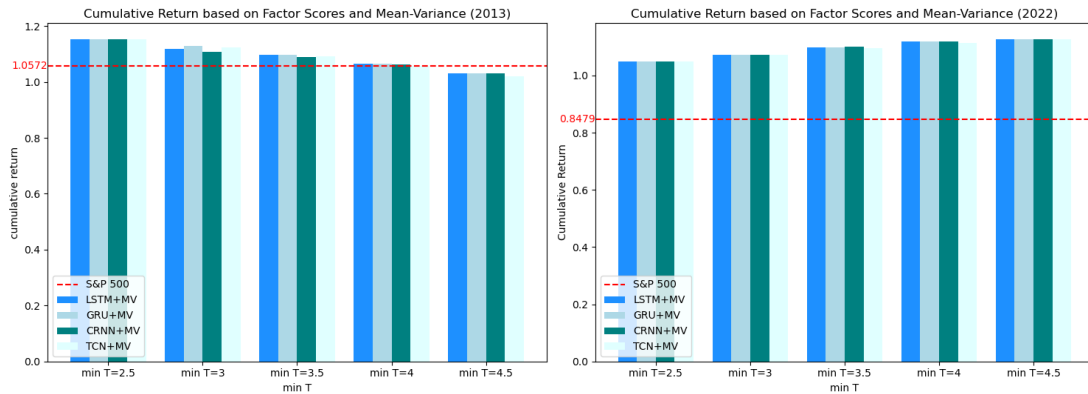


Figure 3: Cumulative returns under factor scores constraints.

5. Conclusion

This article focuses on portfolio construction, combining deep learning algorithms with classic portfolio strategies to construct different portfolio strategies. To assess the effectiveness of the strategy, this paper utilizes market data from both bullish and bearish market conditions, and compares the cumulative return of the portfolio with S&P 500 in a real market environment, and finds that the cumulative returns the strategies combined with the MV are better than those combined with MC. On this basis, this paper selects multiple factors for scoring, and adds factor scores as constraints to the process of portfolio optimization. The results show that the cumulative return has been steadily improved in both bull and bear markets.

This article only considers the investment strategy of buy-and-hold and does not consider the strategy of dynamic weight adjustment as the market changes. At the same time, this article does not consider factors such as transaction costs. These issues are directions for further research.

References

- [1] Markowitz, H. (1952) *Portfolio selection*. *The Journal of Finance*, 7, 77–91.
- [2] Alexander, G. J., and Baptista, A. M. (2002) *Economic implications of using a mean-VaR model for portfolio selection: A comparison with mean–variance analysis*. *Journal of Economic Dynamics and Control*, 26, 1159–1193.
- [3] Rockafellar, R. T., and Uryasev, S. (2000) *Optimization of conditional value-at-risk*. *Journal of Risk*, 2, 21–42.

- [4] Kapsos, M., Christofides, N., and Rustem, B. (2014). Worst-case robust Omega ratio. *European Journal of Operational Research*, 234, 499–507.
- [5] Xu, W.J., Huang, J.L., Fu, Z. N., and Zhang, W. G. (2022) Research on Black-Litterman portfolio model based on financial text sentiment mining—Evidence from the posting text of eastmoney stock forum and the A share market. *Operations Research Transactions*, (04),1-14.
- [6] Wang, W., Li, W., Zhang, N., and Liu, K. C. (2020) Portfolio formation with preselection using deep learning from long-term financial data. *Expert Systems with Applications*, 143, 113042.
- [7] Li, R.Y., and Ye, Z.Q. (2023) Prediction of Fund Returns Based on Machine Learning. *Statistics & Decision* (11),156-161.
- [8] Bessler, W., Taushanov, G., and Wolff, D. (2021) Factor investing and asset allocation strategies: a comparison of factor versus sector optimization. *Journal of Asset Management*, 22(6), 488–506.
- [9] Luo, J., Lin, Y., and Wang, S. (2022) Intraday high-frequency pairs trading strategies for energy futures: evidence from China. *Applied Economics*, 1-15.
- [10] Du, J. (2022) Mean–variance portfolio optimization with deep learning based-forecasts for cointegrated stocks. *Expert Systems with Applications*, 201, 117005.
- [11] Ma, Y. L., Han, R. Z., and Wang, W. Z. (2020) Prediction-based portfolio optimization models using deep neural networks. *IEEE Access*, 8, 115393-115405.
- [12] Zhang, N., Yan, S.B., and Fan, D. (2022) Yield Prediction and Portfolio Model Based on Deep Learning. *Statistics & Decision*, (23),48-51.
- [13] Mang, J., Xie, L., and Xu, H. J. (2023) Integrated deep reinforcement learning portfolio model. *Journal of Computer Applications*.
- [14] Xie, H. L. and Hu, D. (2017) The Application of Multi-factor Quantization Model in Portfolio: The Comparative Research on LASSO and Elastic Net. *Journal of Statistics and Information* (10),36-42.