深度强化学习在金融资产配置中的应用

梁智鹏*†, 蒋康康*†, 陈昊*†, 祝俊浩*†, 李嫣然*†
*似然科技
†中山大学

{*liangzhp*6, *jiangkk*3, *chenhao*348, *zhujh*25, *liyr*8}*@mail*2.*sysu.edu.cn*

***Abstract*—**深度强化学习对超参数、网络结构等因素具有高度敏感性。在本文中，我们实现了两种目前流行的连续深度强化学习算法——深度确定性策略梯度 **(DDPG)** 和近端策略优化 **(PPO)** 并讨论了它们在不同设定下的表现，以便于有兴趣从事相关研究的人员提供调参、特征选择和数据处理方面的帮助。我们还会提出一些深度强化学习在资产配置中的发展方向。

***Index Terms*—**强化学习, 资产配置, 深度学习

I. 引言

在算法交易领域，强化学习在资产配置中的应用越来越广泛。但是，深度学习对于神经网络结构、数据质量等及其敏感。因此，在我们的实验中，我们利用两种深度强化学习算法：深层确定策略梯度（DDPG）和近端策略优化（PPO），探讨了不同优化器和网络结构对交易代理人训练的影响，并分别在中国和美国股票市场的数据集上进行实验。我们的代码已经在 github 开源。[1].

II. 概要

这篇文章主要包括三个领域：资产配置、强化学习、深度学习。首先，资产配置是在一定时期内通过在固定资产组合下动态调整权重以获得高回报以及低风险的方法。在此之前，已有几种主要的方法，包括“跟随赢家”方法，“跟随失败者”的方法，基于“模式匹配”的方法和“元学习算法”。深层加强学习实际上是“模式匹配”和“元学习”的结合 [1]。

强化学习是一种通过与环境交互学习，不断进行反复试验逐步优化，并利用马尔可夫决策过程来解决问题的方法。作为序列决策问题，资产配置问题也可以通过强化学习来解决。Xin Du 等人在强化学习中构建了 Q-Learning 和策略梯度算法，并发现相比于基于价值函数的搜索算法，直接使用强化学习算法（策略搜索）能够更简单地表示问题 [2]。Saud Almahdi 等人扩展了循环强化学习，并以预期的最大回测作为目标函数下建立最优的可变权重组合分配方法 [3]。Xiu Gao 等人分别使用了绝对收益和相对的损失调整后的收益作为评价函数来训练系统并且使用了两组网络。它们发现这样的改进能够在外汇市场上产生更加高的收益 [4]。

由于深度学习的蓬勃发展，它在提取复杂特征方面的能力得到的广泛关注，并且仅需要极其少量的特征工程工作量就能在机器控制、游戏等方面取得很好的成绩并实现端对端的学习 [5]。值函数近似方法是求解大规模动态规划方程的主流方法之一 [6]。深度 Q-learning，使用神经网络代替 Q 值函数并使用回放记忆进行学习，在不改变网络结构的情况下，表现优于传统所有的 Atari 游戏方法 [7]。DDPG 是我们为实验选择的算法之一，它使用 actor-critic 框架而不是仅仅使用 actor 来稳定训练过程，采样效率同时得到了提高 [8]。另一种算法（PPO），是经过推导出的一种单调改进策略的方法 [9]。

由于金融市场数据中复杂的非线性模式和低信噪比，深度强化学习在资产配置中的应用具有相当好的前景。Zhengyao Jiang 给出了一个深度强化学习的框架，并通过实验证明强化学习可以胜过传统的策略 [10]。Yifeng Guo 等改进对数最优策略并将其与强化学习相结合 [12] 。但是，上述大部分实验都是使用的美国的数据，在更具波动性的中国市场中具有较低的参考价值。更重要的是，前人的工作中很少针对资产数量或多个功能组合对最终收益的影响进行讨论。

为了深入了解强化学习在资产配置中的真实性能和缺陷，我们在这个项目中选择了主流的强化学习算法（DDPG 和 PPO），并基于更复杂的国市场股票数据进行不同超参数，网络结构等进行深入实验。

这篇文章的构成如下：在第二部分我们将会正式地建立资产配置模型。我们将会展示交易成本的存在使得整个问题从一个单纯的预测问题、能够用贪心策略求得最优策略，转化为了计算量巨大的动态规划问题。大部分强化学习算法聚焦于游戏和机器人控制上，而我们将会展示一些在金融中的特点，来说明强化学习算法运用在资产配置领域的确需要进行改进。第三部分我们将会进入实验部分，我们将会介绍数据预处理、算法和我们对于不同超参数的选择对最终累计资产收益的影响。第四部分我们将会展示我们的实验结果。第五部分我们将会总结并提出深度强化学习在资产配置方面应用的未来方向。

III. 问题设定

给定一个时期，以一年为例，股票交易者在预算约束条件下，将其财富分配到一组资产中，追求利润最大化。在我们的实验中，我们假设股票市场是连续的，也就是说，每一天的收盘价等于第二天的开盘价。每天交易代理人通过分析数据观察股市然后重新做出投资决策。同时，我们假设代理将会在临近收盘前进行重新配置，这表明所有的交易都可以以收盘价格完成。此外，我们的实验已经考虑了交易成本的影响。

正式来说，我们的资产配置项目有 m + 1 支股票，其中包括 m 支有风险股票和 1 支无风险资产。在没有通货膨胀压力的条件下，我们选择现金作为无风险资产。设 t 个交易时期后，第 i 支股票为 $v_{i,t}^{close}$, 我们将所有资产在时间 t 的价格用向量 $v_{i,t}$ 表示。所有股票在 t 时区的收盘价整合成向量 v_t^{close}。马尔可夫决策过程模型要求下一个状

[1]https://github.com/qq303067814/Reinforcement-learning-in-portfolio-management-

态仅取决于历史状态和动作，因此，(s, a, r, s_{netx}) 将用于描述整个金融资产配置问题。

值得注意的是，在马尔可夫决策过程中，我们使用折扣函数 $\sum_{t=1}^{T} \gamma^t R(s_t, a_t)$ 作为我们的求解对象, 然而，在资产配置问题中，t 时获得的财富将被用于下一次投资，这说明，在时区 T 的财富 $P_T = \prod_{t=1}^{T} P_0 r_t$ 是一个求积形式而非求和形式。这里我们采用取对数的方法将财富总额的求积形式转换为更易于求解的求和形式。

在阐明马尔可夫决策过程之前，我们采用一些符号来简化：定义 $y_t = \frac{v_t}{v_{t-1}} = (1, \frac{v_{1,t}}{v_{1,t-1}}, \ldots, \frac{v_{m,t}}{v_{m,t-1}})^T$ 为价格波动向量；$w_{t-1} = (w_{0,t-1}, w_{1,t-1}, \ldots, w_{m,t-1})^T$ 为分配给每个资产的金额，且可以在第二天再投资。假设初始的投资金额为 P_0。

- 状态 (s): 一个状态包括先前股票价格和一些其他财务指标。

- 行动 (a): 期望的分配权重，$a_{t-1} = (a_{0,t-1}, a_{1,t-1}, \ldots, a_{m,t-1})^T$ 是 $t-1$ 时的分配向量，而我们有 $\sum_{i=0}^{n} a_{it-1} = 1$ 的限制。由于一天中的价格变动，当天开始时的权重 a_{t-1} 将在一天结束时演变为 w_{t-1}：

$$\mathbf{w_{t-1}} = \frac{\mathbf{y_{t-1}} \odot \mathbf{a_{t-1}}}{\mathbf{y_{t-1}} \cdot \mathbf{a_{t-1}}}$$

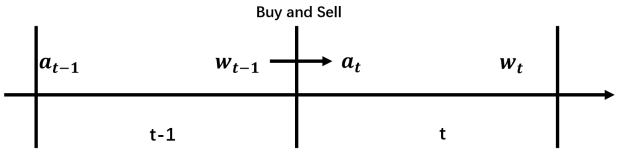

图 1. 资产权重向量的转变

- 回报 (r): 财富的波动减去交易成本。财富的波动为：$a_{t-1}^T \cdot y_{t-1}$，减去交易成本后为：$a_{t-1}^T \cdot y_{t-1}$。具体地，我们设置 $\mu = 0.25\%$。总之，我们可以在时间 t-1 得到奖励：

$$r_t(s_t, a_t) = \log(\mathbf{a_{t-1}} \cdot \mathbf{y_{t-1}} - \mu \sum_{i=1}^{m} |a_{i,t-1} - w_{i,t-1}|)$$

交易成本的引入会暴露出一些传统交易策略例如跟随赢家，跟随输家等的重大短板——即使我们能够准确预测未来的所有股票价格，要想在区间较长时获得最优策略，仍然需要相当大的计算量。我们发现，在不考虑交易成本时，贪心算法可以实现最佳利润。具体而言，最佳分配策略就是将所有财富分配到具有最高预测增长率的股票中。然而，在引入交易成本的条件下，如果交易成本超过了即时回报，则交易成本的存在可能会抵消大部分收益，以使得该行动不再是最优的决策向量。

尽管先前很多文献已经讨论了马尔可夫的决策过程，但资产配置的特点使其很具有挑战性。首先，股票数据中包含的大量噪声会导致价格扭曲。对股票价格、财务指标的观察很难反映下一个状态的情况，这将会使算法的表现效果不尽人意。

其次，不同状态的转换概率仍然是未知的。在我们尝试解决这种高维动态规划问题之前，我们必须先对环境进行学习。虽然买卖股票必须是按手数计算，但在这里我们仍然假设动作空间是连续的。事实上，当财富远远超过股票价格时，这种简化不失一般性。

IV. 深度强化学习

强化学习，尤其是结合了深度学习的深度强化学习，是解决资产分配问题一种较好的备选方案。强化学习是代理和环境通过实验不断交互以达到策略优化的一种机器学习方法，这种方法能够利用较少的信息达到较好的优化效果，因此它对环境的建模要求低，适用于变幻莫测的金融领域。目前，强化学习作为一种新兴的机器学习方法，其表现要优于传统方法，特别是在一些复杂的非线性问题上，比如语音识别以及图像检测等方面。

相比于单一地使用深度学习或者强化学习，深度强化学习吸取了两者的优点：

其一，作为一种完全的人工智能算法，强化学习能分析股票市场的信息输入而输出对应的资产分配决策，它不需要人进行股票价格走势的分析，并且能达到不断自我提升的效果。

其二，由于深度强化学习算法避免了预测未来股票价格的走势这一复杂的过程，因此它能够更简单地进行自我改进。

其三，相较于传统的强化学习算法，深度强化学习利用了神经网络模拟策略函数或者值函数，这不仅可以通过改变神经网络结构提高算法的灵活性，还可以避免“维度诅咒”，使得大规模资产的管理变得可行。

目前，学界和业界提出了一系列不同的强化学习方法，比如 PG, dual Q network, DDPG, PPO 等。我们对 DDPG 以及 PPO 算法进行实验，以测试他们在资产配置中的潜力。

A. DDPG 算法

DDPG 是结合了 Q-learning 和 PG 算法，并且在 DPG 算法的基础上成功地使用神经网络作为值函数估计模型的一种深度强化学习算法 [13]。为了阐明它的思想，我们会简单介绍一下 Q-learning 和 PG 算法，从而引出 DDPG 算法。

Q-learning 是一种基于 Q-函数的强化学习方法。Q 函数指的是对于当前状态 s 以及在对应策略下采取动作 a 的价值评估函数。利用贝尔曼方程的性质，我们有：

$$Q^{\pi}(s_t, a_t) = \mathbb{E}_{r_t, s_{t+1} \sim E}[r(s_t, a_t) + \gamma \mathbb{E}_{a_{t+1} \sim \pi}[Q^{\pi}(s_{t+1}, a_{t+1})]]$$

对于一个确定的策略函数 $\mu : S \rightarrow A$ 上述方程可以变成：

$$Q^{\pi}(s_t, a_t) = \mathbb{E}_{r_t, s_{t+1} \sim E}[r(s_t, a_t) + \gamma[Q^{\mu}(s_{t+1}, \mu(s_{t+1}))]]$$

具体来讲,Q-learning 采取了一个贪心策略：

$$\mu(s) = \arg\max_a Q(s, a)$$

深度强化学习采用了神经网络来估计 Q 函数，并使用回放学习等方法来提高其收敛到最优策略的可能性。在估计 Q 函数时，我们不采用传统的迭代方法，而是选择最小化下面的损失函数。

$$L(\theta^Q) = \mathbb{E}_{s_t \sim \rho^{\beta}, a_t \sim \beta, r_t \sim}[(Q(s_t, a_t | \theta^Q) - y_t)^2]$$

其中

$$y_t = r(s_t, a_t) + \gamma Q(s_{t+1}, \mu(s_{t+1})|\theta^Q)$$

值得一提的是，这里的 y_t 是由另一个 target 网络来给出的，target 网络与 online 网络的使用使得算法收敛的可能性得到提高。解决连续控制问题时，简单地使用 Q-learning 是较为棘手的，因为动作空间过大时将引起“维度诅咒”现象。并且，如果 Q 函数没有较好的性质（比如凸性），那么寻找最优的策略也较为困难。解决连续控制问题时，一般使用的方法为 PG 算法，它可以直接输出一个动作。策略可以计算并且通过设置目标函数进行优化。重新回顾马尔可夫决策过程要解决的问题：找到使得目标函数最大化的最优的策略。为此，我们将问题用参数 theta 表示：

$$\begin{aligned}
\tau &= (s_1, a_1, s_2, a_2, \dots) \\
J(\pi_\theta) &= \mathbb{E}_{\tau\sim p_\theta(\tau)}[\sum_t \gamma^t r(s_t, a_t)] \\
\pi_{\theta^*} &= \arg\max_{\pi_\theta} J(\pi_\theta) \\
&= \arg\max_{\pi_\theta} \mathbb{E}_{\tau\sim p_\theta(\tau)}[\sum_t \gamma^t r(s_t, a_t)] \\
&= \arg\max_{\pi_\theta} \mathbb{E}_{\tau\sim p_\theta(\tau)}[r(\tau)] \\
&= \arg\max_{\pi_\theta} \int \pi_\theta(\tau) r(\tau) d\tau
\end{aligned}$$

在强化学习中，梯度下降是最常用的优化目标函数的方法。假设决策的时间是有限时，我们可以得到下面的方程。它可以表示成下列乘积的形式：

$$\begin{aligned}
\pi_\theta(\tau) &= \pi_\theta(s_1, a_1, \dots, s_T, a_T) \\
&= p(s_1) \prod_{t=1}^{T} \pi_\theta(a_t|s_t) p(s_{t+1}|s_t, a_t)
\end{aligned}$$

然而，这样连乘的形式求导较难。为了让计算更为方便，我们通过一个变换将其梯度变为求和形式：

$$\begin{aligned}
\nabla_\theta \pi_\theta(\tau) &= \pi_\theta(\tau) \frac{\nabla_\theta \pi_\theta(\tau)}{\pi_\theta(\tau)} \\
&= \pi_\theta(\tau) \nabla_\theta \log \pi_\theta(\tau) \\
\nabla_\theta \log \pi_\theta(\tau) &= \nabla_\theta (\log p(s_1) + \sum_{t=1}^{T} \log \pi_\theta(a_t|s_t) + \log p(s_{t+1})) \\
&= \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a_t, s_t)
\end{aligned}$$

因此，我们利用上述公式可以将策略梯度重新表达为：

$$\begin{aligned}
\nabla J(\pi_\theta) &= \mathbb{E}_{\tau\sim\pi_\theta(\tau)}[r(\tau)] \\
&= \mathbb{E}_{\tau\sim\pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)] \\
&= \mathbb{E}_{\tau\sim\pi_\theta(\tau)}[(\sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t))(\sum_{t=1}^{T} \gamma^t r(s_t, a_t))]
\end{aligned}$$

在 DDPG 算法中，需要用到四个网络：online actor,online critic,target actor,target critic。将 Q-Learning 与策略梯度法结合后，actor 网络对应的是动作函数 μ 表示，critic 网络对应的是 Q 函数。代理探索环境，其中 actor 会给出一个连续空间中的动作。Online critic 网络将会计算评估 actor 的动作并更新 online actor 网络，而 target critic 与 target actor 网络用于更新 online critic 的网络。DDPG 更新的方法如下：对于 online actor 其梯度的估计为：

$$\begin{aligned}
\nabla_{\theta^\mu} J &\approx \mathbb{E}_{s_t\sim\rho^\beta}[\nabla_{\theta^\mu} Q(s, a|\theta^Q)|_{s=s_t, a=\mu(s_t|\theta^\mu)}] \\
&= \mathbb{E}_{s_t\sim\rho^\beta}[\nabla_a Q(s, a|\theta^Q)|_{s=s_t, a=\mu(s_t)} \nabla_{\theta^\mu} \mu(s|\theta^\mu)|_{s=s_t}]
\end{aligned}$$

对于 online critic 网络，更新的方法类似。Target actor 以及 target critic 将会缓慢地对 online actor 与 online critic 网络进行更新。具体细节我们放在 Algorithm 处：

Algorithm 1 DDPG

Randomly initialize actor $\mu(s|\theta^\mu)$ and critic $Q(s, a|\theta^Q)$
Create Q' and μ' by $\theta^{Q'} \to \theta^Q$,$\theta^{\mu'} \to \theta^\mu$
Initialize replay buffer R
for $i = 1$ to M **do**
  Initialize a UO process $\mathcal{N}$
  Receive initial observation state s_1
  for $t = 1$ to T **do**
    Select action $a_t = \mu(s_t|\theta^\mu) + \mathcal{N}_t$
    Execute action a_t and observe r_t and s_{t+1}
    Save transition (s_t,a_t,r_t,s_{t+1}) in R
    Sample a random minibatch of N transitions (s_i,a_i,r_i,s_{i+1}) in R
    Set $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$
    Update critic by minimizing the loss:$L = \frac{1}{N}\sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$
    Update actor policy by policy gradient:

$$\begin{aligned}
&\nabla_{\theta^\mu} J \\
&\approx \frac{1}{N} \sum_i \nabla_{\theta^\mu} Q(s, a|\theta^Q)|_{s=s_t, a=\mu(s_t|\theta^\mu)} \nabla_{\theta^\mu} \mu(s|\theta^\mu)|_{s_t}
\end{aligned} \tag{1}$$

    Update the target networks:

$$\theta^{Q'} \to \tau\theta^Q + (1-\tau)\theta^{Q'}$$

$$\theta^{\mu'} \to \tau\theta^\mu + (1-\tau)\theta^{\mu'}$$

  end for
end for

B. Proximal Policy Optimization

大部分策略优化算法都能被分为三大类：(1) 策略迭代 (2) 策略梯度 (3) 无需微分的优化方法。近端策略优化算法 (PPO) 则是第二类算法。因为 PPO 是基于信任区域策略优化 (TRPO) 算法 [14]，我们先介绍 TRPO 然后再介绍 PPO。

TRPO 成功找到策略提升的下界因此策略优化能够采用代理目标函数。这样能够保证每次迭代中单调递增的策略提升。

正式地，让随机策略记为 $\pi : S \times A \to [0, 1]$。随机策略则是动作空间中的概率分布且分布可随着状态改变，以表示所有动作在该状态下的合适程度。我们定义 $\eta(\pi)$ 为策略 π 的价值函数。

$$\eta(\pi) = \mathbb{E}_{s_0,a_0,\ldots}[\sum_{t=0}^{\infty}\gamma^t r(s_t)]$$

$$s_0 \sim \rho_0(s_0), a_t \sim \pi(a_t|s_t), s_{t+1} \sim P(s_{t+1}, a_{t+1}|s_t, a_t)$$

按照状态-动作值函数 Q_π、值函数 V_π 和优势函数的标准定义：

$$V_\pi(s_t) = \mathbb{E}_{a_t,s_{t+1},\ldots}[\sum_{l=0}^{\infty}\gamma^l r(s_{t+l})]$$

$$Q_\pi(s_t, a_t) = \mathbb{E}_{s_{t+1},a_{t+1},\ldots}[\sum_{l=0}^{\infty}\gamma^l r(s_{t+l})]$$

$$A_\pi(s, a) = Q_\pi(s, a) - V_\pi(s)$$

新策略 $\tilde{\pi}$ 超出原策略 π 的预期回报能够用优势函数在时间上的累加和来表示：

$$\eta(\tilde{\pi}) = \eta(\pi) + \mathbb{E}_{s_0,a_0,\cdots\sim\tilde{\pi}}[\sum_{t=0}^{\infty}\gamma^t A_\pi(s_t, a_t)]$$

上述方程可以重新用状态来表示：

$$\begin{aligned}\eta(\tilde{\pi}) &= \eta(\pi) + \sum_{t=0}^{\infty}\sum_{s} P(s_t = s|\tilde{\pi})\sum_{a}\tilde{\pi}(a|s)\gamma^t A_\pi(s, a)\\ &= \eta(\pi) + \sum_{s}\sum_{t=0}^{\infty}\gamma^t P(s_t = s|\tilde{\pi})\sum_{a}\tilde{\pi}(a|s)A_\pi(s, a)\\ &= \eta(\pi) + \sum_{s}\rho_{\tilde{\pi}}(s)\sum_{a}\tilde{\pi}(a|s)A_\pi(s, a)\end{aligned}$$

此处 $\rho_{\tilde{\pi}} = P(s_0 = s) + \gamma P(s_1 = s) + \gamma^2 P(s_2 = s) + \cdots$ 表示给定策略 $\tilde{\pi}$ 对状态 s 在未来无限期的折现访问频率。

但是，由于依赖于策略 $\tilde{\pi}$ 所带来的复杂度使得该等式难以计算。因此 TRPO 提出下列局部近似进行替代。

$$L_\pi(\tilde{\pi}) = \eta(\pi) + \sum_{s}\rho_\pi(s)\sum_{a}\tilde{\pi}(a|s)A_\pi(s, a)$$

作为 TRPO 算法的核心结论之一，策略提升的下界提供了单调策略提升的理论保障：

$$\eta(\pi_{new}) \geq L_{\pi_{old}}(\pi_{new}) - \frac{4\epsilon\gamma}{(1-\gamma)^2}\alpha^2$$

$$\begin{aligned}\epsilon &= \max_{s,a}|A_\pi(s, a)|\\ \alpha &= D_{TV}^{max}(\pi_{old}, \pi_{new})\\ &= \max_{s} D_{TV}(\pi_{old}(\cdot|s)||\pi_{new}(\cdot|s))\end{aligned}$$

$D_{TV}(p||q) = \frac{1}{2}\sum_i |p_i - q_i|$ 表示是两个离散概率分布的总方差散度。

由于 $D_{KL}(p||q) \geq D_TV(p||q)^2$, 我们可以得到下列不等式并用于构建算法。

$$\eta(\tilde{\pi}) \geq L_\pi(\tilde{\pi}) - CD_{KL}^{max}(\pi, \tilde{\pi})$$

$$C = \frac{4\epsilon\gamma}{(1-\gamma)^2}$$

$$D_{KL}^{max}(\pi, \tilde{\pi}) = \max_{s} D_{KL}(\pi(\cdot|s)||\tilde{\pi}(\cdot|s))$$

上述等式的证明都可以在 [14] 中找到。

进一步分析，令 $M_i(\pi) = L_{\pi_i}(\pi) - CD_{KL}^{max}(\pi_i, \pi)$。两个重要性质如下：

$$\eta(\pi_i) = M_i(\pi_i)$$

$$\eta(\pi_{i+1}) \geq M_i(\pi_{i+1})$$

因此，TRPO 论文中给出策略提升的下界：

$$\eta(\pi_{i+1}) - \eta(\pi_i) \geq M_i(\pi_{i+1}) - M_i(\pi_i)$$

因此，通过在每一轮迭代中优化代理目标函数 M_i, 我们能够保证真正的目标 η 不减. 考虑参数化的策略 π_{θ_i}, 策略优化被转化为:

$$\max_{\pi_{\theta_i}}[L_{\pi_{\theta_{i-1}}}(\pi_{\theta_i}) - CD_{KL}^{max}(\pi_{\theta_{i-1}}, \pi_{\theta_i})]$$

但是，从理论结果中得到的惩罚系数 C 会使得策略提升速度过慢。因此在最终的 TRPO 算法中，在详细考虑了目标函数的结构后，新的优化问题被提出：

$$\begin{aligned}&\max_{\pi_{\theta_i}} L_{\pi_{\theta_i}}\\ &s.t. \quad \overline{D}_{KL}^{\rho_{\pi_{\theta_{i-1}}}}(\pi_{\theta_{i-1}}, \pi_{\theta_i}) \leq \delta\end{aligned}$$

$\overline{D}_{KL}^{\rho}(\pi_{\theta_1}, \pi_{\theta_2}) = \mathbb{E}_{s\sim\rho}[D_{KL}(\pi_{\theta_1}(\cdot|s)||\pi_{\theta_2}(\cdot|s))]$

更多近似方法被提出来使得上述优化问题变得可解。回忆原始的优化问题可以被写作：

$$\max_{\pi_\theta}\sum_{s}\rho\pi_{\theta_{i-1}}(s)\sum_{a}\pi_{\theta_i}(a|s)A_{\theta_{i-1}}(s, a)$$

在进行一系列近似包括重要性取样后，最终 TRPO 给出的优化问题为：

$$\begin{aligned}&\max_{\pi_{\theta_i}} \mathbb{E}_{s\sim\rho\pi_{\theta_{i-1}},a\sim q}[\frac{\pi_{\theta_i}(a|s)}{q(a|s)}A_{\pi_{\theta_{i-1}}}(s, a)]\\ &s.t. \quad \mathbb{E}_{s\sim\rho\pi_{\theta_{i-1}}}[D_{KL}(\pi_{\theta_{i-1}}(\cdot|s)||\pi_{\theta_i}(\cdot|s))] \leq \delta\end{aligned}$$

PPO 算法由此提出 [9]：PPO 提出了新的代理函数来简化 TRPO 算法，其中一种函数是“修剪代理函数”，也就是我们在实验中选择的目标函数。让我们先定义 $r(\theta) = \frac{\pi_\theta(a|s)}{\pi_{\pi_{old}}(a|s)}$. 修剪代理函数能够被写为：

$$L^{CLIP}(\theta) = \mathbb{E}[\min(r(\theta)A, clip(r(\theta), 1-\epsilon, 1+\epsilon)A)]$$

新的代理目标函数能够以更为简单的方式约束每次迭代的策略提升幅度，并且经过实验发现这种简化能够在抽样效率方面有所提升。

Algorithm 2 PPO

Initialize actor $\mu : S \to \mathbb{R}^{m+1}$ and $\sigma : S \to diag(\sigma_1, \sigma_2, \cdots, \sigma_{m+1})$
for $i = 1$ to M **do**
  Run policy $\pi_\theta \sim N(\mu(s), \sigma(s))$ for T timesteps and collect (s_t, a_t, r_t)
  Estimate advantages $\hat{A}_t = \sum_{t'>t} \gamma^{t'-t} r_{t'} - V(s_t)$
  Update old policy $\pi_{old} \leftarrow \pi_\theta$
  for $j = 1$ to N **do**
    Update actor policy by policy gradient:

$$\sum_i \nabla_\theta L_i^{CLIP}(\theta)$$

    Update critic by:

$$\nabla L(\phi) = -\sum_{t=1}^{T} \nabla \hat{A}_t{}^2$$

  end for
end for

V. 实验

A. 数据预处理

我们的实验数据来源为 investing[2]、万得[3]。我们分别在两个市场当中挑选低相关甚至是负相关的股票集合，来展示我们的代理在不同资产中的配置能力。为了维持我们的假设，我们选择交易量大的股票来确保我们的行为不会影响市场。在中国股票市场，我们选择了 18 支股票来测试我们代理在大规模资产配置问题上的表现。在美国股票市场上，我们选择了 6 支股票。另外，我们选择过去三年作为我们训练和测试时期。特别地，我们选择 2015 年 1 月 1 日到 2016 年 12 月 31 日作为我们的训练集，2017 年 1 月 1 日到 2018 年 1 月 1 日为回测时期，我们选择的股票代码如表 1。

market	code	market	code
China	000725	USA	AAPL
China	000002	USA	ADBE
China	600000	USA	BABA
China	000862	USA	SNE
China	600662	USA	V
China	002066		
China	600326		
China	000011		
China	600698		
China	600679		
China	600821		
China	600876		
China	600821		
China	000151		
China	000985		
China	600962		

表 I
实验所用的股票代码

[2]https://lpi.invest.com/invest.com&bittrex
[3]http://www.wind.com.cn/

B. 网络结构

受到 Jiang 等人的启发，我们使用被称为“相同独立评估器”（(IEE) 的技巧。“相同独立评估器”是指使用网络独立地评估 m+1 个资产但同时用于评估不同资产的网络参数是共享的。网络每次评估一个股票并且输出一个表示投资于此倾向的标量。接下来这 m+1 个标量将会被 softmax 算法归一化并且作为下一时期的权重向量。IEE 相比于整体的神经网络具有一些十分重要的优势，包括对资产组合规模的可拓展性、数据利用的高效率和对资产组合种类的弹性。鉴于篇幅原因，在此不再赘述，更加深入的解释请查看 [10]。

我们发现在大部分关于深度学习运用于资产配置的工作中，CNN 神经网络都能比 RNN 和 LSTM 表现得更好。但是，和 Jiang 等人的工作不同的是，我们会将为 CNN 神经网络新增了残差块。神经网络的深度对它的表现具有重要影响。但是，传统 CNN 网络会因为深度增加导致的梯度消失或梯度爆炸而无法继续加深。深度残差网络通过加入短路使得不同层级可以直接连接更深层级来解决这个问题。这样可以避免网络由于深度的增加而恶化。深度残差网络已经在图像识别取得很卓越的成绩，并且对深度学习的发展做出了重要贡献。[11]

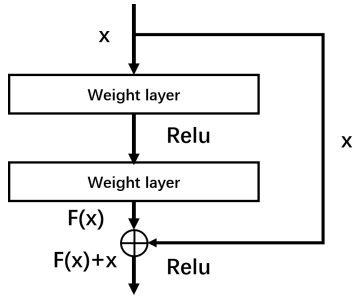

图 2. 残差模块

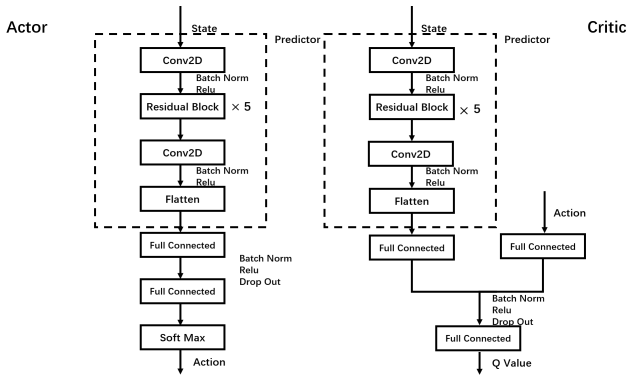

图 3. 实验中在 DDPG 算法中使用的网络结构

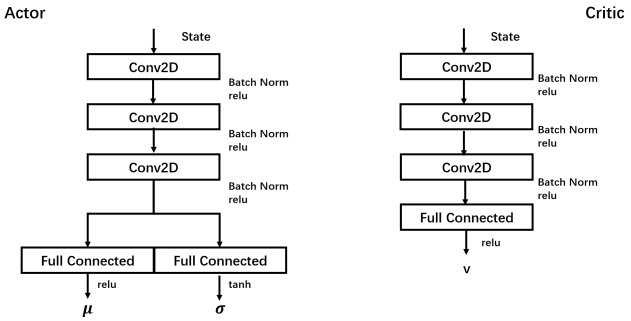

图 4. 实验中在 PPO 算法中使用的网络结构

Algorithm	DDPG		PPO	
	Actor	Critic	Actor	Critic
Optimizer	Adam	Adam	GradientDescent	GradientDescent
Learning Rate	10^{-3}	10^{-1}	10^{-3}	10^{-3}
τ	10^{-2}	10^{-2}	10^{-2}	10^{-2}

表 II
实验中采用的超参数设置

C. 结果

1) 学习率: 学习率在神经网络网络训练中起着至关重要的作用。但是，它极其敏感。高的学习率会使训练时的损失函数值在开始时快速下降，但经常会降低到局部最小值，甚至会使最佳解决方案剧烈振动而无法达到最优值。过低的学习率也会使训练的损失函数值减小得非常缓慢，以至于即使经过长时间的训练都无法达到最优值。只有适当的学习率才能帮助神经网络网络获得满意的训练结果。

因此，我们通过改变 actor 和 critic 的学习率并记录 critic 的 loss 来测试基于 DDPG 的神经网络在不同学习率下的表现。结果表明，在 actor 的学习率不直接控制 critic 训练的情况下，学习率的大小对 critic 的 loss 有显著影响。我们发现，当 actor 学习新的动作时，critic 的 loss 会有所波动。这表明 critic 在遇到新状态时没有足够的泛化能力。只有当 actor 变得稳定时，critic 的 loss 才会减少。

2) 风险: 由于训练数据的局限性，我们的强化学习算法在训练时可能低估了风险，这在实际的交易环境中可能会导致其性能的灾难性恶化。不同的金融方法可以帮助评估当前的投资组合风险，以减轻偏颇的训练数据的影响。受 [3] 和目标函数经过风险调整的 [11] 的启发，我们修改我们的目标函数如下：

$$R = \sum_{t=1}^{T} \gamma^t (r(s_t, a_t) - \beta \sigma_t^2)$$

其中，$\sigma_t^2 = \frac{1}{L} \sum_{t'=t-L+1}^{t} \sum_{i=1}^{m+1} (y_{i,t'} - \overline{y_{i,t'}})^2 \cdot w_{i,t}$ 和 $\overline{y_{i,t'}} = \frac{1}{L} \sum_{t'=t-L+1}^{t} y_{i,t'}$ 用于衡量最后 L 天第 i 支股票回报的波动性。目标函数是通过降低高波动性资产的投资收益来惩罚的，这样会使我们的投资组合暴露在风险中。

然而，结果似乎不太理想。我们同样采用夏普率形式的目标函数进行了实验，但是还是失败了。实际上，即期收益的构造是设计强化学习算法当中其中一个核心话题 [16]。我们的修改似乎使目标函数变得过于复杂。

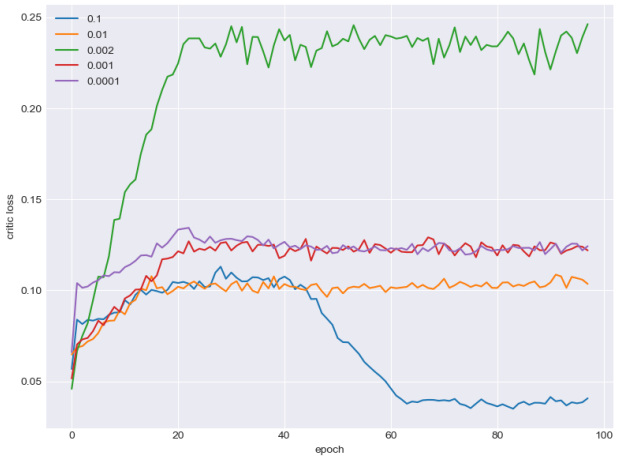

图 5. actor 采用不同学习率下 critic 的损失

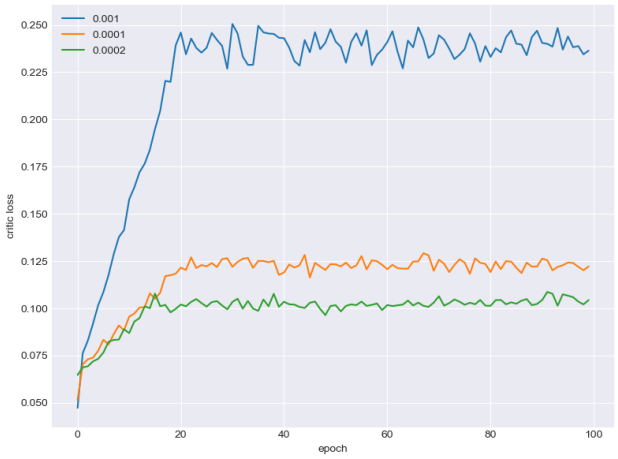

图 6. critic 采用不同学习率下 critic 的损失

3) 特征组合: 据我们所知，先前很少有工作讨论强化学习在资产配置中的特征组合问题。端到端的游戏或机器人控制的输入是像素，而在资产配置中，我们可以考虑资产组合丰富的特征。常见特征包括收盘价，开盘价，当日最高价，当日最低价和交易量。此外，长期分析的财务指标，如市盈率（PE），市净率（PB）也可以为我们提供对市场变动的见解。

但是，添加不相关的特征会增加噪音并使训练变差，权衡相应特征是特征选择的主题。因此，我们在不同的特征组合下进行实验，这些特征为：只有收盘价，收盘价和当日最高价，收盘价和开盘价，收盘价和当日最低价。结果表明，特征组合在训练过程中具有重要意义，而选择收盘价和当日最高价可以帮助代理获得最佳收益。

4) 训练和测试: 经过上述实验，我们得到了一组令人满意的超参数和特征组合。在这样的背景下，我们对中国股票市场和美国股票市场进行了 1000 个阶段的训练。结果表明训练可以增加累积投资组合值（APV），同时降低收益的波动性。

然后我们对美国数据进行代理测试。增强学习代理获得了所有代理中的最高 APV。然而，它的风险（最大回

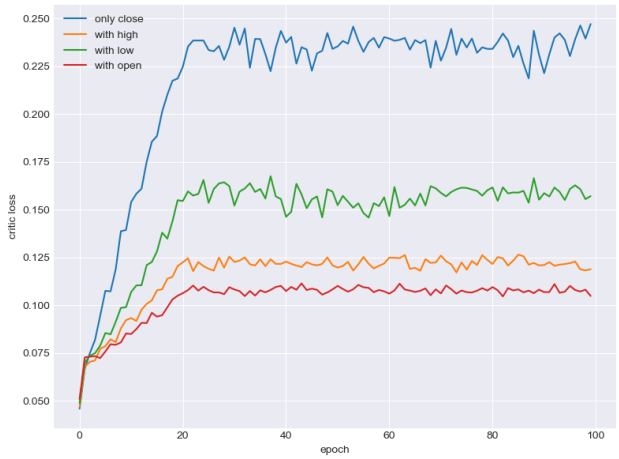

图 7. 不同收益组合对于评价者损失的影响

图 8. 不同特征组合对于收益的影响

撤）也是最大的。PPO 的令人不太满意的结果同样揭示了游戏、机器人控制和资产配置之间的显著的不同。随机策略看起来并不适合不稳定、低信噪比的金融市场环境，尽管它拥有极佳的理论性质——策略的单调提升和更高的采样效率。

	APV(%)	Sharpe Ratio(%)	Maximum Drawdown
DDPG	**159.904**	1.379	0.164
PPO	106.520	0.867	0.107
Winner	66.484	**1.411**	0.160
Loser	57.662	1.150	0.066
UCRP	144.371	0.695	**0.0482**

表 III
不同策略的表现

在中国市场上的回测结果令人沮丧，看起来我们的代理没有学习。我们之后会讨论可能的原因。

图 9. Comparison of portfolio value before and after learning in training data of China stock market

图 10. Comparison of portfolio value before and after learning of America stock market

VI. 未来方向

由于投资组合管理的特性，强化学习在资产配置上还有很多用武之地。在以后的研究中，我们将尝试使用其他指标来衡量资产配置的风险，并致力于将金融学中的知识和传统模型相结合，使强化学习算法在资产配置中的性能更加稳健。具体地说，我们认为基于模型的强化是投资组合管理的一个很好的候选者，而不是无模型的 [17] [18]。在基于模型的强化学习中，使用动态模型来进行预测，并以此来动作选择。定义 $f_\theta(s_t; a_t)$ 为一个训练的离散时间动力学函数，由 θ 参数化，取当前状态 s_t 和动作 a_t，输出在 $t + \delta t$ 时刻的状态的估计。然后，我们可以通过解决以下优化问题来选择动作：

$$(a_t, \ldots, a_{t+H-1}) = \arg \max_{a_t, \ldots, a_{t+H-1}} \sum_{t'=t}^{t+H-1} \gamma^{t'-t} r(s_{t'}, a_{t'})$$

更重要的是，由于神经网络对数据质量敏感，可以利用传统的金融数据降噪方法，如小波分析 [19] 和卡尔曼滤

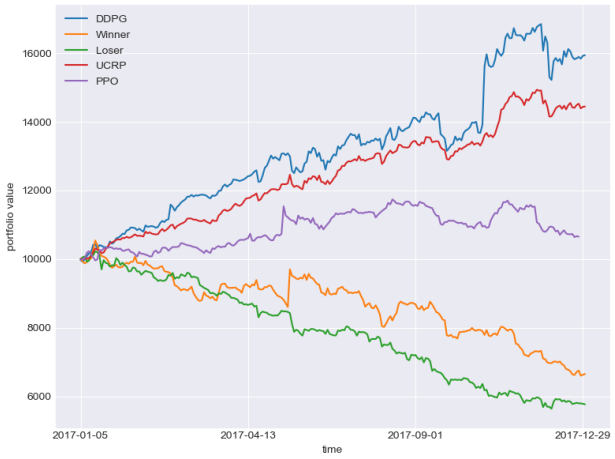

图 11. Backtest on USA stock market

波器 [20] 。数据预处理的另一种方法是将 HMM 与强化学习结合起来，即提取低于波动价格的状态并直接学习 [21]。

我们也可以考虑修改目标函数。一个方向是去适应 risk-adjust 回报。另一个方向是我们在游戏中设计强化学习代理的实验。在游戏中，奖励的机制很简单。例如，在 flappy bird 中，代理在通过柱子时会得到 1 分的奖励，在落地时会得到-1 分的奖励。复杂的目标函数会阻碍代理更好的表现。我们构建了一个简单版本的累积资产值作为目标函数，即采使用赢钱的概率取代绝对收益，但这并不完善。

VII. 结论

这篇文章将连续动作空间的深度强化学习应用于资产配置中。我们比较了 DDPG 算法和 PPO 算法在不同市场、不同超参数等一系列因素下的表现。相比于之前运用强化学习在资产配置中的研究，我们通过设置风险调整后的累计投资组合价值作为我们的目标函数以及不同特征组合作为我们的输入来测试我们的代理。我们的实验结果表明 DDPG 算法得到的策略能够胜过传统资产配置策略。我们发现深度强化学习算法，即便是在很有限的数据和特征中，都能够在一定程度上捕捉市场价格变化特征并且实现自我提升。

但是，强化学习还在资产配置中没有取得像它在游戏设计、机器人控制等领域的卓越成绩。我们对此提出一些想法：

其一，算法需要假设输出的策略以及 critic 中的期望对于神经网络中的参数是二阶可导，因此最优策略都是在对参数二阶可导的策略函数集中寻找，并非在策略集的全体中寻找，这也导致了得到的策略并非是全局最优的。

其二，算法要求同一个状态的转移服从同样的概率分布，但是由于政策变化或者市场的不规范以及政府的操控，股票的状态转移不一定在长时间内服从同一个概率分布，只能在较短的时间内相同。

在我们的实验中，深度强化学习算法是极其敏感以至于它的表现是不稳定的。更重要的是，我们深度强化学习代理资产配置行为发生了退化，也就是同一时间会倾向于只购买一支资产。这也表明要设计出更加有前景的算法需要更多调整和改进。

致谢

我们想要感谢朝旭私募基金刘铭文、中山大学谢铮和付星宇的耐心指导。没有他们的支持，我们无法克服如此多的困难并最终顺利完成项目。

参考文献

[1] Li, Bin, and Steven CH Hoi. ”Online portfolio selection: A survey.” ACM Computing Surveys (CSUR) 46.3 (2014): 35.
[2] Du, Xin, Jinjian Zhai, and Koupin Lv. ”Algorithm trading using q-learning and recurrent reinforcement learning.” positions 1 (2009): 1.
[3] Almahdi, Saud, and Steve Y. Yang. ”An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown.” Expert Systems with Applications 87 (2017): 267-279.
[4] An Algorithm for Trading and Portfolio Management Using Q-learning and Sharpe Ratio Maximization
[5] Li, Yuxi. ”Deep reinforcement learning: An overview.” arXiv preprint arXiv:1701.07274 (2017).
[6] Sutton, Richard S., et al. ”Policy gradient methods for reinforcement learning with function approximation.” Advances in neural information processing systems. 2000.
[7] Mnih, Volodymyr, et al. ”Human-level control through deep reinforcement learning.” Nature 518.7540 (2015): 529.
[8] Lillicrap, Timothy P., et al. ”Continuous control with deep reinforcement learning.” arXiv preprint arXiv:1509.02971 (2015).
[9] Schulman, John, et al. ”Proximal policy optimization algorithms.” arXiv preprint arXiv:1707.06347 (2017).
[10] Jiang, Zhengyao, Dixing Xu, and Jinjun Liang. ”A deep reinforcement learning framework for the financial portfolio management problem.” arXiv preprint arXiv:1706.10059 (2017).
[11] He K, Zhang X, Ren S, et al. Deep residual learning for image recognition[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2016: 770-778.
[12] Guo, Yifeng, et al. ”Robust Log-Optimal Strategy with Reinforcement Learning.” arXiv preprint arXiv:1805.00205 (2018).
[13] Silver, David, et al. ”Deterministic policy gradient algorithms.” ICML. 2014.
[14] John Schulman, Sergey Levine, Philipp Moritz, Michael Jordan, Pieter Abbel: Trust Region Policy Optimization
[15] Jacobsen, Ben, and Dennis Dannenburg. ”Volatility clustering in monthly stock returns.” Journal of Empirical Finance 10.4 (2003): 479-503.
[16] Dewey, Daniel. ”Reinforcement learning and the reward engineering principle.” 2014 AAAI Spring Symposium Series. 2014.
[17] Gu, Shixiang, et al. ”Continuous deep q-learning with model-based acceleration.” International Conference on Machine Learning. 2016.
[18] Nagabandi, Anusha, et al. ”Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning.” arXiv preprint arXiv:1708.02596 (2017).
[19] Rua, António, and Luís C. Nunes. ”International comovement of stock market returns: A wavelet analysis.” Journal of Empirical Finance 16.4 (2009): 632-639.
[20] Faragher, Ramsey. ”Understanding the basis of the Kalman filter via a simple and intuitive derivation.” IEEE Signal processing magazine 29.5 (2012): 128-132.
[21] Serban, Iulian Vlad, et al. ”The Bottleneck Simulator: A Model-based Deep Reinforcement Learning Approach.” arXiv preprint arXiv:1807.04723 (2018).
[22] Rao, Anil V. ”A survey of numerical methods for optimal control.” Advances in the Astronautical Sciences 135.1 (2009): 497-528.