摘 要

在人工智能时代下，计算机设备水平的不断升级以及机器学习技术的快速进步，使量化投资这一现代金融新兴投资方法的发展在硬件、算法及数据层面均得到有效支撑。量化投资以其强大的数据处理能力和精准高效的决策特点愈加受到各类投资机构的青睐和重视。深度强化学习(DRL)是构建量化投资策略的新兴技术，因其建模的机器学习问题高度对齐投资者的目标而表现出色。然而 DRL 的有效应用仍面临着环境状态表达不充分、历史数据稀缺等问题，这些问题容易导致训练不稳定及效率低下问题，进而降低投资者收益。本文以道琼斯工业平均指数市场的股票为研究对象，重点研究量化投资中交易和投资组合管理两个任务，提出数据高效的多模态量化投资改进策略，以期提升策略的盈利水平。本文的主要贡献如下：

1. 提出结合表征学习的多模态策略模型 MMD4。由于 DRL 训练需要智能体与环境交互并获取反应当前市场状况的环境状态，故定义蕴含丰富市场状况信息且便于模型训练的环境状态是 DRL 训练的重点。然而，传统的基于指标所定义的状态并不能充分体现市场复杂的变化规律。基于此，本文首先利用股票历史价格数据生成三种类型图片，以图片和向量指标相结合的多模态数据形式作为环境状态的表示；其次，提出结合 CNN 与 LSTM 的图片处理模块去提取市场复杂的动态特征和时序变化规律；最后，引入 D2RL 表征学习模型更好提取向量状态信息，并设计不同的自监督训练任务辅助训练表征学习模型。测试结果显示，MMD4 模型在所有模型中综合表现最佳，其 Sharpe Ratio 指标在投资组合管理和交易任务上分列第一和第二。

2. 提出结合集成 Q 学习和蒙特卡洛树搜索(MCTS)的 MMD4^{+}策略模型。经本文分析发现，DRL 存在样本效率不高，训练不稳定等问题，而投资任务中复杂的市场环境和历史数据的短缺加剧了这些情况。为缓解上述问题，本文首先提出引入 REDQ 集成框架来稳定训练，通过集成 Critic 函数缓解过高估计问题，增强动作价值估计的准确性和模型鲁棒性。其次，本文使用 MCTS 改进环境交互方式为断点存续交互，利用树结构平衡探索和利用。MCTS 能有效选择价值更高、更具前景的搜索路径，通过增强样本质量的方式来提高样本利用效率。测试结果显示，MMD4^{+}模型在继续提升收益效果同时，能有效降低 Sharpe Ratio 的标准差。

关键词：量化投资；深度强化学习；多模态；集成 Q 学习；蒙特卡洛树搜索

Abstract

In the era of artificial intelligence, the continuous upgrading of computer hardware and the rapid advancement of machine learning techniques have effectively supported the development of quantitative investment, which is an emerging method in modern finance, at the hardware, algorithm, and data levels. Quantitative investment is increasingly favored and emphasized by various investment institutions due to its powerful data processing capabilities and precise, efficient decision-making characteristics. Deep Reinforcement Learning (DRL) is an emerging technology for constructing quantitative investment strategies. It performs well in quantitative finance field because the machine learning problems it models are highly aligned with investors' goals. However, the effective application of DRL still faces issues such as insufficient representation of environmental states and scarcity of historical data. These problems can lead to unstable training and low training efficiency, which in turn reduce investor returns. This paper focuses on the stocks included in the Dow Jones Industrial Average, specifically addressing trading and portfolio management tasks in quantitative investment. It proposes an improved data-efficient multimodal quantitative investment strategy, aiming to enhance the profitability of the strategy. The main contributions of this paper are as follows:

1. Propose a multimodal strategy model named MMD4 that combines state representation learning. Since DRL training requires the agent to interact with the environment and obtain states reflecting current market conditions, defining an environment state that contains rich market information and is conducive to model training is a focal point of DRL training. However, traditional state definitions based on indicators cannot fully capture the complex dynamics patterns of the market. Based on this, the paper first uses historical stock price data to generate three types of images, and combines these images with vector indicators to represent the state of the environment in a multimodal format. Secondly, it introduces a CNN-LSTM based image processing module to extract complex market dynamics and temporal patterns. Finally, it incorporates the D2RL state representation learning model to better assist in extracting vector state information and also designs different self-supervised training tasks to facilitate the training of the state

representation learning model. Test results show that the MMD4 model performs best overall performance among all models, ranking first and second in the Sharpe Ratio for portfolio management and trading tasks, respectively.

2. Propose MMD4^{+} model, which incorporates both ensemble Q-learning and Monte Carlo Tree Search (MCTS). The analysis indicates that DRL confronts issues such as low sample efficiency and unstable training, which are exacerbated by the complex market environment and the shortage of historical data in investment tasks. To mitigate these issues, this paper first introduces the REDQ ensemble framework to stabilize training, using an ensemble of Critic functions to alleviate overestimation issues and enhance the accuracy of action value estimation and the robustness of the model. Secondly, it uses MCTS to improve the mode of interaction between agent and environment as a breakpoint continuation interaction, balancing exploration and exploitation by a tree structure. MCTS effectively selects more valuable and promising search paths, thereby enhancing sample quality and improving sample utilization efficiency. Test results show that MMD4^{+} model effectively reduces the standard deviation of Sharpe Ratio while continuing to improve performance.

Key words: quantitative investment; deep reinforcement learning; multimodal; ensemble Q-learning; Monte Carlo Tree Search

目 录

第 1 章 绪 论

1.1 研究背景及意义

金融是现代国家经济发展的血脉，金融行业是国民经济发展的晴雨表，良好的金融活动能够承担资源配置、市场定价、信息融通等诸多影响资金流通的重要功能。量化金融(Quantitative Finance)是金融业和计算机行业结合的新兴产业，其工作机制为在数学和统计学的基础上，利用计算机对金融数据进行建模，定量分析和预测金融市场状况并做出决策。由于量化金融能够有效避免受到投资者的喜好经验等主观因素的影响，实现低成本、可透明、高效率的资金管理，因而量化金融及相关的量化科技、量化策略近年来逐渐兴起。以 2023 年中国量化科技白皮书介绍的我国公募量化基金发展状况为例进行说明，如图 1.1 所示。图左侧部分列出我国主动量化基金增速进入新常态的原因，包括政府政策倾斜和消费投资增加；投资机构不断推出新的产品；应用新的算法和模型等。中间部分列出量化投资的步骤及使用到的技术。右侧部分列出 2022 年三种主流的量化投资策略，其平均收益均跑赢市场平均水准。由图中内容可以得出结论，我国公募主动量化基金增速正进入到新的发展阶段。2022 年主流量化投资策略均能实现正向收益增长，表明我国使用量化科技和量化策略参与量化金融的群体正在日益壮大。

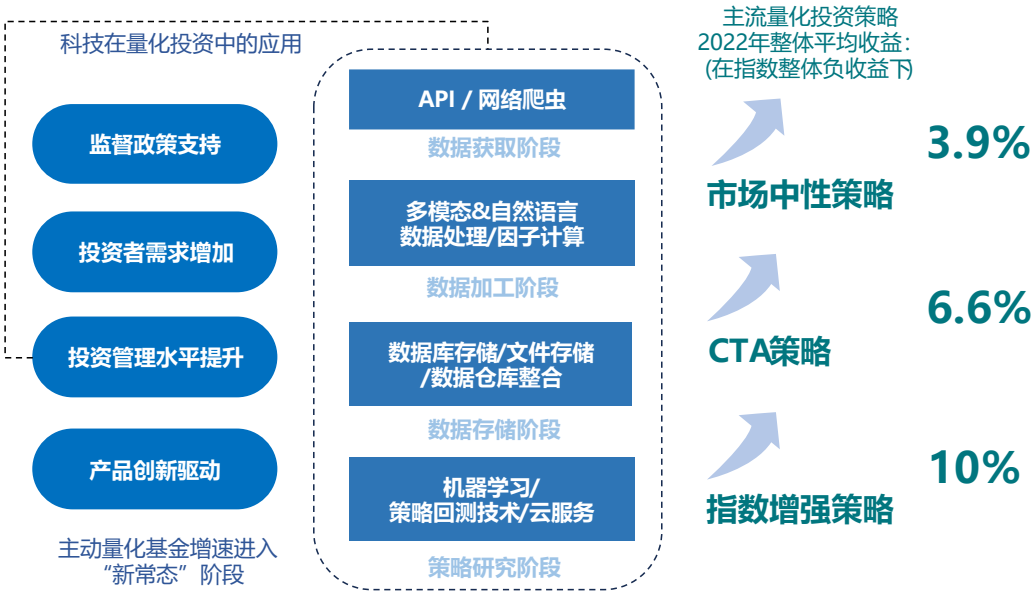


图 1.1 中国量化投资的发展现状

Fig. 1.1 The development of quantitative trading in China

量化投资(Quantitative Investment)是量化金融的重要业务，主要包括交易(Trading)和投资组合管理(Portfolio Management)两个任务。交易任务的执行是基于市场环境，做出能够盈利的具体的买卖交易动作；而投资组合管理任务则需要调整资产配置比例，实现稳健的投资。随着量化金融逐渐受到重视，交易和投资组合管理任务均迎来了新的发展机遇和挑战。无论是国家、投资机构还是个人，都迫切需要协同推进数据、算法和模型的使用，创新量化策略，一方面实现行业的蓬勃发展、创收增效，另一方面也能增强我国金融行业在全球范围内的竞争力，有利于维护我国金融安全，实现金融行业的健康发展。

然而，金融市场环境复杂易变，资产价格不仅会受到资产本身的影响，同时还会被市场非理性情绪波动、国家宏观调控政策、全球化带来的相互依赖关系等诸多因素影响，这给投资者预测市场波动状况并执行交易带来了一般性的困难和挑战。此外，在技术层面上，尽管近年来深度学习技术被用于量化投资中，但是大多数深度学习方法的思路为基于价格预测的有监督学习(Supervised Learning)方法，即根据市场历史信息预测未来资产价格，再将预测值转换为具体的决策动作。然而，这种学习方法存在以下缺陷：

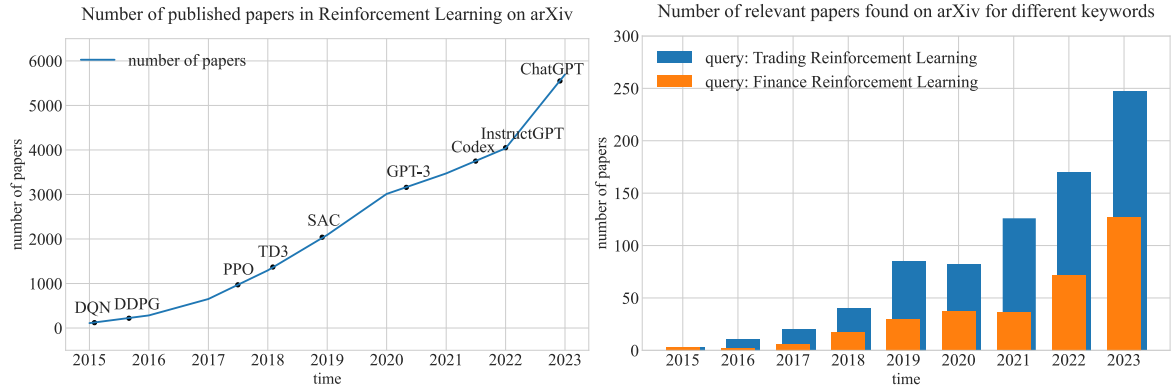
1. 决策质量取决于价格预测结果，容易出现过拟合情况。

2. 需要额外构造映射，来将预测价格映射为具体的买卖交易动作。

3. 需要人工标注精准的交易动作标签，不符合人们在现实中通过交易的盈亏来逐渐调整策略的学习模式。

深度强化学习(Deep Reinforcement Learning, DRL)是另一种构建量化投资策略的方法。它的交互式、试错式的学习特点符合现实世界中生物的学习模式，即智能体(执行策略的对象)和虚拟金融市场环境不断交互，通过尝试各种交易动作去获得市场的反馈，继而不断调整自己的策略。此外，DRL 可以构造端到端的映射，能直接将市场状况信息转换为交易动作。它同时考虑了价格预测和获取收益两部分，不需要建立额外的映射模型，降低了建模和训练的难度。同时，DRL 是近年来人工智能新兴起的研究领域，其快速发展为相关应用提供强有力的工具。图 1.2 给出了相关研究领域中论文发布数量变化情况，绘图数据为利用 arXiv 网站关键词搜索功能搜索出的相关领域历年发表的论文数目。(a)图搜索关键词为 Reinforcement Learning，(b)图搜索关键词在(a)图关键词之前添加了 Trading 或 Finance，(a)图用实心点标注了 DRL 重要模型发布的时间节点。从 (a)图可以看到，自 2015 年 DRL 模型 DQN[1]发布到 2020 年 SAC[2]模型发布，期间 arXiv 每年发表的强化学习(Reinforcement Learning,

RL)相关论文数目快速上升，相关的最优水平(State of the Art, SOTA)模型如深度确定性策略梯度[3](Deep Deterministic Policy Gradient, DDPG)、双延迟深度确定性策略梯度[4](Twin Delayed Deep Deterministic Policy Gradient, TD3)、近端策略优化[5](Proximal Policy Optimization, PPO)相继提出。2022 年 1 月，OpenAI 发布 InstructGPT 模型[6]并给出大语言模型标准的训练流程，其中基于人类反馈的强化学习是训练大语言模型的重要技术，这进一步引发业界和学术界对 RL 的关注，2022 年发表的 RL 相关论文数占据过往总数的三分之一。RL 的快速发展奠定了相关研究的理论基础。(b)图则展示了量化金融结合 RL 的相关论文数量变化，近年发表的论文数量呈翻倍增长态势，这充分说明了该研究的前景值得期待。



(a) (b)

图 1.2 相关领域历年发表的论文数目变化

Fig. 1.2 Changes in the number of papers published in the relevant field over the years

综上，利用 DRL 技术解决交易和投资组合管理等量化投资问题，不仅具有现实意义，更不乏学术科研价值和技术储备，具备良好的可行性基础。本文将聚焦 DRL 技术在量化投资任务中的应用，建模并求解实际的投资问题。

1.2 国内外研究现状

1.2.1 深度强化学习

1. 强化学习发展历程

首先简要梳理 RL 的发展历程，图 1.3 按照四个阶段划分 RL 的发展阶段，描绘了每个阶段 RL 里程碑式的理论和应用成果，每一阶段从代表技术/应用、代表模型/方法、特点和评价这三个角度描述其特点。经典算法和理论如 Q-learning 算法和策略梯度定理(Policy Gradient Theorem, PG)奠定了现代 RL 理论基础，而 2010 年之后出现的 DRL 模型则利用神经网络端到端的、非线性拟合能力，适用于现实中复杂的

任务，下文会详细介绍该部分内容。2022 和 2023 年兴起的大语言模型则使用 RL 训练范式来使模型对其人类价值。

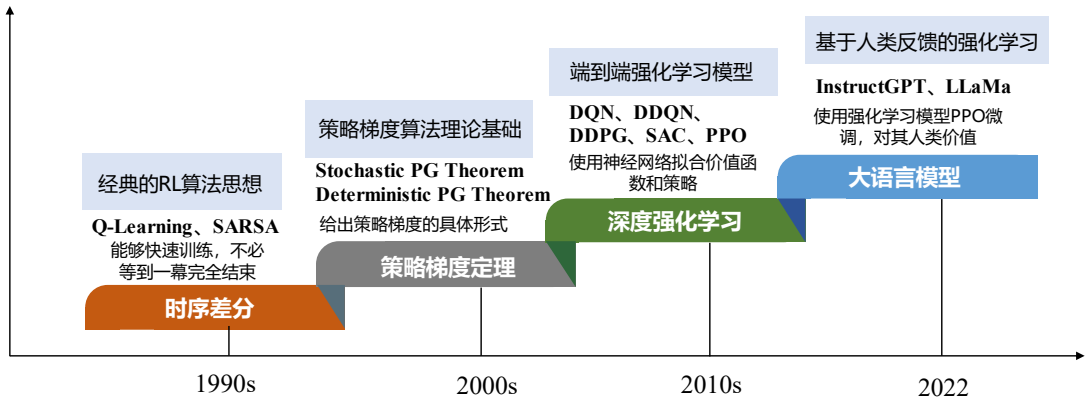


图 1.3 强化学习技术和主流应用发展历程

Fig. 1.3 The development of reinforcement learning techniques and mainstream applications

2. 强化学习的早期发展

1989 年，Chris Wakins[7]在其博士论文中提出 Q-learning 算法，同时证明了算法收敛性，后续现代 Value-Based 以及 Actor-Critic 框架下的 RL 方法都有 Q-learning 的身影。Sutton 和 Barto[8]则奠定了早期 RL 的理论基础。2000 年，Sutton[9]提出了策略梯度定理，这项工作是策略梯度类方法的理论基础，为后续其他方法建立了求解的基本范式。文章指出，Value-Based 类型方法，诸如时序差分(Temporal Difference, TD)系列方法一般适用于离散型动作空间，且趋向于学习到确定性策略。因而，为了求解随机性策略以及适用连续型动作空间的需求，文章提出使用函数逼近手段拟合随机性策略并求解策略梯度。文章使用参数化的函数表示待学习的随机性策略，这里的策略是状态到连续型动作空间的概率分布。根据 RL 极大化累计折扣奖励的目标，文章基于价值函数构造了三种度量策略性能优劣且包含策略参数的目标函数，并且使用随机梯度下降方法优化策略参数。文章给出了三种目标函数下求解策略梯度的理论推导过程，它们都是同种类型的形式。文章还给出了函数逼近情况下，能够满足无偏性质策略梯度的函数形式，以及在这种形式下收敛到局部最优的理论保证。REINFORCE 算法[10]则是策略梯度定理的蒙特卡洛版本应用。其使用蒙特卡洛回报 G_t 估计策略梯度中需要的动作价值，实现简单且保证无偏性。具体而言，策略梯度定理给出了策略梯度的期望形式，而 REINFORCE 算法采用蒙特卡洛算法来采样梯度得到估计值。REINFORCE 需要等到一个 Episode 完全结束之后才能进行训练，后续其他基于策略梯度的方法，尤其是 Actor-Critic 算法能够实现更快速的学习，采样方法也会有所区别。

3. 现代深度强化学习方法(基于值函数的方法)

现代 DRL 的研究始于 2013 年的深度 Q 学习网络[11](Deep Q-learning Network, DQN)。Mnih 等人提出了 DQN 模型来操作 Atari 平台上的各类游戏，该模型使用神经网络来表示动作价值函数(Q 网络)，并且利用 Q-learning 算法训练 Q 网络。DQN 引入关键机制——经验回放(Experience Replay)来保证训练稳定性，它会将过去搜集到的样本放入经验池(Replay Buffer)中保存起来，每次训练时从经验池中随机抽取一小批样本进行训练。经验回放一方面降低了采样轨迹中不同时刻样本的相关性，更贴近于有监督训练的要求；另一方面也提升了样本效率，过去的样本可以被多次训练使用。DQN 是 DRL 研究的开篇之作，是第一个得以成功训练并应用的 DRL 模型。DQN 在 Atari 多个游戏上取得了超越人类专家的水平，展现了 RL 在序列决策控制领域内的潜力。后续学者基于 DQN 提出了一系列重要改进模型。Mnih 等人[1]于 2015 年在 Nature 杂志发表了 DQN 的改进版本，并进一步在 Atari 上进行测试。文章提出了目标网络(Target Network)的概念，每隔固定步数将正在训练的 Q 网络的参数拷贝复制到同样结构的目标 Q 网络上。目标 Q 网络被用来计算 Q 网络更新所使用的标签 TD Target，由于目标 Q 网络能够在一定时间内保证不变，所以 TD Target 也能保持稳定，从而增强训练稳定性。该版 DQN 在 Atari 游戏上取得了更好的效果。van Hasselt 等人[12]于 2015 年提出使用双 Q 学习(Double Q-learning)来改进 DQN，并在 Atari 游戏上进行了实验。该文章的核心想法来源于 van Hasselt[13]2010 年的工作，目的是为了缓解 Q-learning 算法天然具有的过高估计(Over-Estimation)问题，即 Q 网络对优势动作的价值估计过高。双 Q 学习提出使用两个独立的 Q 网络来分别用于动作的选择和动作价值的估计，从而计算 TD Target。van Hasselt 将 DQN 和双 Q 学习结合起来提出了双重深度 Q 学习网络(Double DQN, DDQN)模型。在单独设置的测试游戏环境中，DDQN 相较于 DQN 有效缓解了过高估计问题，并在多项 Atari 游戏上取得了更好的得分。Schaul 等人[14]于 2015 年针对 DQN 的经验回放机制进行了改进，指出经验池中不同样本对训练带来的帮助存在差别，采样时应当更频繁抽取更有助于训练的样本。文章提出有优先级的经验回放(Prioritized Replay)，每次采样时根据样本的 TD error 计算出优先级，TD error 越大表示模型在该样本上表现越差，这样的样本对训练更有帮助，故应当被给予更大优先级。随后利用优先级计算样本分布，按照该分布进行小批量抽样。文章还提出使用 Sum Tree 来提高采样效率。Prioritized Replay 明显提高了训练效率，并在一些 Atari 游戏上取得了更好的效果。Ziyu[15]等人于 2016 年提出 Dueling DQN，并在 Atari 游戏上进行了检测。Dueling DQN 主要利

用‘优势’概念，提出包含两个 channel 的 Dueling 结构的网络分别估计状态价值和优势，两者合并得到动作价值。状态价值和优势分开估计能够让 Q 网络更好明确不同状态下动作价值的估计，在动作价值区分不明显的状态上重点估计状态价值，以此提升 Q 网络的泛化能力，提升训练效率。模型在许多 similar-valued actions[15]上能更好的拟合动作价值，并取得良好效果。

此外，还有其他针对 DQN 的相关改进，例如 Bellemare 等人[16]于 2017 年提出分布式视角的 DQN，认为传统 DQN 通常关注动作价值的估计，忽略了奖励和价值分布的形状和变异。文章提出 Q 网络学习奖励分布，以分布的期望作为动作价值估计。Hessel 等人[17]于 2017 年将过去 DQN 模型所有改进结合起来，提出了多种 DRL 技术混合的 Rainbow DQN 模型。这些技术包括上述提到的双 Q 学习、Prioritized Replay、Dueling Network、Distributional，还包括 N-steps 和 Noisy Nets。Rainbow DQN 在多个标准 RL 环境中得到了显著的性能提升。图 1.4 描绘了 DQN 家族模型的发展历程，经过多次重要的创新之后，DQN 系列模型已成为基于值函数方法中非常重要的模型。

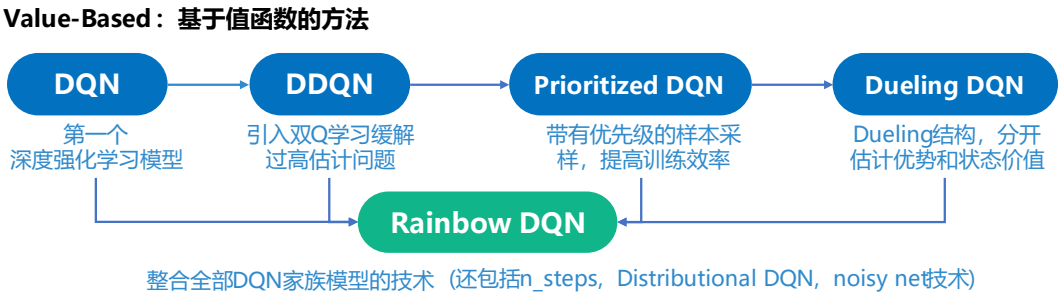


图 1.4 DQN 系列模型发展历程及重要节点

Fig. 1.4 Development history and important nodes of DQN series models

4. 现代深度强化学习方法(基于策略梯度的方法)

DRL 另一类重要方法是基于策略梯度的方法。基于 Sutton 的策略梯度定理，学者们结合现代神经网络提出了一系列基于策略梯度类型的 DRL 方法。

Silver 等人[18]于 2014 年基于策略梯度定理推导出确定性策略梯度(Deterministic Policy Gradient, DPG)定理，并给出一种离轨 Actor-Critic 算法。文章首先从直觉和理论上给出了 DPG 的动机和理论推导，并以 Degris 等人[19]于 2012 年发表的 Off-Policy Actor Critic 定理为基础，提出了应用确定性策略的离轨 Actor-Critic 方法。其次，文章也论证了 DPG 是随机性策略梯度方差趋向于零时的特例，并在实验中于高维动作空间的基准环境里取得了优于随机性策略的效果。以 DPG 为理论基础，Lillicrap 等人[3]于 2016 年提出深度确定性策略梯度(Deep Deterministic Policy Gradient, DDPG)模

型，这是首个基于 DPG 和神经网络成功训练的用于解决连续型动作空间的端到端的 DRL 模型。DDPG 使用多层感知机(Multi-Layer Perception, MLP)拟合 Actor 和 Critic，并使用 Q-learning 和 DPG 算法训练。DDPG 还使用软拷贝机制，使得训练进一步接近于有监督方法，大大稳定了训练过程。为了增强探索，DDPG 在策略输出的动作上添加 OU 噪声，使得智能体在具有惯性性质的环境中有更强的持续探索力度。Fujimoto 等人[4]于 2018 年提出 Clipped Double Q-learning 算法来缓解 Actor-Critic 框架中出现的过高估计问题，给出了 DDPG 的改进版本：双延迟确定性策略梯度(Twin Delayed Deep Deterministic Policy Gradient, TD3)。TD3 使用两个独立的 Q 网络拟合动作价值函数，并使用两个动作价值估计的最小者计算 TD Target，减少过高估计的风险。尽管缺乏理论保证，且可能导致保守的动作价值估计，但 TD3 在实践中表现出良好效果。此外，TD3 也引入其他技巧：包括延迟策略的更新，让 Actor 更新频率慢于 Critic 来保证训练程度一致；引入光滑噪声，添加正态噪声到目标 Q 网络动作上，获取更泛化的动作价值估计。Haarnoja 等人[2]于 2018 年提出了基于最大熵强化学习框架的软演员-评论家(Soft Actor-Critic, SAC)模型。SAC 模型可以从最大熵强化学习角度理解，也可以从 Soft Q-learning 算法理解。对于前者，SAC 模型使用结合策略性能和策略分布熵的目标，并使用自适应优化的温度系数实现了探索和利用的高效权衡。对于后者，SAC 模型中的价值函数形式是传统形式加上策略分布熵，策略分布定义为能量基(Energy-Based)形式，这同样满足贝尔曼方程。Haarnoja 等[20]给出了理论保证：将基于 Soft Q-learning 算法得到的最优价值函数带入到能量基形式策略中所得到的最优策略，等价于最大熵强化学习框架下的最优策略。从而，SAC 给出了新的视角来理解 RL 构建策略的方法。SAC 以其高效的探索机制、自适应的参数寻优、离轨训练等特性实现了快速训练、适应性强、样本利用率高的优势，并且具有良好效果。Schulman 等人[5]于 2017 年提出 PPO 模型，PPO 不仅是 DRL 的经典模型，更是 LLM 的 RLHF 训练过程的核心方法之一[6]。PPO 模型建立在 TRPO[21]模型基础上，后者是前者的理论基础；前者在工程上简化后者，更易于实现。TRPO 为确保优化策略时训练的有效性和稳定提出置信域方法，将策略参数更新的幅度限制在较小范围内。通过理论推导中的多次近似，TRPO 模型求解变成一个带约束优化问题。TRPO 的约束是当前学习的策略和旧策略的 KL 散度，KL 散度不宜过大，从而限制参数更新幅度。然而，TRPO 模型要求解带约束优化问题，没有进一步提出目标函数的梯度，无法执行梯度下降。所以，尽管 TRPO 在理论上非常优雅，但具体应用和求解比较复杂。PPO 为此应运而生。PPO 保持了 TRPO 的核

心思想的同时进行了工程上的简化和近似，并提出了两个版本：自适应惩罚系数和裁剪代理目标函数。第一个版本直接将 TRPO 的 KL 散度约束放进目标函数中作为惩罚项，用惩罚系数来进行权衡，并动态调整惩罚系数。第二个版本构造出原目标函数下界的代理目标函数，当样本趋向于动作非常优势或劣势时，代表正常情况下梯度幅度会很大，此时目标函数会被限制在最低水平，参数更新不会非常激进。

图 1.5 展示了基于策略梯度方法类型的 RL 模型的发展历程，以 Q-learning 算法和策略梯度定理为理论基础，Actor-Critic 框架成为流行训练范式。

图 1.5 基于策略梯度的模型发展历程

Fig. 1.5 The development of models based on policy gradient theorem

5. 其他方面的研究

除了对 RL 理论的直接改进所提出上述经典 DRL 模型以外，还有一些对模型的有效训练方面的研究，例如表征学习(State Representation Learning, SRL)。SRL 的基本思想是构造辅助训练任务来训练特征提取器并将其放置在 Actor 和 Critic 前预先处理环境原始的状态和动作输入，从而 Actor 和 Critic 能够以更有效和易于处理的中间特征作为输入来加速训练和提升效果。相关工作包括 Munk 等人[22]于 2016 年提出的模型学习深度确定性策略梯度(Model Learning Deep Deterministic Policy Gradient, ML-DDPG)，Ota 等人[23,24]于 2020 年提出的在线特征提取网络(Online Feature Extractor Network, OFENet)和对较大神经网络的训练效果的研究，Sinha1 等人[25]于 2020 年提出的用于强化学习的深度稠密结构(Deep Dense Architectures for Reinforcement Learning, D2RL)模型。其他研究还有针对 Q-learning 过高估计问题的研究，例如 Lan 等人[26]于 2020 年提出的 Maxmin Q-learning 算法；提高样本采样效率并在当时实现 SOTA 效果的 Model-Free 方法随机集成双 Q 学习[27](Randomized Ensembled Double Q-learning, REDQ)。

1.2.2 基于深度强化学习的量化投资

随着 DRL 理论的不断进步及其适合解决复杂时序决策问题的特性，DRL 方法被

越来越多应用于量化投资场景中，各类模型、相关理论、交易环境框架被相继提出。

DRL 应用于量化投资已有许多经典案例。Deng 等人[28]首次实现了 DL+RL 构建交易决策。文章提出引入多层感知机增强特征 RNN 的提取能力，帮助 RL 部分更好决策。文章给出了许多 DL 的工程技巧，例如消除数据不确定性的 Fuzzy Learning；针对 DL、RL 和 Fuzzy Learning 模块的不同参数初始化；任务适应性的 BPTT 算法等。Jiang 等人[29]提出以同一独立评估器的集成(Ensemble of Identical Independent Evaluators, EIIE)为核心的 DRL 交易框架。EIIE 是将某种神经网络视作集成单位，以相同参数的神经网络处理每个资产的历史信息，最终集成所有资产的结果来生成资产分配权重。文章还提出了高效向量存储方式和适用于离线和在线的训练方式。文章选择的三种 IIE 模块：卷积神经网络(Convolutional Neural Network, CNN)、循环神经网络(Recurrent Neural Network, RNN)和长短期记忆网络(Long Short Term Memory，LSTM)，均占据所有实验的前三位次。此外，一些多模态、多数据来源策略也被构建出来。Shin 等人[30]提出一种多模态 DRL 方法，结合 CNN 和 LSTM 作为特征提取模块，使用 DQN 作为决策模型进行交易。文章利用股票时序信息生成三种类型图片，并使用结合 CNN 和 LSTM 的模块来处理图像。文章使用 Boltzmann 探索(Softmax 探索)来进行基于概率的动作选择，使得交易动作更趋于连续和平稳。测试结果显示，在韩国综合股价指数下降的测试阶段，模型能实现了利润的大幅提升。许杰等人[31]参考前人提出的多模态方法的工作，提出使用多时刻的多模态数据来更充分地提取广泛特征。其多模态数据为股票因子数据和根据历史价格生成的图片数据，多时刻数据依次送入 LSTM 模块循环处理，并由 DQN 模型决策。Wang 等人[32]提出 DeepTrader 框架，在 AlphaStock[33]基础上添加了 TCN、GCN 和空间注意力机制等技术，来从时间特征、短期和长期空间特征分别改进股票价格部分的特征提取。此外，文章还增加了市场因子来计算市场行业情绪。

在人们逐渐意识到 DRL 解决量化投资问题的优势后，许多综述文章也其进行了广泛的讨论。Millea[34]从 DRL 建模量化投资任务角度讨论了其中涉及的问题，例如问题分类、风险评估(奖励定义)、环境建模、模型选择等。Fischer[35]从模型分类角度，按照 Critic-Only、Actor-Only、Actor-Critic 三方面讨论了不同方法的特点。进一步又根据状态、动作和奖励的建模详细总结每类方法的研究成果。Pricope[36]重点探讨股票交易的研究成果。文章肯定了 DRL 在特定条件下所具有的模式识别能力和预测能力，但同时也给出了一些改进方向，例如需要更多模型和交易者的比较、相同条件下的 SOTA 模型的比较、不同条件下的模型测试等等。Sun 等人[37]对量化投资

的交易、投资组合管理等任务的 SOTA 模型分类进行了讨论，总结了历来的研究成果，指出数据稀疏、风险和盈利平衡、决策解释性等问题，并给出相对应的 Model-Based 方法、Multi-Objective 方法、Hierarchical RL 方法等解决思路。

除模型外，也有针对构建交易框架，创建通用交易环境的相关研究，FinRL[38]即为典型代表。FinRL 提出一套标准的训练量化投资策略的流程，并实现了交易和投资组合管理两个主要任务的虚拟环境，用户可以套用自身数据实现基于数据驱动的 RL 交易环境。此外，还有其他相关研究工作，涉及到集成、对比评估等方面内容。Yang 等人[39]结合 PPO、优势演员-评论家(Advantage Actor Critic, A2C)、DDPG 三种 Actor-Critic 模型提出了一种基于夏普率的集成策略。文章将训练时段划分为多个三个月长度的训练窗口，每个训练窗口都重新训练三种模型并使用验证集来检测三个模型的效果，选择夏普率最高的模型在接下来的训练窗口与环境交互。文章[40, 41]分别在 50 种期货合约和 3 种股票上，对比了 DQN、PG、A2C 之间和 PPO、DQN、SAC 之间的性能。两项工作结果均显示 DQN 效果最好。Yu 等人[42]提出了第一个应用于投资组合管理任务的 Model-Based 方法，文章设计了注入式预测模块(Infused Prediction Module, IPM)预测未来价格，数据增强模块(Data Augmentation Module, DAM)去减轻数据系数问题，行为克隆模块(Behavior Cloning Module)去减少波动性。文章[43, 44]提出了两种多智能体 RL 系统，前者构建独立的智能体生成自身的权重，根据特殊损失增加不同智能体的多样化程度；后者构造进化智能体模块(Evolving Agent Module, EAM)和策略性智能体模块(Strategic Agent Module, SAM)，分别学习市场多样的嵌入和 EAM 的输出结果。

综上所述，DRL 及其在量化投资领域的研究已经积累了大量成果，许多 DRL 模型的提出为求解不同市场环境的投资任务提供了强大工具，各类结果也表明 DRL 方法具有超出传统投资策略的效果。然而，DRL 训练成功需要较为严苛的条件，例如需要设计合理的环境状态来有效表达复杂的市场变化规律；获取的训练数据要充足且分布符合环境的动态转移特性下的分布；精心设置符合任务的超参数等等。这些问题一方面是 DRL 解决量化投资任务所特有的难题，另一方面也是 DRL 训练共有的挑战。过往研究虽然提出了一些针对性的解决方案，如结合市场面和基本面等多方面的因子数据来丰富市场状况信息；利用 Model-Based 方法或构造合成数据训练来提升样本利用效率，但依然在状态提取和模型高效稳定训练等方面需要改进。因而，DRL 在量化投资领域内的工作仍需继续深入研究。

1.3 本文主要研究内容

本文研究内容为量化金融领域内的量化投资问题，该研究主要利用 DRL 技术构建通用高效且具有较强盈利水平的投资策略(RL 构造出的用于决策的行为模式称作策略，除策略外，RL 训练还涉及到如价值函数、智能体和环境交互的交互方式等内容，本文将涉及到 RL 训练的全部内容称为模型，或策略模型)。本文针对 DRL 技术以及其在股票交易和投资组合管理任务中的应用进行了广泛调研，通过文献分析梳理了前人在该领域内的工作成果的同时，也发现了尚有的一些缺陷或不足。在以往的工作中，模型往往针对复杂市场环境信息的处理不够充分，而股票历史数据稀缺等固有难题也限制着投资策略的性能。鉴于此，本文在前人的理论和实践研究基础之上，重点针对市场环境状态特征的表达提取和模型数据利用效率两个角度进行研究，提出结合多模态数据的量化投资策略，并利用集成和 MCTS 等方法提升模型的训练效率和稳定性。本文具体进行了如下两方面的工作：

1. 结合表征学习的多模态投资策略模型(**M**ulti-**M**odal with **D2**RL **DD**PG, MMD4)。现有的 DRL 策略大多利用股票历史价格数据构造出技术指标或统计指标，以此作为市场的环境状态。然而，这种单模态状态虽然能直接反应当前时刻的市场状况，但由于其时间跨度较短，故无法体现长远的动态市场变化规律。本文首先提出增加图片模态数据，丰富环境状态表达，即通过多期历史数据构造三种能反应市场长期发展趋势的图片数据。图片和原始向量形式的环境状态合并，两者共同作为智能体观测的状态，即策略的状态输入。其次，本文构造了结合 CNN 和 LSTM 的图像处理模块，两者分别提取市场整体动态特性和长期时序变化特性，高效处理长期的市场环境信息。最后，本文在扩充状态、增加输入信息的基础上，提出使用表征学习模型 D2RL 预先处理原始向量状态，降低 RL 决策部分的训练难度。同时构造不同的自监督辅助训练任务，帮助训练表征学习模型。本文将多模态环境状态和表征学习结合，一方面能让智能体接收到更完整丰富的市场信息，帮助其理解市场复杂的变化趋势；另一方面也降低了模型的训练难度，避免直接输入原始数据而带来的过多噪声和不稳定问题，进一步提升模型训练效果。最终实验结果表明，本文提出的多模态模型能提升交易任务和投资组合管理任务上的整体盈利水平，取得良好的效果。

2. 结合集成框架和蒙特卡洛树搜索(Monte Carlo Tree Search, MCTS)的数据高效的 MMD4 改进版本投资策略模型(Data-Efficient MMD4, MMD4^{+})。考虑到 RL 与量化投资结合天然所具有的数据短缺及利用低效的问题，本文提出结合集成思想和 MCTS 来提升样本利用效率，对此进行改进。本文首先提出使用 REDQ 集成框架增

加训练时使用的 Critic 数目，该框架通过 In-Target Minimization 优化方式来缓解过高估计问题，继而使得模型能够在保持稳定的情况下执行更多轮训练，从而增加样本利用效率。其次，受到 AlphaGo 模型的启发，本文提出使用 MCTS 来改进智能体和环境的交互方式。AlphaGo 主要利用 MCTS 减少搜索空间的宽度，该技术通过构造一种树结构来评估每个访问状态和动作的价值。在搜索过程中，AlphaGo 根据这些价值决定其搜索策略，从而在探索未知性和利用已知信息方面达到良好平衡，有效减少了搜索量并提高了执行动作的质量。本文了构造蒙特卡洛树来改进智能体和环境的交互过程，以智能体和环境的一段固定长度的交互序列作为树结点，将智能体和环境完整的一幕交互转变为 MCTS 的一次模拟过程。在模拟过程中，模型根据交互的反馈，自适应地寻找具有较高交互价值的状态，从而避免了低价值的交互。这样做一方面通过减少交互次数来提升样本利用效率，另一方面改变了训练样本分布，使得模型能够选取到质量更高的样本，从而改善训练效果。测试结果表明，结合 REDQ 和 MCTS 后，模型在相同交互次数下能实现更优的表现，并且在较少的交互次数下依然能够保障效果的稳定性，这证明了所作改进的有效性。

1.4 本文结构安排

本文共分为五章，总体结构安排如图 1.6 所示。前两章涉及背景介绍、研究内容与基础知识等；第三章和第四章为核心章节，详述本文主要工作内容；第五章总结本文工作，并对未来研究方向进行展望。各个章节具体内容安排如下。

第一章为绪论部分。首先从量化投资和 DRL 研究近年来的快速发展出发，阐述利用 DRL 技术求解量化投资问题的重要意义与价值。其次介绍国内外 DRL 标志性成果及发展历程，同时介绍其在量化投资领域中的应用成果。在章节的最后，明确本文主要研究内容和行文结构。

第二章为预备知识部分。内容围绕 DRL 和量化投资两大领域展开。本章首先简要概述 DRL 的基本概念，其次详细介绍 DRL 经典算法 Q-learning 及其衍生出的 4 种 Actor-Critic 模型，这些模型将在后续章节被使用。章节的最后部分，将介绍两类量化投资任务：交易和投资组合管理，包括它们的交易流程和相关金融指标。

第三章详细叙述了构建多模态策略的流程。章节起始首先介绍研究所使用的道琼斯工业平均指数数据集，并定义了两类数据驱动的虚拟交互环境。本章随后详尽叙述了图片模态数据的生成过程、处理图像数据的 CNN-LSTM 模块架构。以及表征学习模块的搭建和辅助训练方式。此外，本章还给出了综合上述贡献的总体模型框

架 MMD4 及算法流程。章节的最后，通过在测试集上对比 MMD4 模型和基准模型效果，并通过超参数实验和消融实验来证明模型的有效性。

第四章详细叙述了在第三章工作基础上，如何继续改进 MMD4 模型以构建数据高效的策略模型。本章首先分析 RL 存在的样本采样效率底下和训练不稳定等问题，据此提出结合集成框架 REDQ 和 MCTS 的解决方案，旨在针对这些问题进行改进。本章随后介绍了改进后的模型 MMD4^{+}及其算法流程。在章节的最后，通过 MMD4^{+}的对比试验和消融实验，展示了改进模型在样本利用效率的优势。

第五章总结全文，列举并分析全文得出的结论，并对未来可以继续研究的方向进行了展望。

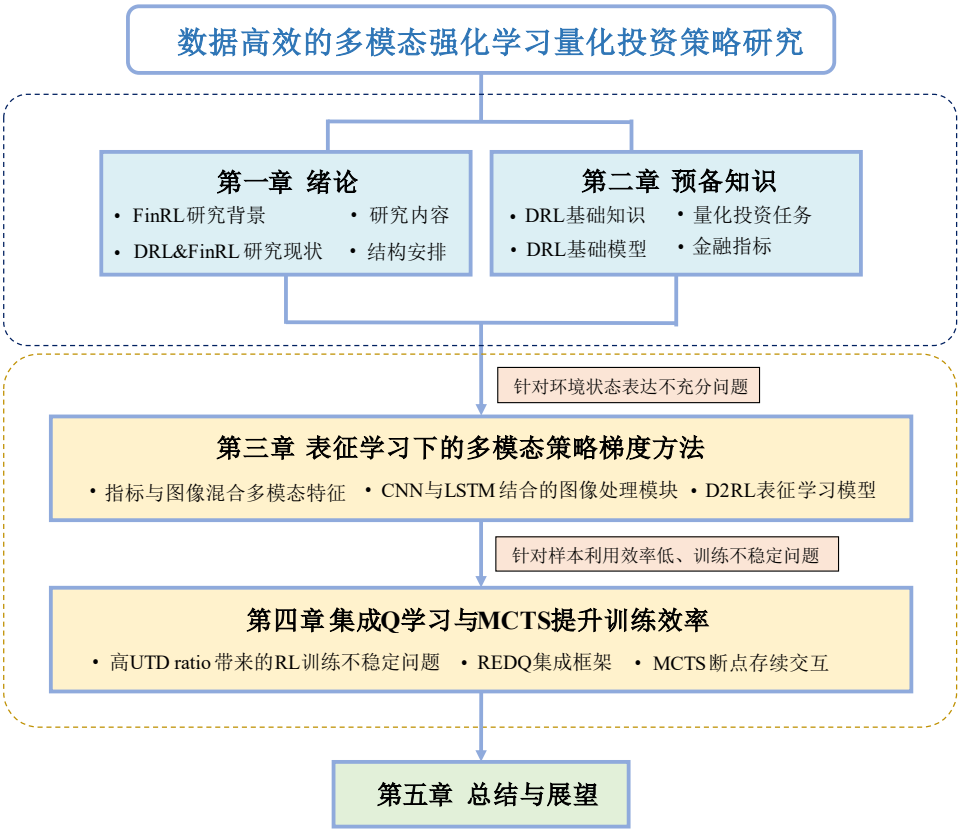


图 1.6 本文内容与结构安排

Fig. 1.6 The content and structure of this paper

1.5 本章小结

本章主要概述了研究背景、研究意义、国内外研究现状和主要研究内容，并总结了全文逻辑构架。尽管国内外已有大量 DRL 在量化投资领域应用的相关研究，但依然存在一些未解决的问题，如状态表达不充分和训练效率低下等。本文将针对这些问题提出具体的解决方案，以期进一步提高模型的性能和实用性。

第 2 章 预备知识

2.1 深度强化学习

2.1.1 深度强化学习概述

强化学习(RL)和有监督学习、无监督学习是三种不同的机器学习范式，有着不同的特点和目标。深度强化学习(DRL)是使用深度学习技术来拟合 RL 中的策略和价值函数的方法，即如果 RL 算法使用神经网络这种参数化的函数来表示策略和价值函数的话，则为 DRL 算法。

RL 的训练范式比较独特，尤其值得和有监督学习进行区分和对比，因为两者都是学习如何在特定场景下做出正确的动作，但是两者的学习目标和学习过程具有明显差异，具体如图 2.1 所示。

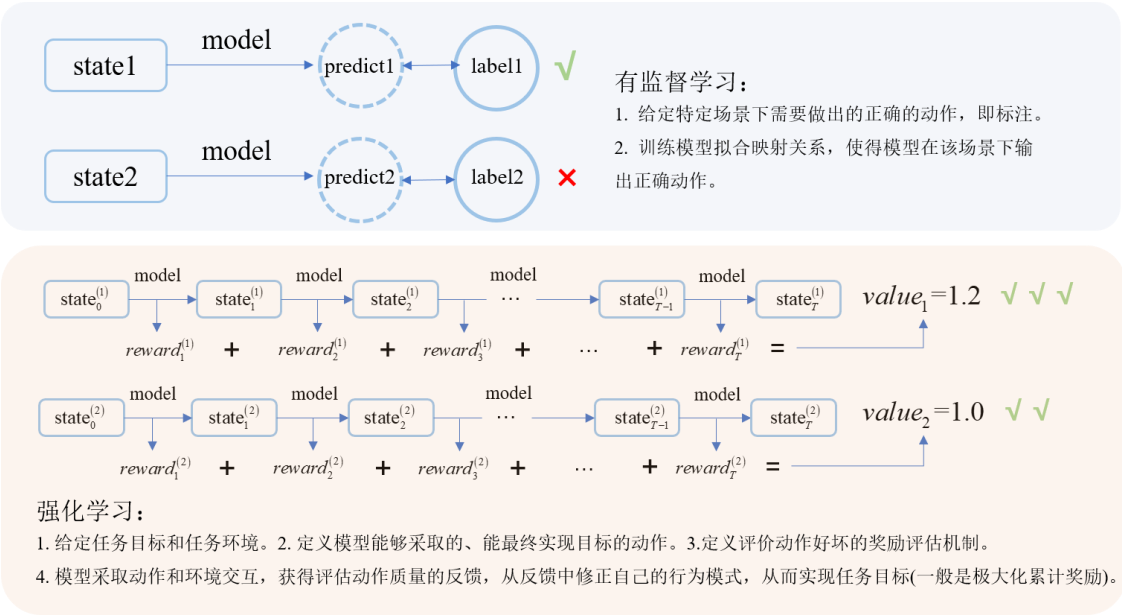


图 2.1 有监督学习和强化学习训练方式对比

Fig. 2.1 Comparison of supervised learning and reinforcement learning training methods

有监督学习本质上引入了一个“教师”来告知模型需要学习的标准内容，然而，构造一个专业的“教师”需要非常多的领域知识以及人工或经济成本，有些复杂的任务甚至无法给出精确的标准动作，典型的如序列控制任务。在一些复杂的，甚至是未知领域中，需要模型能够从自身经验中进行学习，而 RL 能够满足此类要求。从图 2.1 中可以看到，RL 能够在未知的环境中学习并不断修正自己的行为模式。RL

的特点是交互式的、试错的、目标导向的，适合求解复杂未知的任务。由于需要在长期交互中修正自身行为，故 RL 适合求解序列式的任务。RL 的训练就是智能体和环境不断交互的过程。

2.1.2 马尔可夫决策过程和广义策略迭代

1. 马尔可夫决策过程

标准 RL 任务被建模为马尔可夫决策过程(Markov Decision Process, MDP)，MDP 是一种具有马尔可夫性质的随机过程，由于 RL 具有动作概念，所以 MDP 中含有动作这类随机变量。智能体和 MDP 共同给出了一段轨迹序列。MDP 可以用公式 2.1 来建模，该公式也被称为 MDP 的动态特性。

$$p\left(s',r\mid s,a\right)=\Pr(s_{t+1}=s',r_{t+1}=r\mid s_t=s,a_t=a) \tag{2.1}$$

公式 2.1 实际给出了智能体和环境交互的动态变化过程：智能体在当前状态 s_t 下做出动作 a_t，环境将按照条件概率分布进行转移。由于状态和奖励的分布只依赖于前一时刻状态和动作，故具有马尔可夫性质。上述公式描述了环境在外界影响下的变化方式，智能体将在这种随机性中优化策略，来实现 RL 的目标：找到最好的策略，在与环境长期交互中极大化累计折扣奖励 $G_t=\sum_{l=1}\gamma^{l-1}r_{t+l}$ 。

2. 价值函数

RL 使用价值函数作为数学工具，帮助寻找更好的策略。价值函数分为状态价值函数 V 和动作价值函数 Q，它们的定义分别为公式 2.2、2.3：

$$V_\pi\left(s\right)=E_\pi\left[G_t\mid s_t=s\right]=E_\pi\left[\sum_{l=1}\gamma^{l-1}r_{t+l}\mid s_t=s\right] \tag{2.2}$$

$$Q_\pi\left(s\right)=E_\pi\left[G_t\mid s_t=s,a_t=a\right]=E_\pi\left[\sum_{l=1}\gamma^{l-1}r_{t+l}\mid s_t=s,a_t=a\right] \tag{2.3}$$

可以看到，价值函数是奖励之和的期望，表示长期、平均意义的质量评估。状态价值函数 $V\left(s\right)$ 表示状态 s 的价值，即智能体处于状态之 s 后继续遵循策略执行动作平均能够获取的奖励之和。动作价值函数 $Q\left(s,a\right)$ 表示状态 s 下动作 a 的价值，即智能体基于状态 s 做出动作 a 后遵循策略执行动作平均获取的奖励之和。同时可以看到，价值函数依赖于策略 π (因 MDP 形成的轨迹需要动作的参与)。故在引入价值函数之后，RL 的学习目标可以使用价值函数来形式化定义。为了实现这一点，首先需要给出最优价值函数和最优策略的定义，分别如公式 2.4，2.5 所示。要找到一个策略能够获得大量收益，可以使用价值函数比较两个策略的优劣，进而定位最优策略。

$$V_*(s)=\max_{\pi}\left(V_\pi(s)\right),\ \text{for all } s\in S \tag{2.4}$$

$$Q_*(s,a)=\max_{\pi}\left(Q_\pi(s,a)\right),\ \text{for all } s\in S, a\in A \tag{2.5}$$

在给定最优价值函数，尤其是最优动作价值函数之后，可以利用贪心算法直接得到最优策略如公式 2.6 所示。RL 任务即为求解最优策略。

$$\pi_*(s)=\arg\max_{a} Q_*(s,a) \tag{2.6}$$

3. 广义策略迭代

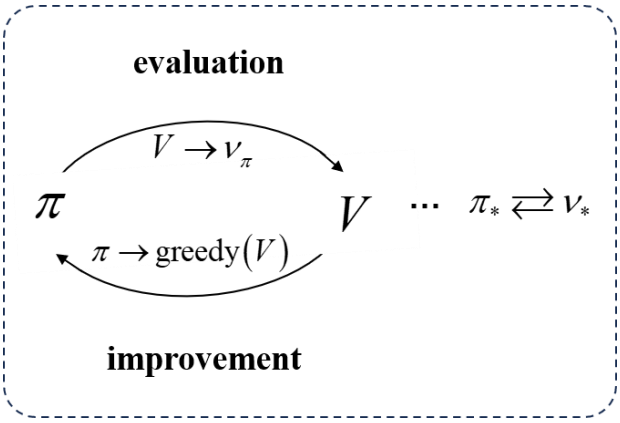


图 2.2 广义策略迭代框架

Fig. 2.2 Generalized policy iteration framework

一般采用广义策略迭代框架寻找最优策略，如图 2.2 所示。广义策略迭代是策略评估(evaluation)和策略提升(improvement)循环交替执行的过程。策略评估为估计策略 π 的价值函数，可使用动态规划或时序差分法；策略提升为基于 t 时刻策略 π_t 的价值函数 Q_{π_t} 获得优于 π_t 的策略，可使用 $\varepsilon-greedy$ 算法如公式 2.7。

$$\pi_{t+1}(s)=\begin{cases}\arg\max_{a} Q_{\pi_t}(s,a), & \text{if } a \le \varepsilon \\ \text{随机挑选的动作, otherwise} & \end{cases},\ a\text{采样自}U(0,1)\text{分布} \tag{2.7}$$

2.1.3 Q-learning 算法和策略梯度方法

1. Q-learning

Q-learning 算法是 Value-Based 类型方法重要的理论基础，也是 DQN 家族模型和 Actor-Critic 类方法训练动作价值函数的算法。Q-learning 算法直接学习最优动作价值函数，其更新迭代动作价值函数的公式如 2.8 所示。

$$Q(s,a)=Q(s,a)+\alpha\left(r+\max_{a'} Q(s',a')-Q(s,a)\right) \tag{2.8}$$

通过不断执行公式 2.8，Q-learning 算法同时执行策略评估和策略提升，直到 Q 函数

收敛，即得到最优动作价值函数，继而得到最优策略。

2. DDPG

DDPG 算法基于 DPG，使用 Actor-Critic 框架训练。Actor 和 Critic 使用神经网络拟合，利用梯度下降法更新参数。DDPG 引入了 OU 噪声增加探索力度，且使用软更新方式来拷贝网络参数到目标网络。算法流程见算法 2.1。

算法 2.1 DDPG

1: 初始化：Critic 网络 $Q\left(s,a;\theta^{Q}\right)$ 和 Actor 网络 $u\left(s;\theta^{u}\right)$，参数分别为 θ^{Q},θ^{u}

2: 初始化目标网络 Q^{-},u^{-}，参数分别为 $\theta^{Q^{-}}\leftarrow\theta,\theta^{u^{-}}\leftarrow\theta^{u}$

3: 初始化：容量为 N 的 Replay Buffer 为 D

4: **for** *episode* = 1, … , E **do**

5: 初始化随机过程 n 为噪声

6: 初始化环境，得到起始状态 s_0

7: **for** $t = 0, \dots, T$ **do**

8: Agent 执行动作：$a_t = u\left(s_t;\theta^{u}\right)+n_t$

9: 环境步进：环境接收动作 a_t，给出奖励 r_{t+1} 和下一时刻状态 s_{t+1}

10: 存储样本：将 $transition=\left(s_t,\ a_t,\ r_{t+1},\ s_{t+1}\right)$ 存入 D 中

11: 抽取样本：从 D 中随机采样 *minbatch* 数量的样本 $\left(s_j,a_j,r_{j+1},s_{j+1}\right)$

12: 计算 *TD Target*：$y_j = r_j+\gamma Q\left(s_{j+1},u(s_{j+1};\theta^{u^{-}});\theta^{Q^{-}}\right)$

13: 训练 Critic：根据目标函数 $\left(y_j-Q\left(s_j,a_j;\theta^{Q}\right)\right)^2$ 执行梯度下降

14: 训练 Actor：根据目标函数 $-Q\left(s_j,u\left(s_j;\theta^{u}\right);\theta^{Q}\right)$ 执行梯度下降

15: 拷贝：更新目标网络参数 $\theta^{Q^{-}}\leftarrow\tau\theta^{Q}+\left(1-\tau\right)\theta^{Q^{-}}$，$\theta^{u^{-}}\leftarrow\tau\theta^{u}+\left(1-\tau\right)\theta^{u^{-}}$

16: **end for**

17: **end for**

3. TD3

TD3 算法是对 DDPG 算法的改进，算法 2.2 是 TD3 的算法流程。TD3 引入两个 Critic 网络计算 TD target，在函数逼近和 Actor-Critic 框架下缓解了过高估计问题。同时，TD3 引入了 smooth noise，能够泛化对动作价值的估计，并且令 Actor 的更新频率降低，增加训练稳定性。

算法 2.2 TD3

1: 初始化：Critic 网络$Q(s,a;\theta_1)$，$Q(s,a;\theta_2)$和 Actor 网络$\pi(s;\phi)$，参数分别为θ_1,θ_2,ϕ

2: 初始化：Critic 和 Actor 目标网络，参数分别为$\theta_1' \leftarrow \theta_1, \theta_2' \leftarrow \theta_2, \phi' \leftarrow \phi$

3: 初始化：容量为 N 的 Replay Buffer 为 D

4: **for** *episode* = 1, … , E **do**

5: 初始化环境，得到起始状态s_0

6: **for** $t = 0, \dots, T$ **do**

7: Agent 执行动作：$a_t = \pi(s_t;\phi) + \varepsilon,\ \varepsilon \sim N(0,\sigma)$

8: 环境步进：环境接收动作a_t，给出奖励r_{t+1}和下一时刻状态s_{t+1}

9: 存储样本：将$transition = (s_t,\ a_t,\ r_{t+1},\ s_{t+1})$存入 D 中

10: 抽取样本：从 D 中随机采样 *minbatch* 数量的样本$(s_j, a_j, r_{j+1}, s_{j+1})$

11: 添加光滑噪声：$\tilde{a}_j = \pi(s_j;\phi') + \varepsilon,\ \varepsilon \sim clip(N(0,\tilde{\sigma}), -c, c)$

12: 计算 *TD Target*：$y_j = r_j + \gamma \min_{i=1,2} Q(s_{j+1}, \tilde{a}_{j+1}; \theta_i')$

13: 训练 Critic：根据目标函数$\left(y_j - Q(s_j, a_j;\theta_i)\right)^2$，$i = 1,2$ 执行梯度下降

14: **if** t mod d **then**

15: 训练 Actor：根据目标函数$-Q(s_j, \pi(s_j;\phi);\theta_1)$执行梯度下降

16: 拷贝：更新目标网络参数$\theta_i' \leftarrow \tau\theta_i + (1-\tau)\theta_i',\ \phi' \leftarrow \tau\phi + (1-\tau)\phi'$

17: **end then**

18: **end for**

19: **end for**

4. SAC

SAC 提出的 Soft Actor-Critic 框架下的广义策略迭代算法，能够得到最大熵强化学习框架下的最优策略。SAC 同时优化奖励和以及策略熵，两部分目标由熵系数来进行权衡，SAC 提出了动态自适应调整熵系数的算法，使得 SAC 对参数并不敏感。SAC 算法流程见算法 2.3。

算法 2.3 SAC

1: 初始化：Critic 网络$Q(s,a;\theta_1), Q(s,a;\theta_2)$和 Actor 网络$\pi(s|a;\phi)$，参数分别为$\theta_1,\theta_2,\phi$

2: 初始化：Critic 和 Actor 目标网络，参数分别为$\theta_1' \leftarrow \theta_1, \theta_2' \leftarrow \theta_2, \phi' \leftarrow \phi$

3: 初始化：容量为 N 的 Replay Buffer 为 D

4: **for** *episode* = 1, … , E **do**

5: 初始化环境，得到起始状态s_0

6: **for** $t = 0, \dots, T$ **do**

7: Agent 执行动作：$a_t = \pi(s_t;\phi)$

8: 环境步进：环境接收动作 a_t，给出奖励 r_{t+1} 和下一时刻状态 s_{t+1}

9: 存储样本：将 $transition = (s_t,\ a_t,\ r_{t+1},\ s_{t+1})$ 存入 D 中

10: 抽取样本：从 D 中随机采样 *minbatch* 数量的样本 $\left(s_j, a_j, r_{j+1}, s_{j+1}\right)$

11: 计算 *TD Target*：

$$y_j = r_j + \gamma\left(\min_{i=1,2} Q\left(s_{j+1}, \tilde{a}_{j+1}; \theta_i'\right) - \alpha \log \pi\left(\tilde{a}_{j+1} \mid s_{j+1}; \phi\right)\right),\ \ \tilde{a}_{j+1} \sim \pi\left(\cdot \mid s_{j+1}; \phi\right)$$

12: 训练 Critic：根据目标函数 $\left(y_j - Q\left(s_j, a_j; \theta_i\right)\right)^2$，$i = 1, 2$ 执行梯度下降

13: 训练 Actor：根据目标函数执行梯度下降：

$$-\min_i Q\left(s_j, a_\phi\left(s_j\right); \theta_i\right) - \alpha \log \pi\left(a_\phi\left(s_j\right) \mid s_j; \phi\right),\ \ a_\phi\left(s_j\right) \sim \pi\left(\cdot \mid s_j\ ; \phi\right)$$

14: 拷贝：更新目标网络参数 $\theta_i' \leftarrow \tau\theta_i + (1-\tau)\theta_i',\ \phi' \leftarrow \tau\phi + (1-\tau)\phi'$

15: **end for**

16: **end for**

5. PPO

PPO 算法通过设置置信域来限制策略参数的更新范围，以此稳定训练。PPO 将问题转为无约束优化问题，采用梯度下降方式更新参数。算法 2.4 给出了 PPO 的裁剪代理目标函数版本的算法流程。

算法 2.4 PPO

1: 初始化：状态价值网络 $V(s;\phi_0)$ 和策略网络 $\pi(s,a;\theta_0)$，参数分别为 θ_0, ϕ_0

2: 初始化：容量为 N 的 Rollout Buffer 为 D

3: **for** $k = 0, 1, 2, \ldots$ **do**

4: 搜集轨迹：Agent 根据策略 $\pi_k = \pi(\theta_k)$ 与环境交互得到一段轨迹 $\{\tau_i\}$

5: 计算 empirical return：$\hat{R}_t = \sum_{l=0} \gamma^l r_{t+l}$

6: 计算 GAE：$\hat{A}_t = \hat{R}_t - V_{\phi_k}\left(s_t\right)$

7: 训练 critic：根据目标函数 $\left(\hat{R}_t - V_{\phi_k}\left(s_t\right)\right)^2$ 执行梯度下降，更新参数 ϕ_{k+1}

8: 训练 actor：根据 PPO clipped 代理目标函数更新参数 θ_{k+1}

$$\theta_{k+1} = \arg\max_\theta \left(\min\left(\frac{\pi_\theta\left(s_t, a_t\right)}{\pi_{\theta_k}\left(s_t, a_t\right)} A^{\pi_{\theta_k}}\left(s_t, a_t\right), clip\left(1-\varepsilon, 1+\varepsilon, \frac{\pi_\theta\left(s_t, a_t\right)}{\pi_{\theta_k}\left(s_t, a_t\right)}\right) A^{\pi_{\theta_k}}\left(s_t, a_t\right) \right) \right)$$

9: 训练状态价值函数：根据目标函数更新参数 ϕ

$$\phi_{k+1} = \arg\min_\phi \left(V_\phi\left(s_t\right) - \hat{R}_t\right)^2$$

10: **end for**

2.2 量化投资主要任务

2.2.1 交易任务

交易任务是量化投资的重要任务，在本文中，它指根据市场环境做出合理的交易决策，并将交易决策转化为每只股票的买卖数目来执行交易。

数学建模描述如下：设在 t 时刻拥有资产 $asset_t$，且资产由两部分组成：t 时刻的现金 $cash_t$ 和 t 时刻所具有的股票价值总额。设 t 时刻所有股票的收盘价向量为 $\boldsymbol{close_t}$，所有股票的持股数目向量为 $\boldsymbol{num_t}$，则 t 时刻所持有的股票价值总额为 $\sum_i close_t^i * num_t^i$，资产总额为 $\sum_i close_t^i * num_t^i + cash_t$。在 t 时刻，智能体根据当前市场状况做出交易动作 a_t，并最终转换为具体的股票买卖数目，因而在 $t+1$ 时刻所持有的现金和股票数目都会变为 $cash_{t+1}$ 与 $\boldsymbol{num_{t+1}}$，根据 $t+1$ 时刻股票收盘价，可以计算出 $t+1$ 时刻所持有的股票价值总额，继而得到 $t+1$ 时刻的资产总额 $asset_{t+1}$。故交易任务的目标就是在测试阶到达终止时刻 T 时，极大化总资产 $asset_t$，模型需要训练出合理的策略，使得策略能在每一时刻做出长远来看具有最大收益的动作。

2.2.2 投资组合管理任务

投资组合管理任务是量化投资中另一重要任务，它区别于交易任务。该任务需要在每时刻重新调整权重向量 $\boldsymbol{w_t}$,来分配当前资产到各支股票，故 $\boldsymbol{w_t}$ 为单位向量，其分量表示某支股票在 t 时刻的投资金额比例。投资组合管理必须消耗掉所有现金。

为了描述清晰，故将投资组合管理的数学建模流程陈列如下。任务的目标即为在 T 时刻，极大化总资产 $asset_T$。

投资组合管理建模流程

1: 初始化：$asset_0$，=1000000\$, $\boldsymbol{w_0} = (1/m, 1/m, ..., 1/m)$。其中 $asset_0$ 为初始资产总额，$\boldsymbol{w_0}$ 为初始分配权重，m 为股票数目。

2: **for** $t = 1, \ldots, T$ **do**

3: Agent 根据状态 s_t 给出新的分配比例 $\boldsymbol{w_t}$

4: 计算 stock return vector $\boldsymbol{r_t} = \left(\frac{v_{1,t} - v_{1,t-1}}{v_{1,t-1}}, \ldots, \frac{v_{m,t} - v_{m,t-1}}{v_{m,t-1}} \right)$，$v_{i,j}$ 是收盘价。

5: 更新投资组合资产总额 $asset_t = asset_{t-1} \times \left(1 + \boldsymbol{w_t^T r_t}\right)$

6: **end for**

2.3 相关金融指标

本节将介绍金融领域评价交易策略的常用指标及主要使用的技术指标。

1. 评价指标

包括累计回报率(cumulative return)、年化回报率(annual return)、夏普率(sharpe ratio)、最大回撤(max drawdown)。令 t 时刻的百分比收益率为 $asset_t / asset_{t-1} - 1$，设 ***returns*** 表示一个量化策略所有交易时刻的百分比收益向量。四种指标的含义和计算公式如表 2.1 所示。

表 2.1 评价指标简介

Table 2.1 Metrics brief introduction

指标名称	指标介绍及公式计算
累计回报率	累计回报率表示策略从开始到投资结束的整个完整交易期间内，所获得的最终资产和初始资产的比例。它考虑了策略在完整投资周期内的整体表现，能够体现出策略的长期投资价值。累计回报率的计算公式如下：$cumulative\ return = (1+returns_1)\times ... \times (1+returns_T) - 1$
年化回报率	年化回报率率是利用累计回报率计算出的每年的平均增长率。它表示了策略平均在每年的增长率，通过将累计回报率年化，投资者能够直观看到过去一年的表现，以及预期的长期增长态势。年化回报率的计算公式如下，其中 $year$ 取值为投资年份：$annual\ return = (cummulative\ return)^{1/year} - 1$
夏普率	夏普率表示策略在投资周期内的平均收益率和收益率标准差的比值。夏普率用于衡量每承担单位风险所带来的超额回报，能够权衡收益和风险，夏普率越高意味着投资具有越强的性价比。夏普率的计算公式如下：$sharpe\ ratio = \bar{R}_p - R_f / \sigma_p \times \sqrt{N}$
最大回撤	最大回撤表示策略在整个投资周期内，收益从峰值跌到谷底的最大幅度。作为一个风险指标，其能够有效衡量策略在形势下行时抵抗风险的能力。最大回撤越小，表示策略在市场下行时表现越稳定，风险越小。最大回撤的计算公式如下:$max\ drawdown = \max_{t\in[0,T]}\left(\max_{\tau\in[0,t]}(asset_\tau) - asset_t\right) / \max_{\tau\in[0,t]}(asset_t)$

2. 技术指标

包含本文将要用到的三种技术指标：MACD、DMI、SSO。

(1). 移动平均收敛发散(MACD, Moving Average Convergence Divergence)

MACD 是一种用途广泛的动量指标，用于跟踪市场趋势。它由三部分组成：MACD 线、信号线、直方图。MACD 线是两个指数移动平均(Exponential Moving Average, EMA)线的差值，一般选择 12 和 26 日 EMA。信号线 *signal* 是 MACD 线的 9 日 EMA 线。直方图绘制 MACD 线和信号线的差异。当 MACD 线从下向上穿过信号线时，表示市场适合买入，反之表示适合卖出。计算公式如公式 2.9：

$$MACD=EMA_{12}\left(prices\right)-EMA_{26}\left(prices\right),\ signal=EMA_{9}\left(MACD\right) \tag{2.9}$$

(2). 方向性运动指数(DMI, Directional Movement Index)

DMI 称为趋向指标，表示市场变化的方向性。它包含三个具体的指标：+DI、−DI 、ADX。+DI 和−DI 用于分析变化的方向，当+DI 大于−DI 时，市场上升趋势可能较强；反之下降趋势较强。ADX 表示市场趋势变化的强度，ADX 越大表示变化趋势越明显，一般 ADX 大于 25 时趋势较强，小于 20 时趋势较弱。DMI 的三个指标的计算流程如下：

DMI 相关指标计算流程

1: 输入：股票日度最高价 $high$, 最低价 low, 收盘价 $close$。 $window$。

2: $tr_{1,t}=high_t-low_t, tr_2=\left|high_t-close_{t-1}\right|, tr_3=\left|low_t-close_{t-1}\right|, \forall t$

3: $dm_{+,t}=\max\left(high_t-high_{t-1},0\right), dm_{-,t}=\left|\min\left(low_t-low_{t-1},0\right)\right|, \forall t$

4: $dm_{+}=\max\left(\max\left(dm_{+}-dm_{-},0\right),dm_{+}\right)$， $dm_{-}=\max\left(\max\left(dm_{-}-dm_{+},0\right),dm_{-}\right)$

5: $atr_t=\left(\sum_{i=t-window+1}^{i=t} tr_i\right)/window, smooth\_dm_{+,t}=\left(\sum_{i=t-window+1}^{i=t} dm_{+,i}\right)/window$

$smooth\_dm_{-,t}=\left(\sum_{i=t-window+1}^{t} dm_{-,i}\right)/window, \forall t$

6: $+DI_t=100*smooth\text{-}dm_{+,t}/atr_t, -DI_t=100*smooth\text{-}dm_{-,t}/atr_t$

7: $DX=\left|(+DI)-(-DI)\right| \Big/ \left((+DI)+(-DI)*100\right)$

8: $ADX_t=\left(DX_{t-window+1}+...+DX_t\right)/window, \forall t$

(3). 随机震荡指标(SSO, Stochastic Oscillator)

SSO 也是一种动量指标，称为随机指标，用来表示资产价格的未来走势。它由%K 线和%D 线两条线组成，每条线又分为快速随机指标和慢速随机指标，即%K_fast、%K_slow 和%D_fast 和%D_slow。使用时，随机指标的交叉点被视为交易信号：当%K_fast 上穿%D_slow 时推荐买入；当%K_fast 下穿%D_slow 时推荐卖

出。它们的计算公式如下：

SSO 相关指标计算流程

1: 输入：股票日度最高价 $high$，最低价 low，收盘价 $close$。 k, d 。

2: $low\_min_t = \min_{i=t,\ldots,t-k+1} low_i$， $high\_max_t = \max_{i=t,\ldots,t-k+1} high_i$

3: $\%K\_fast_t = 100*(close_t - low\_min_t)/(high\_max_t - low\_min_t)$

4: $\%D\_fast_t = \left(\sum_{i=t-d+1}^{i=t} \%K\_fast_i\right)/d$

5: $\%K\_slow = \%D\_fast$， $\%D\_slow_t = \left(\sum_{i=t-d+1}^{i=t} \%K\_slow_i\right)/d$

2.4 本章小结

本章主要介绍了本文研究工作所需要的预备知识，主要涉及两方面的内容：DRL 和量化投资。前者主要包括 DRL 的简介、DRL 的经典算法 Q-learning、4 类 Actor-Critic 模型。Q-learning 算法是常用的控制算法，本文涉及的全部 DRL 模型均建立在 Q-learning 基础上。Q-learning 算法和策略梯度定理相结合，即发展出 Actor-Critic 模型，本文在这些模型基础上继续改进，同时与其进行对比。后者包含交易和投资组合管理任务，两个任务的交易动作和规则略有不同，但都是重要的量化投资任务。此外，本章还介绍了相关金融指标，包含建模涉及到的技术指标及评估模型的指标。下文将在本章介绍的理论基础上继续发展和研究。

第 3 章 表征学习下的多模态策略梯度方法

本章首先详细介绍数据集及数据集预处理过程，展示模型使用的数据形式和内容。其次给出 RL 任务虚拟环境的具体建模流程和一些限制条件，这是模型训练的基础。随后提出一种多模态策略模型，该模型利用结构化数据和图像数据作为输入，利用深度学习技术提取市场信息来帮助智能体进行多股票投资任务决策。模型同时使用表征学习技术去更好地拟合并学习复杂的市场动态状况，并构造辅助任务来训练表征学习模块。在本章最后的实验结果部分，将对比提出的模型与策略梯度方法基准模型以及基于统计的投资方法的测试结果，以证明本章模型的有效性。

3.1 数据集介绍及处理

3.1.1 道琼斯工业指数数据集

本文考虑的金融市场环境为股票市场，具体为美国道琼斯工业平均指数(Dow Jones Industrial Average Market, DJIA)市场(以下简称道指或 DJIA)对应的 30 支股票。美股是当今全球最活跃、最庞大的资本金融市场之一，美股公司的总市值占据全球股票市场总市值相当大的比例。由于美股具有高流动性并能提供大量投资机会，因而是各类投资者和投资机构投资的重点领域，具有较强代表性及研究价值。道指作为最古老的股票市场指数之一，通常被视作美国金融市场繁荣状况的重要参考指标。因而，分析道指市场的波动变化，能够构建出通用的投资策略，从而方便将策略外推应用到更大范围的市场中去。

道指市场的 30 支成分股股票所对应的公司，来自于美国科技、金融、工业、教育医疗等不同重点行业，其整体表现能够反应美国工业和经济的整体发展水平和状况。道指成分股会随着公司发展和市场状况的变化而做出小范围调整，截止到 2024 年 2 月，道指成分股成员如表 3.1 所示。可以看到，道指成分股涵盖了各类重点行业，表中用不同颜色来区分不同行业。所谓道指，即 DJIA，则是综合三十支成分股的价格计算出的股价加权指数，各支股票价格对道指计算的贡献度也随市场变化而动态调整。

表 3.1：道琼斯工业平均指数成分股

Table 3.1 Dow Jones Industrial Average component

道琼斯工业平均指数成分股 (截止到 2024.02)：股票代码-公司中文名称			
科技与电信	金融与保险	医疗保健	消费品与零售	工业与能源	娱乐与传媒
AAPL-苹果	AXP-运通	AMGN-安进	KO-可口可乐	BA-波音	DIS-迪士尼
CRM(Salesforce)	GS-高盛	JNJ-强生	MCD-麦当劳	CAT-卡特彼勒	
CSCO-思科	JPM-摩根大通	MRK-默克	NKE-耐克	MMM-3M	
IBM-国际商业机器公司	TRV-旅行者保险	UNH-联合健康	PG-宝洁	HON-霍尼韦尔	
INTC-英特尔	V-维萨卡		WBA-沃尔格林联合博资	CVX-雪佛龙	
MSFT-微软			WMT-沃尔玛	DOW-陶氏	
VZ-威瑞森			HD-家得宝		

本文使用 OHLCV 类型数据，即以股票的开盘价(Open)、最高价(High)、最低价(Low)、收盘价(Close)、成交量(Volume)为信息所组成的数据形式。本文从雅虎财经网站(https://hk.finance.yahoo.com/)获取所有成分股、DJIA 以及 VIX(恐慌性指数)的 OHLCV 数据，通过调用 Python 的 yfinance 库即可以下载股票在目标时间段的数据。综合考虑数据量和时效性，本文使用时间范围为 2010 年 1 月到 2024 年 1 月的日度数据，30 支股票共计约 10 万条数据。原始数据集的部分展示如表 3.2 所示，表中价格单位为美元，AAPL、MSTF 和 DJIA 分别表示苹果公司、微软公司和道指。

表 3.2 OHLCV 数据展示

Table 3.2 OHLCV data presentation

股票代码	日期	开盘价	最高价	最低价	收盘价	调整收盘价	成交量
AAPL	2010/1/4	7.62	7.66	7.59	7.64	6.48	4.94e8
	2010/1/5	7.66	7.69	7.62	7.65	6.49	6.02e8
MSFT	2010/1/4	30.62	31.1	30.59	30.95	23.47	3.84e7
	2010/1/5	30.85	31.1	30.59	30.96	23.48	4.97e7
DJIA	2010/1/4	10430	10604	10430	10583	10583	1.80e8
	2010/1/5	10584	10584	10522	10572	10572	1.89e8

3.1.2 数据集预处理及分析

获取原始 OHLCV 数据之后，首先需要进行数据预处理工作，以获得后续 RL 环境建模所需要的具体数据内容。数据预处理主要包含两部分工作：数据清洗、技术指标构造。

1. 数据清洗

(1). 使用调整后的收盘价(Adj Close)替代收盘价。OHLCV 类型数据中具有开盘价、收盘价、调整后的收盘价等 5 种价格，但收盘价一般被认为最能体现市场参与者共识，因其反应了一天之内资产的最终评估，不似开盘价那样易受夜间新闻舆情的影响。此外，调整后的收盘价考虑了除权因素，更能准确体现公司股票价值。故本文主要使用调整后的收盘价进行后续计算和分析，下文的收盘价都指调整后的收盘价。

(2). 去除缺失值。由于一些公司上市比较晚，因而需要将由于上市时间段较短而存在较多数据缺失情况的股票去掉。此外，采取直接删除的手段处理极少数的中间数据缺失的情况。经过清洗后，2010 年 1 月到 2024 年 1 月共保留 29 支股票，DOW 股票被去除掉。

2. 技术指标构造

(1). 构造技术指标。技术指标是基于历史价格和交易量数据计算出的反应市场趋势和价格波动的指标，具有更丰富的信息。本文共构造了 8 种技术指标，见表 3.3。

表 3.3 本文构造的技术指标名称

Table 3.3 The name of the technical index constructed in this paper

英文符号	中文	英文符号	中文
MACD	异同移动平均线	BOLL_UB	布林带上轨
BOLL_LB	布林带下轨	RSI_30	相对强弱指数
CCI_30	30 日顺势指标的滑动平均	DX_30	30 日方向指标的滑动平均
Close_30	30 日收盘价的滑动平均	Close_60	60 日收盘价的滑动平均

其中 MACD 和 DX 在第二章已有介绍。布林带指标是一种趋势指标，包含上轨、中轨、下轨。上轨和下轨分别是中轨加减两倍标准差得来，分别表示市场过热或过冷的阈值。RSI 是一种动量指标，能通过价格计算出处于超买或超卖的状态。CCI 是用于确定价格趋势反转的信号。这些指标都是常用且有效的评估指标。

(2). 构造市场预警指标。市场预警指标是指基于更广泛的资产所计算出的反应市

场情绪和稳定性的指标。本文使用两种预警市场指标：VIX 和 Turbulence。VIX 可以直接从雅虎财经获取，Turbulence 的计算方式如公式 3.1：

$$turbulence_t = (y_t - u)\Sigma^{-1}(y_t - u)' \in \mathbb{R} \tag{3.1}$$

其中 $y_t = {price_t}/{price_{t-1}} \in \mathbb{R}^n$ 表示全部股票在 t 时刻的 *returns*，$u \in \mathbb{R}^n$ 表示历史平均 *return*，$\sum \in \mathbb{R}^{n \times n}$ 为 *returns* 的协方差矩阵。预警指标能够预警股票价格的极端波动情况，例如，当预警指标到达设定阈值时，智能体将会终止购买动作并逐渐减持股票。

数据预处理之后，获取到 29 只股票自 2010 年 1 月到 2024 年 1 月的收盘价、成交量数据，DJIA 数据，8 种技术指标和两种预警指标。为了更好地了解数据特点和分布，为下文训练模型提供指导，需要进行进一步的数据分析和展示。

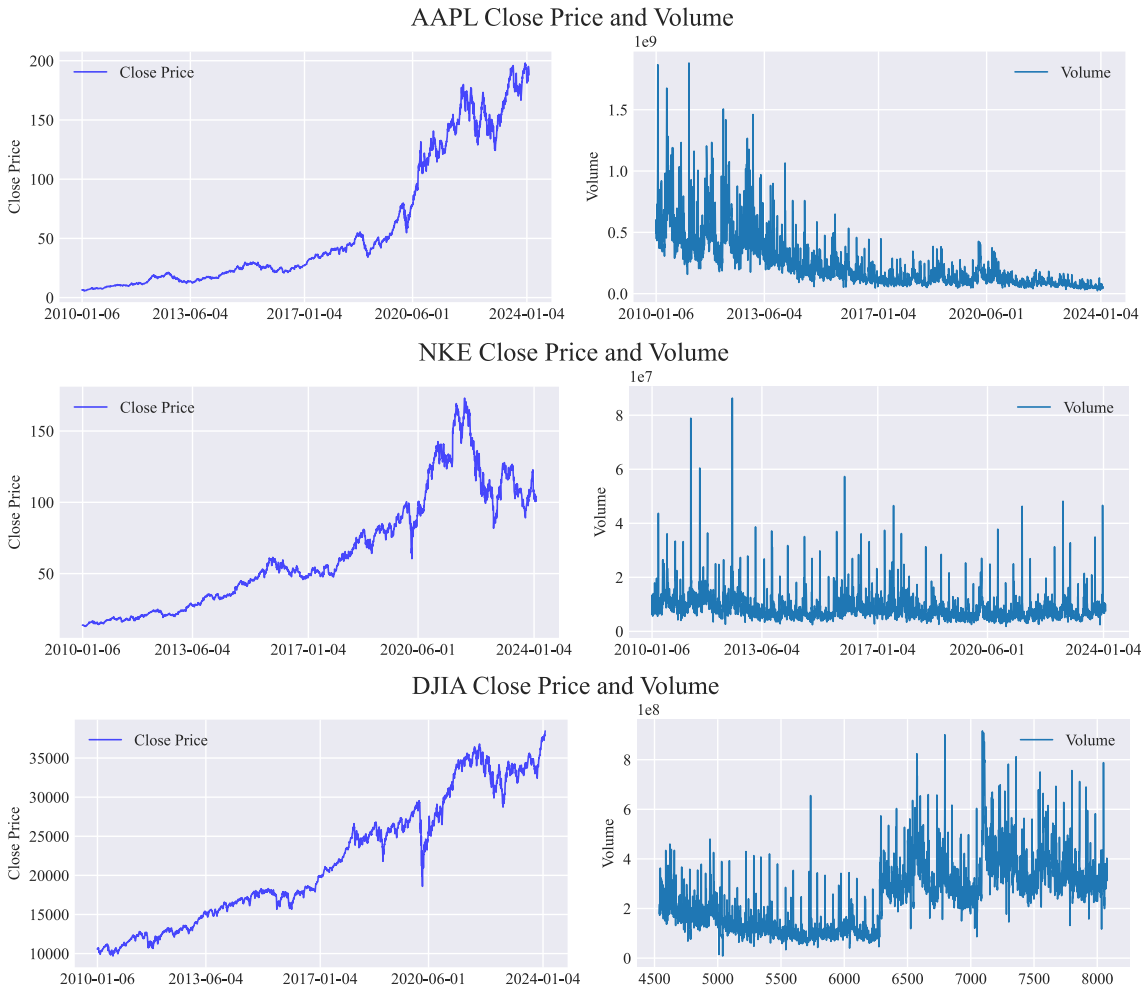


图 3.1 不同证券的收盘价(收盘点数)、成交量的对比

Fig. 3.1 Comparison of closing prices (closing points) and trading volume of different securities

图 3.1 绘制了苹果公司、耐克公司和道指自 2010 年到 2024 年的收盘价(收盘点数)以及成交量的时序变化趋势对比。三种资产分别对应图 3.1 的上中下三部分图，

图左侧为收盘价，图右侧为成交量。从图中可以看到，不同资产的价格变化趋势存在差异，苹果的股价基本呈现上升趋势，即使在 2022 和 2023 年存在短暂下跌，也在 2023 年后半段回升并继续增长，而耐克公司股价在相同时段出现下跌后并未及时恢复。此外，两家公司日成交量情况也不同，苹果的日度成交量持续减少，而耐克则保持基本平稳，这说明金融市场状况复杂多变，不同股票受到宏观经济和自身经营的影响而呈现不同特点。道指综合了 30 支股票的信息，具有一般性和代表性，含有更丰富的信息。不同股票的价格及成交量变化趋势不同，反映了资产间的差异。道指综合全部股票信息，体现了整体市场一般性的变化趋势。

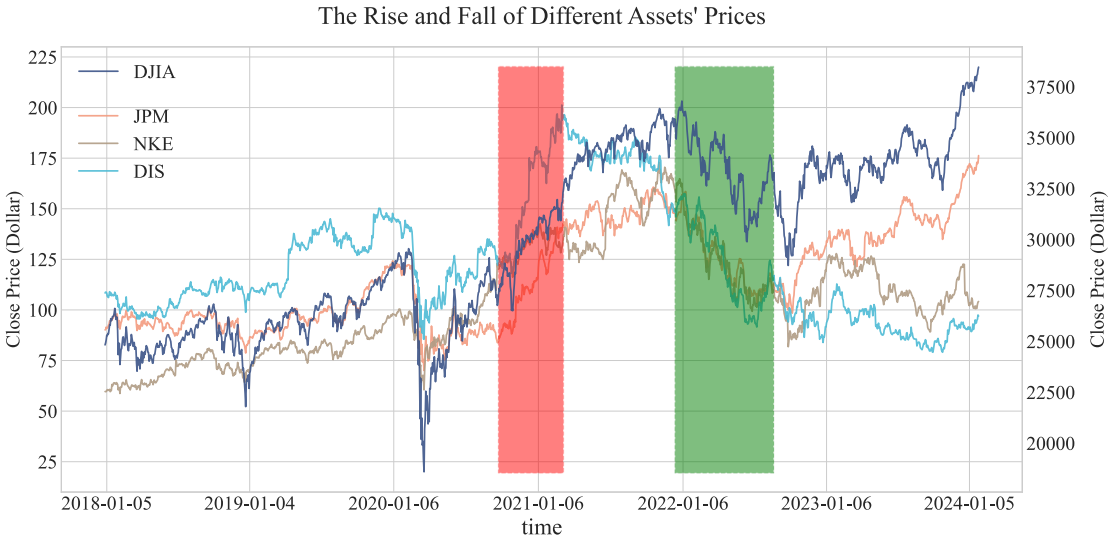


图 3.2 不同证券收盘价的涨跌趋势对照

Fig. 3.2 Comparison of the upward and downward trends of closing prices of different securities

图 3.2 则从资产的趋同性角度来说明数据特点，其绘制了摩根大通、耐克、迪士尼、道指的收盘价的变化趋势。三种股票数据(JPM, NKE, DIS)对应左侧数轴，道指数据(DJIA)对应右侧数轴，绿色和红色覆盖时间段分别表示价格上涨和下跌周期。可以看到，一些股票在相似的大盘环境和宏观经济的影响下，于某些时段呈现出一致的周期性和规律性。例如，绿色(2020.10 到 2021.3)和红色(2021.12 到 2022.8)标注出股票价格明显上涨和下跌的时段，三支股票和道指在对应周期内均呈现出相似的变化趋势。这说明，道琼斯工业平均指数市场数据集具备深入挖掘和提取规律信息的需求和可行性。

3.2 强化学习任务环境建模

RL 训练过程是智能体和环境交互的过程，环境是智能体交互的对象以及获取数据的来源。故 RL 训练任务的首要工作就是创建环境。求解量化投资任务需要创

建虚拟的股票交易市场环境，模拟市场运行状况以供智能体交易。本节将利用处理的数据创建由数据驱动的虚拟环境，使用 Python 语言编写运行。

3.2.1 交易任务

本小节首先介绍环境定义最重要的三个要素：状态、动作、奖励，其次设定交易流程和规则，最后引入环境的交易限制。

1. 状态

在交易任务中，环境在 t 时刻的状态是长度为 $1+N\times2+N\times8$ 的一维向量，N 表示股票数目。这里的数字 1 对应的分量表示 t 时刻的现金；数字 2 对应的长度为 N 的两部分别表示 t 时刻 N 支股票的价格和智能体所持股数；数字 8 对应的部分表示根据每支股票的收盘价计算出的 8 个反映市场状况的金融指标(在 3.1.2 节中介绍)。即，状态包含 4 部分：现金+股价+持股数+技术指标，如图 3.3 所示。

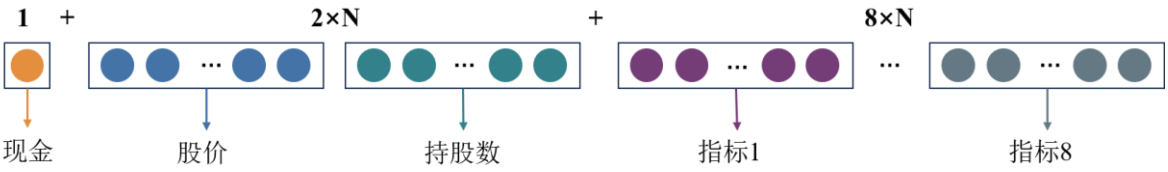


图 3.3 交易任务环境的状态

Fig. 3.3 The state of the trading task environment

2. 动作

环境在 t 时刻的动作是长度为 N 的一维向量，每个分量表示智能体针对对应股票做出的交易动作，其取值范围为$[-1, 1]$。在虚拟环境中，接收到的动作需要乘上 $hmax$ 参数转换为具体的持股数，取 $hmax$ 为 100。故动作取 -1时，表示最大程度卖出，最多卖出 100 股，取 1 时表示最大程度买入 100 股，如图 3.4 所示。

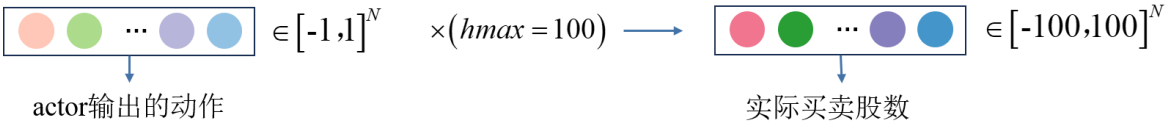


图 3.4 交易任务环境的动作

Fig. 3.4 The action of the trading task environment

3. 奖励

奖励有多种定义方式。本文选择将 t 时刻动作执行完毕后的资产总额 $asset_t$ 和 $t-1$ 时刻资产总额 $asset_{t-1}$ 的差作为奖励，考虑到股票价格较高时奖励会较大，故对奖励进行缩放，实际用于训练的奖励为 $r_t=(asset_t-asset_{t-1})*reward\_scaling$，根据初始资产的量级决定缩放因子 $reward\_scaling$ 大小为 10^{-6}，使得奖励均值处在 0 附近。

4. 交易过程

图 3.5 给出了交易任务的执行流程，它是智能体做出动作、环境步进这两部分循环交替执行的过程。图中左部分是主体流程，为了方便绘制，将其中买入和卖出部分的详细过程单独列在右边，并用绿色和红色虚线框标识。应当留意，买卖需要符合现实逻辑，即考虑当前现金是否充足。此外，交易任务引入了预警机制，当 t 时刻的预警指标取值大于设定的阈值时，会执行特殊的买卖动作。如果是买入动作，则停止买入；如果是卖出动作，则无视出售上限，卖出当前所有股票，以规避可能到来的风险。此外，更新现金时环境会考虑买卖交易的手续费，设置手续费比例 $cost=0.001$，即每卖出 100 美元价值的股票，需要支付 0.1 美元手续费，实际获得现金为 99.9 美元；买入时，购买价值 100 美元的股票，需要支付现金 100.1 美元。最后，为了尽可能实现智能体做出的动作，程序会首先执行卖出操作，然后执行买入操作，防止出现因现金不足而导致智能体无法执行预期购买的情况。

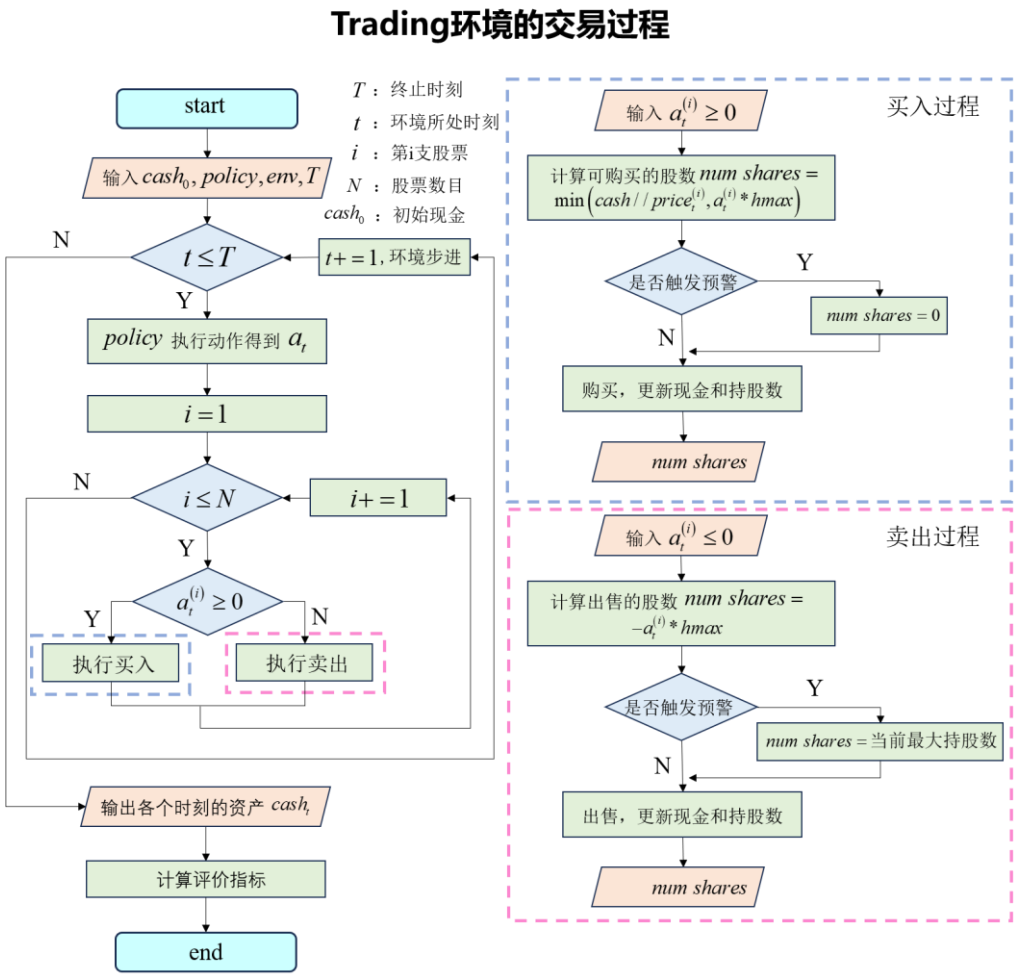


图 3.5 交易任务环境的步进流程图

Fig. 3.5 The flow chart of step process of trading task environment

3.2.2 投资组合管理任务

投资组合管理任务的状态和动作与交易任务不同，但整体交易过程类似。

1. 状态

投资组合管理任务的状态是形状为$(N+4)\times N$的二维矩阵，它包含两部分，上半部分是$N\times N$形状的N支股票收盘价向量之间的协方差矩阵；下半部分是$4\times N$形状的N支股票 4 种技术指标(MACD, RSI, CCI, ADX)。因为投资者通常使用股票价格间的标准差来度量特定的资产分配权重的风险，因而协方差矩阵更适合投资组合管理任务。t时刻环境状态$\boldsymbol{state}_t$的公式见 3.2，其中$\mathrm{cov}_{i,j}$，$1\le i, j\le N$表示股票i和股票j过去 252 天(美股一年的交易日天数)的收盘价构成的向量的协方差，$indicator(k)_i$，$k=1,2,3,4$, $1\le i\le N$表示利用第i支股票t时刻之前的历史价格所计算出的技术指标。

$$\boldsymbol{state}_t=\begin{bmatrix}\begin{bmatrix}\mathrm{cov}_{1,1} & \mathrm{cov}_{1,2} & \cdots & \mathrm{cov}_{1,N-1} & \mathrm{cov}_{1,N}\\ \mathrm{cov}_{2,1} & \mathrm{cov}_{2,2} & \cdots & \mathrm{cov}_{2,N-1} & \mathrm{cov}_{2,N}\\ \vdots & \vdots & \ddots & \vdots & \vdots\\ \mathrm{cov}_{N-1,1} & \mathrm{cov}_{N-1,2} & \cdots & \mathrm{cov}_{N-1,N-1} & \mathrm{cov}_{N-1,N}\\ \mathrm{cov}_{N,1} & \mathrm{cov}_{N,2} & \cdots & \mathrm{cov}_{N,N-1} & \mathrm{cov}_{N,N}\end{bmatrix}\\ \begin{bmatrix}indicator(1)_1 & indicator(1)_2 & \cdots & indicator(1)_{N-1} & indicator(1)_N\\ \vdots & \vdots & \ddots & \vdots & \vdots\\ indicator(4)_1 & indicator(4)_2 & \cdots & indicator(4)_{N-1} & indicator(4)_N\end{bmatrix}\end{bmatrix} \tag{3.2}$$

2. 动作

投资组合管理任务的动作是长度为N的一维向量，向量的每个分量表示总资产分配到对应股票的比例。动作需要$softmax$函数归一化到$[-1,1]$，故动作为单位向量。投资组合管理任务就是不断调整资产分配方案，来实现最终时刻利益最大化。动作的公式表示见公式 3.3，其中$w_{t,i}\in[0,1]$，$1\le i\le N$，表示t时刻资产分配到第i支股票的比例。

$$\boldsymbol{action}_t=\left[w_{t,i},w_{t,2},...w_{t,N}\right]\in[0,1]^N,\ \sum_i w_{t,i}=1 \tag{3.3}$$

3. 奖励

和交易任务一致。

4. 交易过程

智能体在$t+1$时刻给出新的资产分配方案$\boldsymbol{w}_{t+1}$之后，环境会据此计算出新的资产分配方案：$\boldsymbol{w}_{t+1}*asset_t$。随后环境执行步进并根据前后时刻的资产配比，即$\boldsymbol{w}_{t+1}$与$\boldsymbol{w}_t$的差异，依次对各个资产执行卖出和买入操作从而更新资产配比。与交易任务

不同的是，投资组合管理任务买入使用的资金，来自于卖出得到的现金，且这部分现金需要全部使用，而交易任务可以自由使用剩余现金，不必完全用尽。环境步进之后来到 $t+1$ 时刻，此时在新的股票价格 $\boldsymbol{price}_{t+1}$ 下，当前资产配比所具有的资产总额如公式 3.4：

$$asset_{t+1} = asset_t * \boldsymbol{w}_{t+1}^T * \left(\frac{\boldsymbol{price}_{t+1}}{\boldsymbol{price}_t}\right) \tag{3.4}$$

从而，投资组合管理任务的目标即为极大化如公式 3.5 所示的 T 时刻资产总额：

$$asset_T = asset_0 * \left[\boldsymbol{w}_1^T \left(\frac{\boldsymbol{price}_1}{\boldsymbol{price}_0}\right)\right] * \left[\boldsymbol{w}_2^T \left(\frac{\boldsymbol{price}_2}{\boldsymbol{price}_1}\right)\right] * \ldots * \left[\boldsymbol{w}_T^T \left(\frac{\boldsymbol{price}_T}{\boldsymbol{price}_{T\text{-}1}}\right)\right] \tag{3.5}$$

故需要训练最优策略，来持续做出能令收益最大化的资产配比方案。

3.2.3 环境假设

虽然使用数据驱动的方式能够搭建出符合基本市场规律的环境，但虚拟环境和真实股票市场依旧存在差异。为了方便后续建模训练，本文需要在不影响最终效果的前提下简化问题，因而需要做出一定假设。

1. 市场无影响假设

这是指本文的智能体在交易市场中的买卖交易对市场基本没有影响，交易不会引起股票价格的大幅波动。在此前提下，才可以利用历史股票价格来建模环境，使得环境依次步进，给出预期的股票价格变化。由于本文智能体的初始资金较少，且本文选取了市值规模庞大的股票市场作为研究对象，市场流通性较好，因而可以认为假设是合理的。

2. 交易可执行假设

这是指智能体做出的符合环境步进逻辑的合法动作都可以被执行，而不受到现实中其他外部因素影响，不存在交易阻碍。由于在现实正常的市场环境中的绝大部分时间段内，上述条件都可以得到满足，因此可以认为假设是合理的。

3.3 基于表征学习和策略梯度算法的多模态改进策略

本节将提出一种结合表征学习的多模态策略模型，模型的 RL 架构为策略梯度方法，主要用于缓解复杂任务的环境下所出现的状态表征困难、观测信息不完全等问题。该模型特别引了入图像数据，从模态和状态时刻维度双重扩展市场环境状态的表达，同时还集成 CNN 和 LSTM 模块，以抽取市场的动态特征并学习其时序变化规律。提取到的信息与原始环境状态整合后，作为智能体观测的信息输入到 RL

系统中，以指导决策来极大化累计收益。此外，本节还将引入表征学习，构建一个额外的状态特征提取模块，并设计辅助训练任务以优化表征学习模型的训练过程。本节主要创新点如下：

1. 引入图像模态数据，丰富环境状态表达

引入蜡烛时序图、DMI 指标图和 SSO 指标图三种图像模态数据，与向量状态一道共同作为环境状态，以较小的训练代价供智能体观测更多时刻、更全面市场状况。

2. 结合 CNN、LSTM 模块处理图像数据，得到融合特征

结合 CNN 和 LSTM 处理图像数据。CNN 提取市场整体动态变化特性，LSTM 循环学习市场长期时序变化规律。RL 模型使用融合特征进行决策。

3. 结合表征学习模块进一步提取状态特征

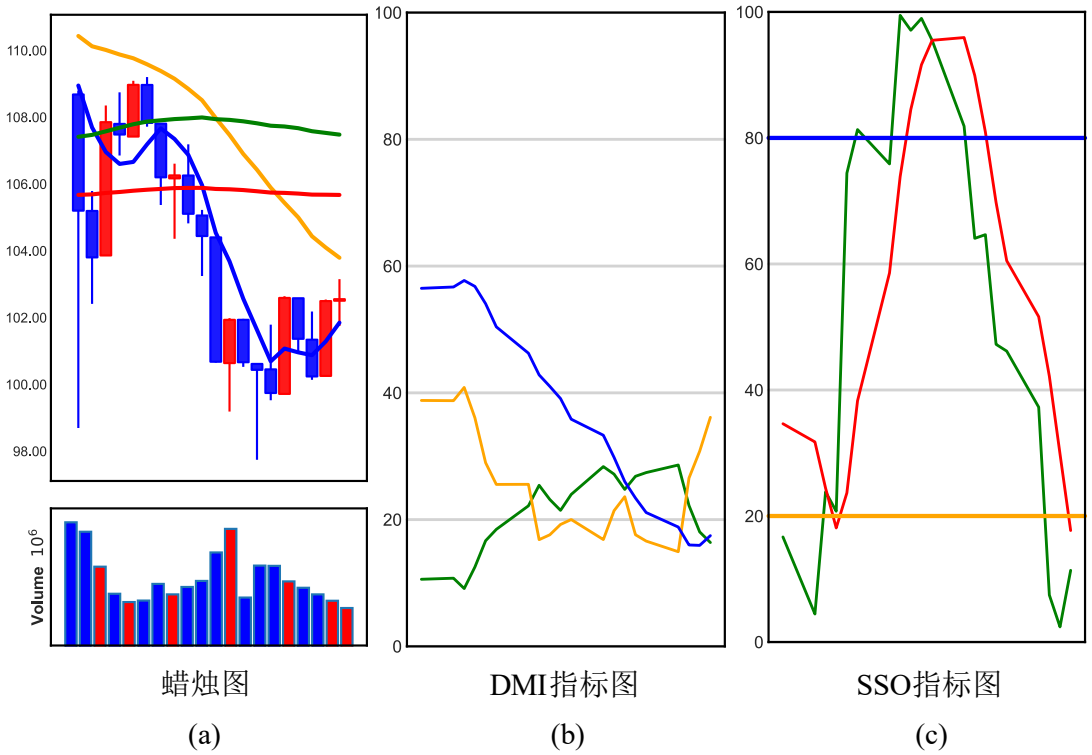
基于表征学习理论，构造表征学习模型，用于进一步处理原始向量环境状态，获得更易于智能体学习的特征表示。针对不同的投资任务，设计不同的自监督辅助训练任务，以帮助训练表征学习模型。

4. 结合基于策略梯度方法的 RL 模型，适应多市场多任务

RL 决策模型采用基于策略梯度算法类型的模型，由于其动作空间连续，故模型能够适应任意资产数量的建模，从而适应不同市场以及交易和投资组合管理等多种投资任务。

3.3.1 多模态策略梯度方法

首先介绍多模态方法使用的图片数据。本文参考文献[31]的工作，使用三种图片作为图像数据：蜡烛图、DMI 指标图、SSO 指标图。从而，每一时刻智能体观测到的环境状态包含两部分：原始的向量环境状态和图像。两种环境状态分别以结构化数据和图像数据的形式来表达，两者互为补充，共同作为智能体在 t 时刻观测到的状态。以描述 2010 年 6 月 4 日的市场状态的图片为例进行说明，如图 3.6 所示。图中(a)、(b)、(c)三幅子图分别为蜡烛图、DMI 指标图、SSO 指标图。图中数据时间跨度为 2010-06-04 及过去 20 个交易日。(a)图蜡烛图的下半部分为成交量图。图片绘制使用 DJIA 的收盘价和成交量数据。



蜡烛图 DMI指标图 SSO指标图

(a) (b) (c)

图 3.6 本文使用的三种类别的图片

Fig. 3.6 This paper uses three kinds of pictures

(a)图的蜡烛时序图包含上下两部分：蜡烛时序图和成交量图。蜡烛图可以表示一段时间内价格波动，由实体和影线构成。实体为蜡烛身，影线为实体上下的竖线。实体、影线的两个边界分别表示当日的收/开盘价和最高/低价。蓝色表示收盘价高于开盘价，意味着价格上涨，市场逐渐走高。下部的成交量图表示每日成交股数量，颜色同理。蜡烛图还绘制了股票价格过去 5、20、60、120 时刻的简单移动平均线。

(b)图的 DMI 指标图表示第二章介绍的 DMI 指标，三条线分别表示 +DI 、 −DI 、ADX 三个指标。(c)图的 SSO 指标图表示第二章介绍的 SSO 指标，两条折线分别表示%K fast 和%D slow 线，两条横线表示预示市场状况转换的临界值。

上述三幅图使用过去多个时刻的常用指标数据绘制图片，实际上是以图片形式扩充了市场状态信息。原本智能体仅能观测到由向量表达的当前时刻市场状态信息，现在能够观测到更遥远时刻的市场状况，形成更连续完整的认知。后续可以用 CNN 和 LSTM 模块有效处理图片数据，提取市场整体动态信息和长期时序变化规律，来帮助 RL 决策的同时也降低模型整体训练难度。

由于本文研究多股票投资任务，所以绘制图片时使用能反映整体市场状况的 DJIA 的收盘价数据，而不是某一支或某几支股票数据。

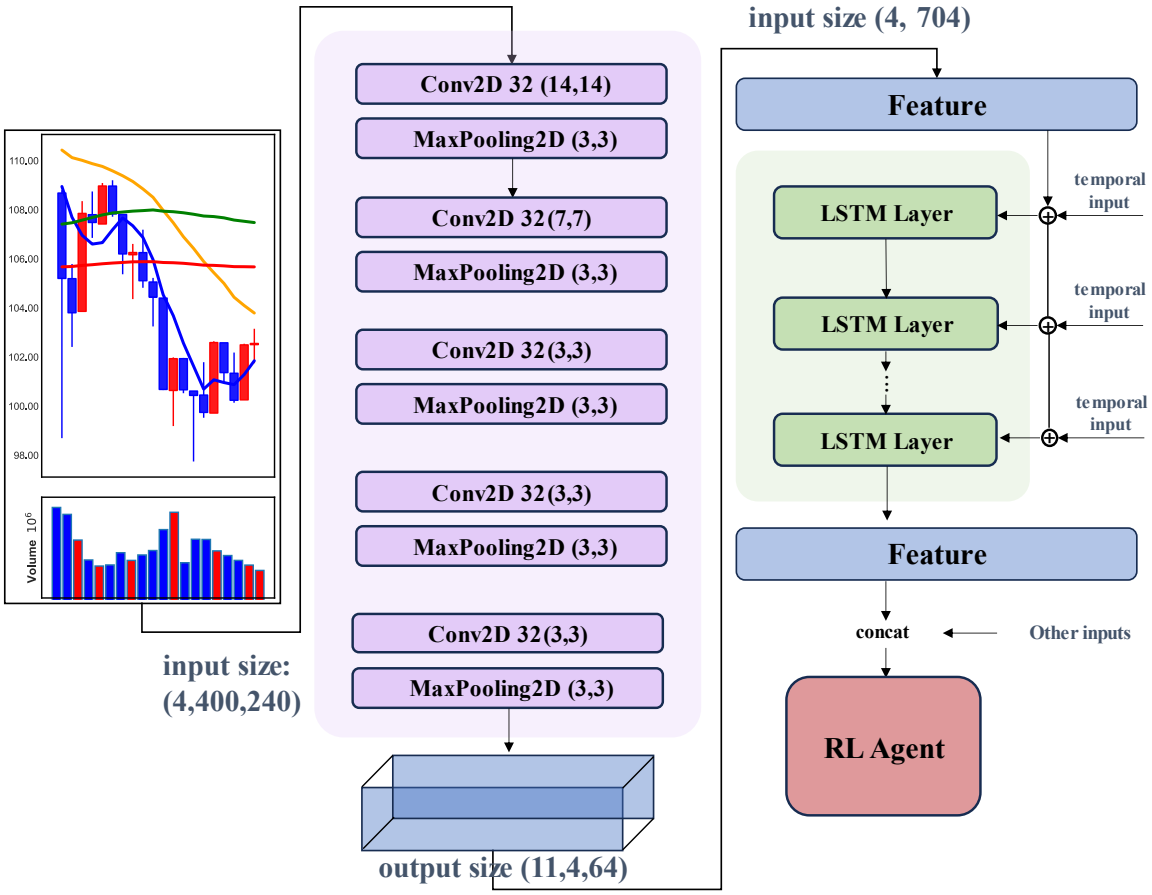


图 3.7 CNN-LSTM 模块结构

Fig. 3.7 CNN-LSTM module structure

用于处理图片数据的 CNN 和 LSTM 模块的结构如图 3.7 所示，图中展示的是一张图片的处理过程。首先对生成的图片进行下采样，处理至(4,400,240)大小。其次由 CNN 模块处理，该模块包含 5 层卷积层和池化层(中间还包含 Batch Normalization 层，为了简便并未画出)，输出尺寸为(11,4,64)的特征图。最终，特征图将按照其列顺序为时间顺序进行拆分，依次送入 LSTM 模块处理，得到图片状态的特征。图片特征将和其他信息一道共同作为 RL 部分的输入。

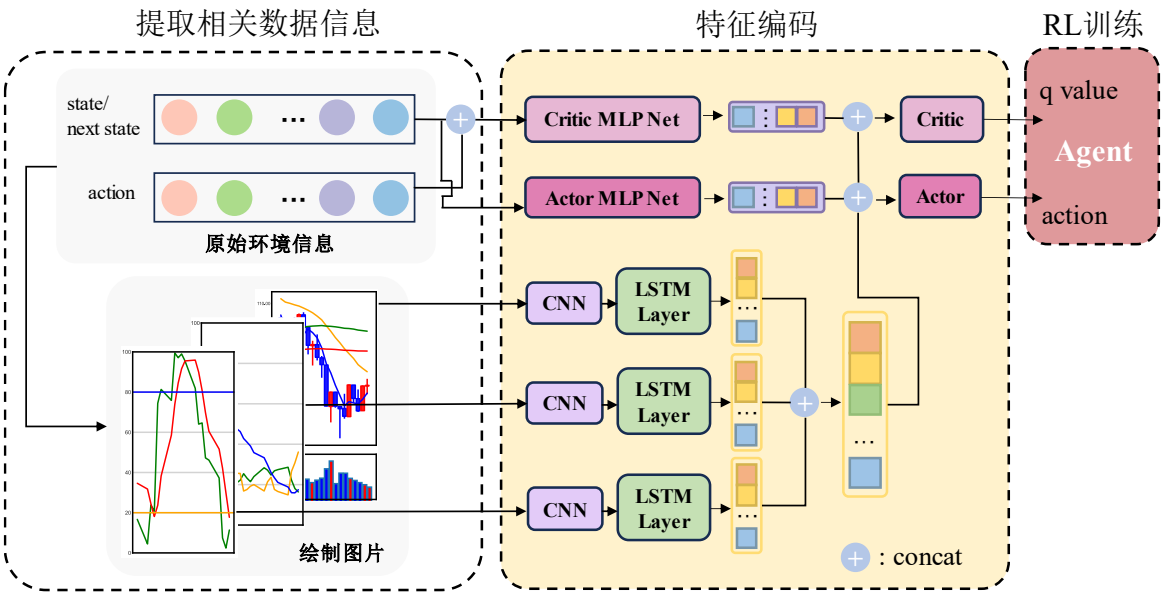


图 3.8 多模态模型整体结构和数据处理流程

Fig. 3.8 Overall structure of multimodal model and data processing flow

图 3.8 给出了全面的数据处理流程和模型架构。图中左半部分为环境的状态部分，此部分输出向量形式的 t 时刻市场状态和图片形式的过去时刻的市场状态。中间部分为特征提取部分，CNN 和 LSTM 模块处理图片，MLP 模块处理向量状态。所有信息综合后，送入 RL 的 Critic 和 Actor，输出动作价值和动作，执行 RL 模型训练。

3.3.2 基于表征学习的辅助特征提取模块

RL 的表征学习是一种学习高效数据表示的方法，它能让 RL 决策部分更好地理解状态，做出决策并获得奖励，继而提升训练效率并增强模型泛化能力。区别于一般的特征提取方法，RL 可以利用任务/环境独有的状态信息来构造辅助训练任务，以帮助训练表征学习模块。由于环境的状态信息具有现实含义，故对此类数据的理解和表达能够帮助智能体理解环境。

本节提出使用表征学习来帮助训练高维向量状态的特征提取部分。加入表征学习模块的整体流程如图 3.9 所示，红色虚线框部分为表征学习的 SRL model。该部分会将原始环境的向量状态提前进行处理，即 $s_t \xrightarrow{SRL\ model} z^{s_t}$，后续的 RL 部分将利用处理过的状态特征来执行决策，而不再是原始的环境状态。SRL model 有两种训练途径：RL 部分训练以及辅助训练任务。辅助训练任务见图中“计算辅助训练任务 loss”。SRL model 会利用内部辅助预测模块，针对处理过的状态特征预测环境的部分原始状态信息，从而执行优化过程。

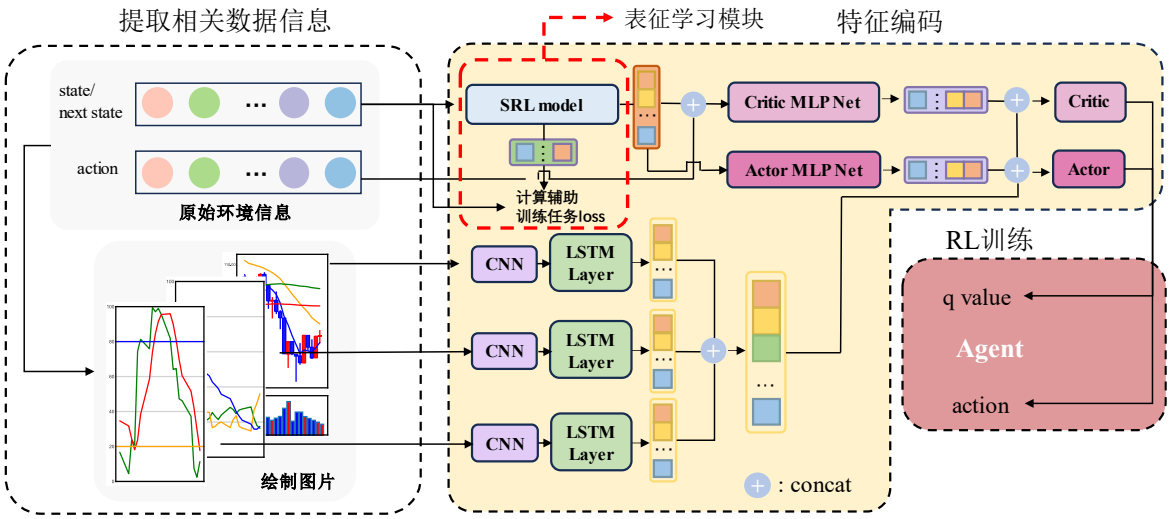


图 3.9 结合表征学习的模型整体结构图

Fig. 3.9 The overall structure of the model with state representation learning

接下来将介绍本文涉及的 SRL model 的具体结构及自监督训练任务。参考文献[25-28]，本文尝试使用三种表征学习模型，如图 3.10 所示。图 3.10 展示出三种结构的模型，标号由 1 到 3 分别为 OFENet、D2RL、DenseNet，它们的区别主要在每层输入和输出的连接方式。OFENet 每一层网络的输出会结合该层的输入，适合处理信息庞杂，观测难度较高的任务。D2RL 是为了解决深度网络在 RL 中的应用问题而提出，每一层网络的输出会结合原始输入，旨在提高信息流通，加速学习过程。DenseNet 将每一层的输出与之前所有层的输入结合起来，每一层结合了之前所有的特征信息，使得特征多次重用。三种网络都被证实能够发挥不同的作用[28]，故本章将对三种网络进行实验分析。参考前人[25,26]工作，本章 SRL model 均使用两层结构。

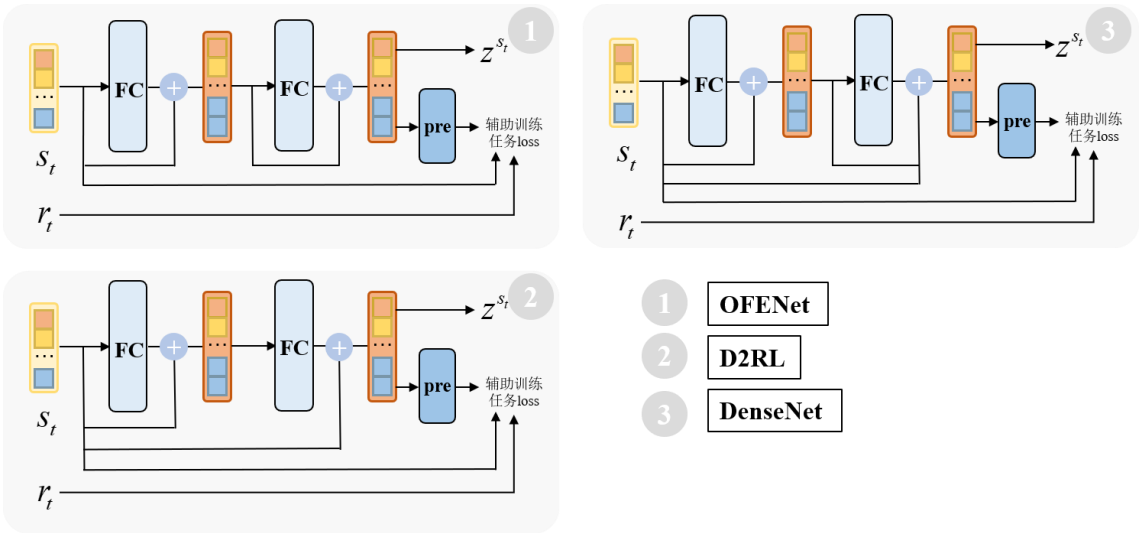


图 3.10 三种表征学习模型结构

Fig. 3.10 Three kinds of state representational learning model structures

参考文献[25]设计自监督训练任务的方式，本节根据量化投资的具体任务设计了两种辅助自监督训练任务。交易任务的辅助训练任务为优化公式 3.6 表示的目标函数，它包含两部分损失。公式中 z^{s_t} 表示 SRL model 处理原始环境状态后得到的状态特征； $f_{pred,s}$ 和 $f_{pred,r}$ 表示 SRL model 的‘预测’模块，来预测环境中具有实际意义的变量，也即图 3.10 中的深蓝色 pre 部分； $s_{t,\,price}$ 表示 t 时刻的状态中的股票价格部分； λ 表示权衡两部分损失的超参数， r_t 表示 t 时刻奖励。投资组合管理任务的辅助训练任务的损失为公式3.7，和前者区别在于第一部分损失中预测的对象不是股票价格，而是股票在 t 时刻的 MACD 指标，即 $s_{t,\,macd}$ 。

$$L_{aux\_trading} = \left\| f_{pred,s}\left(z^{s_t}\right) - s_{t,\,price} \right\|_2^2 + \lambda * \left\| f_{pred,r}\left(z^{s_t}\right) - r_t \right\|_2^2 \tag{3.6}$$

$$L_{aux\_portfolio} = \left\| f_{pred,s}\left(z^{s_t}\right) - s_{t,\,macd} \right\|_2^2 + \lambda * \left\| f_{pred,r}\left(z^{s_t}\right) - r_t \right\|_2^2 \tag{3.7}$$

因后续实验采用 DDPG 模型作为 RL 决策部分，采用 D2RL 模型作为表征学习模块，故本章将提出的模型命名为 **M**ulti-**M**odal with **D2**RL **DD**PG，简称为 MMD4。算法 3.1 给出了 MMD4 的算法流程。本节的主要创新部分在于 13-17 行，其中 13-14 行表示表征学习模型处理原始向量环境状态，得到状态特征，并根据状态特征执行训练过程；15 行进一步处理状态特征和动作，由 MLP 模块给出原始环境状态和动作的聚合特征；16 行由 CNN 和 LSTM 模块处理得到图像特征；17 行合并得到多模态特征，之后 RL 的 Critic 和 Actor 将利用多模态特征来执行决策并训练。

算法 3.1 MMD4

1: 初始化： *policy* 网络，参数为 θ ，它包含 Actor 和 Critic 以及 SRL model 部分。具体包括图片处理部分 $figure_{net}$，这是 Actor 和 Critic 的公共部分；SRL model 提取特征部分 srl_{net}；SRL model 辅助预测部分 srl_{f_pred}；Actor 的特征提取网络 $action_{net}$；Actor 输出动作部分 $actor_{net}$；Critic 的特征提取网络 $value_{net}$；Critic 的输出动作价值部分 $ciritc_{net}$。Actor 和 Critic 各自的整体用 *actor* 和 *critic* 来表示。

2: 初始化目标网络 $policy^{-}$，结构和 *policy* 相同

3: 初始化：容量为 N 的 Replay Buffer 为 D

4: **for** $episode = 1, \ldots, E$ **do**

5: 初始化随机过程 n 为噪声

6: 初始化环境，得到起始状态 s_0

7: **for** $t = 0, \ldots, T$ **do**

8: Agent 执行动作： $a_t = actor\left(s_t\right) + n_t$

9: 环境步进：环境接收动作 a_t ，给出奖励 r_{t+1} 和下一时刻状态 s_{t+1}

10: 存储样本：将 $transition=\left(s_t,\ a_t,\ r_{t+1},\ s_{t+1}\right)$ 存入 D 中

11: **for** $i = 1, \ldots, n\_updates$ **do**

12: 抽取样本：从 D 中随机采样 *minbatch* 数量的样本 $\left(s_j,a_j,r_{j+1},s_{j+1}\right)$，获取对应图片 $figures_j$

13: 将原始环境状态表征为状态特征 $z^{s_j}=srl_{net}\left(s_j\right)$

14: 辅助训练 SRL model：根据损失 $\left\|srl_{f\_pred}\left(z^{s_j}\right)-s_j\right\|_2^2$ 执行梯度下降

15: 得到单模态的状态特征和动作状态合并特征：

$$feature_{action,j}=action_{net}\left(z^{s_j}\right),$$

$$feature_{value,j}=value_{net}\left(concat\left(z^{s_j},a_j\right)\right)$$

16: 处理图片得到图片特征：$feature_{f,j}=figure_{net}\left(figures_j\right)$

17: 合并图片特征，得到多模态特征：

$$feature_{actor,j}=concat\left(feature_{f,j},feature_{action,j}\right)$$

$$feature_{critic,j}=concat\left(feature_{f,j},feature_{value,j}\right)$$

$$action_{pre,j}=actor_{net}\left(feature_{actor,j}\right)$$

$$feature^{pre}_{critic,j}=concat\left(value_{net}\left(concat\left(z_{s_j},action_{pre,j}\right)\right),feature_{f,j}\right)$$

18: 计算 *TD Target*：$y_j=r_j+\gamma critic^{-}_{net}\left(feature^{pre}_{critic,j+1}\right)$

19: 训练 Critic：根据目标函数 $\left(y_j-critic_{net}\left(feature_{critic,j}\right)\right)^2$ 执行梯度下降

20: 训练 Actor：根据目标函数 $critic_{net}\left(feature^{pre}_{critic,j}\right)$ 执行梯度下降

21: 更新目标网络参数：$\theta^{-}\leftarrow\tau\theta+\left(1-\tau\right)\theta^{-}$

22: **end for**

23: **end for**

24: **end for**

3.4 实验结果

3.4.1 基准方法

本文进行对比的基准模型和方法包括两类：基于统计的方法和 DRL 基准模型。本文实现了这些基准方法，方便在本文数据上与本章提出的方法进行对比。

首先介绍交易任务中基于统计的基准方法，此类方法需通过两步产生动作，第一步基于历史数据获得当前时刻的 *signal*，第二步利用缩放映射函数将 *signal* 映射到 $[1,-1]$ 中，以获得合法的执行动作。本文给出了三种交易任务的基准方法，图 3.11 展示了计算 *signal* 的公式以及它们的介绍，图 3.12 给出了缩放映射函数[45]。*Long*

Only 策略意为一直最大程度购买，因而其始终做出数值 1 的动作；*MACD* 策略是基于 MACD 指标获得映射动作，当 MACD 指标绝对值较小时，MACD 变大会鼓励买入，但当绝对值较大时，映射函数会给出保守的或卖出动作；*Immediate* 策略综合长短期收益率的符号做出买入或卖出的动作。

1. 基于统计的方法(交易任务)

策略名称	**Signal**计算方式
Long Only	$signal_t^{(i)}=1$ 每时刻做出最大程度的购买动作。
MACD	$macd_t^{(i)}=MACD(short,long)_t^{(i)}$ $q_t^{(i)}=macd_t^{(i)}/std\left(close_{t,\ldots,t-63}^{(i)}\right)$ $signal_t^{(i)}=q_t^{(i)}/std\left(q_{t,\ldots,t-252}^{(i)}\right)$ 基于MACD指标做出动作。 其中 $MACD(short,long)_t^{(i)}$ 表示t时刻股票i的MACD指标，计算方式见第二章。实际计算时采用三组(*short*, *long*)参数计算结果的均值。$close_{t,\ldots,t-63}^{(i)}$ 表示股票i在t时刻及过去63时刻数据的标准差。$q_{t,\ldots,t-252}^{(i)}$ 同理。
Immediate	$signal_t^{(i)}=(1-w)*sign\left(r_{t,t-252}\right)+w*sign\left(r_{t,t-21}\right)$ 其中 $r_{t,t-252}^{(i)}$ 表示t时刻收盘价和 $t-252$ 时刻收盘价比值。 基于长期(年份)和短期(月份)的收益率做出动作。

图 3.11 交易任务的基准方法计算 *signal* 指标的方式

Fig. 3.11 The method of calculating the signal indicator of benchmark methods in trading task

策略名称	缩放映射函数
Long Only	$f(x)=x$
MACD	$f(x)=x*\exp\left(-x^2/4\right)/0.89$
Immediate	$f(x)=sign(x)$

图 3.12 交易任务的基准方法计算动作的函数

Fig. 3.12 The method of calculating the action of benchmark methods in trading task

2. 基于统计的方法(投资组合管理任务)

其次介绍投资组合管理任务中基于统计的基准方法，此类方法都是按照某种策略进行优化或者依照某种原则来获得分配方案。本文采用 5 种[29,46]投资组合管理任务基准方法，图 3.13 展示了这些基准方法获取动作的方式。其中 UBAH 全称为 Uniform Buy and Hold，意为初始均匀购买并一直持有；UCRP 全称为 Uniform Constant Rebalanced Portfolio，意为初始均匀购买并一直保持资金均匀分配状态；

Best CRP 全称为 Best Constant Rebalanced Portfolio，意为计算出某种最优分配权重；UP 全称为 Universal Portfolio，意为基于统计模拟计算出多种投资组合的收益，根据收益计算这些组合的加权。

策略名称	资金分配方案计算公式
UBAH	$w_0^{(i)}=\frac{1}{m}$ 资金在初始时刻均匀分配，随后一直持有，不干预资金分配方案。
UCRP	$w_t^{(i)}=\frac{1}{m}$ 每时刻调整资金分配，始终保持均匀分配。
Best CRP	$\boldsymbol{w}=\underset{\boldsymbol{w}}{\text{minimize}}\prod_{t=1}^{T}\boldsymbol{w}^T\boldsymbol{r}_t$ 其中 $\boldsymbol{r}_t$ 为t时刻收益率向量，$r_t^{(i)}=\frac{price_t^{(i)}}{price_{t-1}^{(i)}}-1$ 给定历史收益率，求最优的资产分配权重。
Best Stock	$\boldsymbol{w}=(0,0,\ldots,1,0,\ldots,0)$ $i^*=\arg\max_i\prod_{t=1}^{T}\left(1+r_t^{(i)}\right)-1$ 第i个分量为1，其余为0 根据历史数据选择表现最好的一支股票，简单地投资这支股票。
UP	$\boldsymbol{S}_t=\boldsymbol{W}\boldsymbol{R}_t\boldsymbol{S}_{t-1}$ $\boldsymbol{w}_t=\boldsymbol{W}^T\boldsymbol{S}_t$ $\boldsymbol{w}_t=\boldsymbol{w}_t/\sum_{i=1}^{m}w_t^{(i)}$ 模拟各种投资组合情况，计算每种情况的资金增长率，将其作为权重对每种组合加权。 其中 $\boldsymbol{W}$ 为模拟的投资组合权重矩阵。$\boldsymbol{R}_t$ 为t时刻资金增长率，$\boldsymbol{S}_t$ 为t时刻资金累计增长率，也即加权矩阵。

图 3.13 投资组合管理任务的基准方法计算动作的公式

Fig. 3.13 The method of calculating the action of benchmark methods in portfolio task

3. DRL 方法

本文还使用了 4 种 Actor-Critic 类 DRL 模型作为基准方法：DDPG、TD3、SAC、PPO，在第二章已经对其进行了介绍，这里不再陈述。

3.4.2 实验设置

本文将数据集划分为训练集、验证集、测试集三部分，划分方式如图 3.14 所示。图 3.14 中的上图绘制了 DJIA 曲线在不同数据集时间段内的变化，可以看到其在训练和测试集上总体呈现上升趋势，而在验证集上则动荡并趋于下降。图 3.14 的下图则具体给出了三个数据集覆盖的时间范围。三部分数据集覆盖的时间段分别为：2010-01-01 到 2021-13-31；2022-01-01 到 2022-12-31；2023-01-01 到 2024-01-30。

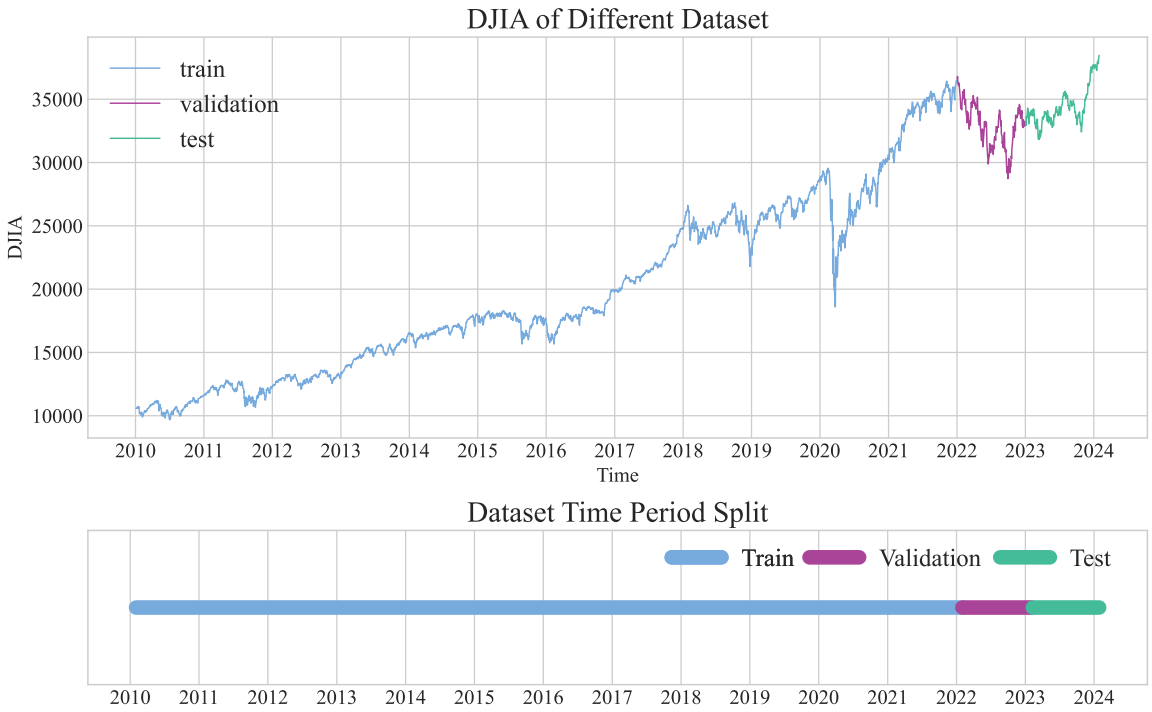


图 3.14 数据集划分方式

Fig. 3.14 Ways to split datasets

实验使用的 Python 版本为 3.7.13，PyTorch 版本为 1.13.1。RL 基准方法的模型结构和超参数采用原论文中的默认设置，但在 *batch_size* 和 *n_updates* 等关键参数上和 MMD4 保持一致。本章提出的 MMD4 方法的超参数设置见表 3.4 所示：

表 3.4 MMD4 模型超参数设置

Tabel 3.4 MMD4 model hyperparameter settings

参数名	参数值	参数含义
episode	10	训练总幕数
batch_size	3(trading), 4(portfolio)	批量大小
lr	5e-8	学习率
buffer_size	1e4	经验池大小
n_updates	3(trading), 5(portfolio)	交互一次训练的次数
gamma	0.99	折扣因子
tau	0.005	目标网络参数软更新参数
srl_hidden_dim	512(trading), 1024(portfolio)	表征学习模型隐藏层维度
lstm_hidden_dim	1024	LSTM 模型隐藏状态维度
activation	ReLU	激活函数

3.4.3 对比试验

首先对比 4 种 DRL 基准模型的测试结果，据此选择合适的模型来构建多模态策略。由于深度学习模型训练具有随机性，故为了更好地反映真实水平，本文所有深度学习模型都训练 5 次，计算 5 次结果的均值作为最终结果。每次训练根据验证集的 Sharpe Ratio 结果来保存最优模型。表 3.5 给出了交易任务和投资组合管理任务下不同模型的测试结果，选取 5 种评价指标进行呈现。Sharpe Ratio 一列包含均值和标准差，其他列为均值结果。下文实验部分以 Trading 表示交易任务，Portfolio 表示投资组合管理任务。表中用红色加粗和蓝色加粗字体分别标识最好效果和次好效果，同时在右上方用数字上标标识排序。例如，对于 Trading 任务的 Cumulative Returns 指标，PPO 和 TD3 模型位列前两位，故 PPO 的结果即用红色标识，TD3 的结果用蓝色标识。除 Sharpe Ratio 的标准差和 Max Drawdown 指标为越小越好外，其他指标都是越大越好。本文之后所有类似的表同理。

表 3.5 DRL 基础模型结果对比

Table 3.5 Comparison Results of Basic DRL Model

任务	模型	评价指标			
		Cumulative Returns	Annual Returns	Sharpe Ratio	Max Drawdown
Trading	DDPG	0.2238	0.2082	1.67 ± 0.323	**0.083**[1]
	TD3	**0.2459**[2]	**0.2287**[2]	**1.74**[2] ± **0.128**[1]	**0.092**[2]
	SAC	0.2410	0.2242	1.71 ± 0.282	0.094
	PPO	**0.2478**[1]	**0.2305**[1]	**1.80**[1] ± **0.276**[2]	**0.092**[2]
Portfolio	DDPG	**0.2236**[2]	**0.2081**[2]	**1.77**[1] ± **0.049**[1]	**0.082**[1]
	TD3	**0.2293**[1]	**0.2134**[1]	**1.67**[2] ± 0.168	0.085
	SAC	0.2112	0.1966	1.62 ± 0.107	**0.082**[1]
	PPO	0.1962	0.1827	1.54 ± **0.060**[2]	0.083

从表 3.5 可以看到，4 种 DRL 模型的效果比较相近，并没有一种模型具有绝对优势。Trading 任务下的 PPO 模型整体效果最好，然而 Portfolio 任务表现很差，因此不选择 PPO。SAC 模型在两个任务中表现均一般，故也被排除。DDPG 模型在 Portfolio 任务中表现最佳，在 Trading 任务中的 Max Drawdown 指标上表现最好，虽

然盈利水平并不突出，但其策略体现出较强稳定性，适合保守型投资者。而尽管 TD3 是 DDPG 的改进版本并显示出某些任务中的优势，鉴于其模型复杂度及未展现出优于 DDPG 的综合水平的表现，因此决定不予选择。

实际上，由于 DDPG 能够输出确定性动作，更适合对动作精确度要求较高的交易任务。相比之下，SAC 和 PPO 等随机性策略模型则有可能引入更多噪声，从而导致效果不佳。综上，本文认为 DDPG 模型更适合研究的市场和所选数据集，因此选择 DDPG 作为多模态策略的核心架构。

表 3.6 三种表征学习模型结果对比

Table 3.6 Comparison Results of Three Representation Learning Models

任务	模型	评价指标			
		Cumulative Returns	Annual Returns	Sharpe Ratio	Max Drawdown
Trading	D2RL	**0.2304**[1]	**0.2144**[1]	**1.77**[1] ± **0.239**[2]	**0.084**[1]
	OFENet	**0.2065**[2]	**0.1921**[2]	**1.57**[2] ± 0.430	0.090
	DenseNet	0.1726	0.1609	1.38 ± **0.154**[1]	**0.086**[2]
Portfolio	D2RL	**0.2286**[2]	**0.2127**[2]	**1.77**[2] ± **0.120**[2]	**0.080**[2]
	OFENet	0.2242	0.2087	1.76 ± **0.073**[1]	0.082
	DenseNet	**0.2524**[1]	**0.2347**[1]	**1.89**[1] ± 0.136	**0.076**[1]

在选择 DDPG 作为多模态模型架构之后，还需要选择合适的表征学习模型。表 3.6 给出了在隐藏层维度均为 512 的情况下，三种表征学习模型的对比结果。D2RL 在两个任务中都展现出最佳的综合表现，故选择 D2RL 作为表征学习模块。进一步分析发现，D2RL、OFENet、DenseNet 三种模型结构的连接稠密度逐步上升，此特性与两个任务不同的输入维度大小相互作用，使得效果呈现一定规律的改变。例如，Trading 任务的原始环境状态维度为 291，较低的连接稠密度如 D2RL 模型即可得到较好效果。更复杂的连接方式反而会导致信息冗余，为模型学习带来困难，使得 OFENet 和 DenseNet 的效果依次降低。相反，Portfolio 任务中，状态维度为 957，更高的维度需要更复杂的连接来有效处理信息，因此 DenseNet 由于其每层输入的合并，能更充分地提取信息，从而表现最佳。

表 3.7 全部模型结果对比

Table 3.7 Comparison Results of All Models

任务	模型	评价指标			
		Cumulative Returns	Annual Returns	Sharpe Ratio	Max Drawdown
Trading	MMD4(ours)	0.2304	0.2144	**1.77**[2] ± **0.239**[2]	0.084
	DDPG	0.2238	0.2082	1.67 ± 0.323	**0.083**[2]
	TD3	0.2459	0.2287	1.74 ± **0.128**[1]	0.092
	SAC	0.2410	0.2242	1.71 ± 0.282	0.094
	PPO	**0.2478**[2]	**0.2305**[2]	**1.80**[1] ± 0.276	0.092
	Long Only*	0.1831	0.1706	1.48	**0.082**[1]
	MACD*	**0.3339**[1]	**0.3098**[1]	1.72	0.119
	Immediate*	0.1993	0.1856	1.24	0.116
Portfolio	MMD4(ours)	**0.2366**[1]	**0.2201**[1]	**1.81**[1] ± 0.243	0.084
	DDPG	0.2236	0.2081	**1.77**[2] ± **0.049**[1]	**0.082**[2]
	TD3	**0.2293**[2]	**0.2134**[2]	1.67 ± 0.168	0.085
	SAC	0.2112	0.1966	1.62 ± 0.107	**0.082**[2]
	PPO	0.1962	0.1827	1.54 ± **0.060**[2]	0.083
	UBAH*	0.1907	0.1776	1.49	0.088
	UCRP*	0.1807	0.1684	1.43	0.087
	Best CRP*	0.1029	0.0961	0.62	**0.082**[2]
	Best Stock*	-0.0144	-0.0135	0.04	0.146
	UP*	0.1823	0.1699	1.45	**0.075**[1]

本章使用 DDPG 和 D2RL 来构建多模态策略模型 MMD4。表 3.7 展示了 MMD4 和全部基准模型的测试结果对比，基于统计的方法用右上角的*符号来标识。由于基于统计的方法不具有随机性，标准差为 0，故在表中不予显示。可以看到，MMD4 模型在两个任务上的综合表现最佳。在 Portfolio 任务中，MMD4 在 Cumulative Returns 和 Annual Returns 两项指标上取得最好的效果，说明模型具有较强盈利水平，能够为投资者带来较高收益。此外，模型的 Max Drawdown 指标也表

现出竞争力，故 Sharpe Ratio 指标亦为最佳，表明模型构建的策略在盈利之余，也能尽量规避市场动荡带来的风险，减少股票价格剧烈下跌带来的损失。在 Trading 任务中，MMD4 模型的 Sharpe Ratio 均值和标准差均位列第二，仅次于 PPO 模型。总体而言，MMD4 展现出了适应多种任务的能力，表现出全面均衡的投资水准。

为了更直观展示 MMD4 模型的特点，图 3.15 和 3.16 使用雷达图对各指标进行了展示。图中指标已进行最小最大值归一化，且为了统一指标偏好，对 Sharpe Ratio Std 和 Max Drawdown 进行取倒数和缩放操作，该处理在图中用右上角的*符号标记。这样处理后，所有指标均遵循越大越好的原则。图 3.16 同理。从图中可以看到，MMD4 模型在 Cumulative Returns、Annual Returns、Sharpe Ratio、Max Drawdown 指标上均有出色表现。然而，由于训练结果之间具有较大差异，导致 Sharpe Ratio 标准差较大，下一章将进行有针对性的改进。

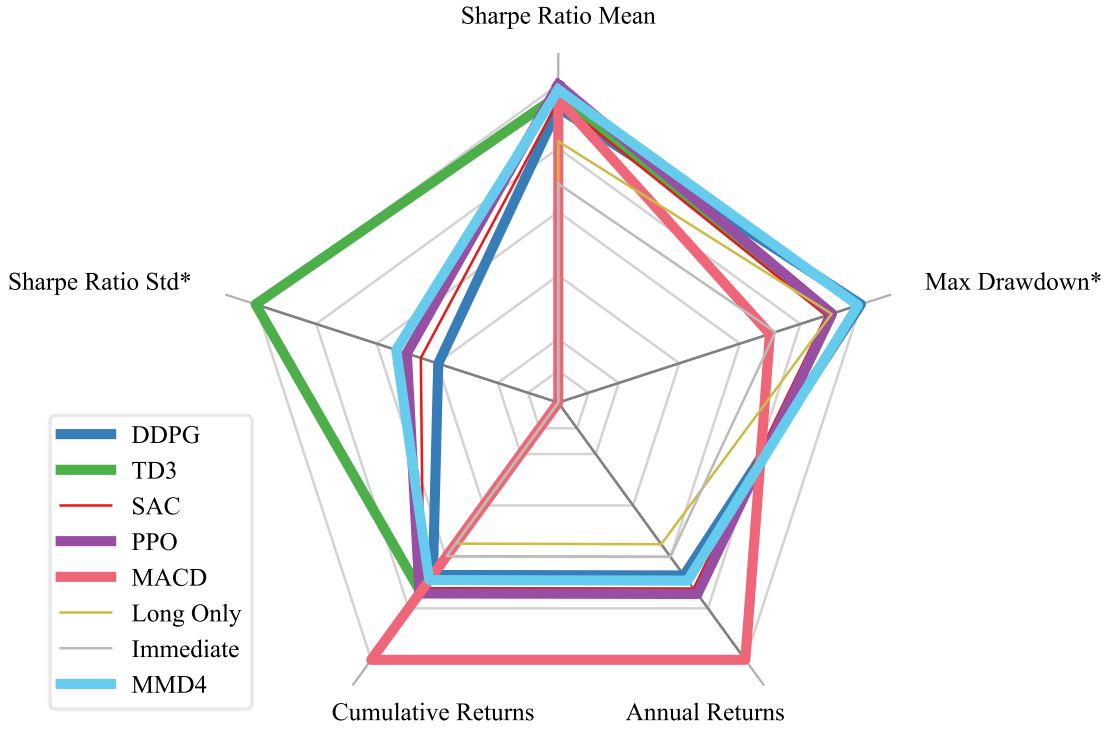


图 3.15 模型在 Trading 任务中的各项指标雷达图

Fig. 3.15 Radar chart of various indicators for the models in the Trading task

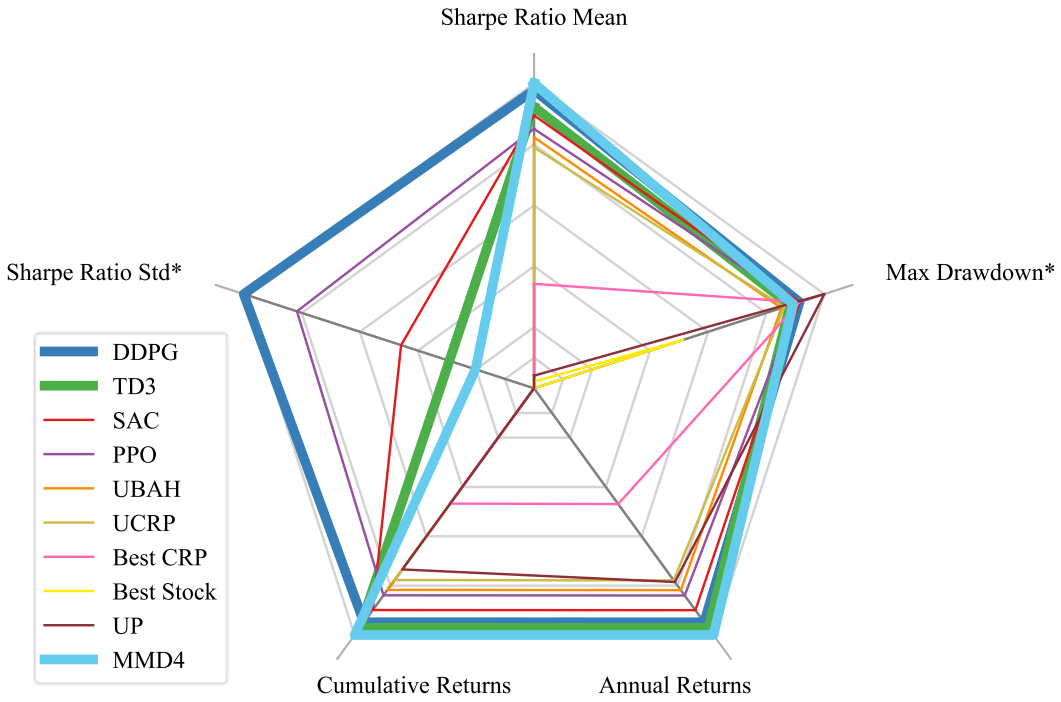


图 3.16 模型在 Portfolio 任务中的各项指标雷达图

Fig. 3.16 Radar chart of various indicators for the models in the Portfolio task

图 3.17 和 3.18 展示了模型在测试集上的资产变化曲线，能直观反应不同模型的收益水平及在特定时间段内投资表现差异。图中增加了用深绿色绘制的 DJIA 的资产曲线，作为展示市场平均繁荣水平的“基准”收益曲线。当策略的累计收益率大于 DJIA 的累计收益率时，认为策略在表现上“跑赢”了大盘平均水平。

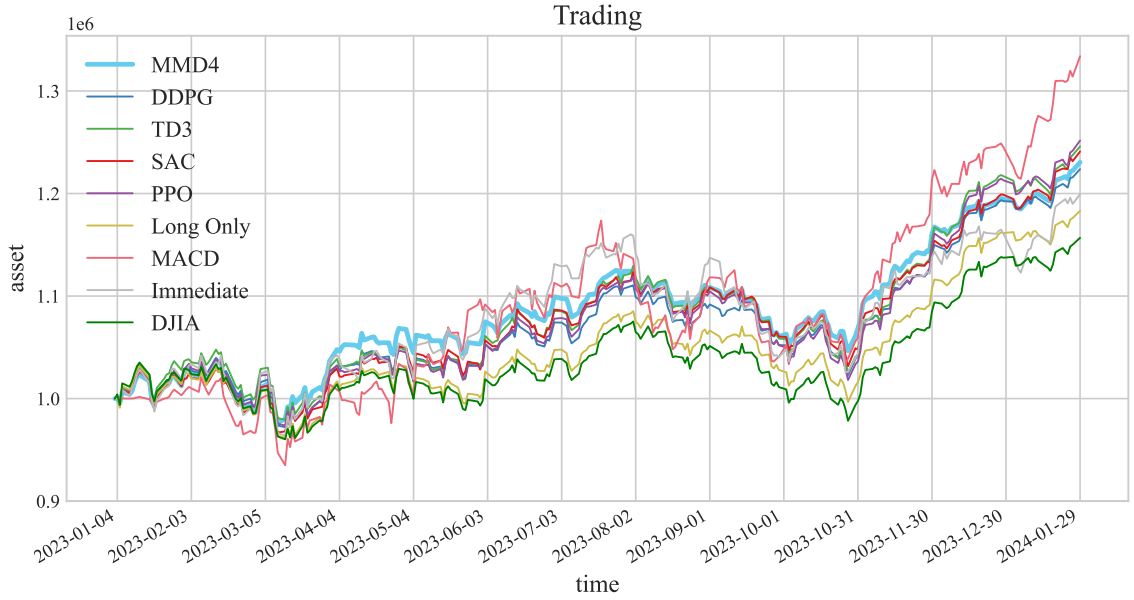


图 3.17 全部模型在交易任务测试集上决策的资产曲线

Fig. 3.17 The asset curves of all models' decisions on the trading task test set

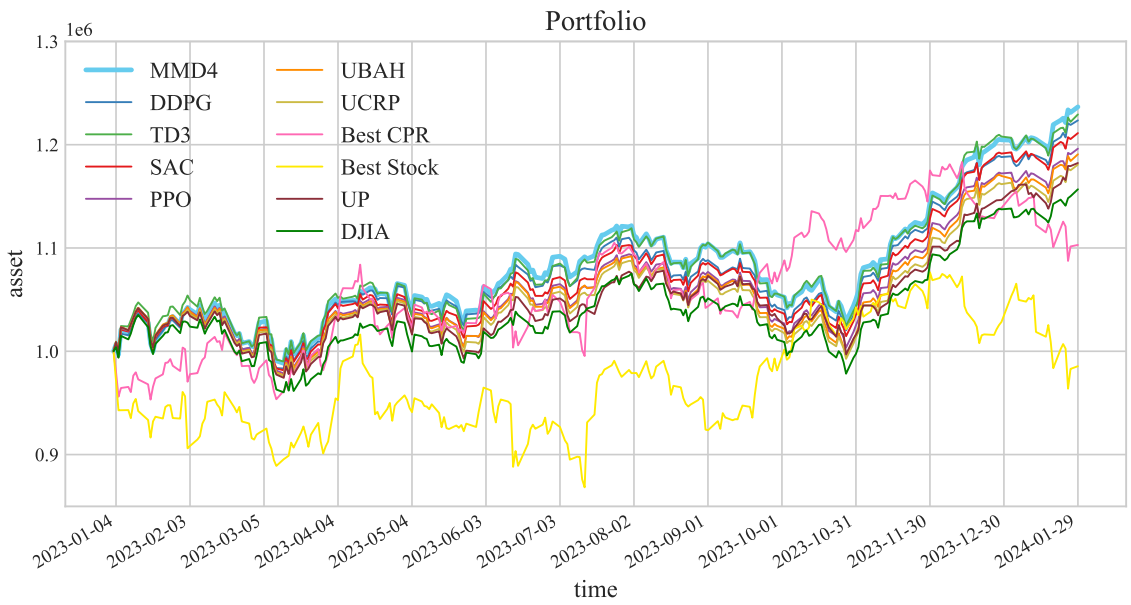


图 3.18 全部模型在投资组合管理任务测试集上决策的资产曲线

Fig. 3.18 The asset curves of all models' decisions on the portfolio task test set

3.17 与 3.18 两图显示，MMD4 模型在两个投资任务中均表现出色。在 Trading 任务中，MMD4 模型能够有效平衡收益和风险，从而在市场剧烈波动期间能够及时调整策略。例如，在 2023 年 2 月到 3 月市场整体下跌期间，MACD 模型遭受到较为严重影响，导致资产水平较高峰期下降较多，而 MMD4 模型则通过及时调整策略，避免了较大的资产缩水。4 到 5 月份情况也同样如此。在 Portfolio 任务中，MMD4 的资产曲线最终位于最高水平，展现出模型良好的收益水准。MMD4 在两个任务下的累计收益率均高过 DJIA 曲线，从而印证本章提出方法的有效性。

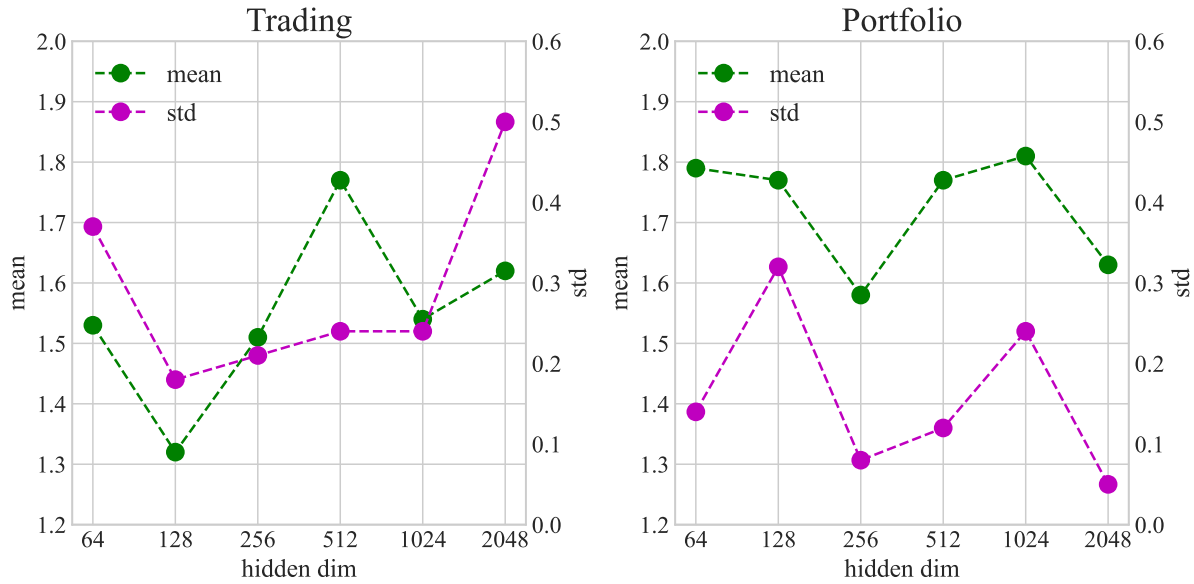


图 3.19 表征学习模型隐藏层维度超参数选择

Fig. 3.19 The selection of hyperparameters for hidden layer dimensions in srl models

最后，本节分析了表征学习模型隐藏层维度(hidden dim)对 MMD4 模型效果的影响。图 3.19 展示了 hidden dim 对 Sharpe Ratio 的均值和标准差的影响，两者均呈现一致的变化趋势，尤其在 Portfolio 任务中表现显著。鉴于盈利是投资的主要目标，故本章选取性能最佳的隐藏层维度：512 和 1204。此外，Portfolio 任务较大的最优 hidden dim 表明，处理较大的环境状态维度时，需要更宽泛的表征学习模型，这进一步验证了前文对表征学习模型的分析结论。

3.4.4 消融实验

本章最后进行了消融实验，以此说明多模态机制和表征学习模块各自对 MMD4 的影响。表 3.8 给出了消融实验 Sharpe Ratio 的均值和标准差结果。MMD4-no-Figure 表示去掉图片输入，变成单模态模型；MMD4-no-Srl 表示去掉表征学习模块，不对原始环境状态向量进行预先处理。

表 3.8 消融实验结果

Table 3.8 Ablation experiments results

任务	模型	评价指标	
		Sharpe Ratio Mean	Sharpe Ratio Std
Trading	MMD4	1.77	0.239
	MMD4-no-Figure	1.52	0.288
	MMD4-no-Srl	1.60	0.658
Portfolio	MMD4	1.81	0.243
	MMD4-no-Figure	1.45	0.128
	MMD4-no-Srl	1.58	0.252

可以看到，两个任务呈现两种类似的趋势。其一为移除图片输入后，模型性能显著下降；其二为去掉表征学习模型后，结果标准差变大。这表明图像输入为智能体提供有助于决策的长期环境状况信息，从而提升训练效果。然而，若仅增加图像输入而不结合表征学习模块，模型对复杂信息的处理能力会不足，增大了 RL 部分的训练难度，导致性能变得极不稳定。有效地处理和整合高维信息，才能减少无关信息的干扰，进而稳定训练过程。因此，图像输入和表征学习模块二者缺一不可，两者结合共同发挥作用才能充分发挥模型的潜力。

3.5 本章小结

本章提出了结合表征学习的多模态策略模型 MMD4。MMD4 模型在原始基于统计和技术指标的向量环境状态的基础上，引入了三类图像数据作为环境状态的额外补充，并使用 CNN 和 LSTM 模型处理图像数据，提取更加丰富的市场状态信息。此外，为了帮助 RL 部分模型训练，MMD4 引入了 D2RL 表征学习模型进一步处理向量部分环境状态，并针对不同投资任务构造特定辅助训练任务来训练 D2RL 模型。MMD4 模型更充分地观测并提取到完整及丰富多样的长期市场变化情况，有利于 RL 部分分析市场环境变化规律，从而做出更有效的决策。

测试结果显示，MMD4 模型策略在两个任务上的盈利水平均位于前列，综合表现超过基准模型。其中 MMD4 模型策略在 Portfolio 任务的 Cumulative Returns、Annual Returns、Sharpe Ratio 指标上取得最优效果，说明模型在盈利和风险管控方面综合能力表现最佳。消融实验结果表明，多模态的环境状态能够显著提升模型盈利水平，表征学习模块能够提升训练稳定性，两者结合能发挥出更好效果。然而，模型 Sharpe Ratio 的标准差结果较大，说明在训练时仍然具有不稳定性问题，且交易任务的效果仍有较大提升空间，第四章将继续做出针对性的改进工作。

第 4 章 集成 Q 学习与 MCTS 提升训练效率

本章将在第三章基础上进一步改进 MMD4 模型，重点针对深度强化学习 Model-Free 类方法所具有的样本采样效率低下和训练不稳定问题进行分析和改进，提出改进模型 MMD4^{+}。Model-Free 类方法虽然简单、直接，但由于需要与环境大量进行交互获取训练样本，故需要较长的交互和训练过程，同时也容易受到环境动态特性改变的影响而出现训练不稳定甚至效果下降的现象。故本章提出使用蒙特卡洛树搜索(Monte Carlo Tree Search, MCTS)来改进智能体和环境交互及获取样本的方式，从交互速度和数据分布方面进行改进，以此提升模型效果。本章还提出使用集成框架，通过集成 Q 网络来改进 Actor-Critic 类型模型的评估过程，进一步缓解 RL 中的过高估计问题，稳定训练过程，提升训练效率。对比试验结果表明，本章提出的方法能够在同等训练资源下实现更加稳定和优异的效果。

4.1 强化学习的训练效率问题

RL 的训练方式为：智能体和不断和环境交互来获取样本，从获取到的样本中学习经验和知识，修正自身行为模式，并继续和环境进行更好地交互。这种独特的训练机制致使其可能会出现如下问题：训练样本稀缺；训练不稳定。

产生上述情况的根本原因是环境所具有的复杂易变的动态特性，也即环境的复杂的状态转移分布。智能体学习的过程就是不断调整自身策略分布，以适应环境的状态转移分布，使得在当前分布下能够最大化累计奖励。然而在现实情况中，环境往往复杂易变，需要模型能够及时根据环境状态转移分布的转变来修正自身过去学习到的经验，否则可能导致训练失败。具体而言，训练失效一方面源自样本稀缺，因为频繁的状态分布变化意味着智能体即使已经通过交互并搜集大量的数据而学习到了一些正确经验，也可能因为环境的微笑改变导致先前学到的经验不再适用，从而导致某种意义上的样本稀缺。另一方面也源于训练的不稳定性，模型探索新的样本所具有的新的奖励会增加训练的方差。

适应环境动态特性也是 RL 中面临的利用和探索的权衡问题，这实际上是 RL 所面临的核心问题。一方面，智能体需要在学习尚未充分情况下保持探索力度，不断和环境交互获取新的样本；另一方面，智能体也需要有效利用现有的样本和自身学习到的经验来做出高质量动作，以获得更多的奖励。探索和利用如若不能有效平衡，

就会影响训练效率和效果。如果探索力度过大，模型会耗费更多时间和环境交互，在降低训练效率的同时也难以获取到具有高奖励值的高质量样本。而如果忽略探索，过分看重利用，则可能导致模型过拟合，训练过程不稳定。

量化投资任务尤其容易出现上述两个问题。主要原因在于：金融市场环境复杂多变，需要智能体不断适应新的市场环境，因而样本利用效率较低。一般解决此类方法的手段为增大 Update-To-Data(UTD) ratio，即模型训练总次数和与环境交互次数的比例。在交互次数不变，即样本总量不变的前提下，简单增加模型训练总次数能够加快训练速度，缓解样本利用效率较低的问题。然而，这会导致模型训练极不稳定，甚至效果下降。为了说明这一点，参考文献[27]的研究，本文给出了模型训练总次数 *n_updates* 的变化对训练稳定性和效果的影响(交互次数不变，都保持在 10 幕)。

图 4.1 和 4.2 给出了 DDPG 模型在测试集上的表现随 *n_updates* 变化的改变。*n_updates* 取值为 1、2、4、8，表示智能体每和环境交互一次，即抽取 *n_updates* 次数的小批量样本执行训练，*n_updates* 越大表示训练程度越大。两图展示了在不同 *n_updates* 参数下各训练 5 次模型后，于测试集上交易所得到的资产均值以及 3 倍残差阴影图，阴影部分越大表示模型越不稳定。由图 4.1 可以看到，随着 *n_updates* 的增大，平均资产曲线水平逐渐下降，说明模型训练效果逐渐变差，而阴影部分却逐渐变大，说明训练不稳定性逐渐上升，且 *n_updates* 为 8 时，红色阴影的覆盖面积最大。图 4.2 也呈现出同样的趋势。这说明，在交易任务和投资组合管理任务中，简单增大 *n_updates* 极容易导致训练不稳定，继而导致模型效果下滑。

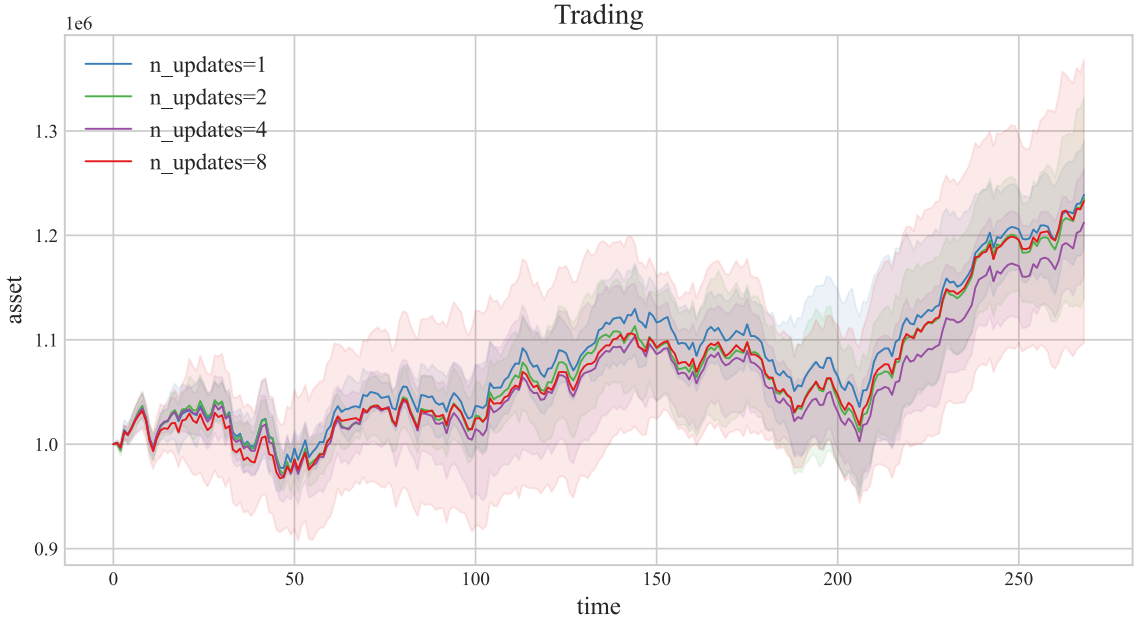


图 4.1 DDPG 模型在交易任务中的表现受 *n_updates* 变化的影响

Fig. 4.1 How *n_updates* changes affect DDPG model performance in trading tasks

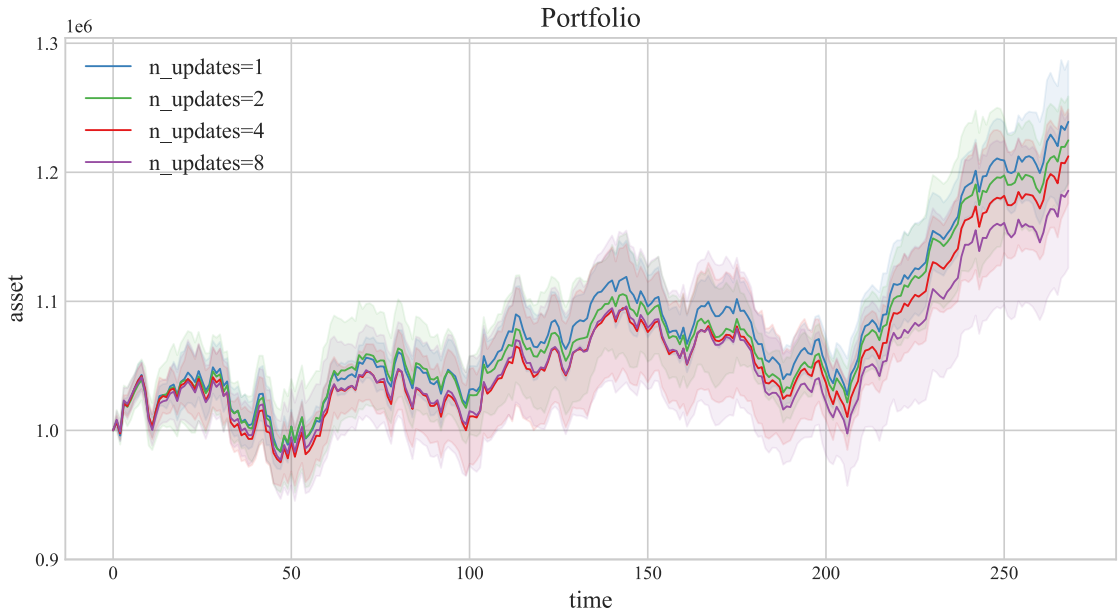


图 4.2 DDPG 模型在投资组合管理任务中的表现受 *n_updates* 变化的影响

Fig. 4.2 How *n_updates* changes affect DDPG model performance in portfolio tasks

图 4.3 更直观给出了测试集上夏普率指标的均值(左侧)和标准差(右侧)受 *n_updates* 变化的影响。可以看到，随着 *n_updates* 的增大，标准差呈现整体上升的趋势，说明模型训练不稳定性逐渐提升，均值呈现下降趋势，说明模型效果也随之变差。上述结果总的表明，单纯通过增大 *n_updates* 的方法并不是有效的提高样本训练效率的方法，容易使得模型训练不稳定，甚至导致效果变差。下文即针对上述问题进行针对性改进。

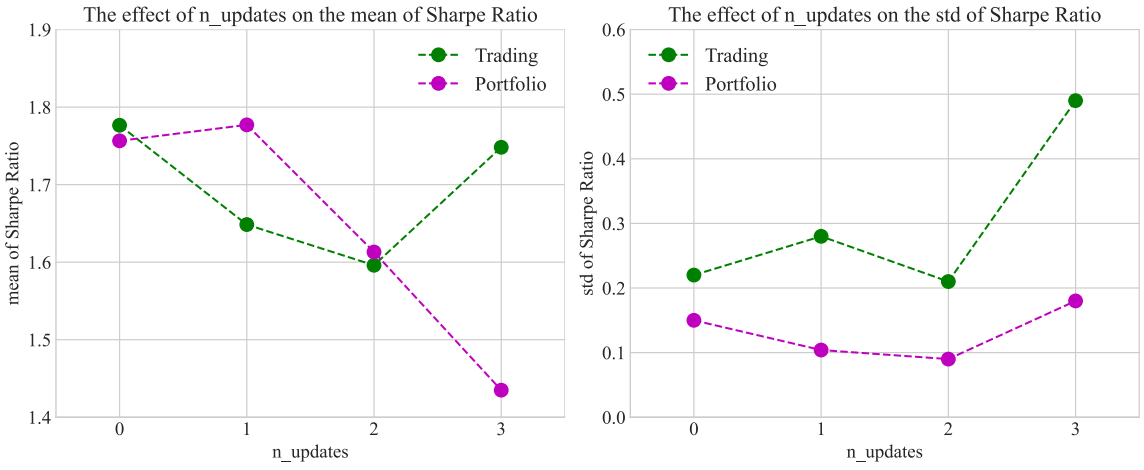


图 4.3 夏普率均值和标准差受 *n_updates* 改变的影响

Fig. 4.3 How *n_updates* changes affect Sharpe Ratio mean and standard deviation

4.2 基于集成框架和蒙特卡洛树搜索的改进策略

本节将提出 MMD4 模型具体改进方案，提出利用 REDQ 集成框架和 MCTS，针对性缓解上述样本利用效率低和训练不稳定问题。具体而言，本章主要创新点如下：

1. 提出使用 REDQ 集成框架缓解过高估计问题，降低 MMD4 训练不稳定问题

将 MMD4 的 Critic MLP 部分和 Critic Net 以及 CNN-LSTM 图片处理模块视作 Critic 部分，按照 REDQ 集成框架集成多个 Critic 计算 TD Target，缓解过高估计问题，提升模型鲁棒性和训练稳定性。

2. 提出利用 MCTS 改进智能体与环境交互方式，增加样本利用效率

将环境交互过程融入到 MCTS 的 Select、Expand、Rollout、Backprop 四阶段的模拟过程中。将原始完整的交互改为断点存续式交互，利用蒙特卡洛树结构平衡智能体的探索和利用过程。通过 MCTS 的 Select 过程选择更有前景、当前更具价值探索的路径，来实现获取更符合当前训练程度的探索或利用方式，在减少不必要的环境交互的同时，获取更高质量的合适的训练样本。

4.2.1 基于 REDQ 框架的 Q 学习方法

随机集成双 Q 学习(REDQ, Random Ensemble Double Q-Learning)是一种 Model-Free 类模型训练的框架，其提出是为了解决 Model-Free 模型所具有的低样本效率和不稳定性问题。REDQ 创造性地使用了集成和 In-Target Minimization 两项技术，同时从理论上分析了集成 Q 网络的个数和每次参与计算的 Q 网络的个数与最终效果之间的关系，能够有效控制模型的训练过程，继而提升最终效果。

本章提出使用 REDQ 框架来训练 MMD4 模型。具体而言，本章将集成 N 个 Q 网络(即 Critic 部分)同时用于计算训练 Q 网络时所使用到的 TD Target。由于计算出来的 TD Target 标签值采取了最小化操作，故能够有效缓解传统方法使用单个 Q 网络时容易出现的过高估计问题。同时，本章并非简单地按照传统集成方法使用所有的 Q 网络来参与计算，而是结合 In-Target Minimization 技术，即在每次计算 TD Target 时，随机选择 Q 网络子集来进行计算。这样做一方面降低了训练复杂度，另一方面也考虑了不同组合网络的估计，避免模型拟合到特定网络的噪声或误差，从而增强了模型训练稳定性。此外，引入 REDQ 框架能够在保证训练稳定前提下增大 UTD 值，从而提高样本利用效率。

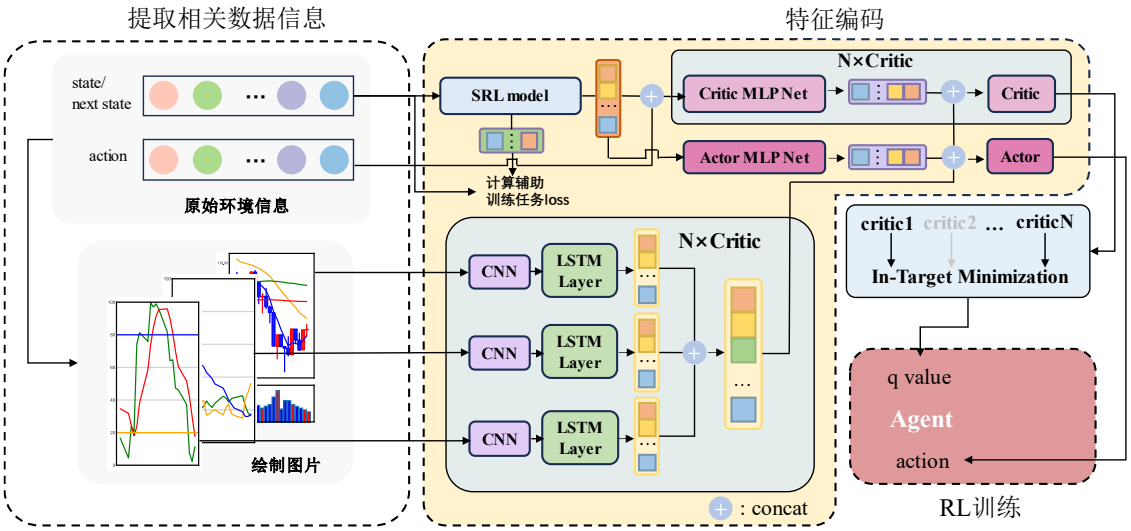


图 4.4 引入 REDQ 集成框架后的模型整体结构图

Fig. 4.4 The overall structure of the model with REDQ

图 4.4 展示了引入 REDQ 之后的 MMD4 模型。图中青色部分表示集成的 N 个 Critic 模型，计算 TD Target 时将从中选取 M 个 Critic 来执行 In-Target Minimization(同样用青色标识，黑色的 Critic 表示被选中，灰色表示没有被选中)，得到的 Q 值标签用于训练所有的 N 个 Critic。

本章还会通过实验结果来从经验上分析与 REDQ 框架相关的多种集成方式的效果的异同，这些集成方式包括：REDQ、Maxmin、Weighted、Average。以下是对四种集成方式的公式介绍。

算法 4.1 REDQ 集成方式计算 TD Target

1: **for** $i = 1, \ldots, n\_updates$ **do**

2: 抽取样本：从 D 中随机采样 $minbatch$ 数量的样本 $\left(s_j, a_j, r_{j+1}, s_{j+1}\right)$

3: 抽取索引：从索引集 $\{1, 2, \ldots, N\}$ 中随机抽取具有 M 个索引的子集 Ω

4: 根据 N 个 Critic，计算 $TD\ Target$：$y_j = r_j + \gamma \min_{i \in \Omega} Q\left(s_{j+1}, u(s_{j+1}; \theta^{u^-}); \theta_i^{Q^-}\right)$

5: 依次训练 N 个 Critic：

根据目标函数 $\left(y_j - Q\left(s_j, a_j; \theta_i^Q\right)\right)^2$ 执行梯度下降，$i = 1, 2, \ldots, N$

6: 训练 Actor：根据目标函数 $-\frac{1}{N}\sum_i Q\left(s_j, u\left(s_j; \theta^u\right); \theta_i^Q\right)$ 执行梯度下降

7: 更新目标网络参数：$\theta_i^{Q^-} \leftarrow \tau\theta_i^Q + (1-\tau)\theta_i^{Q^-}$，$i = 1, 2, \ldots, N$，$\theta^{u^-} \leftarrow \tau\theta^u + (1-\tau)\theta^{u^-}$

8: **end for**

9: **end for**

算法 4.1 给出了 REDQ 集成方式完整的一轮训练过程以及计算 TD Target 的方

法。智能体每和环境交互一次后，即执行一轮训练，包含 *n_updates* 次小批量梯度下降。可以看到，REDQ 的关键特点有二，其一为随机从 N 个集成的 Q 函数中抽取 M 个用于计算 TD Target，取 M 个动作价值的最小者；其二是利用 TD Target 更新全部的 Q 函数。两者结合，即为 In-Target Minimization 技术。REDQ 的稳定特性使得参数 *n_updates* 能够取较大的值，从而加速训练。

表 4.1 给出了其余三种集成方式计算 TD Target 的方式，其余的训练过程和 REDQ 一致。可以看到，Maxmin 取所有 Q 值的最小值来计算 TD Target，但是在训练时会随机抽取部分 Q 网络进行更新。Weighted 是 REDQ 的期望版本，即取组合数 C_N^M，每一种情况对应一个索引子集，计算全部索引子集对应的 REDQ 情形的 TD Target 均值。Average 即为取所有 Q 值的均值。

表 4.1 Maxmin, Weighted, Average 三种集成方法计算公式

Table 4.1 The calculation formulas for Maxmin, Weighted, and Average ensemble methods

集成方法	TD Target 计算公式
Maxmin	$y_j = r_j + \gamma \min_i Q\left(s_{j+1}, u(s_{j+1};\theta^{u^-});\theta_i^{Q^-}\right)$
Weighted	$y_j = r_j + \gamma \frac{1}{\binom{N}{M}} \sum_{\Omega} \min_{i\in\Omega} Q\left(s_{j+1}, u(s_{j+1};\theta^{u^-});\theta_i^{Q^-}\right)$
Average	$y_j = r_j + \gamma \frac{1}{N} \sum_i Q\left(s_{j+1}, u(s_{j+1};\theta^{u^-});\theta_i^{Q^-}\right)$

4.2.2 基于蒙特卡洛树搜索的环境交互方法

MCTS 是一类常用的决策算法，特别适合用于没有明确启发式经验的场景。在深度学习广泛繁荣之前，MCTS 就已经在各类棋类游戏中得到应用，应用于树的上置信界(Upper Confidence bounds applied to Trees, UCT)算法[47]的提出为 MCTS 有效平衡树类探索和利用提供了强大工具。2016 年，DeepMind[48]将 MCTS 和神经网络结合，提出计算机围棋程序 AlphaGo。AlphaGo 使用 CNN 来评估棋局状态，MCTS 为自对弈(Self-Play)提供搜索方案，大大减少评估状态所需要的搜索的宽度。MCTS 在 LLM 中也有应用，文献[49]基于 MCTS 构造名为 Cooperative Inference 的 LLM 推理过程，通过基于 MCTS 的多次模拟来生成多条解答，并从中选取质量最高者作为最终答案。MCTS 在各个领域已经得到广泛应用。

本节提出使用 MCTS 来改进 MMD4 模型，主要改进智能体和环境交互的方式，提升样本利用效率和模型训练效果。传统的智能体和环境交互方式为完整交互，即

智能体从环境初始状态开始和环境交互，直到环境到达终止状态。本节提出使用断点存续交互方式，即利用 MCTS 来指导智能体交互，以这样的方式来和环境执行一段完整的交互过程，称为 MCTS 模拟。MCTS 过程会在多次交互中建立蒙特卡洛树，在每一次 MCTS 模拟中，智能体根据当前蒙特卡洛树来选择环境起始状态，从该状态和环境进行交互，而非完整经历一幕过程。在交互和训练过程中蒙特卡洛树会被逐渐更新和完善(扩展和回溯阶段)，逐渐适应当前训练程度的需要，来平衡对经验的探索和利用，从而提升训练效果和速度。

具体而言，本节提出的 MCTS 方法包含四个步骤：选择(Select)、扩展(Expand)、预演(Rollout)、回溯(Backprop)。图4.5给出了MCTS的完整流程和简单介绍。MCTS的执行基于蒙特卡洛树，图中绿色、红色、蓝色分别表示树结点。灰色表示 Rollout 过程中形成的轨迹状态，并非树的结点。以下是对图中四个步骤的详细介绍。

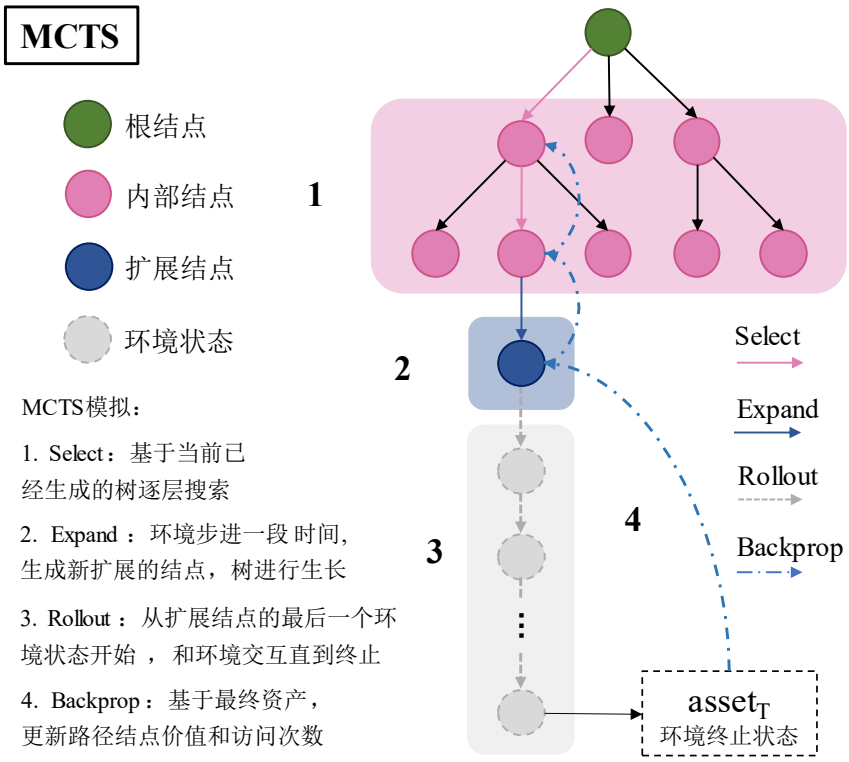


图 4.5 蒙特卡洛树搜索过程

Fig. 4.5 Monte Carlo Tree Search procedure

1. Select

MCTS 模拟第一个步骤为 Select，它指从当前已生成的蒙特卡洛树的根结点(即绿色结点)出发，逐层向下搜索，直到搜索到某一个红色叶结点为止。Select 阶段会返回一个叶结点，该叶结点也是下一阶段的开始。根结点记录环境起始状态，每次模拟都从根结点出发。

本章采用适应轨迹的UCT算法来执行 Select。UCT 算法改进自 Upper Confidence Bounds(UCB)算法，而 UCB 算法一般用于求解多臂赌博机问题，即在每个手臂背后的奖励分布未知情况下，如何分配动作来获得累计最大奖励。UCB 算法采用置信上界思想来平衡探索和利用，即综合每个手臂的选择次数和已经获得的平均奖励来评价该手臂，每次都选择置信上界最大的手臂，随着选择次数越来越多，对手臂的真实价值(奖励分布期望)的估计越来越准确。而 UCT 实际上是 UCB 应用于树搜索问题中的应用，它将孩子结点视作动作，从而能平衡树的探索和利用，适用于围棋等复杂策略游戏。

具体而言，UCT 策略如公式 4.1 所示：

$$UCT(s) = \arg\max_{a} \frac{Q(s,a)}{n_a} + C \cdot \sqrt{\frac{\ln N}{n_a}} \tag{4.1}$$

其中 s 表示树中某个父结点代表的状态；$UCT(s)$表示在 s 状态下 UCT 策略选择的动作(某个子结点)；$Q(s, a)$表示状态 s 下动作 a 的价值；n_a 表示动作 a 被选择的次数；C 为平衡探索和利用的参数；N 为父节点被选择总次数。

本章提出的适应轨迹的 UCT 策略需要解决两个问题：如何表示树中结点；如何计算结点价值 Q。由于量化投资任务一幕时刻较长，将树结点表示成一个时刻环境状态会导致需要模拟很多次才可能构造出有意义的树结构，故本章将树结点表示成多个时刻的交互状态，结点记录下这段交互轨迹最后一个时刻的状态，用于断点恢复。对于第二个问题，本章采用 Rollout 结束后的最终时刻资产作为奖励来回溯更新结点价值。原因在于只有途径的轨迹智能体决策发挥出色，才有可能获得较高的资产，从而资产总额可以评价轨迹中的结点质量。从而，适应轨迹的 UCT 策略公式如 4.2 所示。*parent* 和 *node* 分别表示父结点和待选子结点。

$$UCT^{*}(parent) = \arg\max_{node} \frac{Q(parent, node)}{n_{node}} + C \cdot \sqrt{\frac{\ln N}{n_{node}}} \tag{4.2}$$

可以看到，适应轨迹的 UCT 策略保留了 UCT 的平衡探索和利用的特点：结点质量越高，被选择的可能性越大，从而体现利用；被选择次数越多，下次被选择概率则降低，从而体现探索。此外，针对性地根据量化投资任务的特点修改 UCT 策略，能够使其动态调整创建结点时的交互次数，更具灵活性。

2. Expand

Expand 是模拟的第二个步骤，它指从 Select 阶段返回的叶结点记录的最后时刻环境状态出发，继续和环境交互一段时间，形成一段轨迹及一个新的叶结点。扩展能够为树增加一个新的结点，而 Select 阶段涉及的所有结点，均来自以往的 Expand

阶段。

3. Rollout

Rollout 是模拟的第三个步骤，它指从 Expand 阶段扩展的结点所记录的最后时刻环境状态出发，继续和环境交互到终止状态的过程。图 4.5 中的灰色圆圈即表示这段轨迹中每一时刻的环境状态，它们不构成树的一部分，仅仅是通过蒙特卡洛模拟来获取数据。

4. Backprop

Backprop 是模拟的最后一个步骤，是指根据 Rollout 得到的结果，反向回溯 Select 和 Expand 所经历的所有结点并逐次更新结点信息。根据公式 4.1，结点需要维持价值和被选择次数。价值更新方式为：根据终止时刻资产，计算反向传播到结点时的折扣值。即 $Q(parent, node) = Q(parent, node) + asset_T \times \gamma^i$，其中 i 为结点在反向路径中的位置，需要对资产进行缩放。此外，结点访问次数 n_{node} 需要加一。

MCTS 实际上在每次模拟中改变了智能体获取训练数据的方式，这是通过蒙特卡洛树实现的。树的 Select 策略根据过往交互经验平衡了探索和利用，在 Rollout 过程前，Select 和 Expand 阶段会根据过往经验和树结构来实现更灵活的交互，即断点存续交互。智能体会在选择阶段的最后时刻环境状态上开始 Expand 和 Rollout，由此开始和环境交互获取新的交互数据，而跳过了 Select 部分的时间段，其优势在于能够充分利用过往学习到的知识，选择当前训练程度下，能够综合探索和利用的最佳的探索方向。实际上，本章提出的方法利用选择机制避免了原始交互方式过多考虑环境已知的、对当前训练不重要的信息的弊端，而将精力重点放在还未探索的、更有助于训练的部分。从而，MCTS 有效加速了训练过程，且由于改变了和环境的交互方式而带来训练数据分布的变化，故也能够提升训练效果，提升训练稳定性。

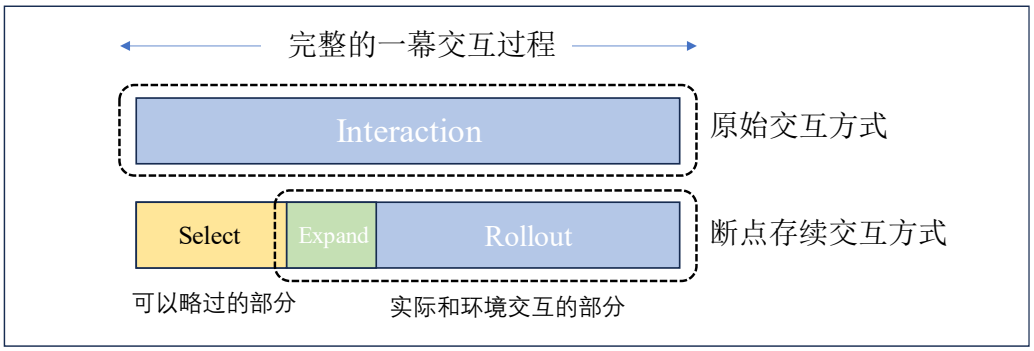


图 4.6 原始交互方式与断点存续交互方式对比

Fig. 4.6 Comparison between original interaction mode and breakpoint continuity interaction mode

图 4.6 展示了上述内容。图中展示了原始的交互方式和断点存续交互方式的区

别。前者为完整的一幕交互过程，后者在 Select 阶段可以利用树结构快速掠过一段交互，而从 Expand 和 Rollout 阶段开始交互，从而改变训练集分布，加快交互速度。

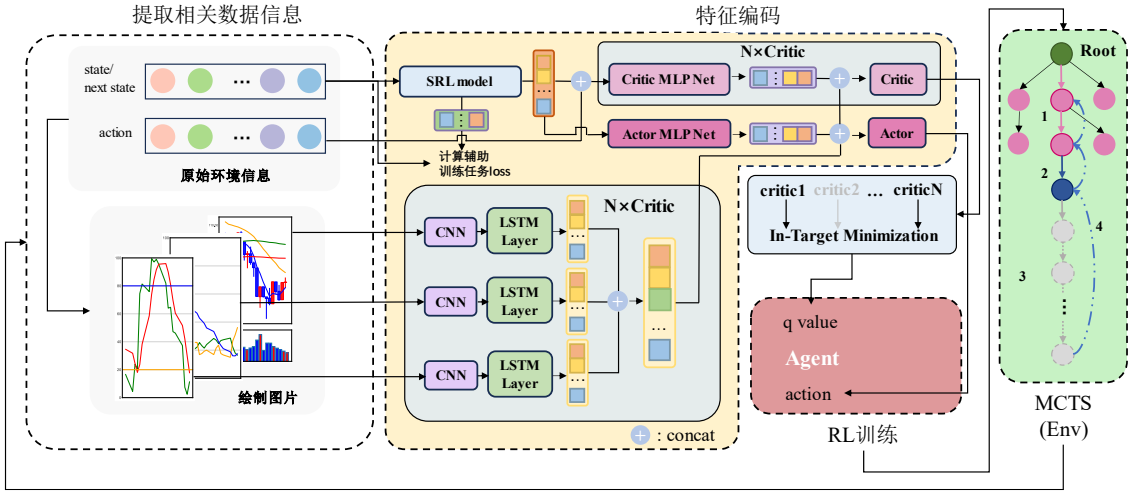


图 4.7 结合蒙特卡洛树搜索的模型整体结构图

Fig. 4.7 The overall structure of the model with MCTS

本文将本章提出的模型命名为 Data-Efficient MMD4，因本章提出的方法重在利用 REDQ 和 MCTS 技术来提升样本效率，继而提高模型效果。本章将模型简记为 MMD4$^+$，以此表示对 MMD4 在样本效率上的改进。图 4.7 给出了 MMD4$^+$的整体结构图，在引入 MCTS 之后，模型在一轮训练之后会利用蒙特卡洛树结构来执行环境交互过程，获取到样本后继续执行下一轮训练。

最后给出 MMD4$^+$模型的算法流程，见算法 4.1。整个流程即为 MCTS 模拟的不断循环的过程，环境交互和模型训练都蕴含在其中。算法 4.1 主要陈列了 REDQ 和 MCTS 的算法流程，第三章的多模态处理部分进行了简化。

算法 4.1 MMD4$^+$

1: 初始化： *policy* 网络，参数为 θ，包含 Actor 和集成 Critic 两部分。集成 Critic 部分含有 N 个 Critic。将 Actor 和 Critic 模型分别记为 *actor*, *critic*。M 表示 In-Target Minimization 参数。目标网络 *policy*$^-$，结构和 *policy* 相同。创建 Replay Buffer 为 D。初始化随机过程 n 为噪声，i = 0。

2: **while** *i* < *total_updates* **do**

3: 　　Select：从根结点出发，根据 UCT 策略，搜索到当前树的一个叶结点作为父结点，记为 *parent*。*parent* 结点记录的环境状态记为 s_s

4: 　　Env 断点存续：将环境恢复至 s_s，$s_t = s_s$

5: 　　Expand：智能体和环境交互 *expand_length* 次数。执行如下循环过程：

6: 　　**for** $k = 1, \ldots,$ *expand_length* **do**

7: Agent 执行动作：$a_t = actor(s_t) + n_t$

8: 环境步进：环境接收动作 a_t，返回 r_{t+1}、s_{t+1}、环境终止状态 *done*, $t = t+1$

9: 存储样本：将 $transition = (s_t, a_t, r_{t+1}, s_{t+1})$ 存入 D 中

10: **for** $k = 1, \dots, n\_updates$ **do**

11: 抽取样本：从 D 中随机采样 *minbatch* 数量的样本 $(s_j, a_j, r_{j+1}, s_{j+1})$，获取对应图片 $figures_j$

12: 表征状态：将原始环境状态表征为状态特征 $z^{s_j} = srl_{net}(s_j)$

13: 辅助训练 SRL model：根据损失 $\left\| srl_{f\_pred}(z^{s_j}) - s_j \right\|_2^2$ 执行梯度下降

14: 合并特征：按照算法 3.1 流程，得到多模态特征：

$feature_{actor,j}$，$feature_{critic,j}$，$feature^{pre}_{critic,j}$

15: 抽取索引：从索引集 $\{1, 2, \dots, N\}$ 中随机抽取具有 M 个索引的子集 Ω

16: 计算 *TD Target*：$y_j = r_j + \gamma \min_{i \in \Omega} critic^{-(i)}_{net}\left(feature^{pre}_{critic,j+1}\right)$

17: 训练 N 个 Critic：

根据目标函数 $\left(y_j - critic^{(i)}_{net}\left(feature_{critic,j}\right)\right)^2$ 执行梯度下降，$i = 1, 2, \dots, N$

18: 训练 Actor：根据目标函数 $-\frac{1}{N}\sum_i critic^{(i)}_{net}\left(feature^{pre}_{critic,j}\right)$ 执行梯度下降

19: 更新目标网络参数：$\theta^- \leftarrow \tau\theta + (1-\tau)\theta^-$

20: 更新总训练次数：$i = i + n\_updates$

21: **end for**

22: **end for**

23: 创建新结点：记录 Expand 阶段结束时环境最后时刻的状态，据此创建新的叶子结点，作为 *parent* 的孩子结点

24: Rollout：从环境当前状态出发，智能体和环境交互，直到环境终止时刻。期间每和环境交互一次，即执行小批量梯度下降 *n_updates* 次，并拷贝参数。流程同 10~22 行。即执行如下循环过程：

25: **while** not *done* **do**

26: Agent 执行动作：$a_t = actor(s_t) + n_t$

27: 环境步进：环境接收动作 a_t，返回 r_{t+1}、s_{t+1}、环境终止状态 *done*, $t = t + 1$

28: 存储样本：将 $transition = (s_t, a_t, r_{t+1}, s_{t+1})$ 存入 D 中

29: 执行训练过程：同 10-22 行

30: **end while**

31: Backprop：获得终止时刻资产 $asset_T$，据此更新沿途结点的价值和访问次数，如 4.2.2 中 4 部分所述。

32: **end while**

4.3 实验结果

4.3.1 基准方法

本章用于对比的基准方法和第三章相同，它们包括基于统计的方法和 4 种 DRL 基准方法，这里不再赘述。

4.3.2 实验设置

本章的数据集划分方式和 Python 版本等设置同第三章。表 4.2 给出了 MMD4$^+$模型的超参数设置，未提及的超参数设置同第三章。

表 4.2 MMD4$^+$模型超参数设置

Tabel 4.2 MMD4$^+$ model hyperparameter settings

参数名	参数值	参数含义
total_updates	90000(trading), 138400(portfolio)	训练总次数
batch_size	3(trading), 4(portfolio)	批量大小
lr	5e-8	学习率
buffer_size	1e4	经验池大小
n_updates	3(trading), 5(portfolio)	交互一次训练的次数
gamma	0.99	折扣因子
tau	0.005	目标网络参数软更新参数
N	2(trading), 3(portfolio)	集成 Critic 数目
M	1(trading), 2(portfolio)	计算 TD Target 所用 Critic 数目
expand_length	200(trading), 50(portfolio)	Expand 长度
children_maximum	3(trading), 5(portfolio)	树结点最大孩子数目
select_prob	0.5(trading), 0.2(portfolio)	树内部搜索，决定 Select 的概率

4.3.3 对比试验

本章创新点为集成和 MCTS 两部分，故首先需要探究合适的集成方案。本节测试了 4.2.1 节中列举的 4 种集成方法，旨在找到适合不同任务特性的集成方式。表 4.3 给出了两个任务下，四种集成方法的实验结果对比。需要注意的是 Trading 任务参数设置为：*N*=3，*M*=2；Portfolio 任务的设置为：*N*=4，*M*=2。两组实验均不使用

MCTS 技术，旨在单纯探究不同集成方式的效果。

表 4.3 集成方法结果对比

Table 4.3 Comparison results of ensemble method

任务	集成方法	评价指标			
		Cumulative Returns	Annual Returns	Sharpe Ratio	Max Drawdown
Trading	REDQ	**0.2354**[2]	**0.2190**[2]	**1.77**[1] ± **0.171**[2]	0.0823
	Maxmin	0.1850	0.1722	1.44 ± 0.410	**0.0730**[2]
	Weighted	0.2291	0.2131	**1.73**[2] ± 0.326	**0.0708**[1]
	Average	**0.2491**[1]	**0.2317**[1]	1.64 ± **0.092**[1]	0.1078
Portfolio	REDQ	0.2010	0.1871	1.56 ± 0.158	**0.0839**[2]
	Maxmin	**0.2227**[1]	**0.2072**[2]	1.58 ± **0.112**[1]	0.0953
	Weighted	**0.2070**[2]	**0.2147**[1]	**1.64**[1] ± **0.207**[2]	0.0895
	Average	0.2026	0.1886	**1.60**[2] ± 0.201	**0.0826**[1]

表 4.3 的结果表明，Trading 任务下 REDQ 集成框架表现出色，其 Sharpe Ratio 均值排名第一，且两项收益率也排名前列。此外，REDQ 的 Sharpe Ratio 标准差也取得较好结果，低于第三章 MMD4 的 0.239 的标准差。这说明，REDQ 的 In-Target Minimization 技术能够有效缓解动作价值过高估计问题，从而获得更精准和稳健的动作价值估计，进一步提高模型训练稳定性。因而，Trading 任务选择 REDQ 集成方式进行建模。对于 Portfolio 任务，由于 Weighted 集成方式的指标综合表现最佳，故选择该方法建模。实际上，Weighted 为 REDQ 的期望版本，它消除了抽取 Critic 子集时的随机性，考虑到更全面的集成种类，因而可以实现更好的效果。

在选择完毕集成方式并按照 4.3.2 节中列出的超参数设置进行训练之后，得到测试结果对比如表 4.4 所示，其中 MMD4$^+$为本章改进的模型。首先，本章改进模型在盈利效果方面进一步提升，这一点从 Trading 任务中模型 Cumulative Return 和 Annual Return 指标的大幅提升可见一斑，相应的 Sharpe Ratio 也显著提高。此外，在 Portfolio 任务上，收益率指标也略微有所提升。其次，模型在训练稳定性方面也具有明显提升，这从两个任务的 Sharpe Ratio 标准差的降幅中可得见。综合来看，MMD4$^+$对模型训练稳定性方面有所改进，这主要得益于集成框架的使用。此外，模型也能够进一步提升盈利效果，这主要是由于通过断点存续交互方式获取到更有

价值的训练样本，从而能够提升训练效果。

表 4.4 全部模型结果对比

Table 4.4 Comparison results of all models

任务	模型	评价指标			
		Cumulative Returns	Annual Returns	Sharpe Ratio	Max Drawdown
Trading	MMD4+(ours)	**0.2824**[2]	**0.2623**[2]	**1.87**[1] ± **0.141**[2]	0.101
	MMD4(ours)	0.2304	0.2144	**1.77**[2] ± 0.239	0.084
	DDPG	0.2238	0.2082	1.67 ± 0.323	**0.083**[2]
	TD3	0.2459	0.2287	1.74 ± **0.128**[1]	0.092
	SAC	0.2410	0.2242	1.71 ± 0.282	0.094
	PPO	0.2478	0.2305	1.80 ± 0.276	0.092
	Long Only*	0.1831	0.1706	1.48	**0.082**[1]
	MACD*	**0.3339**[1]	**0.3098**[1]	1.72	0.119
	Immediate*	0.1993	0.1856	1.24	0.116
Portfolio	MMD4+(ours)	**0.2422**[1]	**0.2252**[1]	**1.82**[1] ± 0.235	**0.079**[2]
	MMD4(ours)	**0.2366**[2]	**0.2201**[2]	**1.81**[2] ± 0.243	0.084
	DDPG	0.2236	0.2081	1.77 ± **0.049**[1]	0.082
	TD3	0.2293	0.2134	1.67 ± 0.168	0.085
	SAC	0.2112	0.1966	1.62 ± 0.107	0.082
	PPO	0.1962	0.1827	1.54 ± **0.060**[2]	0.083
	UBAH*	0.1907	0.1776	1.49	0.088
	UCRP*	0.1807	0.1684	1.43	0.087
	Best CRP*	0.1029	0.0961	0.62	0.082
	Best Stock*	-0.0144	-0.0135	0.04	0.146
	UP*	0.1823	0.1699	1.45	**0.075**[1]

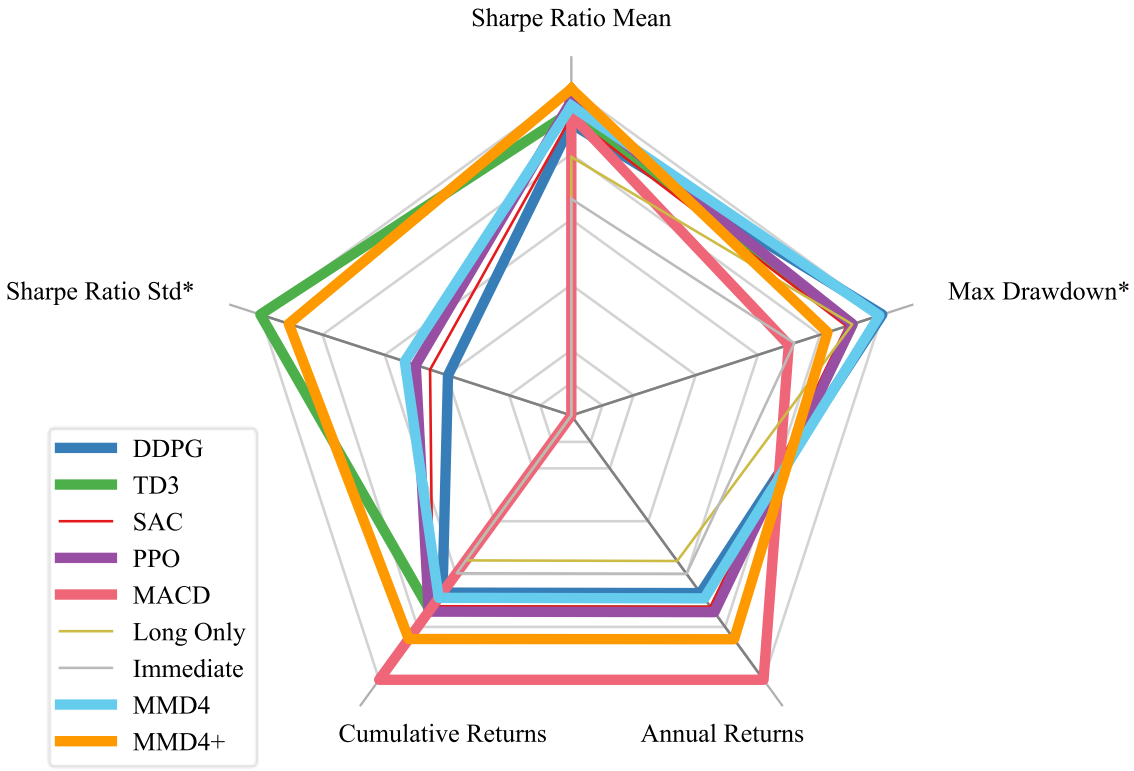


图 4.8 模型在 Trading 任务中的各项指标雷达图

Fig. 4.8 Radar chart of various indicators for the models in the Trading task

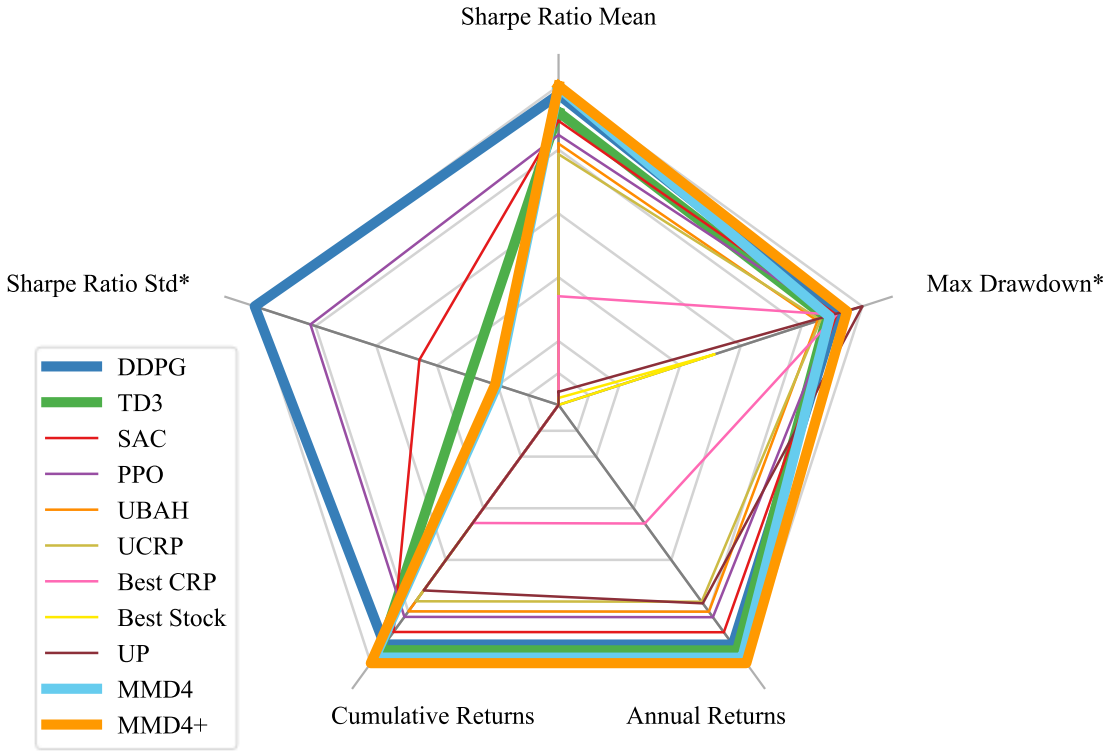


图 4.9 模型在 Portfolio 任务中的各项指标雷达图

Fig. 4.9 Radar chart of various indicators for the models in the Portfolio task

类似第三章，为了直观比较模型之间各项指标的变化，图 4.8 和 4.9 绘制了两个任务下的模型指标的雷达图。其中浅蓝色和橙色分别表示 MMD4 和 MMD4⁺模型。在 Trading 任务中，除 Max Drawdown 指标外，MMD4⁺模型在其他所有指标上均优于 MMD4 模型，尤其是在 Sharpe Ratio 标准差上得到了显著改善。在 Portfolio 任务的雷达图中，除 Sharpe Ratio 标准差外，MMD4⁺模型的其他指标均得到了提升，其整体表现超越其他所有模型，且在标准差上相较 MMD4 也具有一定改进。上述分析结果总的表明，MMD4⁺的改进带来了更优的性能效果。

本节进一步验证了 MMD4⁺模型针对样本利用效率的提升。样本利用效率是指模型利用有限样本执行训练的能力，即如果模型能利用更少的样本来实现相近的效果，则说明其样本利用效率更高。由于 DRL 训练数据来源于智能体和环境交互，因此交互次数越少，获取的样本量就越少。在本文设置中，智能体每进行一次环境交互，即执行 *n_updates* 次数的小批量样本训练，在总训练次数固定的情况下(例如，Trading 任务共计训练 90000 次)， *n_updates* 增大意味着和环境交互的次数减少，从而获取的样本量也更少。因此，为了展示 MMD4⁺模型在样本利用效率上的提升，本章探讨了 *n_updates* 参数设置对训练效果的影响，通过逐步增大 *n_updates* 来减少智能体和环境总交互次数，并观察测试结果的变化。

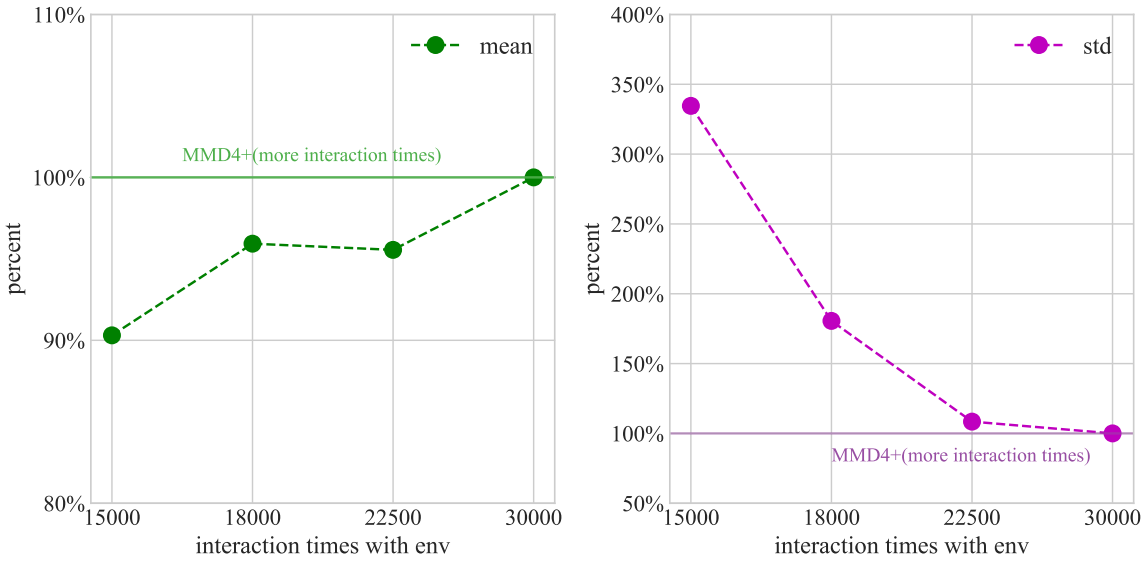


图 4.10 增大 *n_updates*(减少环境交互次数)对模型效果的影响

Fig. 4.10 How model performance changes with an increase in *n_updates* (resulting in a decrease in the number of interactions with the environment)

图 4.10 绘制了在 Trading 任务中，随着 *n_updates* 变化，MMD4⁺模型的测试结果如何变动。左图和右图分别呈现 Sharpe Ratio 的均值和标准差的变化，图中横线

代表 *n_updates* 原始设置为 3 时模型的测试结果，此时作为基准结果进行对比。图中绘制的是 *n_updates* 设置为 4、5、6 时，测试结果相较于基准结果的百分比变化情况，这些设置分别对应环境交互次数为 22500、18000、15000。

可以看到，随着 *n_updates* 的增大，即交互次数的降低，Sharpe Ratio 均值并未出现明显下降，当获取的训练样本量减半时，模型依然保持有原有结果的 90%。此外，在样本量下降近 1/3，即交互次数为 22500 时，Sharpe Ratio 标准差也并未出现显著上升。这些结果表明，MMD4$^+$模型在有效提高样本利用效率的同时，仍能保持模型训练的稳定性。然而，当 *n_updates* 较大时，模型标准差依然位于较大水平，这一现象是由前文叙述的 DRL 的固有特性决定的。即便如此，MMD4$^+$在一定程度内依旧证明了本章提出方法的有效性。

4.3.4 实例测试

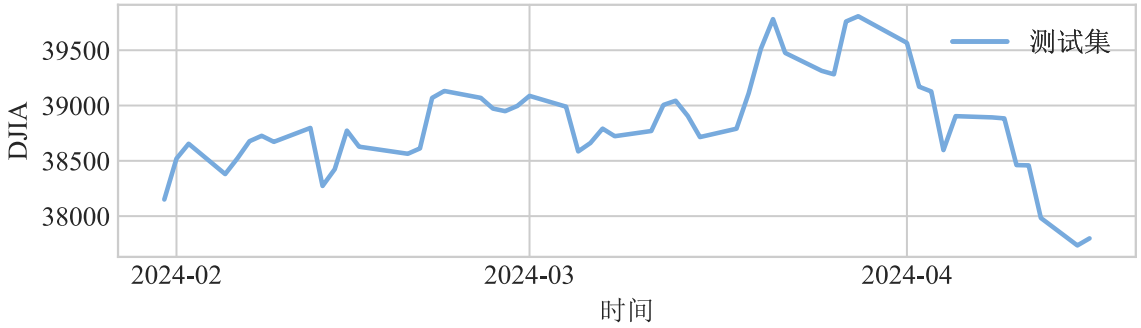


图 4.11 新测试集的 DJIA 曲线

Fig. 4.11 DJIA curve for the new test set

为了说明模型的泛化性与普适性，本节利用最新时段的股票数据来测试模型的交易表现。测试时间段为数据集以外的最新时段：2024 年 1 月 31 日至 2024 年 4 月 16 日。绘制此时间段的 DJIA 曲线如图 4.11 所示，可以看到市场表现并不理想，DJIA 并未呈现上升态势。虽然大盘状况可能影响模型表现，使其难以展现出优异的盈利能力，但仍可以比较模型指标的相对表现来评估模型性能。

测试结果如表 4.5 所示，其显示出 MMD4$^+$模型在新时段仍具有较强的综合表现。在 Trading 任务中，MMD4$^+$模型在三项关键指标上均表现最佳，盈利能力大幅超越 MMD4，这充分证明了本章提出的方法在提升模型盈利水平方面的显著效果。在 Portfolio 任务中，MMD4$^+$模型的表现在全部模型中仍位居前列，且在 DRL 模型中仅次于 TD3 模型。综合两个任务的表现可以认为，本章提出的模型具有良好适应能力。

表 4.5 全部模型在新测试时期结果对比

Table 4.5 Comparison of results for all models during the new testing period

任务	模型	评价指标			
		Cumulative Returns	Annual Returns	Sharpe Ratio	Max Drawdown
Trading	MMD4⁺(ours)	**-0.0022**[1]	**-0.0075**[1]	**-0.107**[1] ± 0.850	**0.048**[2]
	MMD4(ours)	-0.0360	-0.1620	-1.65 ± **0.425**[2]	0.061
	DDPG	-0.0216	-0.0965	-0.96 ± 0.967	0.056
	TD3	-0.0092	-0.0423	-0.34 ± 0.553	0.051
	SAC	**-0.0034**[2]	**-0.0160**[2]	**-0.12**[2] ± **0.376**[1]	**0.045**[1]
	PPO	-0.0219	-0.0979	-0.90 ± 0.886	0.056
	Long Only*	-0.0119	-0.0564	-0.52	0.055
	MACD*	-0.0113	-0.0536	-0.45	0.060
	Immediate*	-0.0147	-0.0693	-0.62	0.053
Portfolio	MMD4⁺(ours)	-0.0148	-0.0689	-0.69 ± **0.470**[2]	0.055
	MMD4(ours)	-0.0137	-0.0614	-0.65 ± 0.977	0.055
	DDPG	-0.0241	-0.1091	-1.15 ± 0.839	0.061
	TD3	-0.0105	-0.0487	-0.42 ± 0.522	0.057
	SAC	-0.0183	-0.0837	-0.85 ± 0.725	0.059
	PPO	-0.0121	-0.0575	-0.57 ± **0.049**[1]	0.055
	UBAH*	**-0.0088**[2]	**-0.0419**[2]	**-0.40**[2]	**0.054**[2]
	UCRP*	-0.0108	-0.0513	-0.50	0.055
	Best CRP*	-0.0402	-0.1803	-1.35	0.073
	Best Stock*	-0.1274	-0.4843	-2.93	0.164
	UP*	**-0.0057**[1]	**-0.0273**[1]	**-0.25**[1]	**0.052**[1]

图 4.12 以直观的方式展现了模型综合性能的相对排名。图 4.12 的左图和右图分别对应 Trading 和 Portfolio 任务，图中 x 和 y 轴线分别表示两个模型能力维度，即基于 Cumulative Return 和 Annual Return 计算出的盈利能力维度，以及基于 Sharpe Ratio 计算出的盈利-抗风险综合能力维度。模型越靠近右上角，说明其综合能力越强。为

方便评价，图中标出基于统计的方法和 DRL 模型的平均结果，以两者为界限，将模型划分为强模型、中间模型、弱模型，并以不同深度颜色加以区分。特别的，MMD4⁺模型用红色显著标出。

从图中可以观察到，DRL 类模型整体表现优于基于统计的方法，其中 MMD4⁺模型在两个任务中均处于强模型范畴，且在 Trading 任务中表现最佳。而作为 MMD4⁺模型基础架构的 DDPG 模型在两个任务中反而表现较差，在 Trading 任务中甚至排名垫底，这进一步体现出 MMD4 和 MMD4⁺模型改进的有效性。同时，基于统计的方法在 Portfolio 任务中也表现出与 DRL 模型相媲美的竞争力，其中 UP 和 UBAH 方法表现出色。此外，图中还用蓝色线标识了 DJIA 的收益表现，可以看到，MMD4⁺模型均能够跑赢大盘。而 DJIA 在 Trading 任务中明显处于较低水准，说明相较于 Portfolio 任务而言，模型在 Trading 任务中发挥更出色，能够以较大幅度超越大盘水平。

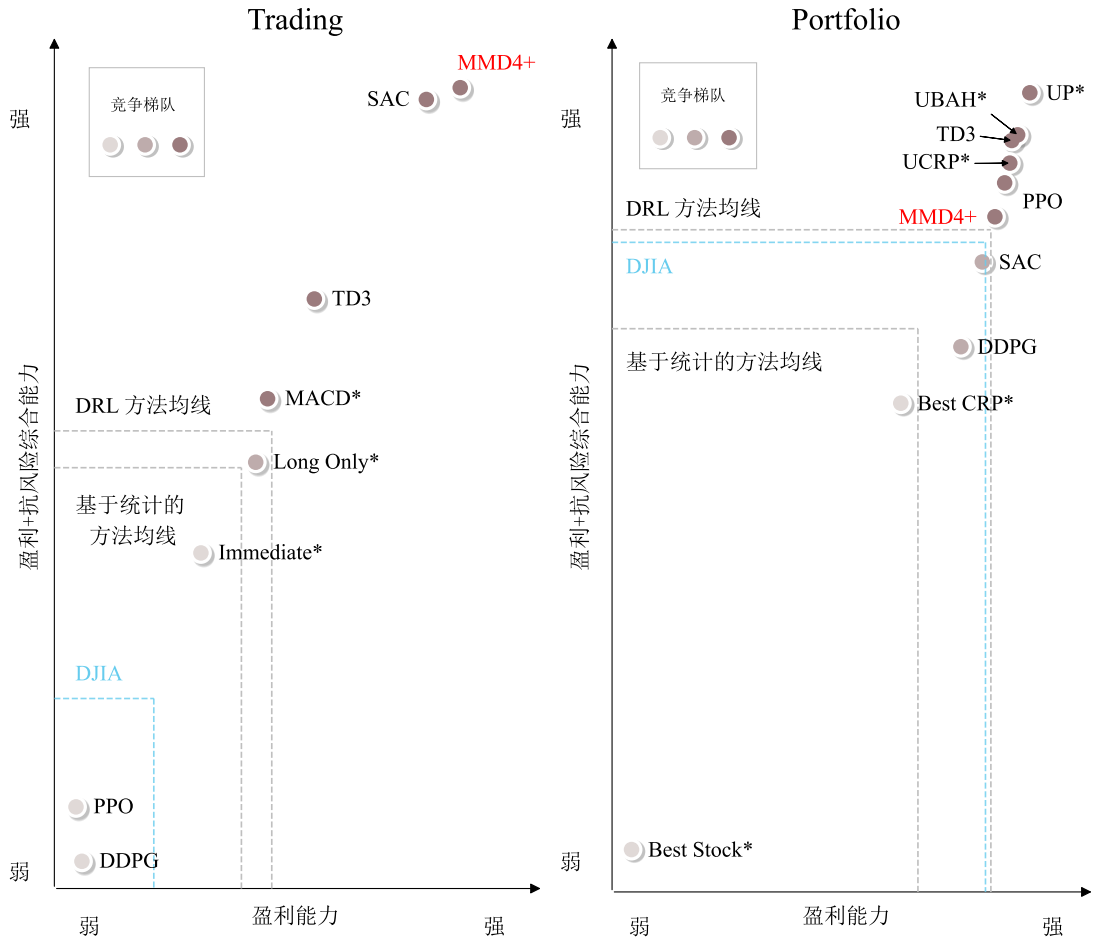


图 4.12 模型在新测试时段的不同维度的能力表现

Fig. 4.12 Performance of the model in different dimensions of competence during the new test period

4.3.5 消融实验

最后，本节进行了针对 MMD4$^+$模型的消融实验，表 4.6 展示了实验结果。首先，模型在去掉集成和 MCTS 任意一项后，效果均不如原始模型。其次，去掉 MCTS 后的模型的 Sharpe Ratio 水平明显下降，去掉集成后的模型的 Sharpe Ratio 的标准差则明显上升，这说明 MCTS 能够通过提升样本质量来改进模型效果，而集成则重点稳定模型训练过程。上述结果表明，集成与 MCTS 两者结合，能够发挥最好的效果。

表 4.6 消融实验结果

Table 4.6 Ablation experiments results

任务	模型	评价指标	
		Sharpe Ratio Mean	Sharpe Ratio Std
Trading	MMD4$^+$	1.87	0.141
	MMD4$^+$-no-Ensemble	1.64	0.400
	MMD4$^+$-no-MCTS	1.37	0.380
Portfolio	MMD4+	1.82	0.235
	MMD4$^+$-no-Ensemble	1.74	0.138
	MMD4$^+$-no-MCTS	1.57	0.145

4.4 本章小结

本章在第三章基础上进一步改进了 MMD4，重点缓解训练不稳定和样本利用效率低的问题。首先，通过实验分析确认了量化投资中 DRL 训练存在的不稳定问题，指出 DRL 训练需要适应环境动态特性的变化，合理平衡探索和利用，以提高样本利用效率。根据上述分析，本章提出一种数据高效的改进策略模型 MMD4$^+$，其利用集成框架来缓解过高估计问题从而稳定训练；并通过 MCTS 优化和环境的交互方式，采用断点存续交互方式和树结构去探索更有价值的路径，提高样本质量和利用效率。测试结果表明，MMD4$^+$在盈利水平和训练稳定性方面较 MMD4 有进一步改善，在两项任务上均取得了最优 Sharpe Ratio 取值和较低的标准差。此外，*n_updates* 的实验表明，MMD4$^+$能够在减少和环境交互的情况下保持 90%以上的 Sharpe Ratio 水平。消融实验进一步印证了 MCTS 和集成分别在提升盈利和稳定训练方面的关键性作用。同时，新的交易时段的测试结果也印证了 MMD4$^+$的适用性和泛化能力。

第 5 章 总结与展望

5.1 总结

本文主要将 DRL 技术应用于量化投资领域中的交易和投资组合管理任务，构建了以多模态数据输入、稳定训练、高效数据利用为特点的投资策略模型 MMD4 和 MMD4^{+}。研究中主要采用了强化学习和深度学习相关理论、算法和模型，具体包括策略梯度算法、卷积神经网络、循环神经网络等，同时也运用了表征学习、集成以及 MCTS 等现金技术。在对 DRL 和量化投资进行了相关调研和分析之后，本文不仅梳理了关键理论和实际应用成果，同时也明确指出了现有研究的不足，例如环境状态表征的不充分、数据利用效率的低下、训练过程中的不稳定性等。本文在前人工作基础上，采用了针对性的技术进行改进，并在道琼斯工业平均指数市场的股票上进行实验验证。测试结果取得了良好的效果，验证了本文方法的有效性。

本文首先调研了 DRL 的发展及其在量化投资领域内的应用，指出了 DRL 在量化投资中的发展前景以及存在的问题。DRL 自 2015 年 DQN 发布后进入快速发展阶段， DQN 系列模型和 Actor-Critic 类模型相继提出。深度学习和强化学习理论的结合使得 DRL 具有处理高维复杂数据的能力，因而其在量化投资领域内的应用也具备可通用和扩展的潜力。同时由于 DRL 的训练范式适合投资者追求长期稳定收益的目标，故在许多量化投资任务中表现出色。然而，DRL 依旧存在环境状态表征不充分问题，且金融市场复杂多变的特点对数据的高效利用和模型稳定的训练提出了挑战，这些问题在过去并未得到良好解决。

本文据此提出构建多模态策略来提升对环境状态的表达感知能力，提出了结合表征学习的多模态模型策略 MMD4。MMD4 模型利用图片和基于指标的向量这两种模态的数据共同作为环境状态表达，在能够准确表示当下环境状态的同时，也能直观高效的表示长期市场的变化规律。MMD4 模型利用 CNN-LSTM 模块处理图片，并使用表征学习模型 D2RL 帮助处理状态信息。本文对 MMD4 模型进行了详细实验分析，确定以综合表现最好的 DDPG 模型作为 DRL 的算法主体框架，选择了 D2RL 这种从输入输出连接的疏密性上最适合本文任务的表征学习模型。在对比实验中，MMD4 模型在 Sharpe Ratio 指标上表现出色，具有较为优秀的盈利水平和抗风险能力，且能够跑赢大盘水平。然而，模型多次训练结果的波动性较大，显现出训练不稳定等问题。

随后，为了提高样本利用效率和模型稳定性，本文对 MMD4 模型进行了改进，提出了数据高效的模型 MMD4⁺。MMD4⁺利用集成 Q 学习框架来降低训练不稳定性，并通过多个 Critic 函数计算更准确和稳健的动作价值。此外，MMD4⁺通过 MCTS 方法改进了智能体和环境的交互方式，提出了一种新的断点存续的交互方法。这种方法利用树结构自适应地选择探索路径，减少了不必要的交互，提高了样本质量和利用效率。MMD4⁺模型在交易任务和投资组合管理任务上分别选择了 REDQ 和 Weighted 集成方式，并取得了比 MMD4 模型更优的效果。具体表现在，模型取得了 Sharpe Ratio 指标的最好的效果，且降低了标准差，显示出了模型训练过程中稳定性的增强。此外，模型能够利用更少的环境交互次数和更少的样本，取得依旧具有竞争力的表现。在获取的样本量减少一半的情况下，模型的 Sharpe Ratio 仅损耗在 10% 以内，说明模型能够有效提升样本利用效率。

5.2 展望

DRL 作为量化投资领域新兴的应用技术，其应用依然处于起步探索的阶段，未来仍需要进行大量研究工作。本文所提出的 MMD4 及 MMD4⁺模型旨在从两个方面对该领域的工作做出一些贡献：环境状态表达、数据的高效利用。这些工作取得了一定效果，但依然具有可以继续深入研究的价值和空间，如：

1. 添加文本形式的市场情绪信息

本文新增加了图片模态数据，用以表达市场动态变化规律和长期时序演变规律。此方式主要以图像曲线的变化趋势来体现，所呈现的规律反映了多种因素的综合影响，涉及的信息来源复杂多样。然而，金融市场的波动也易受政府、企业和民众舆情的影响，这些通常以新闻标题、股民评论、论坛讨论等文本形式呈现。未来研究可从舆情角度进一步丰富和细化市场状态表达，增加文字模态数据。通过引入如 Bert、GPT、GLM、LLaMa 等模型处理文本信息，以助理智能体做出更精准的决策。

2. 进一步改进环境建模，包括更稳健的奖励定义、精细化的向量状态中指标的选取和预处理、引入更多现实市场环境的限制等

本文奖励定义较为简单和直观，即连续两次交易后持有资产的差值(进行了缩放)，这种定义可能受到通货膨胀影响，使得模型在早期训练时获得的平均奖励过小，而在接近当前时期则获得的奖励平均过大。为避免这一问题，可以探究更合适的奖励定义方式，避免直接使用资金的绝对值作为奖励。例如，本文建议探究使用 Sharpe Ratio 作为奖励，因其利用风险进行了规则化。但需要注意，更换奖励意味着

智能体追寻达成的目标发生改变，从而改变构建策略的基准。此外，本文选择常见的技术及金融指标作为向量状态，未来可以根据不同市场的特征进行更精细的测试挑选。引入 ICA、PCA、LDA 等降维方法对上述指标进行预处理，可能会使输入更适应模型参数的初始化，获得更稳定的模型训练过程。同时，考虑到现实市场的交易限制，未来还需要完善虚拟环境交易规则。

3. 丰富模型验证和对比的方式，包括在更多市场上验证效果以及加入人类专家投资者的对比基线

本文已在道琼斯工业平均指数股票市场上训练并验证模型，未来可以在更多市场验证模型的效果，如纳斯达克市场或中国股市，以证明模型通用性。此外，若条件允许，可以在真实的市场环境中与人类专家投资者进行对比，以增加模型的实际应用价值。

参考文献

[1] Mnih V, Kavukcuoglu K, Silver D, et al. Human-level control through deep reinforcement learning[J]. Nature, 2015, 518(7540): 529-533.

[2] Haarnoja T, Zhou A, Hartikainen K, et al. Soft Actor-Critic Algorithms and Applications[J]. arXiv preprint arXiv:1812.05905, 2018.

[3] Lillicrap T P, Hunt J J, Pritzel A, et al. CONTINUOUS CONTROL WITH DEEP REINFORCEMENT LEARNING[J]. arXiv preprint arXiv:1509.02971, 2015.

[4] Fujimoto S, Hoof H, Meger D. Addressing Function Approximation Error in Actor-Critic Methods[C]//International Conference on Machine Learning. 2018, 80: 1587-1596.

[5] Schulman J, Wolski F, Dhariwal P, et al. Proximal Policy Optimization Algorithms[J]. arXiv preprint arXiv:1707.06347, 2017.

[6] Ouyang L, Wu J, Jiang X, et al. Training language models to follow instructions with human feedback[J]. Advances in Neural Information Processing Systems, 2022, 35: 27730-27744.

[7] WATKINS C. Learning from Delayed Rewards[D]. Cambridge: University of Cambridge, 1989.

[8] Sutton R S, Barto A G. Reinforcement Learning: An Introduction[M]. Cambridge: MIT Press, 2018.

[9] Sutton R S, McAllester D, Singh S, et al. Policy Gradient Methods for Reinforcement Learning with Function Approximation[J]. Advances in Neural Information Processing Systems, 1999, 12: 1058-1063.

[10] Williams R J. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning[J]. Machine Learning, 1992, 8: 229-256.

[11] Mnih V, Kavukcuoglu K, Silver D, et al. Playing Atari with Deep Reinforcement Learning[J]. arXiv preprint arXiv:1312.5602, 2013.

[12] Hasselt H. Double Q-learning[J]. Advances in Neural Information Processing Systems, 2010, 23: 2613-2621.

[13] Van Hasselt H, Guez A, Silver D. Deep Reinforcement Learning with Double Q-

Learning[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2016, 30(1): 2094-2100.

[14] Schaul T, Quan J, Antonoglou I, et al. PRIORITIZED EXPERIENCE REPLAY[J]. arXiv preprint arXiv:1511.05952, 2015.

[15] Wang Z, Schaul T, Hessel M, et al. Dueling Network Architectures for Deep Reinforcement Learning[C]//International Conference on Machine Learning. 2016, 48: 1995-2003.

[16] Bellemare M G, Dabney W, Munos R. A Distributional Perspective on Reinforcement Learning[C]//International Conference on Machine Learning. 2017, 70: 449-458.

[17] Hessel M, Modayil J, Van Hasselt H, et al. Rainbow: Combining Improvements in Deep Reinforcement Learning[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2018, 32(1): 3215-3222.

[18] Silver D, Lever G, Heess N, et al. Deterministic Policy Gradient Algorithms[C]// International Conference on Machine Learning. 2014, 32(1): 387-395.

[19] Degris T, White M, Sutton R S. Off-Policy Actor-Critic[J]. arXiv preprint arXiv:1205.4839, 2012.

[20] Haarnoja T, Tang H, Abbeel P, et al. Reinforcement Learning with Deep Energy-Based Policies[C]//International Conference on Machine Learning. 2017, 70: 1352-1361.

[21] Schulman J, Levine S, Abbeel P, et al. Trust Region Policy Optimization[C]//International Conference on Machine Learning. 2015, 37: 1889-1897.

[22] Munk J, Kober J, Babuška R. Learning State Representation for Deep Actor-Critic Control[C]//Conference on Decision and Control. 2016: 4667-4673.

[23] Ota K, Oiki T, Jha D, et al. Can Increasing Input Dimensionality Improve Deep Reinforcement Learning?[C]//International Conference on Machine Learning. 2020, 119: 7424-7433.

[24] Ota K, Jha D K, Kanezaki A. Training Larger Networks for Deep Reinforcement Learning[J]. arXiv preprint arXiv:2102.07920, 2021.

[25] Sinha S, Bharadhwaj H, Srinivas A, et al. D2RL: DEEP DENSE ARCHITECTURES IN REINFORCEMENT LEARNIN[J]. arXiv preprint arXiv:2010.09163, 2020.

[26] Lan Q, Pan Y, Fyshe A, et al. MAXMIN Q-LEARNING: CONTROLLING THE ESTIMATION BIAS OF Q-LEARNING[J]. arXiv preprint arXiv:2002.06487, 2020.

[27] Chen X, Wang C, Zhou Z, et al. RANDOMIZED ENSEMBLED DOUBLE Q-LEARNING: LEARNING FAST WITHOUT A MODEL[J]. arXiv preprint arXiv:2101.05982, 2021.

[28] Deng Y, Bao F, Kong Y, et al. Deep Direct Reinforcement Learning for Financial Signal Representation and Trading[J]. IEEE Transactions on Neural Networks and Learning Systems. 2016, 28(3): 653-664.

[29] Jiang Z, Xu D, Liang J. A Deep Reinforcement Learning Framework for the Financial Portfolio Management Problem[J]. arXiv preprint arXiv:1706.10059, 2017.

[30] Shin H G, Ra I, Choi Y H. A Deep Multimodal Reinforcement Learning System Combined with CNN and LSTM for Stock Trading[C]//Information and Communication Technology Convergence. 2019: 7-11.

[31] 许杰, 祝玉坤, 邢春晓. 基于深度强化学习的金融交易算法研究[J]. 计算机工程与应用, 2022, 58(07): 276-285.

[32] Wang Z, Huang B, Tu S, et al. DeepTrader: A Deep Reinforcement Learning Approach for Risk-Return Balanced Portfolio Management with Market Conditions Embedding[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2021, 35(1): 643-650.

[33] Wang J, Zhang Y, Tang K, et al. AlphaStock: A Buying-Winners-and-Selling-Losers Investment Strategy using Interpretable Deep Reinforcement Attention Networks [C]//Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. 2019: 1900-1908.

[34] Millea A. Deep Reinforcement Learning for Trading—A Critical Survey[J]. Data, 2021, 6(11): 119.

[35] Fischer T G. Reinforcement learning in financial markets - a survey[R]. FAU Discussion Papers in Economics, 2018.

[36] Pricope T V. Deep Reinforcement Learning in Quantitative Algorithmic Trading: A Review[J]. arXiv preprint arXiv:2106.00123, 2021.

[37] Sun S, Wang R, An B. Reinforcement Learning for Quantitative Trading[J]. ACM Transactions on Intelligent Systems and Technology, 2023, 14(3): 1-29.

[38] Liu X Y, Yang H, Chen Q, et al. FinRL: A Deep Reinforcement Learning Library for Automated Stock Trading in Quantitative Finance[J]. arXiv preprint arXiv:2011.09607,

2020.

[39] Yang H, Liu X Y, Zhong S, et al. Deep Reinforcement Learning for Automated Stock Trading: An Ensemble Strategy[C]//Proceedings of the First ACM International Conference on AI in Finance. 2020: 1-8.

[40] Zhang Z, Zohren S, Roberts S. Deep Reinforcement Learning for Trading[J]. arXiv preprint arXiv:1911.10107, 2019.

[41] Yuan Y, Wen W, Yang J. Using Data Augmentation Based Reinforcement Learning for Daily Stock Trading[J]. Electronics, 2020, 9(9): 1384.

[42] Yu P, Lee J S, Kulyatin I, et al. Model-based Deep Reinforcement Learning for Dynamic Portfolio Optimization[J]. arXiv preprint arXiv:1901.08740, 2019.

[43] Lee J, Kim R, Yi S W, et al. MAPS: Multi-agent Reinforcement Learning-based Portfolio Management System[J]. arXiv preprint arXiv:2007.05402, 2020.

[44] Huang Z, Tanaka F. MSPM: A modularized and scalable multi-agent reinforcement learning-based system for financial portfolio management[J]. PlOS ONE, 2022, 17(2): e0263689.

[45] Wood K, Roberts S, Zohren S. Slow Momentum with Fast Reversion: A Trading Strategy Using Deep Learning and Changepoint Detection[J]. arXiv preprint arXiv:2105.13727, 2021.

[46] Lim B, Zohren S, Roberts S. Enhancing Time Series Momentum Strategies Using Deep Neural Networks[J]. arXiv preprint arXiv:1904.04912, 2019.

[47] Kocsis L, Szepesvári C. Bandit Based Monte-Carlo Planning[C]//European Conference on Machine Learning. 2006, 4212: 282-293.

[48] Silver D, Huang A, Maddison C J, et al. Mastering the Game of Go with Deep Neural Networks and Tree Search[J]. Nature, 2016, 529(7587): 484-489.

[49] Zhu X, Wang J, Zhang L, et al. Solving Math Word Problems via Cooperative Reasoning induced Language Models[J]. arXiv preprint arXiv:2210.16257, 2022.

致 谢

感谢最终有幸抵达这个地方，能够稍微停下来喘口气，歇歇脚，回过头检查来时走过的路。断断续续写了近四个月，即使单论这段动笔的日子也不能否认是段不短的时间，更何况从一开始它即代表着三年或可说是七年的难以过滤的岁月沉淀。尽管往事如在眼前，但身上的一些变化告诉我，七年对一个人来说，仍是相当漫长的一段时光，长到若事先知晓，或许会畏惧止步。没有人能一生下来就具有翻山越岭的能力，我能有幸领略这一路的风景，并抵达自己人生的一个小小地标，离不开很多爱我的人和我爱的人的帮助。爱带给我许多鼓励、安慰或批评，大多数时候也许只是默默的关注，但都足够支撑自己内心的疲惫、不安和虚空，恰如水中的粗石细沙之光样，将偶有出现在我眼前漂浮不散的一团团黑暗给照亮和驱散。

感恩我的父母，你们永远是我最爱也最值得依赖的人。许多时候，当我变得忧郁焦虑，我常常会想和你们聊一通电话。亲近的人，熟悉的话语总能快速抚平我的情绪，你们是我无助彷徨时的底气和力量，也是我的安全感和对未来的希望。

感谢我的研究生指导老师邵新慧教授，您是我科研路上的光辉榜样。对于做学问所应秉持的踏实务实、严谨追求的学习作风与灵活思辨的精神态度，您总能用亲切随和，亦庄亦谐的妙语赠我以警示和启迪。“灰心流泪你的果，创造光明你的因”。感恩薪火中您点亮的我身上的小小光芒，感谢留在我心史中的您温暖的眼光。

感谢我实习期间组内的 Leader 和同事们。你们给了我对职场最初的美好印象。记得有时我们在深夜沟通工作，交流对职业发展道路的看法，那让我经历了许多认真思考人生的重要时刻。你们的专注和进取心使我看到了自己未来的无限可能，向上的理想，登峰造极的努力，对技术的信仰，一步步占据、证实和鼓励着我对未来的想象。

此外，还要感谢周末常一起打羽毛球的同学们。具有相同爱好的人聚在一起不会有太大压力，我们一起追逐着简单的快乐，健康优美的身心。愿我们永远记得那些欢声笑语。感谢陈奕迅和他唱的歌曲。我把他们也视作我生命的一部分，太多美好的情感和情到深处的感悟来自于对他们的欣赏。每首歌曲都是送给我的情书，让我对生活的每一面都充满爱意和欣喜。

再次通读之后，感慨良多。限于自身水平，虽然文章终究会有纰漏和欠缺，但当数万文字像砖块一样最终笨拙地垒起一个小院时，围着院子走一圈，也足以让人

感叹。每一处思考都是一段学习的时光，每一行笔记和代码，都能引发或美好或痛苦的记忆。实事求是地讲，我为一直以来所坚持下来的探索热情和忠纯志虑而感到自豪，同时也为用力过猛而导致的偏差感到遗憾。人总是希望能尽可能扩展自己的生命，让自身以某一种形式、某一种状态凝固下来，成为另一种存在和延续。我不敢妄想这篇文章能带来多少实质性的影响，在如水的时间里能留下些什么。但我仍然感谢自己花费生命中一段时光书写了这样一篇文章，能让自己问心无愧说一句，我已经尽我所能了。如果有人之后看到我的文章，可以真诚地认为停留在我的文章上的几秒并不是白白浪费，那么通过长长的时间，跨过遥远的空间，我会认为我和另外一时另外一地的人，彼此生命流注，无有阻隔。我会向他含泪致敬，并永远永远地为他而祝福。