HYPERGRAPH-BASED REINFORCEMENT LEARNING FOR STOCK PORTFOLIO SELECTION

Xiaojie Li ⋆*, Chaoran Cui* ⋆*, Donglin Cao* ⋆*, Juan Du* †*, Chunyun Zhang* ⋆

⋆School of Computer Science and Technology, Shandong University of Finance and Economics
†School of Finance, Shandong University of Finance and Economics

ABSTRACT

Stock portfolio selection is an important financial planning task that dynamically re-allocates the investments to stock assets to achieve the goals such as maximal profits and minimal risks. In this paper, we propose a hypergraph-based reinforcement learning method for stock portfolio selection, in which the fundamental issue is to learn a policy function generating appropriate trading actions given the current environments. The historical time-series patterns of stocks are firstly captured. Then, different from prior works ignoring or implicitly modeling stock pairwise correlations, we present a HyperGraph Attention Module (HGAM) in the portfolio policy learning, which utilizes the hypergraph structure to explicitly model the group-wise industry-belonging relationships among stocks. The attention mechanism is also introduced in HGAM that quantifies the importance of different neighbors regarding the target node to aggregate the information on the stock hypergraph adaptively. Extensive experiments on the real-world dataset collected from China's A-share market demonstrate the significant superiority of our method, compared with state-of-the-art methods in portfolio selection, including both online learning-based methods and reinforcement learning-based methods. The data and codes of our work have been released at `https://github.com/lixiaojieff/stock-portfolio`.

***Index Terms*—** Stock portfolio selection, reinforcement learning, portfolio policy, hypergraph attention networks

1. INTRODUCTION

Stock market is currently one of the most important investment channels for wealth accumulation. Many machine learning-based methods [1] have been developed to help investors make profitable trading decisions in stock markets. A straightforward idea is to predict the exact prices or movement directions of stocks in the future [2]. Apparently, the investment returns based on such models largely depend on the accuracy of stock price prediction, which, however, is not a trivial matter due to the noisy, stochastic, and chaotic nature of the financial market [3]. Moreover, how to convert price predictions to trading signals remains an open issue. On the other hand, stock portfolio selection has recently attracted extensive attention of investors and researchers. Portfolio Selection (PS) is a continuous decision-making process that dynamically re-allocates an amount of funds to stock assets to achieve the purpose of suppression of risks and maximization of rewards [4].

Generally, existing PS methods can be divided into two categories, i.e., Online Learning (OL)-based methods and Reinforcement Learning (RL)-based methods. Representative examples of OL-based methods include Constant Rebalanced Portfolio (CRP) [5], Anti Correlation (Anticor) [6], and Online Network Steps (ONS) [7], which aim to maximize the expected log-return in sequential decision-making. However, these methods need to manually extract stock features, which requires a considerable amount of domain knowledge and engineering skills. In another research line, RL-based methods resolve the continuous decision-making problem by learning a comprehensive policy function, which generates appropriate trading actions given the current environments to maximize reward signals. For example, Jiang et al. [3] proposed a RL framework with the Ensemble of Identical Independent Evaluators (EIIE) topology, in which the policy function is implemented by a Convolutional Neural Network (CNN) [8], a vanilla Recurrent Neural Network (RNN) [9], and a Long Short-Term Memory (LSTM) [10], respectively. Intuitively, the portfolio policy learning is cast as a time series modeling task in [3], for which only the temporal features of individual stocks are exploited, but the complex correlations between stocks are ignored. To solve this issue, Xu et al. [11] presented a Relation-Aware Transformer (RAT) to learn all stock pairwise correlations after capturing sequential patterns for PS. However, implicitly modeling the underlying correlations between stocks is a rather tough problem.

At present, little work has explicitly introduced the relationship information between stocks into the portfolio policy learning. The rise of graph representation learning [12] inspired us to construct a graph containing all stocks, with stocks as nodes and relationships between stocks as edges, and then further apply node embedding methods on the graph to learn stock representation. However, we notice that stocks

Chaoran Cui is the corresponding author.

are often correlated through group-wise relationships instead of pairwise ones as described in previous work [13, 14]. For example, multiple stocks in the same industry may have similar intrinsic attributes [15]. Hypergraph is a generalization of graph, in which a hyperedge can connect any number of vertices. It has proven to be effective in a wide range of applications, including image classification [16], object segmentation [17], and multimodal learning [18]. Recently, Hypergraph Convolutional Networks (HGCNs) [19] has been proposed to learn the stock representation considering the group-wise relationship information [20]. However, HGCNs treat the stock neighbors in a hyperedge equally, resulting in inaccurate information propagation to the target node.

In this paper, we introduce a HyperGraph Attention Module (HGAM) to the policy function learning for stock portfolio selection. To our knowledge, this is the first work that utilizes the hypergraph structure to model the stock group-wise relationships and treats them as an injection of explicit knowledge into PS issues. Specifically, after modeling the stock temporal patterns based on LSTM, we deploy HGAM to model the group-wise relationships among stocks for PS. Remarkably, HGAM introduces an attention mechanism to augment the HGCNs, thereby quantifying the importance of different neighbors of the target node to aggregate the information on the stock hypergraph adaptively. In this way, HGAM can explicitly model the stock correlations to guide more profitable portfolio selection. Extensive experiments on real-world dataset collected from China's A-share market demonstrate the significant superiority of our method, compared with both state-of-the-art OL-based methods, as well as RL-based methods for PS.

2. PROBLEM FORMULATION

Input Information. Given a set of n stocks $\mathcal{N}$, we collect the historical price records over a lookback window of m trading days for each stock, i.e., $\mathcal{X} = \{\boldsymbol{X}_1, \boldsymbol{X}_2, \ldots, \boldsymbol{X}_m\}$, where $\boldsymbol{X}_t = [\boldsymbol{x}_1^t, \boldsymbol{x}_2^t, \ldots, \boldsymbol{x}_n^t]^T \in \mathbb{R}^{n \times d}$ and $\boldsymbol{x}_i^t$ is d-dimensional price features of stock i at the t-th trading day.

Moreover, we introduce the hypergraph structure to explicitly model the group-wise industry-belonging relationships among stocks. Let $\mathcal{G} = (\mathcal{N}, \mathcal{E})$ be the industry-belonging hypergraph, where $\mathcal{N}$ is the node set, and $\mathcal{E}$ is the set of hyperedges, each of which connects multiple stocks belonging to a specific industry.

Reinforcement Learning. The goal of reinforcement learning (RL) is to find an optimal strategy that allows an agent to take action to obtain as many rewards as possible from the current environment. This process can be defined as a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{R})$. Specifically, the agent observes a state $\boldsymbol{s}_t = (\boldsymbol{X}_t, \mathcal{G}, \boldsymbol{a}_{t-1}) \in \mathcal{S}$, and takes action $\boldsymbol{a}_t = \pi(\boldsymbol{s}_t) \in \mathcal{A}$. Afterwards, the agent could receive a reward $r_t \in \mathcal{R}$. Concretely, the action is assigned by a portfolio vector $\boldsymbol{a}_t = [a_{t,1}, a_{t,2}, \ldots, a_{t,n}]^T \in \mathbb{R}^n$, where $a_{t,i}$ indicates the invested wealth proportion regarding stock i at the t-th trading day and $\sum_{i=1}^{n} a_{t,i} = 1$. The key challenge in RL is how to design the policy function π, which determines the action $\boldsymbol{a}_t$ given the current state $\boldsymbol{s}_t$ and aims at maximizing the accumulated reward $\mathcal{R}$.

3. FRAMEWORK

In practice, both the historical time-series patterns and the potential relationships between stocks are significant for decision-making in stock portfolio selection. Therefore, it is necessary to simultaneously utilize these two types of information in the portfolio policy learning.

3.1. Temporal Pattern Modeling

In this paper, we use the LSTM network to capture the temporal dynamics of stocks from time-series data. Specifically, we input the historical sequential features $\boldsymbol{x}_i^t$ of stock i at the t-th trading day into the LSTM network, and treat the hidden state $\boldsymbol{h}_i$ as the sequential embedding of stocks, i.e.,

$$\boldsymbol{h}_i = LSTM(\boldsymbol{x}_i^1, \boldsymbol{x}_i^2, \ldots, \boldsymbol{x}_i^m) \tag{1}$$

where $\boldsymbol{h}_i \in \mathbb{R}^{d_t}$ and d_t denotes the temporal embedding size. The above process can be applied to each stock asset separately, and we thus yield the time-series representations of all n stocks, which are denoted as $\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_n$.

3.2. Hypergraph Attention Module

In addition, we consider how to explicitly inject the group-wise relationship information into the portfolio policy learning. As analyzed above, Hypergraph Convolutional Networks (HGCNs) [19] only transmit the relationship information between nodes equivalently, while ignoring the varying degrees of influence between different nodes. To address this problem, we propose a HyperGraph Attention Module (HGAM), which introduces an attention mechanism to replace the fixed standardized operation in HGCNs and achieves better neighbor information aggregation.

HGAM aims to learn the different importance of stock neighbors, and adaptively determines the optimal way of information propagation in the stock hypergraph. For stock i and its neighbor $j \in \mathcal{N}_i$, we quantify the degree to which i is close to j based on their time-series embeddings, i.e.,

$$d(\boldsymbol{h}_i, \boldsymbol{h}_j) = \delta\left(\boldsymbol{b}^T \left[\boldsymbol{P}\boldsymbol{h}_i, \boldsymbol{P}\boldsymbol{h}_j\right]\right), \tag{2}$$

where $\boldsymbol{P}$ is a projection matrix to be learned, $\boldsymbol{b}$ is a shared attention vector, $[\cdot]$ denotes the concatenation operation, and $\delta(\cdot)$ denotes a nonlinear activation function like LeakyReLU. Then, we normalize the importance between node pairs to get the neighbor weight α_{ij} :

$$\alpha_{ij} = \frac{\exp\left(d(\boldsymbol{h}_i, \boldsymbol{h}_j)\right)}{\sum_{k \in \mathcal{N}_i} \exp\left(d(\boldsymbol{h}_i, \boldsymbol{h}_k)\right)}. \tag{3}$$

Note that the weight coefficient α_{ij} is asymmetric which means stocks i and j make different contributions to each other. Finally, we explicitly inject the relationship information into the portfolio policy learning in the sense that reformulating the embedding of stock i by aggregating all its neighbors' embeddings with the corresponding weights:

$$\boldsymbol{h}_i' = \delta\left(\sum_{j\in\mathcal{N}_i}\alpha_{ij}\boldsymbol{P}\boldsymbol{h}_j\right). \tag{4}$$

Denote by the output of HGAM $\boldsymbol{H}' = [\boldsymbol{h}_1', \boldsymbol{h}_2', \ldots, \boldsymbol{h}_n']^T \in \mathbb{R}^{n\times d_r}$, where d_r denotes the relational embedding size.

3.3. Decision-making Layer

Following [11], we introduce short sale into the decision making, and perform three independent softmax fully-connected output heads. Concretely, one head outputs an initial portfolio vector $\hat{\boldsymbol{a}}_t = FC(\boldsymbol{H}'||\boldsymbol{a}_{t-1})$, where FC is a fully-connected layer, $\boldsymbol{a}_{t-1}$ represents the portfolio vector at the previous time, and $||$ denotes the concatenate operation. Analogously, another head outputs a short sale vector $\hat{\boldsymbol{a}}_t^s$, while the last one outputs a reinvestment vector $\hat{\boldsymbol{a}}_t^r$. As such, the final portfolio vector is decided by $\boldsymbol{a}_t = \hat{\boldsymbol{a}}_t - \hat{\boldsymbol{a}}_t^s + \hat{\boldsymbol{a}}_t^r$. It still satisfies the requirement that the sum of the elements of $\boldsymbol{a}_t$ is equal to 1. We follow the prior research [21] to adopt a recursive mechanism, which concatenates the portfolio vector from the previous time $\boldsymbol{a}_{t-1}$ into the current feature maps $\boldsymbol{H}'$. In this way, the decision-making layer considers the previous action, so that it can be learned to avoid the heavy transaction costs of frequently adjusting portfolio vectors.

3.4. Learning with Reinforcement Learning

The policy network will be constructed by the above framework. An explicit reward function is given under the RL problem formulation. Here, we first denote the price change of all stocks by a price relative vector $\boldsymbol{y}_t := \frac{\boldsymbol{p}_t^c}{\boldsymbol{p}_{t-1}^c} \in \mathbb{R}^n$ regarding day t, where $\boldsymbol{p}_t^c$ indicates the close price. Considering the transaction remainder factor μ_t, i.e., the reduction of the portfolio value after paying commission fees [3], the average log-return of portfolios will be adjusted as:

$$R\left(\boldsymbol{s}_1, \boldsymbol{a}_1, \cdots, \boldsymbol{s}_{t_f}, \boldsymbol{a}_{t_f}\right) := \frac{1}{t_f}\sum_{t=1}^{t_f+1}\ln\left(\mu_t\boldsymbol{y}_t\cdot\boldsymbol{a}_{t-1}\right) \tag{5}$$

where t_f denotes the length of the whole portfolio selection process. We adopt the classical direct policy gradient algorithm [21] to train the policy network by maximizing the reward function.

4. EXPERIMENTS AND RESULTS

4.1. Experiments Setup

Datasets. In our experiment, we used a financial data API [1] to collect the historical price and industry relationship data of stocks from China's A-share market. Specifically, we selected the stocks that have more than 98% of all trading days price records between 01/04/2013 and 12/31/2019, obtaining 758 stocks. For each stock, we extracted its six price features of each day, including the open price, high price, low price, close price, trading volume, and trading value. Moreover, we introduced four technical indicators, i.e., 5-, 10-, 20-, and 30-day moving averages, to capture the past weekly and monthly trends. We grouped all stocks into 104 industry categories according to the Shenwan Industry Classification Standard[2]. The trading records of all stocks were chronologically split into three time periods in the ratio of 8:1:1 for training, validation, and testing, respectively.

Baseline. We compare our method with state-of-the-art methods, including (1) OL-based methods: CRP [5], Anticor [6], and ONS [7]; and (2) RL-based methods: EIIE-CNN [3], EIIE-LSTM [3], and RAT [11].

Measure Metrics. Following [3], we use three metrics to evaluate the performance. First, Accumulated Portfolio Value (**APV**) is the most intuitive measure by calculating the accumulated portfolio value over the period of testing, when considering the transaction cost $\mu_t = 0.25\%$. Mathematically, $APV = S_m = S_0\prod_{t=1}^{m}\left(\mu_t\boldsymbol{y}_t\cdot\boldsymbol{a}_{t-1}\right)$, where $S_0 = 1$ denotes the initial wealth. Besides, we take the risk factor into account, which can be assessed by the metrics of Sharpe ratio (**SR**) [22] and Maximum Drawdown (**MDD**) [23].

Implementation Details. All methods were implemented with Pytorch[3]. The number of hidden layers and cells per layer in the LSTM model is set to 2 and 32, respectively. The size of stock embeddings output by HGAM was set to 16. We set the initial learning rate to 10^{-3} for all layers. Our network was trained 600 epochs with the mini-batch Adam optimizer, where batch size is 32. The length of the historical price records m is set to 20.

4.2. Results and Analysis

4.2.1. Profitability

Table 1 reports the profitability comparison of different methods. To make it more intuitive, Fig. 1 plots the accumulated portfolio value curves for different methods. To observe the overall volatility of the stock market, a passive buy-and-hold trading strategy is also added as a benchmark for the comparison. In the buy-and-hold strategy, investors are assumed to purchase all stocks with the same proportion at the beginning

[1] https://tushare.pro/document/2

[2] http://www.swsindex.com/idx0530.aspx

[3] https://pytorch.org

Table 1. Profitability of different methods.

Algorithm	APV	SR	MDD
Buy-and-Hold	0.911	-0.304	0.376
CRP [5]	1.009	0.053	0.173
Anticor [6]	1.002	0.019	0.115
ONS [7]	0.998	-0.144	0.291
EIIE-CNN [3]	5.301	0.502	**0.052**
EIIE-LSTM [3]	7.084	<u>0.787</u>	0.105
RAT [11]	<u>11.489</u>	0.486	0.177
HGAM	**29.386**	**3.112**	<u>0.058</u>

The best result in terms of each metric is indicated in bold, and the second best one is underlined.

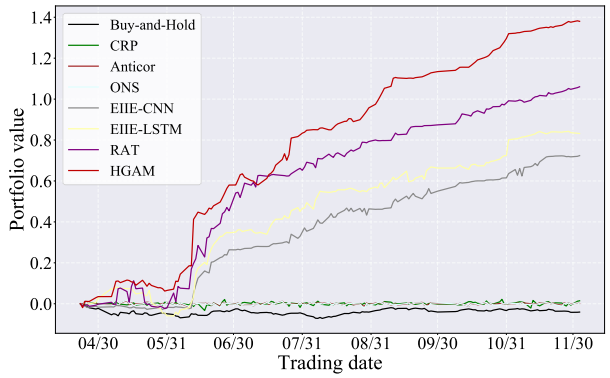

Fig. 1. Comparison of portfolio value of different methods during testing. The curves are plotted in log-10 scale.

of testing period, and hold them until they are sold at the end of testing period. From the results, we can make the following observations:

All portfolio selection methods beat the buy-and-hold strategy, indicating that portfolio selection is beneficial to helping investors to make profitable trading decisions. OL-based methods fail to perform well on account of the inability to learn the crucial sequential features for stocks. As for RL-based methods, the EIIE equipped with CNN or LSTM has shown significant progress. Moreover, RAT has also made remarkable gains in modeling temporal patterns and stock correlations via exploring specific attention modules. It proves the effectiveness and superiority of deep RL in PS. However, they are still inferior to the proposed method HGAM, since their policy functions suffer from the limited capabilities in explicitly modeling group-wise relationships.

Overall, HGAM outperforms the state-of-the-art OL-based methods as well as RL-based methods, which demonstrates the strong profitability of our method in PS. Especially in terms of APV, the final reward of HGAM is more than twice that of the runner-up. Also, we can see that HGAM shows a stronger power of risk management, that is, it can achieve more stable profits under the same risk. Therefore, we receive satisfying improvement by explicitly modeling

Table 2. Profitability performance of different relationship modeling modules.

Algorithm	APV	SR	MDD
GCN [2]	7.999	0.887	0.193
HGCN [19]	<u>13.568</u>	<u>1.736</u>	<u>0.098</u>
HGAM	**29.386**	**3.112**	**0.059**

the relationship information in the policy network. From such results, we can conclude that the industry-belonging relationships among stocks is important for PS because they may reveal macro market trends and contribute to reducing investment risk by adjusting portfolio diversification.

4.2.2. Ablation Studies

Ablation studies are also conducted to investigate the efficacy of the key relationship modeling in our approach. The results are reported in Table 2. Specifically, GCN model uses a LSTM network to encode the historical price features, and the results are then fed into a graph convolutional layer to learn based on the pairwise relationships between stocks. HGCN model only transforms the graph into a hypergraph convolution layer to model the group-wise relationships, and the rest is the same as HGAM. As can be seen, HGCN performs significantly better than GCN, suggesting the effectiveness of using the hypergraph structure to model the group-wise relationships rather than pairwise ones. In addition, HGAM still shows the best results, which indicates the attention mechanism is conducive to more accurately model the complicated group-wise industry-belonging relationships between stocks, thereby improving profitability.

5. CONCLUSION

In this paper, we explicitly introduce the collective relationships between stocks into policy learning for stock portfolio selection. Specifically, after modeling the stock temporal patterns based on LSTM, we deploy a HyperGraph Attention Module (HGAM), which serves as the policy network for PS to explicitly model the group-wise industry-belonging relationships among stocks. Extensive experiments demonstrate that HGAM offers a significant improvement in PS, maximizing the accumulated portfolio value while controlling risk and transaction costs.

6. ACKNOWLEDGEMENTS

This work is supported by the National Natural Science Foundation of China (62077033, 61701281), the Shandong Provincial Natural Science Foundation Key Project (ZR2020KF015), and the Fostering Project of Dominant Discipline and Talent Team of Shandong Province Higher Education Institutions.

7. REFERENCES

[1] Shunrong Shen, Haomiao Jiang, and Tongda Zhang, "Stock market forecasting using machine learning algorithms," *Department of Electrical Engineering, Stanford University, Stanford, CA*, pp. 1–5, 2012.

[2] Yingmei Chen, Zhongyu Wei, and Xuanjing Huang, "Incorporating corporation relationship via graph convolutional neural networks for stock price prediction," in *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*, 2018, pp. 1655–1658.

[3] Zhengyao Jiang, Dixing Xu, and Jinjun Liang, "A deep reinforcement learning framework for the financial portfolio management problem," *arXiv preprint arXiv:1706.10059*, 2017.

[4] Markowitz and H. Harry, "Portfolio selection: Efficient diversification of investment," *Journal of the Institute of Actuaries*, 1992.

[5] Thomas M Cover and David H Gluss, "Empirical bayes stock market portfolios," *Advances in Applied Mathematics*, vol. 7, no. 2, pp. 170–181, 1986.

[6] A. Borodin, R. El-Yaniv, and V. Gogan, "Can we learn to beat the best stock," *The Journal of Artificial Intelligence Research*, vol. 21, pp. 579–594, 2004.

[7] Amit Agarwal, Elad Hazan, Satyen Kale, and Robert E Schapire, "Algorithms for portfolio management based on the newton method," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 9–16.

[8] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[9] Mike Schuster and Kuldip K Paliwal, "Bidirectional recurrent neural networks," *IEEE Transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[10] Sepp Hochreiter and Jürgen Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[11] Ke Xu, Yifan Zhang, Deheng Ye, Peilin Zhao, and Mingkui Tan, "Relation-aware transformer for portfolio policy learning," in *Proceedings of the 29th International Joint Conference on Artificial Intelligence*, 2020, pp. 4647–4653.

[12] Thomas N Kipf and Max Welling, "Semi-supervised classification with graph convolutional networks," *arXiv preprint arXiv:1609.02907*, 2016.

[13] Fuli Feng, Xiangnan He, Xiang Wang, Cheng Luo, Yiqun Liu, and Tat-Seng Chua, "Temporal relational ranking for stock prediction," *ACM Transactions on Information Systems*, vol. 37, no. 2, pp. 1–30, 2019.

[14] Raehyun Kim, Chan Ho So, Minbyul Jeong, Sanghoon Lee, Jinkyu Kim, and Jaewoo Kang, "Hats: A hierarchical graph attention network for stock movement prediction," *arXiv preprint arXiv:1908.07999*, 2019.

[15] Chaoran Cui, Xiaojie Li, Juan Du, Chunyun Zhang, Xiushan Nie, Meng Wang, and Yilong Yin, "Temporal-relational hypergraph tri-attention networks for stock trend prediction," *arXiv preprint arXiv:2107.14033*, 2021.

[16] Heyuan Shi, Yubo Zhang, Zizhao Zhang, Nan Ma, Xibin Zhao, Yue Gao, and Jiaguang Sun, "Hypergraph-induced convolutional networks for visual classification," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 10, pp. 2963–2972, 2018.

[17] Yuchi Huang, Qingshan Liu, and Dimitris Metaxas, "Video object segmentation by hypergraph cut," in *Proceedings of the 2009 IEEE conference on computer vision and pattern recognition*. IEEE, 2009, pp. 1738–1745.

[18] Eun-Sol Kim, Woo Young Kang, Kyoung-Woon On, Yu-Jung Heo, and Byoung-Tak Zhang, "Hypergraph attention networks for multimodal learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 14581–14590.

[19] Yifan Feng, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao, "Hypergraph neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2019, pp. 3558–3565.

[20] Ramit Sawhney, Shivam Agarwal, Arnav Wadhwa, and Rajiv Ratn Shah, "Spatiotemporal hypergraph convolution network for stock movement forecasting," in *Proceedings of the 20th IEEE International Conference on Data Mining*, 2020, pp. 482–491.

[21] John Moody and Matthew Saffell, "Learning to trade via direct reinforcement," *IEEE transactions on neural Networks*, vol. 12, no. 4, pp. 875–889, 2001.

[22] William F Sharpe, "The sharpe ratio," *Journal of portfolio management*, vol. 21, no. 1, pp. 49–58, 1994.

[23] Malik Magdon-Ismail and Amir F Atiya, "Maximum drawdown," *Risk Magazine*, vol. 17, no. 10, pp. 99–102, 2004.