SAC-Based Deep Reinforcement Learning for Portfolio Selection

Jintao Ge, [1] **Shuheng Chen,** [1] **Qianwen Mao,** [1] **Weili Zhou,** [1] **Wei Ding** [1]

School of Informatics Xiamen University
{30920221154268, 30920221154264, 30920221154255, 30920221154289, 30920221154266}@stu.xmu.edu.cn

Abstract

Portfolio Selection (PS) is a fundamental financial planning task that aims to identify a strategy for dynamically allocating wealth among a set of portfolio assets to maximize long-term return. However, it is difficult to design a profitable strategy in a complex and dynamic stock market. In this paper, we propose a PS model using the Soft Actor-Critic (SAC) Deep Reinforcement Learning (DRL) framework. Specifically, we utilize Long Short-Term Memory (LSTM) to extract both price series patterns and asset correlations from portfolio series, while using the DRL model to generate the portfolio weight. Furthermore, we investigate the effect of including stock movement prediction indicators in the state representation. We formulate experiments to evaluate our DRL models on real data from the U.S., Japanese and British stocks, against benchmarks including state-of-the-art online portfolio selection approaches, using measures consisting of Average daily yield, Sharpe ratio, Sortino ratio, and Maximum drawdown. Our experiments show that the SAC-based trading strategy is profitable, robust, and risk-aware, as compared to those of the baselines. Moreover, the introduction of additional financial indicators in the state representation is found to have a positive effect overall.

Introduction

Portfolio selection is the task of determining how to optimally allocate funds of a finite budget into a range of financial assets (Filos 2019). As the saying goes, “Don’t put all your eggs in one basket.” From the concept of portfolio selection was proposed (Markowitz 1952), it has been a very popular research topic (Samuelson 1975; Rockafellar and Uryasev 2002; Chu, Tsai, and Pan 2006; Karimkashi and Kishk 2010; Yu et al. 2022). Throughout literature, two major paradigms for investigating the portfolio selection problem are identified. These are the Mean Variance Theory (Markowitz 1952) originating from the finance community, and the Kelly Criterion (Kelly Jr. 1956; Cover and Ordentlich 1996) originating from information theory.

However, only the Kelly Criterion fits the online scenario and incorporates the online machine learning perspective, consisting of multiple periods or steps (Gunjan and Bhattacharyya 2022). Following Kelly Criterion, many kinds of portfolio selection methods have been proposed, including online learning and reinforcement learning based methods.

Online learning based methods provide the optimal per-trade position size to maximize the expected log-return. The pioneering studies include Exponential Gradient (EG) (Helmbold et al. 1998), Online Netwon Step (ONS) (Hazan and Seshadhri 2009), Universal Portfolios (UP) (Cover 1991), and Uniform Constant Rebalanced Portfolios (UCRP) (Cover and Gluss 1986). Recently, several methods exploit the mean reversion property to select the portfolio, e.g., Passive Aggressive Mean Reversion (PAMR) (Li et al. 2012), Robust Median Reversion (RMR) (Huang et al. 2016), Online Moving Average Reversion (OLMAR) (Li and Hoi 2012), and Weighted Moving Average Mean Reversion (WMAMR) (Gao and Zhang 2013). Nonetheless, all the above methods ignore sequential features and rely solely on handcrafted features such as moving averages and stochastic indicators. Consequently, they may perform unsatisfactorily as a result of poor representation. Furthermore, many of the above methods assume no transaction cost. Such a cost will bring biases into the estimation of accumulative returns, and thus affects the practical performance of these methods (Zhang et al. 2020).

Reinforcement learning (RL) based methods, on the other hand, use RL algorithms for optimizing specific utility functions and making comprehensive portfolio policies (Neuneier 1995; Neuneier and Mihatsch 1998). Recently, some studies (Lee et al. 2020; Aboussalah and Lee 2020; Betancourt and Chen 2021) apply deep reinforcement learning to portfolio selection, where they use deep neural networks to extract patterns. We found that these methods use only the most basic trading indicators in the state space, such as *Open*, *Close*, *High*, *Low* and *Volume*, etc. However, more complex features, such as stock movement prediction indicators used in online learning based methods, could be introduced into the state space to improve model performance.

To address the aforementioned problems, in this paper, we propose a PS model using the SAC algorithm (Haarnoja et al. 2018), which achieves sample-efficient learning by introducing entropy maximization. Furthermore, we utilize LSTM (Hochreiter and Schmidhuber 1997) to extract both price series patterns and asset correlations from portfolio series, while using the SAC model to generate the portfolio weight. It is worth mentioning that we consider practi-

Copyright © 2023, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

cal trading constraints, such as transaction costs, to stably train an autonomous agent whose investment decisions are risk-aware yet profitable. Finally, we add stock movement prediction indicators to the state space to improve the performance of the model.

These models are compared with the online learning based methods mentioned above, using different time window parameters, on real data from the U.S., Japanese and British stocks. Experimental results show that the SAC-based trading strategy is profitable, robust, and risk-aware, as compared to those of the baselines. Further, adding additional financial indicators to the state space has a positive effect as a whole. Our main contributions are summarized as follows.

- We apply the state-of-the-art SAC deep reinforcement learning framework to the domain of portfolio selection, expanding the application scenarios of the algorithm.
- Our proposed PS model applies stock movement prediction indicators to the DRL state format. To the best of our knowledge, this is the first attempt to use the functionality of the online portfolio selection in the state format to help DRL agent.
- Extensive experiments on real-world datasets demonstrate the effectiveness and superiority of the proposed method in terms of profitability, cost-sensitivity and representation abilities.

Related Work

With the availability of large-scale market data, it's natural to employ deep learning (DL) model which can exploit the potential laws of market in PS. DL-based portfolio selection systems can provide users with financial services and investment advice by employing low-cost and easy-to-use algorithms (Noonpakdee 2020). Meanwhile, the application of deep learning algorithms can balance the risk and return of investments, improving the portfolio to a large extent (Deng, Xu, and Wu 2021). Previous studies (Heaton, Polson, and Witte 2017; Ban, El Karoui, and Lim 2018; Chaouki et al. 2020) have demonstrated the effectiveness of neural network (NN) models in predicting asset prices and allocating assets. However, Given the nonlinear, dynamic, and chaotic nature of the stock market, models are usually unstable and sensitive to parameters. What's even more crucial is, DL models which have no interaction with the market have a natural disadvantage in decision making problem like PS.

DRL is the combination of reinforcement learning and deep neural networks for deep learning, extending to tasks with high-dimensional input and action spaces. Today, a large number of state-of-the-art algorithms in the field of DRL are widely used in portfolio selection and optimization. For instance, (Buehler et al. 2019) present a DRL framework to hedge a portfolio of derivatives under transaction costs, where the framework does not depend on specific market dynamics. (Jiang, Xu, and Liang 2017) use the model-free Deep Deterministic Policy Gradient (DDPG) (Lillicrap et al. 2015) to dynamically optimize cryptocurrency portfolios. Further, (Yu et al. 2019) propose a deep reinforcement technique in which investment decisions and actions are made periodically based on the current global objective.

Nevertheless, these methods above ignore the learning of sequential features and do not control costs during optimization, leading to limited representation abilities and performance. In contrast, our method not only learns good feature representation based on SAC framework but is also sensitive to cost.

Proposed Solution

The complexity of the stock market presents volatility, vulnerability, and uncertainty. Deep reinforcement learning agents can make a dynamic adjustment at any time according to changes in the environment, which can be successfully applied to stock portfolio selection. In this paper, we propose a PS model using the SAC DRL framework, as illustrated in Figure 1. To extract more feature patterns to make accurate decisions, We integrate stock movement prediction indicators with the prices of assets for state augmentation. Next, we adopt the soft actor-critic algorithm based on the augmented state for learning the policy of PS. Specifically, we will detail these method components in the following subsections.

Problem Settings

Consider a portfolio selection task over a financial market during n periods with $m+1$ assets, including one cash asset and m risk assets. On the tth period, we denote the prices of all assets as $p_t \in \mathbb{R}_+^{(m+1)\times d}$, where each row $p_{t,i} \in \mathbb{R}_+^d$ indicates the feature of asset i, and d denotes the number of prices. Specifically, we set $d = 4$ in this paper. That is, we consider four kinds of prices, namely the opening, highest, lowest and closing prices. One can generalize it to more prices to obtain more information. The price series is represented by $P_t = \{p_{t-k}, .., p_{t-1}\}$, where k is the *window length*. The window length is a configurable parameter that denotes the number of past time steps considered relevant for each state. We experiment using the window sizes $3, 7$ and 11.

The price change on the tth period is specified by a *price relative vector* $x_t = \frac{p_t^c}{p_{t-1}^c} \in \mathbb{R}_+^{m+1}$, where p_t^c is the closing price of assets. Assuming there is no inflation or deflation, the cash is risk-free with invariant price, i.e., $\{\forall t \mid x_{t,0} = 1\}$, and it has little influence on the learning process. We thus exclude the cash asset in the input, i.e., $P_t \in \mathbb{R}^{m\times k\times 4}$. The price relative vector can be used to calculate change in *total portfolio value* in a period. For example, given that p_{t-1} is the portfolio value at the beginning of period t, without taking transaction cost into consideration, p_t is calculated as follows:

$$p_t = p_{t-1} x_t^\top a_t \tag{1}$$

where a_t is the *portfolio weight vector* at the beginning of period t. When making decisions, the investment decision is specified by the portfolio weight vector $a_t = [a_{t,0}, a_{t,1}, a_{t,2}, \dots, a_{t,m}] \in \mathbb{R}^{m+1}$, where $a_{t,i} \geq 0$ is the proportion of asset i, and $\sum_{i=0}^{m} a_{t,i} = 1$. Here, the portfolio decision contains the proportion of all assets, including

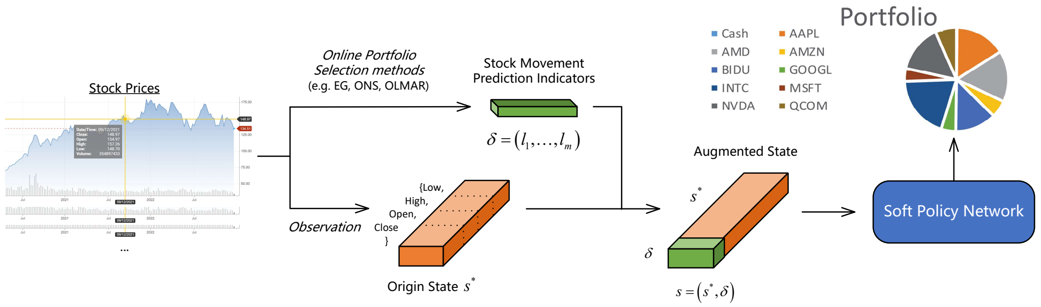

Figure 1: The framework of our proposed state-augmented SAC-based portfolio selection method. The price information of multiple stocks in the market constitutes the original state s^*. Meanwhile, online portfolio selection methods such as EG, ONS, and OLMAR extract stock movement prediction indicators δ from the price series for state enhancement. Finally, the soft policy network will generate the portfolio weight vector from the augmented state s.

the cash $a_{t,0}$. The initial portfolio weight vector a_0 is set to $(1, 0, ..., 0)$, and the elements in the portfolio weight vector at any period a_t, always sum up to one. Therefore, the *logarithmic rate of return* for period t is

$$l_t = \ln \frac{p_t}{p_{t-1}} = \ln x_t^\top a_t \tag{2}$$

Hence, assuming no transaction cost, the *final portfolio value* is

$$S_n = S_0 \exp\left(\sum_{t=1}^{n} l_t\right) = S_0 \prod_{t=1}^{n} x_t^\top a_t \tag{3}$$

where S_0 is the initial investment amount. This is set to 1 throughout all our experiments. However, in the real market, adjusting the portfolio's asset allocation is usually not free, which would introduce transaction cost. After considering transaction cost c_t, S_n is calculated as follows:

$$S_n = S_0 \prod_{t=1}^{n} x_t^\top a_t \left(1 - c_t\right) \tag{4}$$

There are two general assumptions (Li and Hoi 2012; Zhang et al. 2020) in this task: (i) *perfect liquidity*: all market assets are liquid enough to make every trading at the last price immediately possible when an order is placed; (ii) *zero-market-impact*: the investments done by our trading agents are so small that they have no effect on the market.

Markov Decision Process for Portfolio Selection. We formulate the investment process as a generalized Markov Decision Process by $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R})$. Specifically, on the tth period, the agent observes a state $s_t = P_t \in \mathcal{S}$, and takes an action $a_t \sim \pi(\cdot \mid s_t) \in \mathcal{A}$, which determines the reward $r_t = x_t^\top a_t \in \mathcal{R}$, while the next state is a stochastic transition $s_{t+1} \sim \mathcal{T}\left(s_t\right)$. When considering the transaction cost, the reward will be adjusted as $r_t^c = r_t * (1 - c_t)$, where c_t is the proportion of transaction cost.

Soft Actor Critic (SAC)

The Soft Actor Critic (SAC) (Haarnoja et al. 2018) is an off-policy algorithm developed for maximum entropy reinforcement learning. Compared to the DDPG (Lillicrap et al. 2015), the SAC uses stochastic policy, which has certain advantages over deterministic policy. The SAC requires the actor to maximize the entropy of reward expectation and strategy distribution at the same time. The introduction of maximum entropy enhances action exploration ability, enabling the exploration of more stock decisions and achieving more stable performance under complex circumstances.

The iterative process of the SAC is divided into soft policy evaluation and soft policy improvement. For fixed strategy π, its soft Q value can be iterated by Bellman backup operator $\mathcal{T}^\pi$:

$$\mathcal{T}^\pi Q\left(\mathbf{s}_t, \mathbf{a}_t\right) \triangleq r\left(\mathbf{s}_t, \mathbf{a}_t\right) + \gamma \mathbb{E}_{\mathbf{s}_{t+1} \sim p}\left[V\left(\mathbf{s}_{t+1}\right)\right] \tag{5}$$

where

$$V\left(\mathbf{s}_t\right) = \mathbb{E}_{\mathbf{a}_t \sim \pi}\left[Q\left(\mathbf{s}_t, \mathbf{a}_t\right) - \alpha \log \pi\left(\mathbf{a}_t \mid \mathbf{s}_t\right)\right] \tag{6}$$

is the soft state value function. The hyperparameter α measures the relative importance of entropy for reward. In practice, tractable policies are preferred. Thus, we additionally restrict the policy to set of policies Π that can correspond to a parameterized family of distributions such as Gaussians. The soft policy is updated as follows:

$$\pi_{\text{new}} = \arg\min_{\pi' \in \Pi} D_{KL}\left(\pi'\left(\cdot \mid \mathbf{s}_t\right) \| \frac{\exp\left(Q^{\pi_{\text{old}}}\left(\mathbf{s}_t, \cdot\right)\right)}{Z^{\pi_{\text{old}}}\left(\mathbf{s}_t\right)}\right) \tag{7}$$

where $Z^{\pi_{\text{old}}}\left(\mathbf{s}_t\right)$ is the partition function used to normalize the distribution of Q values. Different from the usual off-policy method used to maximize the Q value, the policy of the SAC is updated in the direction of an exponential distribution proportional to Q. In practice, to facilitate the processing of the policy, we still output the policy as a Gaussian distribution and minimize the gap between the two distributions by minimizing KL divergence.

By using soft policy evaluation and soft policy improvement repeatedly and alternately, the final policy will converge to the optimal value. The learning objective of the SAC is as follows:

$$\pi^* = \arg\max_{\pi} \sum_t \mathbb{E}_{\left(\mathbf{s}_t, \mathbf{a}_t\right) \sim \rho_\pi}\left[r\left(\mathbf{s}_t, \mathbf{a}_t\right) + \alpha \mathcal{H}\left(\pi\left(\cdot \mid \mathbf{s}_t\right)\right)\right] \tag{8}$$

Table 1: The company name abbreviation of 30 stocks selected in the U.S., Japanese, and British markets respectively.

Market	Company Symbol
The U.S. stock (30 stocks)	AAPL, AMGN, AXP, BA, CAT, CRM, CSCO, CVX, DIS, DOW, GS, HD, HON, IBM, INTC, JNJ, JPM, KO, MCD, MMM, MRK, MSFT, NKE, PG, TRV, UNH, V, VZ, WBA, WMT
The Japanese stock (30 stocks)	Advantest Corp, Alps Electric, Amada, Casio, Chiba Bank, Chugai Pharmaceutical, Concordia Financial Group, Dainippon Screen Mfg, DIC Corp, Eneos Holdings Inc, Fujikura Ltd, GS Yuasa Corp, Hitachi, Hitachi Zosen Corp, Isuzu Motors Ltd, J.Front Retailing Co, Japan Steel Works Ltd, JR West Japan, KDDI, Keisei Electric Railway Co, Konami Corp, Maruha Nichiro Corp, Mazda, Minebea Mitsumi Inc, Mitsubishi Motors, Mitsui chemical, NEC, Sony, Tokyo Electric Power, Yamaha
The British stock (30 stocks)	ABDN, AZN, BATS, BGUK, BLND, BP, CPG, CRH, EDEV, ENT, FLTRF, HIK, HLMA, ICP, JETJ, LSEG, MRON, NG, NWG, OCDO, POLYP, PSHP, RMV, RR, RTO, SGE, SMDS, SMT, SSE, TW

The randomness of the optimal control policy controlled by α is determined by the following formula:

$$\alpha_t^* = \arg\min_{\alpha_t} \mathbb{E}_{\mathbf{a}_t \sim \pi_t^*} \left[-\alpha_t \log \pi_t^* \left(\mathbf{a}_t \mid \mathbf{s}_t; \alpha_t \right) - \alpha_t \overline{\mathcal{H}} \right] \tag{9}$$

Relative to the deterministic policy, the stochastic policy of the SAC also requires entropy maximization, which means that the neural network needs to explore all possible optimal paths. This can produce the following advantages. (i) The policy will learn many ways to complete tasks through maximum entropy, which is more conducive to learning new tasks. (ii) Clearly, the policy's stronger exploration ability makes it easier to find better modes under multimodal rewards. For example, stock decision-making agents should not only obtain high returns but also reduce trading risks. (iii) The policy is more robust and generalizable by exploring various optimal possibilities in different ways, so it is easier to adjust in the face of interference. For example, when facing different stock markets, agents can make different decisions in dealing with different environments.

Finally, in order to extract patterns from a portfolio series, we use the same approach used in prior works, based on a Long Short-Term Memory (LSTM) predictor (Zhang et al. 2020; Aboussalah and Lee 2020). Both the actor and critic networks, for our SAC framework, utilize the same configuration.

Experiments

In this section, we compare our proposed SAC-based portfolio selection method with other methods on real data from the U.S., Japanese and British stocks. We will summarize these three datasets, describe the training process, define the evaluation metrics, introduce the baseline PS methods for comparison and perform extensive experiments to validate the effectiveness of state augmentation.

Datasets

We first explain the selection of stock data used for our trading strategy. As is well known, 30 Dow Jones stocks cover representative companies from many different industries, mainly including financial services, pharmaceutical industry, information technology, etc., and can reflect the state of the U.S. stock market to a certain extent. Therefore, these data are useful to train the robustness, effectiveness, and universality of our proposed model. Thus, these 30 Dow Jones stocks are very suitable for the training and testing of our proposed strategy.

In addition to 30 U.S. stocks, we also choose 30 Japanese and British stocks for our experiments to verify the generality and applicability of the model. The company names of the stocks are shown in Table 1. Along the timeline of the original datasets, we partition the data samples for 1985/01/07 to 2006/11/02 as a training set and those for 2006/11/03 to 2010/06/29 as a testing set. Namely, datasets are split in a 6:1 ratio for training and testing, respectively. Our selected ratio allows for a great number of training steps, whilst still leaving an adequate number of testing steps.

Training

Training was done over 500 episodes, each consisting of 1000 steps. At the start of each episode, the agent is placed at a random point within the training subset. This starting step is to allow for the training steps to be completed. When the networks are to be trained, the training is done with a mini-batch of 128, sampled uniformly from a replay buffer consisting of the agents' history.

In order to avoid overfitting, we include a *value function threshold* parameter which terminates the training phase when a reward value threshold is exceeded for a number of consecutive episodes. For example, if the rewards exceed the threshold value in 10 successive episodes, training will stop before reaching the defined 500 episodes. The reward value limit is selected for each dataset based on the best possible baseline performance.

Evaluation Metrics

We evaluate our portfolio optimisation models with the following criteria:

- **Average Daily Yield (ADY):** The mean of all the returns obtained. A higher value is better.
- **Sharpe Ratio (SHR):** The Sharpe ratio (Sharpe and Pnces 1964) is the average return earned in excess of the risk-free rate per unit of volatility or total risk. It is used to compare the portfolio's return to its risk and is defined

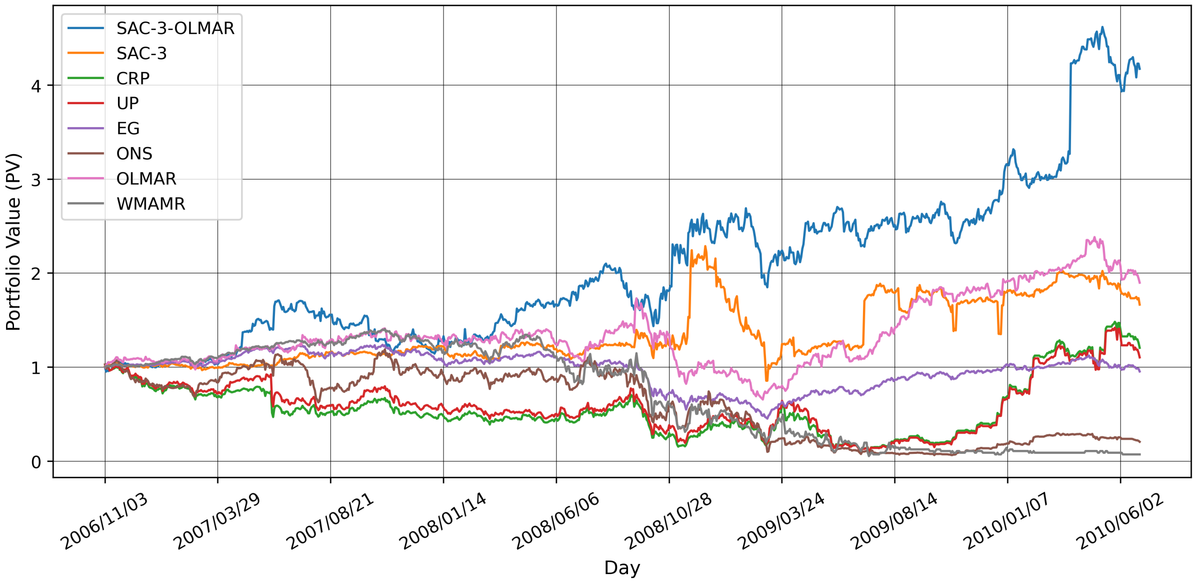

Figure 2: Portfolio Value (PV) of different methods on 30 U.S. stocks.

as follows:

$$\text{SHR} = \frac{R_p - R_f}{\sigma_p} \tag{10}$$

where R_p is the return of the portfolio, R_f is the riskfree rate and σ_p is the standard deviation of the portfolio's excess return.

- **Sortino Ratio (SOR):** Very similar to the Sharpe ratio, but instead penalises only the downside deviation α_d, the risk of losing value (Sortino and Price 1994). The formula for Sortino ratio is as follows:

$$\text{SOR} = \frac{R_p - R_f}{\alpha_d} \tag{11}$$

- **Maximum Drawdown (MDD):** The maximum loss from a peak to a trough, before a new peak is attained. In this case a lower value is better. The formula for MDD is as follows:

$$\text{MDD} = \frac{\text{Trough Value} - \text{Peak Value}}{\text{Peak Value}} \tag{12}$$

- **Portfolio Value (PV):** The total wealth an agent has after the last step has been completed. Expressed as a ratio to the initial portfolio value of 1.

Baselines

- **CRP**. Constant rebalanced portfolio (CRP) (Cover and Gluss 1986) is an investment strategy which keeps the same distribution of wealth among a set of assets from day to day. That is, the proportion of total wealth in a given asset is the same at the beginning of each day.
- **UP**. The full title of UP (Cover 1991) is μ-Weighted Universal Portfolio, μ denoting the distribution on the space of valid portfolio $\triangle_m$. As (Li et al. 2012) describe, this strategy can be interpreted as a historical performance weighted average of all valid constant rebalanced portfolios.
- **EG**. Exponentiated Gradient (EG) (Helmbold et al. 1998) is an online investment algorithm that achieves almost the same wealth as the best constant-rebalanced portfolio determined in hindsight from the actual market outcomes.
- **ONS**. Online Newton Step (ONS) (Hazan and Seshadhri 2009) is an investment algorithm that first combines optimal logarithmic regret bounds with efficient deterministic computability.
- **OLMAR**. On-Line Moving Average Reversion (OLMAR) (Li and Hoi 2012) is a method that exploits moving average reversion to overcomes the limitation of single-period mean reversion assumption.
- **WMAMR**. Weighted Moving Average Mean Reversion(WMAMR) (Gao and Zhang 2013) is a method which fully exploits past period price relatives using equalweighted moving averages and then learns portfolios by online learning techniques.

Experiment 1: Comparison of SAC-Based Model with Baselines

Our SAC-Based model was trained on the U.S., Japanese and British stock datasets using different window lengths (3, 7, 11). Next, These trained models were evaluated on the remaining $\frac{1}{6}$ of the datasets from 2006/11/03 to 2010/06/29. The datasets proved to be a difficult test for our models due to the ”Great Recession” stock market crash of 2008. Accordingly, the training subset consists of another crash in 1987, which could help train our models. The results of the best performing models with their respective time windows are shown in Table 2, 3 and 4. It is worth noting that the

Table 2: Performance Comparisons on 30 U.S. stocks. (The best value for each criteria is in boldface)

Model	ADY(%)	SHR(%)	SOR(%)	MDD(%)	PV
SAC-3-OLMAR	**0.273**	**7.348**	**10.857**	**36.112**	**4.165**
SAC-3	0.229	3.825	4.905	47.354	1.831
CRP	0.225	3.909	4.769	90.344	1.314
UP	0.210	3.410	4.204	90.381	1.207
EG	0.023	0.831	0.944	66.165	0.958
ONS	0.049	0.466	0.527	95.208	0.251
OLMAR	0.232	3.817	5.696	74.801	1.951
WMAMR	0.007	0.085	0.091	96.395	0.094

Table 3: Performance Comparisons on 30 Japanese stocks. (The best value for each criteria is in boldface)

Model	ADY(%)	SHR(%)	SOR(%)	MDD(%)	PV
SAC-11-OLMAR	**0.268**	**5.274**	**10.571**	55.801	**7.295**
SAC-11	0.224	4.851	8.323	**47.125**	6.206
CRP	0.018	0.517	0.915	63.208	0.958
UP	0.015	0.459	0.813	61.514	0.814
EG	0.013	0.426	0.906	59.323	0.920
ONS	0.008	0.016	0.045	82.317	0.107
OLMAR	0.195	1.752	3.270	95.131	2.580
WMAMR	0.039	0.377	0.355	88.203	0.354

Table 4: Performance Comparisons on 30 British stocks. (The best value for each criteria is in boldface)

Model	ADY(%)	SHR(%)	SOR(%)	MDD(%)	PV
SAC-11-OLMAR	**0.313**	**7.257**	**14.785**	**36.273**	**4.151**
SAC-11	0.270	6.003	10.207	78.132	3.207
CRP	0.013	0.656	1.016	65.732	0.970
UP	0.020	0.768	1.003	63.215	0.985
EG	0.013	0.696	0.944	60.117	0.927
ONS	0.003	0.027	0.085	96.181	0.079
OLMAR	0.137	3.580	4.674	90.585	1.879
WMAMR	0.084	0.871	0.418	93.132	0.334

model names are concatenated using the RL algorithm (i.e. SAC) and the window length (e.g. 3).

We observed that individually, SAC with a window length of 11 performed best overall on all criteria in the Japanese and British stock markets compared to online learning based methods. Similarly, SAC with a window length of 3 performed best overall in the U.S. stock market. However, *SAC-3* failed to exceed all the baselines (less than OLMAR) for two important indicators, ADY and PV, which suggests that our model still has room for further improvement.

Experiment 2: Comparison of State Augmented Model with Baselines

Our models may be improved in a variety of ways. Without any external components, the three essential elements of the RL framework are the state, action, and reward. The form of the action, which is a list of weights assigned to portfolio assets, is essential, and this structure is to remain in any future model. The forms of the state and reward, on the other hand, can be adjusted in ways that could introduce improvements. However, altering the reward function also implies a shift in the optimization objective, but this is something we do not expect to happen. For past PS methods, the state format consists only of raw market prices, which may not provide enough information to make accurate decisions. To this end, we try to improve the model performance by enhancing state representation.

From the experimental results in Table 2, 3 and 4, we observe that OLMAR is the best performing online learning based method. OLMAR exploits moving average reversion to overcome the limitation of single-period mean reversion assumption. Therefore, it is very natural for us to consider using OLMAR to extract stock movement prediction indicators from the price series for state enhancement.

To assess whether OLMAR can help our model, we included the OLMAR prediction function inside the state of our SAC model, using the same window size. The experimental results show that the state-augmented model performs best on all three datasets, as seen in Table 2, 3 and 4. It is worth mentioning that the state-augmented model *SAC-3-OLMAR* achieved the most amazing performance in the U.S. stock market, and its final portfolio value is more than twice that of *SAC-3*, as seen in Figure 2.

Conclusion

In this work, we propose a portfolio selection model using the SAC framework that can extract features with LSTM while generating portfolio weights by DRL. Next, we introduce the OLMAR prediction function within the state to further enhance state representation. Experimental results for 90 stocks (the U.S., Japanese and British markets) demonstrate that the SAC-based trading strategy is profitable, robust, and risk-aware, as compared to those baselines. Further, adding additional financial indicators to the state space is found to have a positive effect overall.

In the future, we will explore more strong decision-makers with good performance and integrate them into our strategies. Additionally, we will focus on other factors that influence stock trading, such as social news, sentiment, and politics. Finally, we will study ways to lower the annual volatility and investment risk of our DRL methods under the conditions of high returns.

References

Filos, A. 2019. Reinforcement learning for portfolio management. *arXiv preprint arXiv:1909.09571*.

Markowitz, H. 1952. Portfolio Selection. *The Journal of Finance*, 7(1): 77–91.

Samuelson, P. A. 1975. Lifetime portfolio selection by dynamic stochastic programming. *Stochastic optimization models in finance*, 517–524.

Rockafellar, R. T.; and Uryasev, S. 2002. Conditional value-at-risk for general loss distributions. *Journal of banking & finance*, 26(7): 1443–1471.

Chu, S.-C.; Tsai, P.-W.; and Pan, J.-S. 2006. Cat swarm optimization. In *Pacific Rim international conference on artificial intelligence*, 854–858. Springer.

Karimkashi, S.; and Kishk, A. A. 2010. Invasive weed optimization and its features in electromagnetics. *IEEE transactions on antennas and propagation*, 58(4): 1269–1278.

Yu, X.; Wu, W.; Liao, X.; and Han, Y. 2022. Dynamic stock-decision ensemble strategy based on deep reinforcement learning. *Applied Intelligence*, 1–19.

Kelly Jr., J. L. 1956. A New Interpretation of Information Rate. *Bell System Technical Journal*, 35(4): 917–926.

Cover, T. M.; and Ordentlich, E. 1996. Universal portfolios with side information. *IEEE Transactions on Information Theory*, 42(2): 348–363.

Gunjan, A.; and Bhattacharyya, S. 2022. A brief review of portfolio optimization techniques. *Artificial Intelligence Review*, 1–40.

Helmbold, D. P.; Schapire, R. E.; Singer, Y.; and Warmuth, M. K. 1998. On-line portfolio selection using multiplicative updates. *Mathematical Finance*, 8(4): 325–347.

Hazan, E.; and Seshadhri, C. 2009. Efficient learning algorithms for changing environments. volume 382, 50.

Cover, T. M. 1991. Universal portfolios. *Mathematical finance*, 1(1): 1–29.

Cover, T. M.; and Gluss, D. H. 1986. Empirical Bayes stock market portfolios. *Advances in Applied Mathematics*, 7(2): 170–181.

Li, B.; Zhao, P.; Hoi, S. C.; and Gopalkrishnan, V. 2012. PAMR: Passive aggressive mean reversion strategy for portfolio selection. *Machine learning*, 87(2): 221–258.

Huang, D.; Zhou, J.; Li, B.; Hoi, S. C.; and Zhou, S. 2016. Robust median reversion strategy for online portfolio selection. *IEEE Transactions on Knowledge and Data Engineering*, 28(9): 2480–2493.

Li, B.; and Hoi, S. C. 2012. On-line portfolio selection with moving average reversion. *arXiv preprint arXiv:1206.4626*.

Gao, L.; and Zhang, W. 2013. Weighted moving average passive aggressive algorithm for online portfolio selection. In *2013 5th International Conference on Intelligent Human-Machine Systems and Cybernetics*, 327–330. IEEE.

Neuneier, R. 1995. Optimal asset allocation using adaptive dynamic programming. *Advances in Neural Information Processing Systems*, 8.

Zhang, Y.; Zhao, P.; Li, B.; Wu, Q.; Huang, J.; and Tan, M. 2020. Cost-sensitive portfolio selection via deep reinforcement learning. *IEEE Transactions on knowledge and data engineering*.

Neuneier, R.; and Mihatsch, O. 1998. Risk sensitive reinforcement learning. *Advances in Neural Information Processing Systems*, 11.

Aboussalah, A. M.; and Lee, C.-G. 2020. Continuous control with stacked deep dynamic recurrent reinforcement learning for portfolio optimization. *Expert Systems with Applications*, 140: 112891.

Lee, J.; Kim, R.; Yi, S.-W.; and Kang, J. 2020. MAPS: Multi-Agent reinforcement learning-based Portfolio management System. *arXiv preprint arXiv:2007.05402*.

Betancourt, C.; and Chen, W.-H. 2021. Deep reinforcement learning for portfolio management of markets with a dynamic number of assets. *Expert Systems with Applications*, 164: 114002.

Haarnoja, T.; Zhou, A.; Abbeel, P.; and Levine, S. 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, 1861–1870. PMLR.

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

Noonpakdee, W. 2020. The adoption of artificial intelligence for financial investment service. In *2020 22nd International Conference on Advanced Communication Technology (ICACT)*, 396–400. IEEE.

Deng, Y.; Xu, H.; and Wu, J. 2021. Optimization of blockchain investment portfolio under artificial bee colony algorithm. *Journal of Computational and Applied Mathematics*, 385: 113199.

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

Ban, G.-Y.; El Karoui, N.; and Lim, A. E. 2018. Machine learning and portfolio optimization. *Management Science*, 64(3): 1136–1154.

Chaouki, A.; Hardiman, S.; Schmidt, C.; Sérié, E.; and De Lataillade, J. 2020. Deep deterministic portfolio optimization. *The Journal of Finance and Data Science*, 6: 16–30.

Buehler, H.; Gonon, L.; Teichmann, J.; and Wood, B. 2019. Deep hedging. *Quantitative Finance*, 19(8): 1271–1291.

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

Lillicrap, T. P.; Hunt, J. J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; and Wierstra, D. 2015. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*.

Yu, P.; Lee, J. S.; Kulyatin, I.; Shi, Z.; and Dasgupta, S. 2019. Model-based deep reinforcement learning for dynamic portfolio optimization. *arXiv preprint arXiv:1901.08740*.