DeepTrader: A Deep Reinforcement Learning Approach for Risk-Return Balanced Portfolio Management with Market Conditions Embedding

Zhicheng Wang[1,2], **Biwei Huang**[3], **Shikui Tu**[1,2], **Kun Zhang**[3], **Lei Xu**[1,2]
[1]Department of Computer Science and Engineering, Shanghai Jiao Tong University
[2]Centre for Cognitive Machines and Computational Health (CMaCH), Shanghai Jiao Tong University
[3]Department of Philosophy, Carnegie Mellon University
{wwwangzhch, tushikui, leixu}@sjtu.edu.cn, biweih@andrew.cmu.edu, kunz1@cmu.edu

Abstract

Most existing reinforcement learning (RL)-based portfolio management models do not take into account the market conditions, which limits their performance in risk-return balancing. In this paper, we propose *DeepTrader*, a deep RL method to optimize the investment policy. In particular, to tackle the risk-return balancing problem, our model embeds macro market conditions as an indicator to dynamically adjust the proportion between long and short funds, to lower the risk of market fluctuations, with the negative maximum drawdown as the reward function. Additionally, the model involves a unit to evaluate individual assets, which learns dynamic patterns from historical data with the price rising rate as the reward function. Both temporal and spatial dependencies between assets are captured hierarchically by a specific type of graph structure. Particularly, we find that the estimated causal structure best captures the interrelationships between assets, compared to industry classification and correlation. The two units are complementary and integrated to generate a suitable portfolio which fits the market trend well and strikes a balance between return and risk effectively. Experiments on three well-known stock indexes demonstrate the superiority of *DeepTrader* in terms of risk-gain criteria.

Introduction

Portfolio management aims to allocate resources to obtain optimal returns while avoiding the risk at the same time. To overcome the weakness of human decision making, a great quantity of investment strategies have been proposed (Grinblatt, Titman, and Wermers 1995; Malkiel 1989; Jegadeesh and Titman 1993). These traditional strategies usually perform well in some cases but fail in others as they may not properly adapt to the changing market environment.

Machine learning was adopted to build the trading and portfolio system. For example, the system could be trained by RL (Moody et al. 1998) or by maximizing the expected profit via the so-called adaptive supervised learning decision networks (Xu and Cheung 1997; Hung, Cheung, and Xu 2003). The system could also be built by directly maximizing a portfolio objective, e.g., Sharpe ratio, where the temporal structures of stock returns were learned by a temporal factor analysis (TFA) model which is based on the arbitrage pricing theory (APT) in finance, and the portfolio weights are represented by an extended normalized radial basis function (ENRBF) (Chiu and Xu 2003, 2004).

Efforts have been made in deep reinforcement learning (DRL) for portfolio management with promising results (Deng et al. 2016; Jin and El-Saawy 2016; Jiang, Xu, and Liang 2017). Similar to games, portfolio management also interacts with the environment (financial market) and maximizes the cumulative rewards (returns). However, the abovementioned methods share some common limitations. First, they overlook the intrinsic connection and interplay among different stocks. The price fluctuation of one stock may strongly imply the trend of a set of related stocks. Recently, Wang et al. (2019) alleviated this limitation by introducing the self-attention mechanism (Vaswani et al. 2017) across different assets. Nevertheless, the stock interrelationships are computed based on the similarity between the input sequences along a local region, which may omit the interrelations over a long time span. Second, existing DRL methods usually neglect market conditions, such as stock indexes and how many stocks rise or fall, and then just treat the input financial signals independently from the rest of the market, leading to limited performance on the risk-return balancing.

In this paper, we propose a DRL-based model called *DeepTrader* for portfolio management. *DeepTrader* mainly includes two units to handle the problems of cross-asset interrelationship learning and risk-return balancing, respectively. One unit, which is called *asset scoring unit*, takes individual stock data as input and learns to represent them as "winner scores" for each asset. The winner score indicates how likely a stock is going to rise in the future. To better encode the interrelationships among all stocks, we construct a graph based on different dependence characterizations to capture interrelationships hierarchically, whether long or short, spatial or temporal. The other unit, known as *market scoring unit*, leverages the market sentiment indicators as input and then embeds financial situations as an indicator to adjust the proportion of long and short funds in every trading period. This dynamic adjustment allows the model to reduce the risk induced by complicated market ups and downs in a timely and effective manner. Moreover, while using the price-rising rate as the reward function in the asset

scoring unit, we adopt the negative of the maximum drawdown (MDD), which is a useful indicator of downside risk, as the reward function in the market scoring unit. The two units complement each other and are naturally integrated for portfolio generation.

Policy gradient is used to optimize the investment policy in an end-to-end manner. The optimization of discrete action space for asset scoring unit (choosing which stocks to invest) and continuous action space for market scoring unit (judge the financial condition) are integrated into one loss function. We conduct experiments on three well-known stock indexes. The results show that the investment policy optimized by *DeepTrader* fits the market trend well and strikes a good balance between the risk and return. *DeepTrader* outperforms both traditional trading strategies and other DRL-based ones, in terms of risk-gain criteria including annualized Sharpe ratio, Calmar ratio, and Sortino ratio.

Our contributions are summarized as follows:

i) We propose *DeepTrader*, a DRL-based method for risk-return balanced portfolio management. Market conditions are novelly taken into account as an independent profit-risk balancing module, while the cross-asset interrelationship extraction is enhanced by learning and using a graph structure to characterize interrelationships between stocks.

ii) Experiments on three stock indexes demonstrate the superiority of *DeepTrader* in balancing risk and return, especially in the period of subprime mortgage crisis and the recovery period. Ablation studies further confirm the effectiveness of the key components in *DeepTrader* and the effectiveness of using learned causal structure to encode the interrelationships between assets.

Related Works

Traditional Investment Strategies. Momentum trading (Hong and Stein 1999) and mean reversion (Poterba and Summers 1988) are two well-known traditional investment strategies. The former type selects investments based on their recent performance. It includes Cross Sectional momentum (Jegadeesh and Titman 2002), Buying-Winner-Selling-Loser (Jegadeesh and Titman 1993), and Time Series Momentum (Moskowitz, Ooi, and Pedersen 2012). The latter type purchases stock with price lower than long-mean and sells it with price higher than the average price. Traditional investment strategies usually perform well only in some specific situations.

RL in Portfolio Management. By virtue of the strong ability of deep learning on feature representation, recent models combine deep neural networks with RL in trading (Deng et al. 2016; Almahdi and Yang 2017; Jiang, Xu, and Liang 2017; Ye et al. 2020; Wang et al. 2019). For example, FDDR (Deng et al. 2016) consists of an RNN structure for feature learning and a RL part for self-taught reinforcement trading. Liu et al. (2020) enhanced the exploration-exploitation efficiency by combining RL and imitation learning techniques in their GRU-based neural network. The two above methods focus on giving the long, hold and short signal for single financial asset, lacking the ability to allocate funds among multiple assets. Later, Jiang, Xu, and Liang (2017) introduced various DNN structures combined with Deep Deterministic Policy Gradient to dynamically optimize cryptocurrency portfolios. Ye et al. (2020) extended this by incorporating heterogeneous data (e.g. news) to enhance robustness against environment uncertainty. However, above two works do not take short operation, which causes enormous losses in a bear market. Wang et al. (2019) solves this issue by introducing attention mechanism to model the relative price relations for buying winner and selling losers strategy.

Preliminary

Problem Setup

Portfolio management is a sequential decision-making process of allocating resources into a set of financial assets according to current market conditions, aiming to maximize the return while constraining risks. Such a decision-making process naturally fits into the framework of a Markov Decision Process $M = \langle \mathcal{S}, \mathcal{A}, P, R \rangle$, where $\mathcal{S}$ is the state space and $\mathcal{A}$ the action space. When action $a_t \in \mathcal{A}$ is executed, $s_t \in \mathcal{S}$ (the state at time t) changes according to the transition distribution $s_{t+1} \sim P(s_{t+1}|s_t, a_t)$. Subsequently, the agent receives a reward $r_t = R(s_t, a_t, s_{t+1})$. The agent's goal is to learn a policy π, which can cast an investment strategy $a \in \mathcal{A}$ over all stocks, to maximize the expected return $J = \mathbb{E}_{p(\tau)}[\sum_{t=1}^{T} \gamma^{t-1} r_t]$ over trajectories induced by the policy.

Trading Procedure

Consider a scenario that contains both long and short operations. At the end of $t-1$ holding period, a trader holds C_{t-1}^0 risk-free assets (cash) and $\boldsymbol{b}_{t-1}^+ = \{b_{t-1,1}^+, b_{t-1,2}^+, ..., b_{t-1,N}^+\} \in \mathbb{R}^N$ volume of stocks on a long position. In addition, the trader owes the stock brokerage $\boldsymbol{b}_{t-1}^- \in \mathbb{R}^N$ volume of stocks. Based on close price $\boldsymbol{P}_{t-1}^{(c)} \in \mathbb{R}^N$, long portfolio vector $\boldsymbol{\omega}_t^+$, short portfolio vector $\boldsymbol{\omega}_t^-$ and the total ratio of assets used for short position ρ_t, he/she will follow the steps below to finish the t period: 1) sell all stocks on a long position ($\boldsymbol{b}_{t-1}^+$) and get cash; 2) buy the stock ($\boldsymbol{b}_{t-1}^-$) borrowed at the beginning of $t-1$ period and return it to the stock brokerage; 3) mortgage stocks from a broker according to ρ_t and $\boldsymbol{\omega}_t^-$ and sell them immediately; 4) purchase based on long proportion $\boldsymbol{\omega}_t^+$.

Methodology

Overview of DeepTrader

An overview of *DeepTrader* is given in Figure 1. The input sequential signal $\boldsymbol{\mathcal{X}}_t = [\boldsymbol{x}_{t_1}, ..., \boldsymbol{x}_{t_K}] = \{\boldsymbol{\mathcal{X}}_t^a; \boldsymbol{\mathcal{X}}_t^m\}$ consists of stock indicators $\boldsymbol{\mathcal{X}}_t^a$ and market indicators $\boldsymbol{\mathcal{X}}_t^m$. The two parts are fed into asset scoring unit and market scoring unit, respectively. The asset scoring unit maps input signals $\boldsymbol{\mathcal{X}}_t^a$ and the graph structure $\boldsymbol{A}$ into winner scores $\boldsymbol{v}_t$. This unit consists of stacked spatial-TCN blocks with residual connections (He et al. 2016), with each spatial-TCN block constructed by a dilated causal convolution with spatial attention and graph convolution. On the other side, a novel

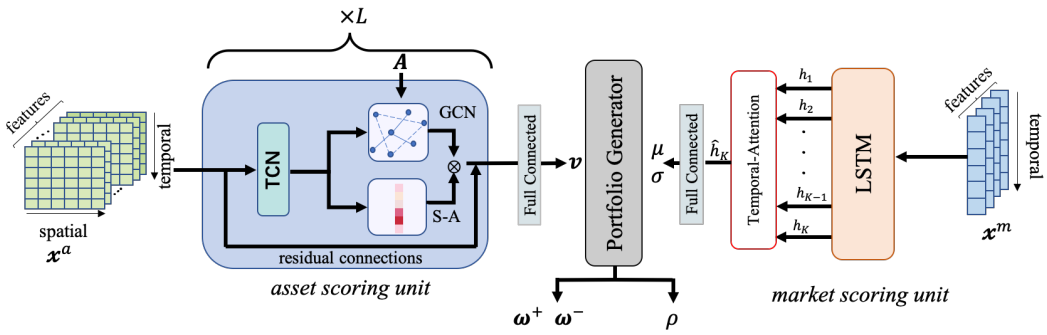

Figure 1: DeepTrader framework. The left part is the asset scoring unit consisting of L stacked spatial-TCN blocks with residual connections, which takes $\boldsymbol{\mathcal{X}}^a$ and $\boldsymbol{A}$ as inputs and outputs $\boldsymbol{v}$. The right part is the market scoring unit, which is responsible for translating $\boldsymbol{\mathcal{X}}^m$ into μ and σ, with $\widetilde{\rho} \sim N(\mu, \sigma^2)$. The third part is a portfolio generator which turns $\boldsymbol{v}$ and $\widetilde{\rho}$ into final investing decisions.

market scoring unit takes $\boldsymbol{\mathcal{X}}_t^m$ as input and turns it into parameters of the Gaussian distribution: mean μ and standard deviation σ, of $\widetilde{\rho} \sim N(\mu, \sigma^2)$. The third component is a portfolio generator which calculates the investment proportions $\boldsymbol{\omega}_t^+$, $\boldsymbol{\omega}_t^-$ and ρ_t according to $\boldsymbol{v}_t$ and $\widetilde{\rho}$. In the rest part of this section, we omit the subscript t for simplicity.

Asset Scoring Unit

The asset scoring unit contains three key components: temporal convolution layer, spatial attention mechanism, and graph convolution layer, which are used to capture temporal and spatial interrelations among stocks, both long and short.

Temporal Convolution Layer. We use the dilated temporal convolution layer (TCN) (Yu and Koltun 2015) to handle temporal relations in long range sequences. Opposed to RNN-based approaches, it facilitates parallel computation and alleviates the gradient exploration as well as the vanishing issue. We denote the input of l^{th} block by $\boldsymbol{H}^{l-1} \in \mathbb{R}^{C \times N \times K_{l-1}}$, where C is the dimension of hidden features, N the number of stocks, and K_{l-1} the temporal length of the $l-1^{th}$ block. After conducting TCN operation along the temporal dimension, we denote the output of this layer by $\hat{\boldsymbol{H}}^l \in \mathbb{R}^{C \times N \times K_l}$.

Spatial Attention Mechanism. To model short-term spatial properties, we adopt an attention mechanism (Feng et al. 2017) to adaptively search the correlations between stocks. The core idea is to assign different weights to each vector at each time step. These weights reflect the relativities between two different stocks. Specifically, given the output $\hat{\boldsymbol{H}}^l \in \mathbb{R}^{N \times C \times K_l}$ from the latest TCN layer (i.e., the TCN in l^{th} block), we compute a weight vector:

$$\boldsymbol{S}^l = \boldsymbol{V}_s \cdot \text{sigmoid}((\hat{\boldsymbol{H}}^l \boldsymbol{W}_1)\boldsymbol{W}_2(\boldsymbol{W}_3 \hat{\boldsymbol{H}}^{l\intercal(1,2)})^\top + \boldsymbol{b}_s), \quad (1)$$

where $\boldsymbol{W}_1 \in \mathbb{R}^{K_l}$, $\boldsymbol{W}_2 \in \mathbb{R}^{C \times K_l}$, $\boldsymbol{W_3} \in \mathbb{R}^C$, and $\boldsymbol{V}_s \in \mathbb{R}^{N \times N}$ are free parameters, $\boldsymbol{b}_s \in \mathbb{R}^{N \times N}$ is the bias vector, and the superscript $\intercal(1,2)$ denotes the transpose for the first two dimensions. Each entry is further normalized via softmax: $\boldsymbol{S}_{i,j}^l = \frac{\exp(\hat{\boldsymbol{S}}_{i,j}^l)}{\sum_{v=1}^N \exp(\hat{\boldsymbol{S}}_{i,v}^l)}$, to represent the correlation between node i and node j.

Graph Convolution Layer. While the performance of individual stocks has changing volatility, the overall performance of the industry usually better reflects the economic situation and the trend of social hot spots in the future. Previous works often fail to take this into consideration. To this end, we model the long-range relationships by graph convolution networks (GCNs) to better guide our decision making. GCNs give an essential operation to capture the dependence of nodes in a graph via message passing, and thus both the edge and neighborhood information are integrated into its state representation.

We consider the following ways to characterize the structural information between stocks: (1) Stock industry classifications. (2) Correlations between stock returns. (3) Partial correlations between stock returns. (4) Causal relationships, with the causal structure between stocks identified by the PC algorithm (Spirtes, Glymour, and Scheines 2001).

We notice that with stock industry classifications alone some dependence relations may be missed; for example, *Amazon.com* is classified to Retail Trade, but it also benefits from the dividends of mobile Internet. To avoid this issue, we formulate our graph convolution layers as (Wu et al. 2019):

$$\boldsymbol{Z}^l = \sum_{q=0}^{Q} \tilde{\boldsymbol{A}}^q \hat{\boldsymbol{H}}^l \boldsymbol{\Theta}_{1,q} + \tilde{\boldsymbol{A}}_c \hat{\boldsymbol{H}}^l \boldsymbol{\Theta}_2, \quad (2)$$

where $\hat{\boldsymbol{H}}^l$ denotes the input signals of the l^{th} block, $\tilde{\boldsymbol{A}} = \boldsymbol{A}/rowsum(\boldsymbol{A})$, and $\boldsymbol{\Theta}_1, \boldsymbol{\Theta}_2 \in \mathbb{R}^{K_l \times K_l}$ are learnable parameters in GCN. $\tilde{\boldsymbol{A}}_c$ is used to capture the correlation with $\tilde{\boldsymbol{A}}_c := SoftMax(\text{ReLU}(\boldsymbol{E}\boldsymbol{E}^\top))$, where $\boldsymbol{E} \in \mathbb{R}^N$ as a learnable parameter; it is initialized randomly and learned through gradient descent. Q is a hyperparameter to balance the information from $\tilde{\boldsymbol{A}}$ and $\tilde{\boldsymbol{A}}_c$.

For the structural information characterized by correlation, partial correlation, and the causal relation, the graph convolution layer is formulated as follows:

$$\boldsymbol{Z}^l = \sum_{q=0}^{Q} \tilde{\boldsymbol{A}}^q \hat{\boldsymbol{H}}^l \boldsymbol{\Theta}_1, \quad (3)$$

that is, the second part in Eq. (2) is no longer necessary, because the dependence information is already estimated from the data.

Furthermore, to integrate long-term and short-term correlations simultaneously and adjust the weight matrix dynamically, we accompany $\boldsymbol{Z}^l$ with spatial attention weight $\boldsymbol{S}^l$ and the final output of l^{th} block is rewritten as: $\boldsymbol{H}^l = \boldsymbol{S}^l \times \boldsymbol{Z}^l \oplus \boldsymbol{H}^{l-1}$, where $\oplus$ represents the residual connections (He et al. 2016). At the end of the asset scoring unit, we use a fully connected layer to translate the hidden state $\boldsymbol{H}^L$ into asset scores $\boldsymbol{v}$:

$$\boldsymbol{v} = \text{sigmoid}(\boldsymbol{W}_L \cdot \boldsymbol{H}^L + \boldsymbol{b}_L). \tag{4}$$

Market Scoring Unit

Unlike other types of signals, financial data contains large amount of unpredictable uncertainties due to the random gambling and emergencies (Deng et al. 2016). This makes it infeasible to accurately judge the rise and fall of stocks based on historical observation. In previous RL-based investment models, the ultimate investment strategy is only based on the analysis of each stock while ignoring the market changes, which usually effectively guide a trader's investment. Hence, conforming to the market will be a better investment strategy in many situations. When the stock market falls, experienced investors tend to spend more money on short-selling; otherwise they are more willing to go long. Previous work lacked this ability to adjust investment decisions dynamically to real-time market conditions.

To balance returns and risks, inspired by human investment behavior, we propose the market scoring unit in this study. It takes market sentiment indicators as inputs and dynamically adjusts the allocation of funds.

This market scoring unit first uses a Long Short-Term Memory (Hochreiter and Schmidhuber 1997) network to recursively extract the sequential representation of input $\boldsymbol{\mathcal{X}}^m$:

$$\boldsymbol{h}_k = \text{LSTM}(\boldsymbol{h}_{k-1}, \boldsymbol{x}_k^m), k \in [1, K],$$

where $\boldsymbol{h}_k$ denotes the hidden state encoded by LSTM at step k. The last hidden state $\boldsymbol{h}_K$ can be seen as a global representation of the input signal. However, earlier information may not be effectively modeled over a long time span. To model these properties, inspired by successful applications of attention in NLP, we adopt a temporal attention mechanism to adaptively model the nonlinear relations, with attention weights being calculated as:

$$e_k = \boldsymbol{V}_e^\top \tanh(\boldsymbol{U}_1[\boldsymbol{h}_k; \boldsymbol{h}_K] + \boldsymbol{U}_2 \boldsymbol{x}_k^m), \alpha_k = \frac{\exp(e_k)}{\sum_{i=1}^K \exp(e_i)},$$

where $\boldsymbol{V}_e \in \mathbb{R}^C$, $\boldsymbol{U}_1 \in \mathbb{R}^{C \times 2C}$, and $\boldsymbol{U}_2 \in \mathbb{R}^{C \times C}$ are parameters to learn. Accordingly, the last hidden state is recalculated as: $\hat{\boldsymbol{h}}_K = \sum_{k=1}^K \alpha_k \cdot \boldsymbol{h}_k$. The hidden state is further represented by μ and σ:

$$\mu, \sigma = \boldsymbol{U}_m \cdot \hat{\boldsymbol{h}}_K + \boldsymbol{b}_m, \tag{5}$$

Portfolio Generator

After obtaining the asset scores $\boldsymbol{v}$, we use the softmax function to transfer $\boldsymbol{v}$ to portfolio weights, similar to Wang et al. (2019). More specifically, we first sort all stocks in descending order based on $\boldsymbol{v}$. Then we select the top G stocks as a winner set $\mathcal{V}^+$ to go long and the bottom G stocks as a loser set $\mathcal{V}^-$ to go short. We obtain the proportions $\boldsymbol{\omega}^+$ and $\boldsymbol{\omega}^-$ as follows:

$$\omega_i^+ = \begin{cases} \frac{\exp(\boldsymbol{v}_i)}{\sum_{j \in \mathcal{V}^+} \exp(\boldsymbol{v}_j)} & i \in \mathcal{V}^+ \\ 0 & i \notin \mathcal{V}^+ \end{cases} \quad \omega_i^- = \begin{cases} \frac{\exp(1-\boldsymbol{v}_i)}{\sum_{j \in \mathcal{V}^-} \exp(1-\boldsymbol{v}_j)} & i \in \mathcal{V}^- \\ 0 & i \notin \mathcal{V}^-. \end{cases}$$

Generating ρ from the market scoring unit can be seen as selecting a value in continuous action space $\mathcal{A}^m \in [0, 1]$. We assume that a random variable $\widetilde{\rho}$ follows normal distribution: $\widetilde{\rho} \sim N(\mu, \sigma^2)$, with μ and σ being derived from the market scoring unit. We sample the value $\widetilde{\rho}$ based on the above normal distribution in training phase for exploration and use μ as $\widetilde{\rho}$ in testing phase for exploitation. Then we clamp this value into the range $[0, 1]$: $\rho = \text{clamp}(\widetilde{\rho})$. After the above process, we complete the trading period according to the procedure defined in Section *Trading Procedure*.

Optimization via Reinforcement Learning

As the portfolio management can be formulated as a Markov decision process, we use policy gradient to optimize the investment policy $\pi(a|\mathcal{X}^a, \mathcal{X}^m; \boldsymbol{\theta})$ in an end-to-end manner.

The policy π consists of two parts: (1) $\pi^a(i|\mathcal{X}^a; \theta^a)$ chooses which stocks to investigate according to stock indicators $\mathcal{X}^a$ in asset scoring unit, and (2) $\pi^m(\tilde{\rho}|\mathcal{X}^m; \theta^m)$ judges the financial condition according to market indicators $\mathcal{X}^m$ in market scoring unit. The two parts are integrated into one loss function for optimization.

Specifically, for the former part, the policy π^a is defined by the portfolio weights (Jiang, Xu, and Liang 2017): $\pi^a(i|\mathcal{X}^a, \theta^a) := \frac{\exp(v_i(\theta^a))}{\sum_{n=1}^N \exp(v_n(\theta^a))}$, for $i = 1, \cdots, N$, where $v_i(\theta^a)$ is the ith asset score (the output of asset scoring unit). The rate of return for holding period t is $r_t = \boldsymbol{y}_t \cdot \boldsymbol{\pi}_{\theta^a}^a - 1$, where $\boldsymbol{y}_t = \boldsymbol{P}_{t+1}^{(c)} / \boldsymbol{P}_t^{(c)}$ is the price rising rate. Given the initial investment amount C_0, thus, the cumulative wealth of a trajectory τ is $C_{|\tau|} = C_0 \prod_{t=0}^{|\tau|} (r_t + 1) = C_0 \prod_{t=0}^{|\tau|} \boldsymbol{y}_t \cdot \boldsymbol{\pi}_\theta^a$.

In this way, the optimization objective of asset scoring unit is switched to maximize the logarithmic cumulated wealth for given trajectories:

$$\nabla J^a(\theta) = \sum_{\tau \sim \pi_\theta} \sum_{t=0}^{|\tau|} \log(\boldsymbol{y}_t \nabla \boldsymbol{\pi}_\theta^a). \tag{6}$$

We then follow the standard Gaussian Policy to optimize market scoring unit, with $\pi^m(\tilde{\rho}|\boldsymbol{\mathcal{X}}^m; \theta^m) = \frac{1}{\sqrt{2\pi}\sigma(\theta^m)} \exp(-\frac{(\tilde{\rho} - \mu(\theta^m))^2}{2\sigma^2(\theta^m)})$, where $\mu(\theta^m)$ and $\sigma(\theta^m)$ are the outputs of market scoring unit, determined by parameters θ^m. Hence, given a reward R_t, the optimization objective of market scoring unit $\nabla J^m(\theta)$ can be calculated as:

$$\nabla J^m(\theta_m) = \sum_{\tau \sim \pi_\theta} \sum_{t=0}^{|\tau|} R_t \nabla \log(\pi_\theta^m). \tag{7}$$

The overall objective function is a weighted sum of Eq. (6) and Eq. (7):

$$\nabla J(\boldsymbol{\theta}) = \nabla J^a(\theta_a) + \iota \nabla J^m(\theta_m)$$
$$= \sum_{\tau \sim \pi_\theta} \left[\sum_{t=0}^{|\tau|} \log(\boldsymbol{y}_t \nabla \boldsymbol{\pi}_\theta^a) + \iota \sum_{t=0}^{|\tau|} R_t \nabla \log(\pi_\theta^m) \right], \quad (8)$$

where ι is used to control the different learning rate for each part. These two parts are optimized by gradient ascent $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \eta \nabla J(\boldsymbol{\theta})$ simultaneously.

Experiments

To comprehensively evaluate *DeepTrader*, we conduct experiments on the constituent stocks of three well-known stock indices, aiming to answer the following questions: **Q1:** How is the performance of *DeepTrader*, especially during some special financial events, such as subprime mortgage crisis? **Q2:** Are the key components in *DeepTrader*, such as attention-aware GCN and market scoring unit, necessary for improving performance? **Q3:** How does the choice of reward function for market scoring unit affect the investment performance? **Q4:** Does the graph structure in GCN affect the investment performance?

Index	Num. of stocks	Training	Test
DJIA	30	1971-1999	2000-2018
HSI	49	1990-2006	2007-2019
CSI100	80	2005-2012	2013-2019

Table 1: Training/Test splitting for different datasets.

Experimental Setup

Datasets. To reduce human intervention and preference, the stock data used in our experiments are from the constituent stocks of Dow Jones Industrial Average (DJIA) in the U.S. market, Hang Seng Index (HSI) in Hong Kong market, and CSI 100 Index in Chinese A-share market. They are the most credible stock indices in their own markets. 1/20 stock(s) is/are deleted from HSI/CSI100 because of data incompleteness.

Comparative Methods. Five related methods in literature are included for comparisons with our model, i.e., *Market*, Buying-Loser-Selling-Winner (*BLSW*) (Poterba and Summers 1988), *CSM* (Jegadeesh and Titman 1993), *EIIE* (Jiang, Xu, and Liang 2017), and *AlphaStock* (AS) (Wang et al. 2019). Here, *Market* is a simple Buy-And-Hold strategy. *BLSW* is a strategy based on Mean Reversion (Poterba and Summers 1988). *CSM* is a classic momentum strategy. *EIIE* and *AS* are two recently developed RL-based methods. Among the five methods, *BLSW*, *CSM*, and *AS* can perform short operation.

We also implement several variants or simplified versions of *DeepTrader* (DT) to investigate the roles of several key components in ablation study, including *DeepTrader-NS* (DT-NS) and *DeepTrader-NM* (DT-NM), where "*N*" means No or removing some components from *DeepTrader*. Specifically, *DT-NS* removes the spatial attention mechanism and GCN layer from asset scoring unit, while *DT-NM* removes market scoring unit. We also investigate the influences of different reward functions and graph structures in GCN. By default, we use MDD as the reward function and *stock industry classification* to derive the graph structure.

Evaluation Measures Six metrics are used in our experiments, which can be divided into three categories: i) profit criterion, including Annualized Rate of Return (ARR); ii) risk criterion, including Annualized Volatility (AVol) and Maximum DrawDown (MDD); iii) risk-profit criterion, including Annualized Sharpe Ratio (ASR), Calmar Ratio (CR), and Sortino ratio (SoR). For AVol and MDD, the lower the better, while for the rest, the higher the better. The computational details for each metric are given in Appendix.

Results on DJIA, HSI, and CSI 100

We report the results of risk-return measures by *DeepTrader* and other comparative methods in Table 2.

Performance on DJIA. Overall, *DeepTrader* (DT) achieves the best performance in obtaining a high return rate while keeping the risk at a low level, i.e., it has the best risk-return balance as indicated by ASR, SoR, and CR. From the table, we can see that although AlphaStock (AS) is a close competitor in controlling the risk with a slightly higher AVol, DT has much better MDD. Note that the AVol indicator takes both downside and upside volatility into account, while MDD only considers the downside risk. Since investigators usually care more about downside volatility and capital preservation, we use MDD as the reward function in the market scoring unit and obtain the lowest (best) MDD in trading. Moreover, we found that RL-based methods, i.e., *EIIE*, *AS*, and our *DeepTrader*, outperform the traditional baselines in risk-return balancing.

Performance on HSI. From the middle column in Table 2, generally speaking, we observed that DRL-based methods perform better over traditional investment strategies. For example, *DeepTrader* achieves the best performance in risk-gain criteria, while *BLSW* trips up in Hong Kong market with negative ARR and highest MDD (note for MDD, the lower the better). *AlphaStock* also performs well in risk control, with the best AVol value but lower ARR value than *EIIE* and *DeepTrader*. Although *EIIE* obtains a remarkable ARR, its performance is not remarkable when risk indicators, i.e., AVol and MDD, are considered. The above observations are basically consistent with the results on DJIA in U.S. market. Besides, we found that all DRL-based methods perform significantly better on HSI than on DJIA, which may be because HSI has more stocks for operation. *CSM* and *BLSW* get opposite performance in Hong Kong and U.S. stock markets, meaning that their generalization abilities are not good.

Performance on CSI 100. Note that the market scoring unit is not applicable on CSI 100, because short position is not allowed in Chinese A-share market; accordingly all capitals are invested on long position. As shown in the right column of Table 2, the lack of short sales does not have an obvious side-effect on the results, and our *DeepTrader*-based methods remain the best in risk-profit balance. Specifically, even without the market scoring unit, *DeepTrader-NM* still

Dataset	DJIA						HSI						CSI100					
Models	ARR(%)	AVol	ASR	SoR	MDD(%)	CR	ARR(%)	AVol	ASR	SoR	MDD(%)	CR	ARR(%)	AVol	ASR	SoR	MDD(%)	CR
Market	4.17	**0.158**	0.264	0.787	49.93	0.084	4.53	0.222	0.204	0.707	56.37	0.080	10.49	0.255	0.411	2.542	39.54	0.265
BLSW	6.57	0.279	0.235	1.112	72.72	0.090	-10.01	0.294	-0.343	-1.213	90.97	-0.111	28.25	0.494	0.572	4.205	29.29	0.964
CSM	-3.57	0.248	-0.144	-0.422	88.31	-0.040	7.66	0.295	0.260	0.858	53.25	0.144	19.17	0.411	0.466	1.753	61.73	0.311
EIIE	9.96	0.185	0.537	1.707	45.98	0.217	**23.40**	0.363	0.645	2.523	50.26	0.466	25.07	**0.235**	1.067	3.949	29.04	0.863
AS	9.13	**0.158**	0.579	2.187	26.66	0.343	14.51	**0.183**	0.793	3.007	17.39	0.834	21.85	0.241	0.922	4.012	19.93	1.096
DT-NS	10.21	0.163	0.628	2.448	26.33	0.388	19.03	0.211	0.902	4.462	19.93	0.955	30.48	0.286	1.066	5.438	**16.24**	1.877
DT-NM	11.49	0.180	0.638	2.507	31.24	0.368	20.72	0.223	0.930	4.572	20.17	1.027	**35.67**	0.324	**1.100**	**6.059**	18.13	**1.967**
DT	**12.35**	0.172	**0.718**	**2.782**	**22.61**	**0.546**	21.85	0.209	**1.044**	**5.140**	**17.11**	**1.277**	-	-	-	-	-	-

Table 2: The results in different stock market. Notice that the complete DeepTrader(DT) is not applicable to CSI 100.

achieves better MDD, compared to the other five comparisons. We further notice that all methods on CSI 100 achieve higher ARR than that on DJIA and HSI. This might be attributable to the strong performance of Chinese stock market over the past few years which are taken as testing set (year 2013-2019). Moreover, the testing set does not contain the 2008 subprime mortgage crisis, which may be another reason why all the methods get higher ARR on CSI 100.

Performance at Subprime Mortgage Crisis

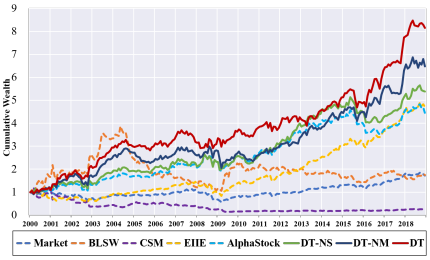

Figure 2: The cumulative wealth on DJIA.

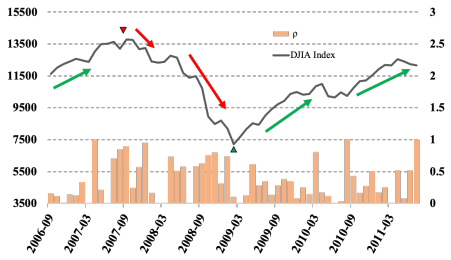

Figure 3: DJIA index and its corresponding ρ from Sep. 2006 to Jul. 2011. The mean values of the DJIA index for each month are displayed. Red and green triangles indicate the beginning and end of subprime crisis respectively.

It should be noted that the backtesting time of the DJIA data ranges from Jan. 2000 to Dec. 2018, which covers several well-known financial events, including the subprime mortgage crisis from 2007 to 2009. Since the ultimate goal of portfolio management is to increase wealth in the long run, we show the curves of cumulative wealth in folds in Figure 2. The results are consistent with Table 2. We can see that *DeepTrader* increases the wealth smoothly across diverse market states and surpasses the five competitors after 2009, which is the end year of subprime mortgage crisis. As shown in Figure 3, the DJIA index drops down significantly from 2007 to 2008, leading to a difficult scenario for most investment methods. Although *BLSW* accumulates wealth fast from 2000 to 2004, then it declines till the end of the crisis. *Market* basically maintains at the starting capital level, while *CSM* loses wealth almost all the time. *EIIE* grows well in bull market along the time of economic recovery to catch up with *AlphaStock* which balances between profit and risk well most of the time. Generally speaking, DRL-based approaches generate more stable returns during the 19 years' investment than traditional ones.

Models	ARR(%)	AVol	ASR	SoR	MDD(%)	CR
DT-RoR	**15.60**	0.204	**0.766**	**2.788**	45.52	0.343
DT-SR	14.36	0.205	0.700	2.498	46.06	0.312
DT-MDD	12.35	**0.172**	0.718	2.782	**22.61**	**0.546**
DT-CR	12.02	0.185	0.648	2.598	31.10	0.387

Table 3: The effects of different rewards.

The roles of asset scoring unit and market scoring unit in *DeepTrader* are demonstrated by the curves of *DT-NS* and *DT-NM* in Figure 2. The asset scoring unit enables *DT-NM* to accumulate wealth faster than *DT-NS* and *AlphaStock* during the economic recovery process. *DT-NM*, with market scoring unit being removed, declines more than *DT* when encountering the 2008 crisis. It can be further demonstrated by the values of ρ in Figure 3 that the market scoring unit dynamically increases the proportion of short funds (mostly at $\rho > 0.5$) to handle market depression, and then decreases the proportion (mostly at $\rho < 0.5$) when the market booms up. This indicates that our model can strike a good balance between risk and return, and accumulate wealth to a high level in a smooth and stable way.

Ablation Study

Effectiveness of spatial-GCN. As observed from Table 2, *DeepTrader* as well as *DeepTrader-NM* outperforms *DeepTrader-NS* in all the three markets in terms of ARR, because the spatial-GCN enhances the dependence relationships between stocks and captures the rising activities well. Different from *AlphaStock*, our spatial-GCN mechanism can

Dataset	DJIA						HSI						CSI100					
Models	ARR(%)	AVol	ASR	SoR	MDD(%)	CR	ARR(%)	AVol	ASR	SoR	MDD(%)	CR	ARR(%)	AVol	ASR	SoR	MDD(%)	CR
DT-precision	10.46	0.178	0.587	2.405	31.53	0.332	20.37	**0.200**	1.020	4.217	22.34	0.912	34.74	**0.317**	1.096	5.770	20.61	1.686
DT-correlation	10.56	0.188	0.561	2.444	24.94	0.423	20.56	0.204	**1.066**	4.243	20.03	1.026	34.25	0.322	1.064	5.735	20.94	1.636
DT-causal	**13.37**	0.178	**0.750**	**3.034**	**18.58**	**0.719**	**22.62**	0.225	1.009	**5.211**	**15.90**	**1.422**	35.21	0.319	**1.103**	5.891	19.82	1.776
DT-industry	12.35	**0.172**	0.718	2.782	22.61	0.546	21.85	0.209	1.044	5.140	17.11	1.277	**35.67**	0.324	1.100	**6.058**	**18.13**	**1.967**

Table 4: The effects of different graph structures in GCN.

capture both long and short dependencies at the same time in a hierarchical way. It enables the model to invest on the most promising stocks and accumulate more wealth than *AlphaStock*, as shown in Figure 2.

Effectiveness of market scoring unit. The market scoring unit is designed to balance benefits and risks. The results indicate that the market scoring unit improves the performance of *DeepTrader* from *DeepTrader-NM* in terms of risk-return balancing indicators: ASR, SoR, and CR. We attribute this improvement to the dynamic adjustment of the ratio between long and short funds granted by ρ, which is an embedding of market conditions. This unit helps increase ρ in the downturn to control losses, while decreasing it in the rally to ensure more profits, as demonstrated in Figure 3.

Effects of Reward Functions

In default *DeepTrader*, we have adopted MDD as the reward function for the market scoring unit to help control the risk under a low level. In addition to MDD, other measures, such as RoR, SR, and CR, are alternative choices for the reward function. For example, SR was adopted by TFA-based model (Chiu and Xu 2004) as the optimization objective or *AlphaStock* (Wang et al. 2019) as the reward in RL learning. In Table 3, we investigate the effects of different choices of the reward function in the market scoring unit on DJIA. Computational details of different rewards are given in Appendix, as well as the results on HSI and CSI 100.

First, *DT-RoR* and *DT-SR* are more profitable with high ARR, but they both lead to high MDD values (larger than 45%), which is an unacceptable risk for many investors. Moreover, with SR as the reward function, which is supposed to help achieve better ASR, *DT-SR* does not lead to the best ASR. It is also noted from Table 2 that *AlphaStock*, which uses SR as the reward, achieves 0.579 in ASR which is much lower than *DeepTrader* with any reward functions in Table 3. Second, *DT-MDD* achieves significant improvement in MDD, more than DT with other profit-related rewards (RoR, SR and CR) does. This is mainly because the market is generally rising, making it easier to increase revenue than to control risk by giving a larger ρ in most cases.

To sum up, *DT-MDD* is the best to control the risk measured by MDD, while keeping a reasonably high return. By adopting different reward functions for the market scoring unit, our model is also flexible to adapt to preferences of the investors, towards high-yield or low-risk.

Effects of Graph Structures in GCN

We further exploit different ways to construct the graph structure in GCN and examine which one performs the best (Table 4). Particularly, we consider *DeepTrader* with the graph structure given by the precision matrix (DT-precision), correlation matrix (DT-correlation), learned causal structure (DT-causal), and industry classification (DT-industry). Interestingly, we find that with causal structure, the performance on DJIA and HSI has a significant improvement, no matter in profit criterion (e.g., ARR), risk criterion (e.g., MDD), or risk-profit criterion (e.g., SoR and CR). Instead, correlation and precision-based graph structures do not perform well, even worse compared to that derived from industry classification. On CSI 100, the overall performance with different graph structures is relatively close. This may be due to the fact that Chinese A-share market does not allow short selling, which limits the ability of DeepTrader to adjust its investment plan based on the relevant information it captured.

This finding shows the importance of the graph structure in GCN, as well as the superiority of using causal structure for it. Specifically, causal relations provide precise structural information of the information propagation process among stocks. It pinpoints the key connectivity characteristics including directions and effectively removes spurious correlations between stocks (Huang et al. 2020), in contrast to correlation and precision. The relatively sparse causal structure after removing spurious correlations also makes optimization easier. Moreover, the (qualitative) causal structure is usually more stable over time or across different conditions, and thus although there may exist distribution shifts in testing data, the causal structure derived from training data still keeps most of the structural information.

Conclusion

In this paper, we have proposed a DRL-based framework *DeepTrader* for portfolio management. It contains not only an asset scoring unit that ranks the rising probability for each stock, but also a novel market scoring unit that embeds the market condition to adjust the ratio between the long and short funds. Moreover, asset scoring unit is enhanced by a graph-based stock interrelationship learning layer, while the negative of the maximum drawdown is adopted as the reward function for the market scoring unit to improve the risk control capability. The two units complement to each other and are integrated to generate a risk-return balanced portfolio. The computed investment strategy can smoothly accumulate wealth in the long run to a high level, under complicated and difficult financial situations such as financial crisis. Comparative experiments and ablation studies on three representative financial markets confirm the superiority of our proposed model.

Acknowledgements

This work was supported by National Science and Technology Innovation 2030 Major Project (2018AAA0100700) of the Ministry of Science and Technology of China, and Shanghai Municipal Science and Technology Major Project (2021SHZDZX0102). Kun Zhang would like to acknowledge the support by the United States Air Force under Contract No. FA8650-17-C-7715. Shikui Tu and Lei Xu are corresponding authors.

References

Almahdi, S.; and Yang, S. Y. 2017. An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown. *Expert Systems with Applications* 87: 267–279.

Chiu, K.-C.; and Xu, L. 2003. Optimizing financial portfolios from the perspective of mining temporal structures of stock returns. In *International Workshop on Machine Learning and Data Mining in Pattern Recognition*, 266–275. Springer.

Chiu, K.-C.; and Xu, L. 2004. Arbitrage pricing theory-based Gaussian temporal factor analysis for adaptive portfolio management. *Decision Support Systems* 37(4): 485–500.

Deng, Y.; Bao, F.; Kong, Y.; Ren, Z.; and Dai, Q. 2016. Deep direct reinforcement learning for financial signal representation and trading. *IEEE transactions on neural networks and learning systems* 28(3): 653–664.

Feng, X.; Guo, J.; Qin, B.; Liu, T.; and Liu, Y. 2017. Effective Deep Memory Networks for Distant Supervised Relation Extraction. In *IJCAI*, 4002–4008.

Grinblatt, M.; Titman, S.; and Wermers, R. 1995. Momentum investment strategies, portfolio performance, and herding: A study of mutual fund behavior. *The American economic review* 1088–1105.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hochreiter, S.; and Schmidhuber, J. 1997. Long short-term memory. *Neural Computation* 9(8): 1735–1780.

Hong, H.; and Stein, J. C. 1999. A unified theory of underreaction, momentum trading, and overreaction in asset markets. *The Journal of finance* 54(6): 2143–2184.

Huang, B.; Zhang, K.; Zhang, J.; Ramsey, J.; Sanchez-Romero, R.; Glymour, C.; and Schölkopf, B. 2020. Causal discovery from heterogeneous/nonstationary data. *Journal of Machine Learning Research* 21(89): 1–53.

Hung, K.-k.; Cheung, Y.-m.; and Xu, L. 2003. An extended ASLD trading system to enhance portfolio management. *IEEE Transactions on Neural Networks* 14(2): 413–425.

Jegadeesh, N.; and Titman, S. 1993. Returns to buying winners and selling losers: Implications for stock market efficiency. *The Journal of finance* 48(1): 65–91.

Jegadeesh, N.; and Titman, S. 2002. Cross-sectional and time-series determinants of momentum returns. *The Review of Financial Studies* 15(1): 143–157.

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

Jin, O.; and El-Saawy, H. 2016. Portfolio management using reinforcement learning. *Stanford University* .

Liu, Y.; Liu, Q.; Zhao, H.; Pan, Z.; and Liu, C. 2020. Adaptive Quantitative Trading: An Imitative Deep Reinforcement Learning Approach. In *AAAI*, 2128–2135.

Malkiel, B. G. 1989. Efficient market hypothesis. In *Finance*, 127–134. Springer.

Moody, J.; Wu, L.; Liao, Y.; and Saffell, M. 1998. Performance functions and reinforcement learning for trading systems and portfolios. *Journal of Forecasting* 17(5-6): 441–470.

Moskowitz, T. J.; Ooi, Y. H.; and Pedersen, L. H. 2012. Time series momentum. *Journal of financial economics* 104(2): 228–250.

Poterba, J. M.; and Summers, L. H. 1988. Mean reversion in stock prices: Evidence and implications. *Journal of financial economics* 22(1): 27–59.

Spirtes, P.; Glymour, C.; and Scheines, R. 2001. *Causation, Prediction, and Search*. Cambridge, MA: MIT Press, 2nd edition.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Wang, J.; Zhang, Y.; Tang, K.; Wu, J.; and Xiong, Z. 2019. AlphaStock: A Buying-Winners-and-Selling-Losers Investment Strategy using Interpretable Deep Reinforcement Attention Networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 1900–1908. ACM.

Wu, Z.; Pan, S.; Long, G.; Jiang, J.; and Zhang, C. 2019. Graph WaveNet for Deep Spatial-Temporal Graph Modeling. *arXiv preprint arXiv:1906.00121* .

Xu, L.; and Cheung, Y.-m. 1997. Adaptive supervised learning decision networks for traders and portfolios. In *Proceedings of the IEEE/IAFE 1997 Computational Intelligence for Financial Engineering (CIFEr)*, 206–212. IEEE.

Ye, Y.; Pei, H.; Wang, B.; Chen, P.-Y.; Zhu, Y.; Xiao, J.; and Li, B. 2020. Reinforcement-learning based portfolio management with augmented asset movement prediction states. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 1112–1119.

Yu, F.; and Koltun, V. 2015. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122* .