证券研究报告

发布时间：2024-03-01

证券研究报告 /金融工程研究报告

强化学习与基于 RRL 的因子合成方法

---机器学习系列之四

报告摘要：

强化学习是人工智能领域的重要研究方向之一，随着 AlphaGo 的成功以及在大模型上的关键应用，强化学习广泛地走进了人们的视野。强化学习是对状态、动作与奖励的建模，机制为基于状态来选择动作并获取奖励，其通过 agent 与环境的交互获取经验，然后利用经验不断优化动作选择的策略，其目标是得到最优策略。从最初的 Markov 决策过程建模到经典的强化学习算法，如 Monte-Carlo 方法、时间差分算法包括 Q-learning 和 SARSA 等 value-based 方法，这些算法推动了强化学习早期的发展。而后 policy-based 方法的出现以及与深度学习的结合，进一步推进了强化学习融入人工智能前沿的进程，从策略梯度算法的提出到 Actor-Critic 架构，其中包括 A2C、A3C、PPO 以及确定性策略中的 DDPG、TD3 等高效的算法。

强化学习在量化领域同样有着广泛的应用，包括组合优化、算法交易以及衍生品对冲等方面。本报告着眼于强化学习在因子合成方面的应用，使用简单高效的循环强化学习 RRL 对基于日内量价序列的 Logsig-Alpha 系列因子进行合成。

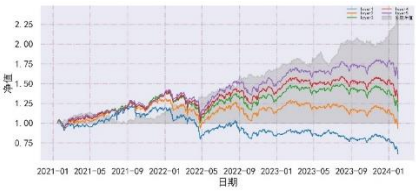
循环强化学习 RRL 是一类直接强化学习算法，它直接对策略进行建模，通过一个可微的目标函数优化整个模型，并且可以建模连续的动作与状态，相比于其他经典算法更加灵活。这里的 RRL 模型包含两个模块，信息融合模块以及策略网络模块，前者使用 RNN 融合当前信息与历史信息作为状态，后者基于状态选择动作。

在基于 RRL 的因子合成方法中，每个时间点的信息选择为待合成因子的多个 Rank IC 以及 ICIR 指标，该时间点的状态定义为当前信息与历史信息的融合；基于状态选择的动作为因子的权重向量；目标函数基于合成因子的整体表现以及与基准的偏差进行设计。计算目标函数，并利用梯度上升算法优化参数。

从测试结果来看，基于 RRL 的合成因子表现显著优于细分因子。另外，相比于传统的因子合成方法，如等权、IC 加权、ICIR 加权以及最大化预期 IC，其表现更为优异，合成因子 2021 年 1 月到 2024 年 2 月的月度调仓回测结果为 Rank IC：13.61%，ICIR：1.09，五分组多头年化超额：15.77%，多空 Sharpe Ratio：2.6。RRL 因子合成的优势在于以下几个方面：第一，相比于传统仅考虑单期单因子单指标的方法，基于 RRL 的因子合成方法同时考虑了待合成因子的多个 Rank IC 和 ICIR 指标，且包含历史数据，也就是说其依据指标的时间序列与期限结构来得到权重，信息输入更加全面。第二，其目标函数可以灵活设定，可以根据不同的偏好选择不同的优化目标，具有一定的可扩展性。第三，RRL 因子合成模型架构简单高效，参数量较少，其训练过程无需额外生成数据序列。

风险提示：以上分析基于模型结果和历史测算，存在模型失效风险。

RRL 合成月度因子分层回测结果



相关报告

《基于营业利润计算过程的财务质量研究》

--20240223

《基于 CNN-Transformer 的深度学习模型探究》

--20240220

《上月红利、Beta、价值因子表现较优》

--20240203

《可转债风险模型构建与应用》

--20240129

《雪球产品敲入规模分布估算和市场影响点评》

--20240124

证券分析师：王琦

执业证书编号：S0550521100001

021-61002390 wangqi_5636@nesc.cn

研究助理：贾英

执业证书编号：S0550122060006

13666061675 jiaying@nesc.cn

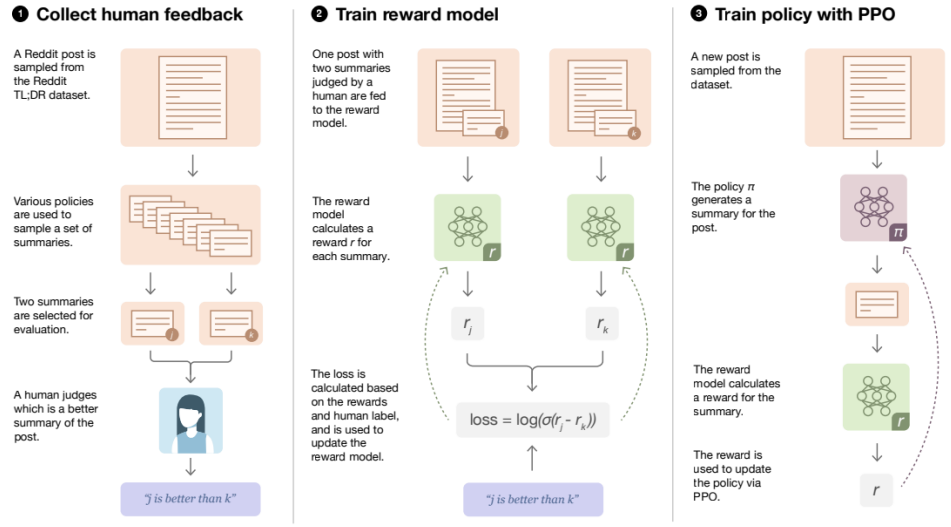
目 录

图表目录

1. 引言

强化学习（Reinforcement learning，RL）是人工智能领域重要的研究方向之一，近年来随着 AlphaGo 的成功广泛地进入了大众视野，另外在大语言模型（Large Language Models，LLMs）中强化学习也起到了关键的作用，例如人类反馈强化学习（Reinforcement learning from human feedback，RLHF）将人类的反馈纳入大模型的训练过程，成为一种新的训练范式。强化学习在量化投资领域中同样有着丰富的应用场景，如算法交易与组合优化等。

图 1：人类反馈强化学习的流程图



数据来源：《Learning to summarize from human feedback》

在机器学习系列报告前几篇中，我们主要聚焦监督学习（Supervised learning）以及生成模型（Generative model）。《机器学习发展历程与量化投资的展望——机器学习系列之一》全面地介绍了机器学习的基本概念与理论，以及其各个分支的发展历程；《基于 cVAE 的数据增强对下行风险预测的提升——机器学习系列之二》介绍了一类生成模型——条件变分自动编码器（cVAE），学习指数风险指标的联合分布用于数据增强，辅助提升下行风险预测模型的性能；《基于 Logsig-RNN 的高频数据低频化选股因子——机器学习系列之三》利用 Log-signature 变换融合日内高频信息，将其转化为日度特征，再利用循环神经网络（Recurrent neural network，RNN）强大的时间序列处理能力生成选股因子。本篇报告则从强化学习的角度入手，探索一类直接强化学习（Direct reinforcement learning）方法，即循环强化学习（Recurrent reinforcement learning，RRL）在因子合成方面的应用，其相比于传统的合成方法考虑了更丰富的信息，线性加权的方式也让模型的可解释性更强，同时可扩展的信息输入和目标函数（Objective function）也让模型更具有灵活性。

本报告的内容安排如下：

第 2 章全面介绍强化学习的基本概念和经典算法，总结算法的发展历程，并且引入循环强化学习模型；第 3 章首先扩展前期报告中的深度学习因子，然后介绍基于 RRL 的因子合成方法，包括模型设定与训练方式，并且将生成的结果与多种传统的因子合成方法进行对比。第 4 章总结。

2. 强化学习理论与应用简介

强化学习主要研究代理(Agent)如何在所处环境(Environment)中采取行动(Action)使得能够获得最大的奖励(Reward)。强化学习可以模仿人类学习或决策的方式，基于现有的经验在环境中进行交互，其侧重于在交互中进行以目标为导向的学习。

图 2：强化学习示意图

数据来源：东北证券

通常来说，代理需要在某种状态（State）下基于策略（Policy）来采取行动，在获得奖励的同时也进入了下一个状态，代理需要在新的状态下继续采取行动，如此往复至结束。在因子合成的场景中，市场作为投资者的交互环境，因子当前或历史的表现作为状态，因子对应的权重作为行动，合成因子下一期的表现作为奖励，策略即为如何基于当前信息或历史信息选择最优的合成因子权重。

2.1. 从 Markov 决策过程到强化学习

Markov 决策过程（Markov decision process，MDP）是强化学习的数学基础。作为一个离散时间的随机控制过程（Stochastic control process），它对于状态的转移以及决策的制定进行建模。Markov 决策过程是对 Markov 链（Markov chain）的扩展，不同之处是它引入了动作和奖励的机制。

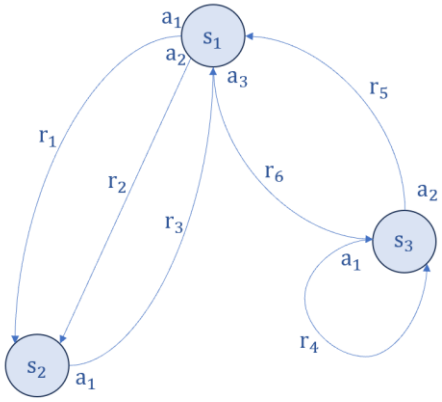
Markov 决策过程是一个四元组$(\mathcal{S},\mathcal{A},\mathcal{P},\mathcal{R})$。其中$\mathcal{S}$表示所有状态的集合，称作状态空间（State space）。$\mathcal{A}$表示所有动作的集合，称作动作空间（Action space）。$\mathcal{P}$表示状态转移概率，具体来说，$P_{s'|s,a} := \mathbb{P}\{S_{t+1} = s'|S_t = s, A_t = a\}$表示在$t$时刻$s$状态下选择动作$a$转移到$s'$状态的概率。$\mathcal{R}$表示状态转移后的期望奖励，具体来说，$R_{s,a} := \mathbb{E}[R_{t+1}|S_t = s, A_t = a]$表示在$t$时刻$s$状态下选择动作$a$转移到$s'$状态得到奖励$R_{t+1}$的期望。

策略π表示基于状态选择动作的概率，即$\pi(a|s) = \mathbb{P}\{A_t = a|S_t = s\}$。Markov 决策过程的目标就是选择最优的策略来最大化回报，这里回报定义为$G_t = \sum_{i=0}^{\infty}\gamma^i R_{t+i+1}$，即为未来奖励的折现值。为更好的描述优化目标，状态价值函数（State-value function）和 Q 函数（Q-function）的定义被引入，两者分别表示在某种策略下，状态对应的期望回报以及状态与动作对应的期望回报，定义如下：

$$v_\pi(s) = \mathbb{E}[G_t|S_t = s],$$

$$q_\pi(s,a) = \mathbb{E}[G_t|S_t = s, A_t = a].$$

图 3：Markov 决策过程示意图



数据来源：东北证券

两者分别满足 Bellman 期望方程（Bellman expectation equation）：

$$v_\pi(s) = \mathbb{E}_\pi[R_{t+1} + \gamma v_\pi(S_{t+1})|S_t = s],$$

$$q_\pi(s,a) = \mathbb{E}_\pi[R_{t+1} + \gamma q_\pi(S_{t+1}, A_{t+1})|S_t = s, A_t = a].$$

事实上，MDP 的目标是找到最优的策略π^*使得在该策略下，在任意状态时状态价值函数为最大值，或在任意状态和动作选择时 Q 函数为最大值，即

$$v_{\pi^*}(s) = \max_\pi v_\pi(s), \quad \forall s \in \mathcal{S},$$

$$q_{\pi^*}(s,a) = \max_\pi q_\pi(s,a), \quad \forall s \in \mathcal{S}, a \in \mathcal{A}.$$

据此可以导出 Bellman 最优方程（Bellman optimality equation），即

$$v_{\pi^*}(s) = \max_{a \in \mathcal{A}} \left\{ R_{s,a} + \gamma \sum_{s' \in \mathcal{S}} P_{s'|s,a} v_{\pi^*}(s') \right\},$$

$$q_{\pi^*}(s,a) = R_{s,a} + \gamma \sum_{s' \in \mathcal{S}} P_{s'|s,a} \max_{a' \in \mathcal{A}} q_{\pi^*}(s',a').$$

Bellman 期望方程表示在某个策略下价值函数满足的关系，当 MDP 的动态特性确定时，便可以得到解析解，其表示在某个策略下各状态对应的价值，或是策略对应的 Q-table，用于策略的评估。而 Bellman 最优方程表示在最优策略情况下价值函数满足的关系，其数值解一般可以由动态规划（Dynamical programming，DP）给出，用于得到最优策略。

MDP 是一个参数模型，需要建模并确定参数后才能求解，然而在实际问题中状态转移概率等参数通常难以事先给定，这给模型的应用带来了困难。另外，动态规划的效率受到维数的限制，计算成本随状态数量呈指数级增长。强化学习也使用 MDP 表示环境，但不同的是，它允许无模型（Model-free）环境且更加通用。代理通过与环境的交互获得信息尽可能探索新状态（Exploration），同时基于已有经验学习最优策略（Exploitation），被称为 Exploration vs Exploitation trade-off。

2.2. 算法分类与简介

根据不同的学习目标，强化学习算法可以分为基于价值的（Value-based）算法和基于策略的（Policy-based）算法，也分别称作间接强化学习与直接强化学习。另外还有一类将两者相结合的 Actor-Critic 类算法。

2.2.1. Value-based 算法

基于价值的算法通常依赖于对价值函数的估计，通过迭代间接地得到最优策略，也称作 critic-based 方法。这类算法的每步迭代通常包括策略评估（Policy evaluation）和策略改进（Policy improvement），前者评估当前策略对应的价值函数，后者基于得到的价值函数对原始策略进行改进。

(1) Monte-Carlo 方法

一个自然的想法是用 Monte-Carlo 方法估计状态价值函数或者 Q 函数，即每次迭代中通过模拟估计回报的条件期望，这样便可以实现策略评估，再基于ε-greedy 方法以及价值函数得到新的策略，重复以上迭代直至收敛。这里ε-greedy 方法是一个引入随机性的决策方法，它在大概率选到价值最高的动作时，也有一定概率随机选择，可以帮助探索更多的状态。Monte-Carlo 方法简单直观，但其计算成本太高且效率低下使其难以应用于复杂问题中。

(2) Q-learning

时间差分学习（Temporal-difference learning，TD-learning）[1]通常被用于定义在离散空间的优化问题中，与 Monte-Carlo 方法不同，其只需要一个时间步就可以完成一次策略迭代。一个代表性的算法是 Q-learning [2]，它建立了一个包含所有状态和动作对应 Q-value 的 Q-table。策略是基于 Q-table 的ε-greedy 方法，很大程度上依赖于 Q-table 的构建与更新，其更新方式如下：

$$q(S_t,A_t) \leftarrow q(S_t,A_t) + \alpha\left(R_{t+1} + \gamma \max_{a\in\mathcal{A}} q(S_{t+1},a) - q(S_t,A_t)\right),$$

可以看出，更新方式类似于 Bellman 最优方程，每步将$R_{t+1} + \gamma \max_{a\in\mathcal{A}} q(S_{t+1},a)$作为更新目标，$\alpha$作为学习率。

(3) SARSA

另一个具有代表性的算法为 SARSA [3]，它与 Q-learning 类似，但在更新方式上有一定区别，其更新方式如下：

$$q(S_t,A_t) \leftarrow q(S_t,A_t) + \alpha\big(R_{t+1} + \gamma q(S_{t+1},A_{t+1}) - q(S_t,A_t)\big),$$

在更新时，更新目标为$R_{t+1} + \gamma q(S_{t+1},A_{t+1})$，其中$A_{t+1}$是基于当前策略$\pi$选择的，这种使用同一套策略更新 Q 函数与动作选择的方式称为同轨（On-policy）。而 Q-learning 在更新过程中，在更新目标的动作选择上使用 greedy 方法即选择 Q-value 最大的动作，这跟需要更新的策略不一致，这种方式称作离轨（Off-policy）。Q-learning 相对 SARSA 而言更为激进，更侧重于对环境的探索与交互。

(4) DQN

上述算法均是在离散状态离散动作下定义的，为了解决连续状态下不适用的问题，一类结合神经网络（Neural network）的算法被提出，即为深度强化学习（Deep reinforcement learning），其中一个重要的代表是 Deep Q-Network（DQN）[4]。DQN 同样是 value-based 算法，它使用神经网络去拟合 Q 函数，换句话说，它将 Q-learning 中的 Q-table 替换成了 Q-network 使得状态可以是连续的。另外 DQN 是 off-policy 的算法，它引入了经验回放（Experience replay）技巧，即将与环境交互得到经验(s,a,r,s')先进行保存，然后随机抽取 mini-batch 对 Q-network 进行更新，这样大幅提升了样本的利用率。DQN 在训练时引入了目标网络（Target network）$\hat{q}$使得训练过程更稳定，目标网络不参与梯度下降，其网络参数为滞后的 Q-network 参数值θ^-，每隔固定的步数目标网络就会复制一次 Q-network 的参数 i.e. $\theta^- \leftarrow \theta$。这样在很大程度上可以固定学习目标，更有利于模型的训练，DQN 的损失函数（Loss function）为

$$L(\theta) = \mathbb{E}\left[\left(r + \gamma \max_{a'} \hat{q}(s', a'; \theta^-) - q(s, a; \theta)\right)^2\right].$$

(5) Double DQN

由于 DQN 在每次更新时，Q 值都是以下一个状态s'的最大 Q 值来估计的，但下一个状态的 Q 值同样也是这样估计的，这就会导致估计的 Q 值可能有偏大的情况，并且这个偏差可能不均匀。Double DQN [5]的出现在一定程度上缓解了这一问题，它同样包括 Q-network 与目标网络两部分，目标网络参数设定更新与 DQN 一致。与 DQN 的主要区别在于网络作用与学习目标，Q-network 用于选择动作，目标网络用于计算 Q 值。Double DQN 的损失函数为

$$L(\theta) = \mathbb{E}\left[\left(r + \gamma \hat{q}\left(s', \underset{a'}{argmax}\, q(s', a'; \theta); \theta^-\right) - q(s, a; \theta)\right)^2\right].$$

2.2.2. Policy-based 算法

2.2.2.1. 策略梯度算法

Value-based 算法通过价值函数的估计或拟合间接得到最优策略。Policy-based 算法则直接拟合策略函数，比如通过神经网络$\pi(a|s;\theta)$来表示在状态s下选择动作a的概率，选择目标函数$J(\theta)$来评估策略表现，然后通过策略梯度算法（Policy gradient）优化目标函数。当$J(\theta)$被定义为初始状态的价值函数时，策略梯度定理（Policy gradient theorem）[6]表明

$$\nabla J(\theta) \propto \mathbb{E}_\pi[q(s,a)\nabla \log \pi(a|\mathrm{s};\theta)].$$

REINFORCE [7]使用回报G来近似$q(\mathrm{s},\mathrm{a})$，G可以由 Monte-Carlo 方法得到，所以其损失函数可以写为：

$$L(\theta) = -\mathbb{E}_\pi[G \log \pi(a|\mathrm{s};\theta)].$$

REINFORCE 是 on-policy 算法，在实际训练时，使用策略π生成一个 episode，倒序遍历每一步来计算对应的G，对数概率可以由神经网络给出，这样便可以计算出损失函数来更新网络。

2.2.2.2. Actor-Critic 架构

前文中介绍的 value-based 类方法有一些局限性。首先其不能直接得到动作值输出，难以扩展到连续动作空间上。另外还存在高偏差（High bias）问题，即估算得到的价值函数与真实价值函数之间存在着误差。而 policy-based 类方法同样有着一些局限，如上小节介绍的 REINFORCE 需要对进行大量的采样，但不同 trajectory 之间的差异可能是巨大的，结果就引入了高方差（High variance）和梯度噪声（Noisy gradients），可能影响模型训练的稳定性。本节介绍的 Actor-Critic 架构结合了两类方法的特点，并在一定程度上缓解了两类方法的局限性。

(1) Actor-Critic

经典的 Actor-Critic 算法在用神经网络$\pi(a|s;\theta)$拟合策略的同时，用另一个神经网络$q(s,a;w)$拟合策略梯度定理中的 Q 函数，前者根据状态选择动作，后者评估动作选择的好坏。

根据策略梯度定理，策略网络的损失函数为：

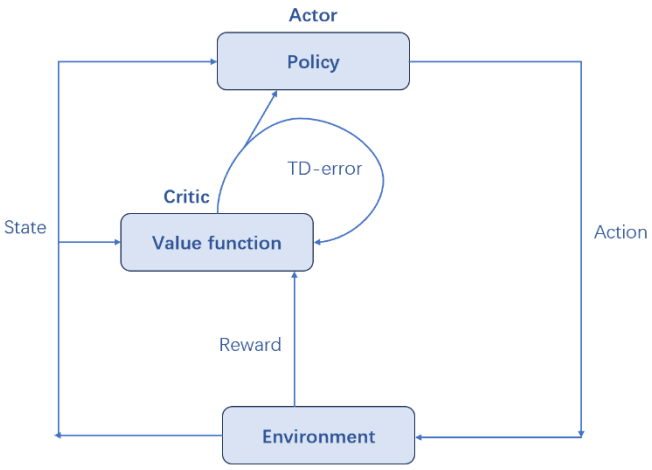
$$L_1(\theta) = -\mathbb{E}_\pi[q(s,a;w)\nabla\log\pi(a|s;\theta)].$$

而评估网络使用 SARSA 算法，其损失函数为：

$$L_2(w) = \mathbb{E}_\pi\left[\left(r + \gamma\hat{q}(s',a';w^-) - q(s,a;w)\right)^2\right].$$

这里与 DQN 类似引入了目标网络来降低偏差，目标网络的参数滞后于评估网络，隔固定步数更新一次。经典的 Actor-Critic 算法同样是 on-policy 的，使用策略π与环境交互，得到四元组(s,a,s',r)，计算L_1用来更新策略函数，然后同样用策略π在状态s'下做出决策a'但不执行，最后计算 TD-error 和L_2用于更新评估网络。

图 4：Actor-Critic 架构示意图



数据来源：东北证券

(2) A2C

为了减小策略梯度中的估计方差，在经典的 Actor-Critic 算法中引入优势函数（Advantage function）得到一个改进版本，即 Advantage Actor-Critic 算法（A2C）。优势函数的定义为某个动作在特定状态下相对于该状态下平均动作价值的优势，即

$$A(s,a) := q(s,a) - V(s) \approx r + \gamma V(s') - V(s).$$

A2C 将目标函数中的$q(s,a)$替换成优势函数$A(s,a)$，由于其近似版本仅依赖于状态价值函数V，所以 A2C 使用一个评估网络$V(s;w)$来近似价值函数，而不是经典 Actor-Critic 算法中的 Q 函数，衡量动作的相对表现，避免了由于当前采用糟糕的策略而对某些动作进行惩罚。A2C 策略网络的损失函数为

$$L_1(\theta) = -\mathbb{E}_\pi\left[\left(r + \gamma\hat{V}(s';w^-) - V(s;w)\right) log\pi(a|s;\theta)\right].$$

评估网络的损失函数为

$$L_2(w) = \mathbb{E}_\pi\left[\left(r + \gamma\hat{V}(s';w^-) - V(s;w)\right)^2\right].$$

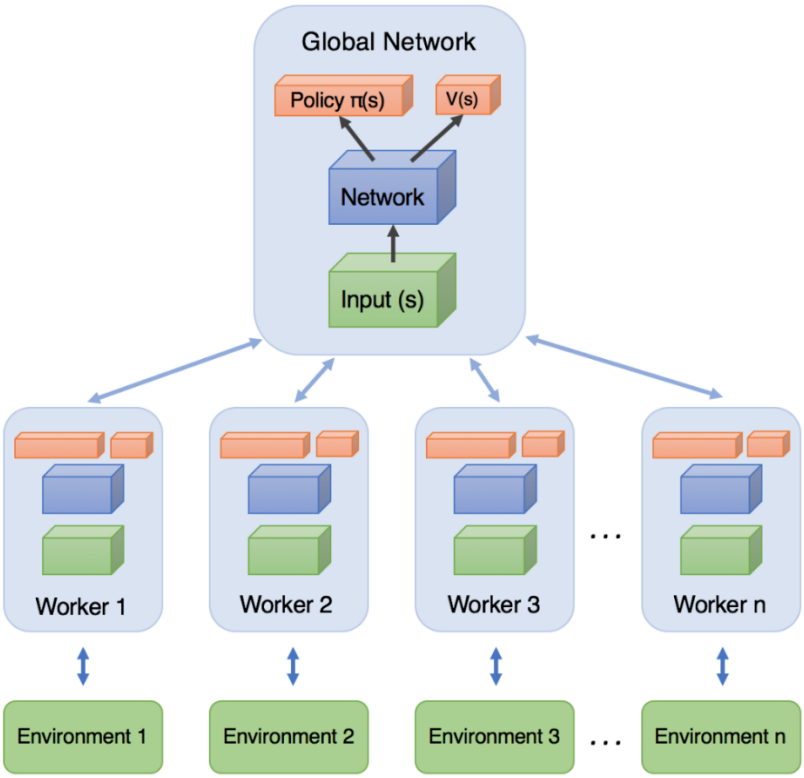
与之前算法一致 A2C 使用了目标网络，训练方式与经典 Actor-Critic 算法一致。在更新策略网络参数时，优势函数会影响更新方向，当优势函数大于零时说明动作选择相对平均具有优势，就会使得参数沿着对数似然的梯度方向更新，即使得策略网络增大在特定状态下该动作选择的概率；当优势函数小于零时则相反。

(3) A3C

Asynchronous Advantage Actor-Critic 算法（A3C）[8]在 A2C 的基础上引入了并行架构，多线程去与环境交互从而探索更多可能的状态与动作，充分利用计算资源，可以在多核 CPU 上实现与 GPU 训练相似的效果。

A3C 的架构由一个 global network 与多个 worker 组成，每个 worker 各自包含一套 A2C 网络，worker 之间是并行独立的。各自与自身的环境交互收集经验，起到与经验缓存与回放类似的效果。A3C 是通过异步更新的方式进行的，换句话说，当某个 worker 采集到足够的经验后，就会用自身的损失函数计算梯度并将梯度上传给 global network 来更新其参数，这个操作叫做 push。再下一次运行之前 global network 会将其参数回传给这个 worker 使其与 global network 同步，这个操作叫做 pull。

图 5：A3C 并行架构示意图



数据来源：https://aemah.github.io/2018/06/26/A3C/

(4) PPO

为了解决传统策略梯度方法对步长选择的敏感性以及采样效率低下等局限性，OpenAI 在 2017 年提出了近端策略优化算法(Proximal policy optimization, PPO)[9]。另外经典的 Actor-Critic 算法包括 A2C 算法均为 on-policy 的，PPO 通过重要性采样（Importance sampling）将其转为 off-policy 提高数据利用效率。

所谓重要性采样，就是借助一个可以采样的分布对其它分布下的函数期望进行估计。

$$\mathbb{E}_{x\sim p}[f(x)] = \mathbb{E}_{x\sim q}\left[f(x)\frac{p(x)}{q(x)}\right].$$

但是由于采样得到的样本有限，所以p和q的分布差异不能太大才能得到可靠的结果。

考虑 A2C 算法策略网络的梯度

$$\nabla J(\theta) = \mathbb{E}_{\pi_\theta}[A(s,a)\nabla \log \pi(a|s;\theta)],$$

应用重要性采样技巧使用旧策略$\pi_{\theta'}$进行采样，策略网络的梯度可以重写为

$$\nabla J(\theta) = \mathbb{E}_{\pi_{\theta'}}\left[\frac{\pi(a|s;\theta)}{\pi(a|s;\theta')}A(s,a)\nabla \log \pi(a|s;\theta)\right].$$

所以策略网络的目标函数为

$$J(\theta) = \mathbb{E}_{\pi_{\theta'}}\left[\frac{\pi(a|s;\theta)}{\pi(a|s;\theta')}A(s,a)\right].$$

另外 PPO 需要限制π_θ与$\pi_{\theta'}$的差异，这种约束体现在两种形式上，前者在目标函数中加入 KL 散度（Kullback-Leibler Divergence）约束（PPO1），后者直接在目标函数中进行限制（PPO2）。两种方式的损失函数如下：

$$L^{KL}(\theta) = -\mathbb{E}_{\pi_{\theta'}}\left[\hat{A}(s;w)\frac{\pi(a|s;\theta)}{\pi(a|s;\theta')}\right] + KL(\pi_\theta \| \pi_{\theta'}),$$

$$L^{clip}(\theta) = -\mathbb{E}_{\pi_{\theta'}}\left[\min\left(\hat{A}(s;w)\frac{\pi(a|s;\theta)}{\pi(a|s;\theta')}, \hat{A}(s;w)clip\left(\frac{\pi(a|s;\theta)}{\pi(a|s;\theta')}, 1-\varepsilon, 1+\varepsilon\right)\right)\right],$$

其中，$\hat{A}(s;w) = r + \gamma\hat{V}(s';w^-) - V(s;w)$近似优势函数，$clip(\cdot, 1-\varepsilon, 1+\varepsilon)$表示把值限制在$1-\varepsilon$与$1+\varepsilon$之间。

评估网络的损失函数跟 A2C 相同，即

$$L_2(w) = \mathbb{E}_{\pi_{\theta'}}\left[\left(r + \gamma\hat{V}(s';w^-) - V(s;w)\right)^2\right].$$

PPO 是 off-policy 算法，训练时通常使用旧策略与环境交互得到大量的经验(s,a,s',r)，然后从经验中不断抽取 mini-batch 计算策略网络与评估网络的损失函数，对网络进行更新。评估网络同样使用目标网络的形式。在网络更新后可以使用新策略再次大量收集经验，如此循环。

(5) DDPG

前述算法均是对随机性策略建模，而深度确定性策略梯度算法（Deep deterministic policy gradient, DDPG）[10]关注确定性策略，即依据状态直接得到确定性动作的策略，它可以作为 DQN 在连续动作情况下的扩展。

DDPG 包含四个网络，即策略网络$\pi(s;\theta)$和评估网络$q(s,a;w)$及其对应的目标网络$\hat{\pi}(s;\theta^-)$和$\hat{q}(s,a;w^-)$。评估网络的损失函数和 DQN 类似，即

$$L_2(w) = \mathbb{E}\left[\left(r + \gamma\hat{q}(s', \hat{\pi}(s';\theta^-);w^-) - q(s,a;\theta)\right)^2\right].$$

另外由确定性策略梯度定理，

$$\nabla J(\theta) = \mathbb{E}[\nabla_a q(s,a)\nabla_\theta \pi(s;\theta)].$$

所以策略网络的损失函数可以简化为

$$L_1(\theta) = -\mathbb{E}[q(s,\pi(s;\theta);w)].$$

DDPG 是 off-policy 算法，为了充分与环境交互得到经验，在采集经验 exploration 过程中给策略加上了动作噪声，即在生成动作为确定性动作加上一个 Ornstein-Uhlenbeck 过程，这个操作类似于 DQN 中的ε-greedy 采样策略。在采集足够多的经验后，通过 mini-batch 方法计算评估网络和策略网络的损失函数并更新网络。最后用 soft-update 方法更新目标网络参数，即每次更新时目标网络参数只按照比例更新一小部分，这样可以大大提高学习的稳定性。所有网络更新结束之后，再次采样循环以上步骤。

(6) TD3

Twin delayed DDPG（TD3）是在 DDPG 的基础上进一步改进得到的。为了缓解 DDPG 中高估 Q 值的可能情况，TD3 使用两套评估网络进行拟合，每个评估网络都有自身对应的目标网络。

在训练时，与 DDPG 一样首先通过加入噪声的动作与环境交互充分探索动作与状态空间，在收集足够的经验后从中采样 mini-batch 来更新评估网络。

两个评估网络的损失函数分别为

$$L_2^i(w_i) = \mathbb{E}\left[\left(r + \gamma \min_{j=1,2} \hat{q}_j\left(s', \tilde{a}; w_j^-\right) - q_i(s, a; w_i)\right)^2\right], \quad i = 1,2.$$

其中，$\tilde{a} = \hat{\pi}(s'; \theta^-) + \varepsilon$，$\varepsilon \sim clip(\mathcal{N}(0, \sigma), -c, c)$。这里计算目标时加噪是为了让训练更加稳健。评估网络更新多次之后，对策略网络进行更新，这种延迟更新可以在一定程度上避免策略的盲目迭代。

策略网络的损失函数为

$$L_1(\theta) = -\mathbb{E}\left[\min_{i=1,2} q_i(s, \pi(s; \theta); w_i)\right].$$

在策略网络更新完之后，同样通过 soft-update 方法更新目标网络。所有更新完成后循环以上步骤。

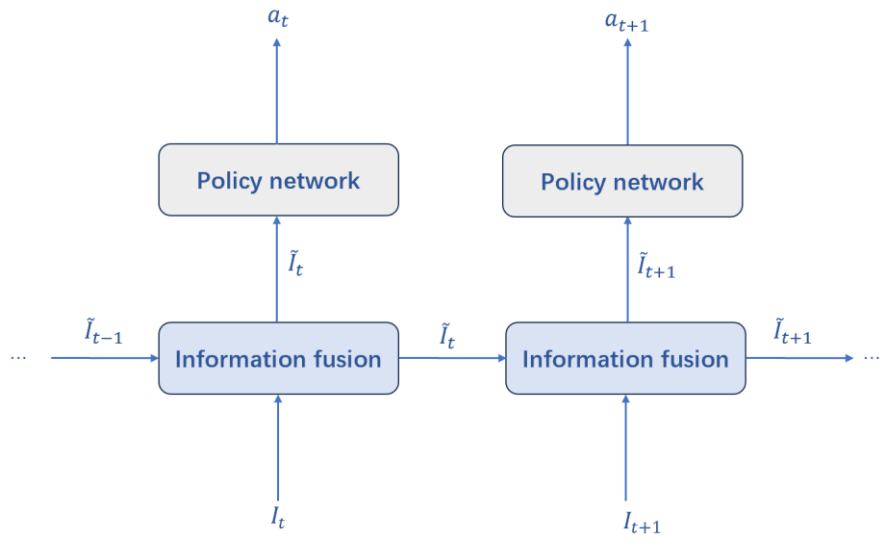
2.3. 循环强化学习

循环强化学习（RRL）在[12]中首次被提出，它是一类 policy-based 直接强化学习算法。通常其不需要估计价值函数，直接对策略进行建模，即从参数化的策略族中定义一系列连续的动作$a = \pi(s;\theta)$，只需要一个带有潜在变量θ的可微目标函数就可以进行优化[13]。

循环强化学习相比于 value-based 间接方法更适用于量化交易问题，一方面是优化的目标更加灵活，即可以选择不同的目标函数，另一方面其对市场状态和动作的描述均是连续的。另外 RRL 相比于其他的 policy-based 算法，其架构简单高效，仅建模策略网络，用灵活的目标函数驱动策略优化，并且对数据量要求不高，能够适用于金融序列数据。

如果我们使用[14]中的构建方式，RRL 的状态由当前时刻信息与可获得的历史信息组成，即$s_t = \tilde{I}_t = f(I_t, \tilde{I}_{t-1})$，这使得我们可以用循环神经网络的思想构建模型，即将历史信息编码在隐变量（Latent variable）中。每个时刻依据模型得到动作$a_t = \pi(s_t;\theta)$获得奖励r_t得到下一个状态s_{t+1}。如此往复至一个 episode 完成后，计算某个定义好的损失函数$L(\theta)$并更新参数。

图 6：RRL 架构示意图



数据来源：东北证券

这样的 RRL 架构包含两部分，信息融合部分与策略网络部分，前者是 RNN 架构来融合历史信息与当前信息，后者是一个 MLP 输入融合后的信息（状态）输出动作。损失函数的设计是多样的，一般基于每一步动作产生的奖励来定义。另外这个架构具有两个退化版本，第一种是将状态定义为当前时刻的信息，认为动作仅依赖于当前时刻的信息，这样信息传递将不需要使用 RNN 进行建模，整个架构只包含策略网络模块；第二种是将信息融合部分与策略网络部分整合，认为动作依赖于当前时刻信息与上一步的动作，整个架构只包含 RNN 模块，将每步的动作作为隐变量进行传递，这个版本可以参考[12]。

此架构相比于退化版本有一定的优势，一般来说信息过程是 non-Markovian 过程，所以状态的定义应该尽可能包含历史信息，这样更加贴合真实情况。

2.4. 强化学习在量化投资上的应用简介

强化学习在量化投资中具有广泛的应用，主要分布于组合优化、算法交易、衍生品对冲等几个方面。

组合优化方面，[15]使用循环强化学习算法训练了一个组合优化系统，它考虑了不同的目标函数，比如 P&L、utility、Sharpe ratio 等，另外系统考虑了交易费用、冲击成本等因素。[16]为组合优化引入了一个新的深度强化学习的框架。

算法交易方面，[17]使用 Q-learning 在风险资产与无风险资产之间进行轮动，实现了对标的的择时。[18]使用基于 LSTM 的循环强化学习方法成功地建立算法交易模型，并利用市场数据测试了模型的适用性与稳健性。[19]使用了 Double DQN 算法实现了下单的优化。

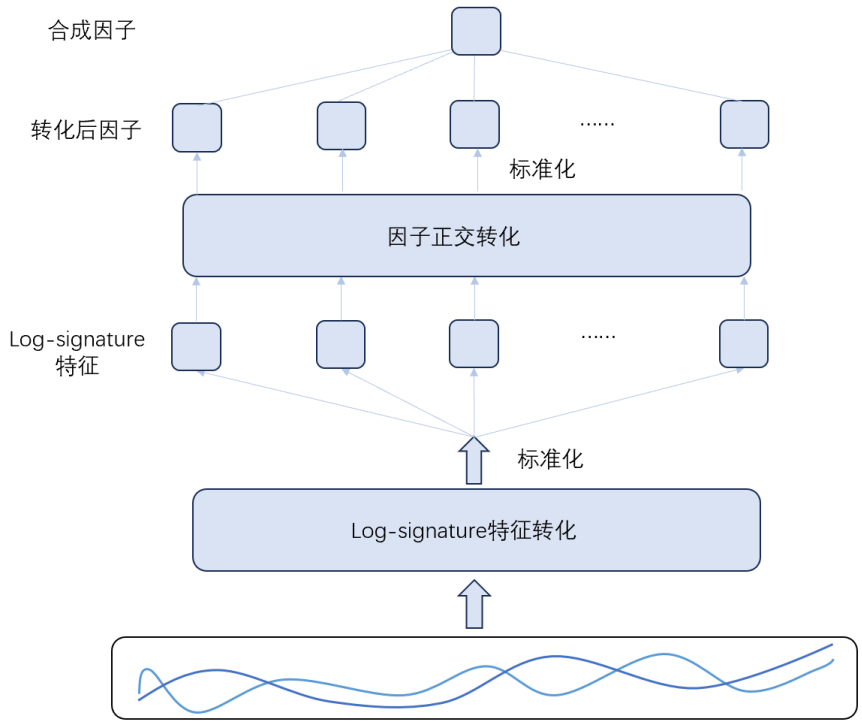
衍生品对冲方面，[20]应用 Q-learning 在 Black-Scholes 假设和无交易费用假设下构建了期权对冲策略。[21]应用循环强化学习的方法提出了一个衍生品组合对冲的框架，该框架考虑了市场摩擦（Market friction）包括交易成本与冲击成本，流动性和风险限制等。[14]在[21]的基础上进一步扩展。[22]考虑在 non-Markovian 市场环境下应用循环强化学习进行衍生品组合的对冲。

3. 应用：循环强化学习做因子合成

3.1. 因子生成

首先对前期报告《日内成交量分布因子及 Logsig-Alpha 因子生成——因子选股系列之六》中的因子进行扩展，根据不同的分钟级量价信息利用 Logsig-Alpha 分别构建低频化选股因子。Logsig-Alpha 是一个从序列到因子的因子生成器，它可以端到端的将序列或数据流转化为因子。它分为两个模块：其一为 log-signature 计算模块，将原始的序列转化为一个特征集，再进行标准化操作后得到基础因子；其二为因子正交化模块，它是一个 MLP 架构，旨在保留因子选股能力的同时降低因子之间的相关性，将正交转换后的因子进行标准化后输出，或是等权合成。架构简单高效，易于操作，详见前述报告。

图 7：Logsig-Alpha 因子生成器的架构



数据来源：东北证券

与前述报告同样的方式，我们使用窗口期为 20 个交易日的 5min 对数成交量序列，进行 lead-lag 变换后序列变为两条，再使用 10 作为截断阶数将序列转化为对应的 log-signature 特征集。Lead-lag 变换优势在于可以提取到序列前后项形成的变差（Variation）信息，其变换形式如下：

对于原始长度为$N+1$的d维序列X，其 lead-lag 变换包含两个长度为$2N+1$的d维序列X^{lead}和X^{lag}：

$$X_{t_j}^{lead} = \begin{cases} X_{t_i} , & if \quad j = 2i \\ X_{t_i} , & if \quad j = 2i-1 \end{cases}.$$

$$X_{t_j}^{lag} = \begin{cases} X_{t_i} , & if \quad j = 2i \\ X_{t_i} , & if \quad j = 2i+1 \end{cases}.$$

每个样本中，上述计算好的 log-signature 作为特征，$t+1$至$t+21$的收益率在经过行业市值中性化和标准化之后作为标签。从 2018 年开始到 2024 年 2 月按年滚动训练更新模型，每次考虑前 4 年的数据，前 3 年作训练集，后 1 年作验证集。在训练时，随机从训练集中采样日期，将当日内处理后全市场数据作为一个 batch 输入模型优化参数。另外设置早停机制（Early stopping），即在验证集上的表现持续低于最优的一轮时停止训练，防止过拟合（Overfitting）。训练时对模型用不同的随机种子训练三次，最终生成因子为三个模型的均值，提升模型的稳健性（Robustness）。

优化目标为加权 IC 再加上转化后因子相关系数的惩罚项，损失函数定义如下：

$$L(X_t,Y_t)=-(\rho^w(X_t,Y_t)-\lambda\parallel M_t\parallel_{L^2}),$$

其中，$X_t=(X_t^1,\cdots,X_t^n)$为t日n个股票的合成因子值，$Y_t=(Y_t^1,\cdots,Y_t^n)$为对应的标签，M_t为合成前各因子的相关系数矩阵，λ为对相关性的厌恶系数，来控制生成因子相关性的强弱。选择优化加权 IC 的目的是降低结果的空头效应，其形式如下：

$$\rho^w(X_t,Y_t)=\frac{\sum_{i=1}^n w_i X_t^i Y_t^i-\left(\sum_{i=1}^n w_i X_t^i\right)\left(\sum_{i=1}^n w_i Y_t^i\right)}{\sqrt{\sum_{i=1}^n w_i\left(X_t^i\right)^2-\left(\sum_{i=1}^n w_i X_t^i\right)^2}\sqrt{\sum_{i=1}^n w_i\left(Y_t^i\right)^2-\left(\sum_{i=1}^n w_i Y_t^i\right)^2}},$$

这里的权重由生成的因子值决定，将其映射到 0 到 1 之间再进行归一化，使得在计算相关系数时对较大的因子值分配更大的权重，即最终优化目标更偏向多头表现。

按上述方式滚动训练得到月频因子 Logsig-Alpha-v。也可以将输入扩展到分钟级价格序列，比如使用窗口期为 20 个交易日的 5min 收盘价序列，数据预处理与对数成交量不同，首先除以最后一日的收盘价进行标准化，之后进行 lead-lag 变换，目的是提取价格序列的变差信息，其中可能含有高频收益率的波动等各阶矩信息，这样处理后训练得到月频因子 Logsig-Alpha-c。另外，可以同时输入窗口期为 20 个交易日的 5min 收盘价序列与开盘价序列，同样除以最后一日收盘价做标准化，然后将其转换为特征集，这样的输入设定希望可以捕捉到日内反转等更高阶的信息，训练得到月频因子 Logsig-Alpha-oc。最后，可以将输入改为最高价和最低价序列，预处理同 Logsig-Alpha-oc 一致，希望捕捉到收益的振幅或更高阶风险类的信息，训练得到月频因子 Logsig-Alpha-hl。

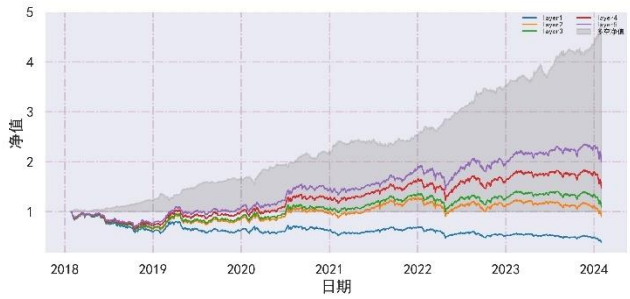
下面是 Logsig-Alpha 系列因子测试结果，其中年化超额基准为全 A 等权。

表 1：Logsig-Alpha 系列月度因子测试结果

因子名称	Rank IC	ICIR	多头年化收益	多头年化超额	多头换手率	多空年化收益	多空年化波动率	多空Sharpe Ratio	多空最大回撤
Logsig-Alpha-v	10.87%	1.17	12.64%	12.99%	1.39	31.01%	9.56%	3.24	9.76%
Logsig-Alpha-c	10.76%	0.94	8.74%	9.09%	1.38	24.06%	12.94%	1.86	18.39%
Logsig-Alpha-oc	10.22%	0.80	7.06%	7.41%	1.19	18.50%	13.66%	1.35	19.03%
Logsig-Alpha-hl	10.07%	0.95	8.47%	8.82%	1.42	20.58%	11.04%	1.86	11.47%

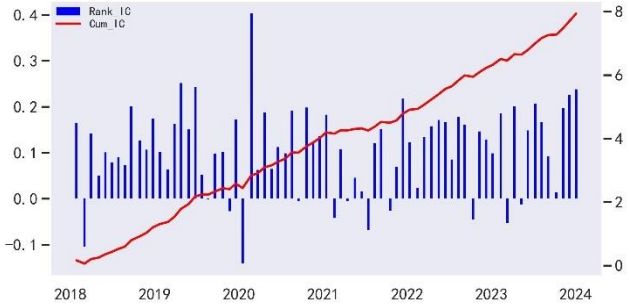
数据来源：东北证券，ricequant

图 8：Logsig-Alpha-v 月度因子分层回测结果



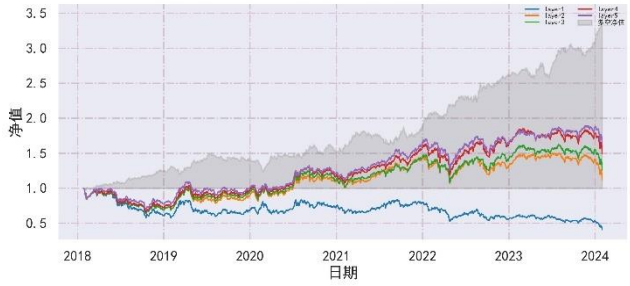
数据来源：东北证券，ricequant

图 9：Logsig-Alpha-v 月度因子 Rank IC



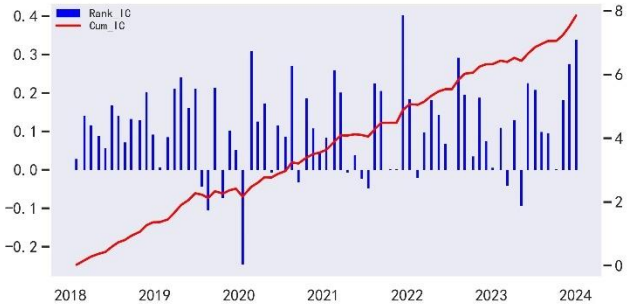
数据来源：东北证券，ricequant

图 10：Logsig-Alpha-c 月度因子分层回测结果



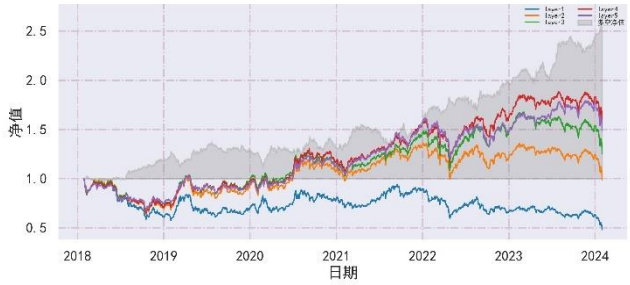
数据来源：东北证券，ricequant

图 11：Logsig-Alpha-c 月度因子 Rank IC



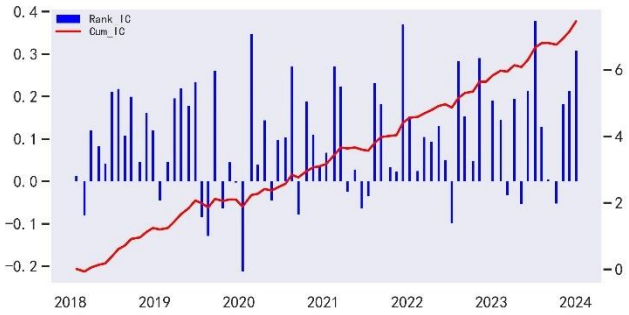
数据来源：东北证券，ricequant

图 12：Logsig-Alpha-oc 月度因子分层回测结果



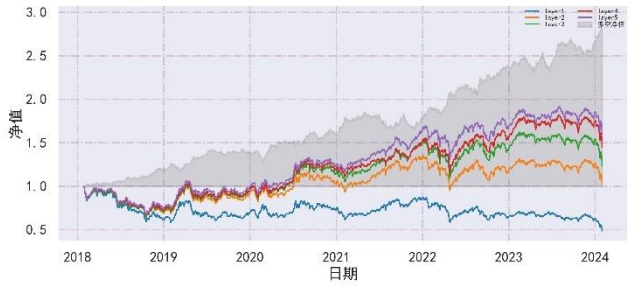
数据来源：东北证券，ricequant

图 13：Logsig-Alpha-oc 月度因子 Rank IC



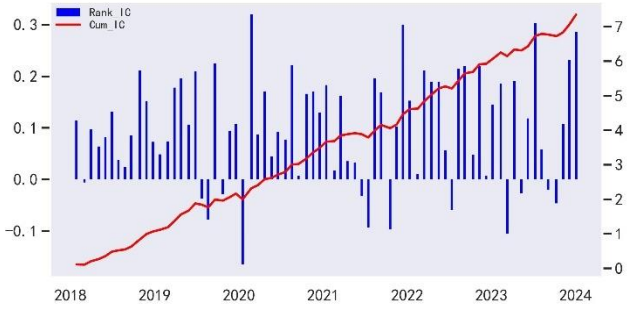
数据来源：东北证券，ricequant

图 14：Logsig-Alpha-hl 月度因子分层回测结果



数据来源：东北证券，ricequant

图 15：Logsig-Alpha-hl 月度因子 Rank IC



数据来源：东北证券，ricequant

表 2：Logsig-Alpha-v 月度因子分年度测试结果

year	Rank IC	ICIR	多头年化收益	多头年化超额	多空年化收益	多空年化波动率	多空Sharpe Ratio
2018	10.06%	1.28	-26.96%	3.22%	27.81%	7.30%	3.81
2019	11.45%	1.32	44.58%	17.13%	36.77%	9.48%	3.88
2020	11.93%	0.92	33.91%	20.16%	33.64%	11.50%	2.93
2021	6.47%	0.70	29.30%	5.22%	17.58%	10.28%	1.71
2022	11.95%	1.77	2.01%	11.68%	39.21%	8.59%	4.56
2023	12.29%	1.30	16.73%	9.96%	23.87%	9.36%	2.55
汇总	10.87%	1.17	12.64%	12.99%	31.01%	9.57%	3.24

数据来源：东北证券，ricequant

表 3：Logsig-Alpha-c 月度因子分年度测试结果

year	Rank IC	ICIR	多头年化收益	多头年化超额	多空年化收益	多空年化波动率	多空Sharpe Ratio
2018	11.33%	2.32	-24.64%	5.54%	28.90%	10.52%	2.75
2019	8.80%	0.72	31.86%	4.41%	16.33%	11.30%	1.45
2020	9.45%	0.64	20.28%	6.53%	8.97%	14.69%	0.61
2021	11.15%	0.78	35.73%	11.65%	25.55%	15.55%	1.64
2022	11.95%	1.29	-2.60%	7.07%	38.38%	13.00%	2.95
2023	9.94%	0.88	12.34%	5.57%	18.27%	11.10%	1.65
汇总	10.76%	0.94	8.74%	9.09%	24.06%	12.95%	1.86

数据来源：东北证券，ricequant

表 4：Logsig-Alpha-oc 月度因子分年度测试结果

year	Rank IC	ICIR	多头年化收益	多头年化超额	多空年化收益	多空年化波动率	多空Sharpe Ratio
2018	10.35%	1.16	-27.52%	2.66%	22.52%	11.28%	2.00
2019	7.12%	0.51	27.73%	0.28%	10.61%	12.17%	0.87
2020	8.35%	0.55	19.58%	5.83%	2.41%	15.43%	0.16
2021	10.89%	0.77	38.50%	14.42%	21.75%	16.96%	1.28
2022	10.27%	0.92	-8.17%	1.50%	24.81%	13.43%	1.85
2023	12.59%	0.95	18.90%	12.13%	24.12%	11.73%	2.06
汇总	10.22%	0.80	7.06%	7.41%	18.50%	13.67%	1.35

数据来源：东北证券，ricequant

表 5：Logsig-Alpha-hl 月度因子分年度测试结果

year	Rank IC	ICIR	多头年化收益	多头年化超额	多空年化收益	多空年化波动率	多空Sharpe Ratio
2018	8.90%	1.51	-27.63%	2.55%	22.93%	8.77%	2.61
2019	9.15%	0.90	34.52%	7.07%	18.92%	9.41%	2.01
2020	11.01%	0.91	25.31%	11.56%	15.87%	11.93%	1.33
2021	8.14%	0.66	32.73%	8.65%	13.01%	12.97%	1.00
2022	12.16%	1.20	-0.28%	9.39%	34.86%	11.56%	3.02
2023	9.52%	0.76	9.17%	2.40%	12.44%	10.58%	1.18
汇总	10.07%	0.95	8.47%	8.82%	20.58%	11.05%	1.86

数据来源：东北证券，ricequant

Logsig-Alpha 生成的月度因子整体表现不错，2024 年来因子多空收益持续上升，且相对全 A 等权组合有一定超额。下面测试因子与常见高频因子之间的相关性。

表 6：Logsig-Alpha 系列因子与常见高频因子的相关性

	日内反转	Logsig-Alpha-c	Logsig-Alpha-hl	Logsig-Alpha-oc	Logsig-Alpha-v	对数成交量日内偏度
日内反转	1.00	-0.40	-0.36	-0.40	-0.27	0.19
Logsig-Alpha-c	-0.40	1.00	0.49	0.59	0.36	-0.31
Logsig-Alpha-hl	-0.36	0.49	1.00	0.70	0.35	-0.29
Logsig-Alpha-oc	-0.40	0.59	0.70	1.00	0.31	-0.28
Logsig-Alpha-v	-0.27	0.36	0.35	0.31	1.00	-0.29
对数成交量日内偏度	0.19	-0.31	-0.29	-0.28	-0.29	1.00
已实现峰度	0.26	-0.27	-0.35	-0.27	-0.36	0.32
已实现偏度	0.43	-0.26	-0.36	-0.25	-0.32	0.27
已实现波动	0.47	-0.63	-0.53	-0.60	-0.35	0.41
UOIDR	-0.19	0.24	0.32	0.22	0.35	-0.18
早盘成交量波动稳定性	0.16	-0.37	-0.35	-0.29	-0.41	0.25
早盘成交量占比稳定性	0.24	-0.37	-0.38	-0.32	-0.41	0.27

数据来源：东北证券，ricequant

表 7：Logsig-Alpha 系列因子与常见高频因子的相关性（接上表）

	已实现峰度	已实现偏度	已实现波动	UOIDR	早盘成交量波动稳定性	早盘成交量占比稳定性
日内反转	0.26	0.43	0.47	-0.19	0.16	0.24
Logsig-Alpha-c	-0.27	-0.26	-0.63	0.24	-0.37	-0.37
Logsig-Alpha-hl	-0.35	-0.36	-0.53	0.32	-0.35	-0.38
Logsig-Alpha-oc	-0.27	-0.25	-0.60	0.22	-0.29	-0.32
Logsig-Alpha-v	-0.36	-0.32	-0.35	0.35	-0.41	-0.41
对数成交量日内偏度	0.32	0.27	0.41	-0.18	0.25	0.27
已实现峰度	1.00	0.69	0.44	-0.50	0.35	0.57
已实现偏度	0.69	1.00	0.41	-0.36	0.27	0.41
已实现波动	0.44	0.41	1.00	-0.34	0.38	0.46
UOIDR	-0.50	-0.36	-0.34	1.00	-0.40	-0.44
早盘成交量波动稳定性	0.35	0.27	0.38	-0.40	1.00	0.84
早盘成交量占比稳定性	0.57	0.41	0.46	-0.44	0.84	1.00

数据来源：东北证券，ricequant

从上表可以看出，首先在 Logsig-Alpha 系列因子中，基于成交量的因子与其余三个基于价格的因子相关性较低，另外基于最高最低价的因子与基于开盘收盘价的因子相关性较高。在系列因子与常见高频因子相关性的比较中可以发现，Logsig-Alpha-v 更偏向成交量稳定性相关因子，Logsig-Alpha-c 偏向收益波动相关因子，Logsig-Alpha-oc 偏向日内反转因子，Logsig-Alpha-hl 偏向已实现高阶矩类因子以及风险稳定性相关因子。

3.2. 模型设定

Logsig-Alpha 系列因子含有的 alpha 信息并不相同，信息融合也就是因子合成方法的选择是一个重要的问题。本节应用循环强化学习 RRL 对 Logsig-Alpha 系列因子进行合成，并与传统的合成方法进行对比。

定义状态$s_t = \tilde{I}_t = f(I_t, \tilde{I}_{t-1})$为当前步信息与历史信息的融合，这个融合是通过RNN进行的。这里I_t指第t步四个因子的最近 20 日 Rank IC，最近 10 日 Rank IC 以及最近 20 日中的 5 日频 ICIR，这里的指标是第t步可获得的最新值，均用历史数据计算，不含有未来数据。$\tilde{I}_{t-1}$指前一步 Rank IC 信息与历史 Rank IC 信息的融合。事实上，由于需合成因子均为月频，每步间隔 20 个交易日。这样的状态定义包含因子当前与历史的指标，另外选用了多个指标进行描述，补充了因子最新的表现与稳定性，相当于在优化因子权重时参考了 Rank IC 的时序变化与期限结构，这相对于仅考虑单时点单指标的传统因子权重优化方法来说更加全面。

定义连续的确定性动作$w_t = \pi(s_t; \theta)$表示因子的权重向量，其中$\pi(\cdot; \theta)$表示策略网络。定义奖励r_t为使用该权重合成因子的 20 日 Rank IC。下一步即$t+1$步的状态s_{t+1}由 20 个交易日之后的数据重新计算。

在训练时，每一个 episode 进行 12 步，步长为 20 个交易日，采样的一条 trajectory 包含一年的数据。每一步输出的因子权重可以用来计算合成因子$X = \{X_t\}_{t=1,\cdots,T}$，其中$X_t = \sum_{i=1}^{N} w_t^i X_t^i$。然后计算合成因子序列对应的 Rank IC 序列$IC = \{IC_t\}_{t=1,\cdots,T}$。

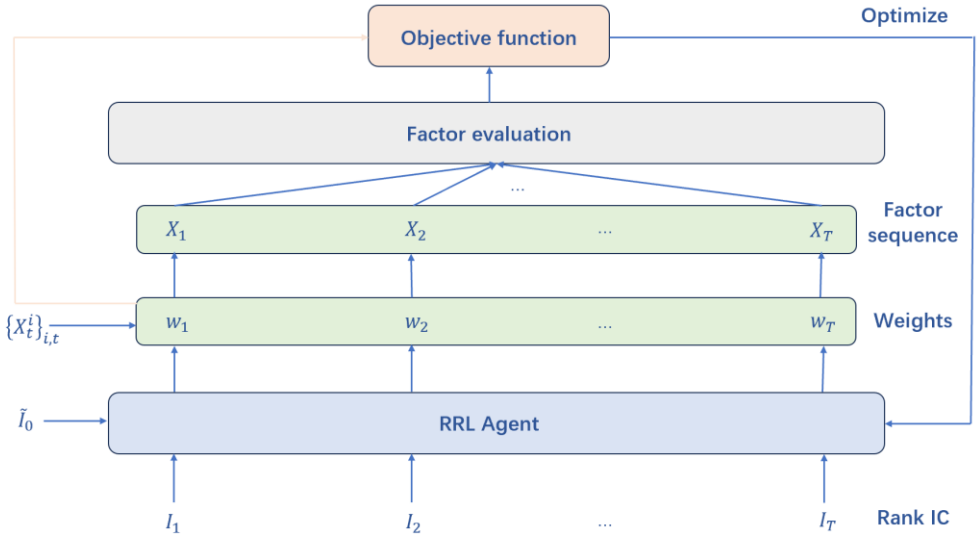
目标函数整体考察了合成因子在该 episode 中的表现，其定义如下：

$$J(\theta) = IC_T + \alpha \frac{\bar{IC}}{\sigma(IC)} - \beta \frac{1}{NT} \sum_{t=1}^{T} \sum_{i=1}^{N} \left(w_t^i - \frac{1}{N} \right)^2.$$

其中，IC_T是最后一步合成因子的 Rank IC，这一项在目标函数中作为 terminal reward，目的是提升考虑全局信息时的因子表现。$\frac{\bar{IC}}{\sigma(IC)}$是合成因子在整个 episode 中的 ICIR，在目标函数中作为 running reward，目的是提升因子的稳定性。$\frac{1}{NT} \sum_{t=1}^{T} \sum_{i=1}^{N} \left(w_t^i - \frac{1}{N} \right)^2$度量因子权重与基准等权之间的差异，在目标函数中作为 running penalty，目的是防止权重过于集中在某个因子上，给出一个基准，起到平衡的作用。

基于 RRL 的因子合成流程示意图如下：

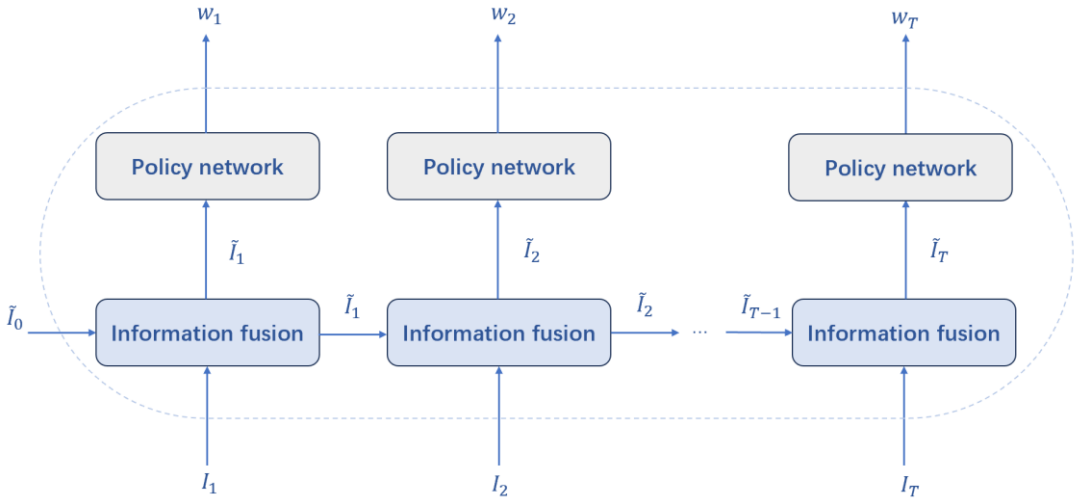
图 16：基于 RRL 的因子合成模型流程



数据来源：东北证券

其中，RRL Agent 的模型结构如下：

图 27：RRL Agent 模型示意图



数据来源：东北证券

流程主要包括以下几个步骤：

(1) Agent 与环境交互

首先选定起始时间点，计算起始信息，即待合成因子当日可获得的多项 Rank IC 与 ICIR 指标。设定历史信息的初始值为 0 向量，与初始信息进行融合得到第一步的状态并输入到策略网络中，得到因子权重。然后到达下一步，每步步长为 20 个交易日，每个 episode 执行 12 步即一年。在新的时间点重新计算信息集，并将其与上一步输出的历史信息进行融合得到新的状态，输入到策略网络中得到第二个时间点的因子权重。如此循环，直至结束。

(2) 因子合成与评估

一个 episode 完成后，根据权重向量序列在每一步对因子进行线性合成，得到合成后因子序列。然后计算 IC、ICIR 等指标对合成因子序列进行评估。

(3) 目标函数与优化

利用合成因子序列的评估指标与权重向量序列计算目标函数，最大化目标函数通过梯度上升算法优化网络参数。

在实际应用中，信息融合部分为一个少量参数的单层 GRU，策略网络为一个线性层加一个 Softmax 操作层，前者降低模型过拟合风险，后者使得输出的权重向量符合要求。在训练时，从 2021 年开始每半年的首个交易日利用前期的数据进行优化，每一个 episode 从可选日期中随机选择一个作为起始点，使得结束点在该日之前且据该日至少 20 个交易日，避免数据泄露。抽取 1000 条 trajectory，对每条 trajectory 计算目标函数并更新一次参数。

在应用训练好的模型于某个交易日对因子进行合成时，首先以当前交易日为结束点向前倒推 12 步，步长为 20 个交易日，作为一条 trajectory。将其输入给 RRL Agent，取结束点对应的因子权重进行线性合成。这样在因子的合成过程中考虑了更长期的历史表现，相比于只依赖当期表现的方法更优。

另外需要提到的一点是，基于 RRL 的因子合成模型参数量较少且最终输出线性权重，模型可以理解为是利用因子 Rank IC 序列以及期限结构的优化器，其优化目标函数的方法可以类比于最大化 IC、最大化 ICIR 的因子合成方法。另外线性的权重也使得因子的合成过程具有较强的解释性。

3.3. 结果对比

使用前一节的模型设定对 Logsig-Alpha 系列因子进行合成，从 2021 年开始模型每半年优化一次。基准合成方法为等权、IC 加权、ICIR 加权以及最大化预期 IC。

其中，IC 加权使用当期各因子可获得 IC 归一化后进行加权；ICIR 加权使用窗口期为 6 个月的 IC 序列计算 ICIR 然后归一化加权；最大化预期 IC 的权重获取过程如下：

假设某时刻共有N个因子，X_i表示第i个因子在股票池上的因子值，R为当前股票池的预期收益，w_i为第i个因子的权重，那么合成因子的预期 IC 为

$$\rho\left(\sum_{i=1}^{N} w_i X_i, R\right) = \frac{\sum_{i=1}^{N} w_i \, Cov(X_i, R)}{\sigma(\sum_{i=1}^{N} w_i X_i)\sigma(R)} = \frac{\sum_{i=1}^{N} w_i \, IC_i}{\sqrt{W^\top \Sigma W}}.$$

其中，W是权重向量，Σ是待合成因子的方差协方差矩阵，最后一个等式成立是因为事先对因子进行过标准化，即$\sigma(X_i) = 1$。最大化预期 IC 方法具有解析解

$$W = \Sigma^{-1} IC.$$

这里的IC为N个因子的预期 IC 向量，为了简便这里直接用滚动 6 月窗口期月频 IC 均值来估计。各合成方法得到因子的测试结果如下：

表 8：RRL 算法合成方法与传统合成方法对比

因子名称	Rank IC	ICIR	多头年化收益	多头年化超额	多头换手率	多空年化收益	多空年化波动率	多空Sharpe Ratio	多空最大回撤
等权	13.60%	1.07	16.08%	15.39%	1.23	34.06%	13.62%	2.5	13.34%
ICIR加权	12.46%	1.03	14.86%	14.17%	1.25	30.97%	12.40%	2.5	12.62%
IC加权	12.54%	0.92	15.64%	14.95%	1.33	33.04%	13.27%	2.49	12.88%
最大化预期IC	11.29%	0.95	13.50%	12.81%	1.37	30.70%	11.78%	2.61	10.78%
RRL	13.61%	1.09	16.46%	15.77%	1.24	34.79%	13.37%	2.6	13.49%

数据来源：东北证券，ricequant

这里的测试时间区间均是从 2021 年 1 月至 2024 年 2 月初。下面是 Logsig-Alpha 在对应回测区间中的测试结果：

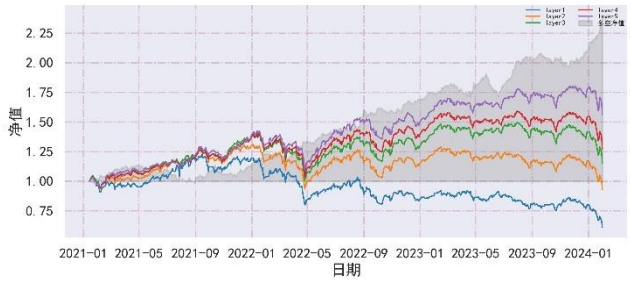
表 9：Logsig-Alpha 系列因子在对应区间的测试结果

因子名称	Rank IC	ICIR	多头年化收益	多头年化超额	多头换手率	多空年化收益	多空年化波动率	多空Sharpe Ratio	多空最大回撤
Logsig-Alpha-c	11.77%	0.98	12.09%	11.40%	1.44	31.15%	13.36%	2.33	10.73%
Logsig-Alpha-hl	10.26%	0.87	9.85%	9.16%	1.38	19.77%	11.69%	1.69	10.79%
Logsig-Alpha-oc	11.97%	0.93	11.30%	10.61%	1.37	27.15%	13.88%	1.96	13.26%
Logsig-Alpha-v	9.46%	1.08	12.63%	11.94%	1.28	25.28%	8.94%	2.83	6.84%

数据来源：东北证券，ricequant

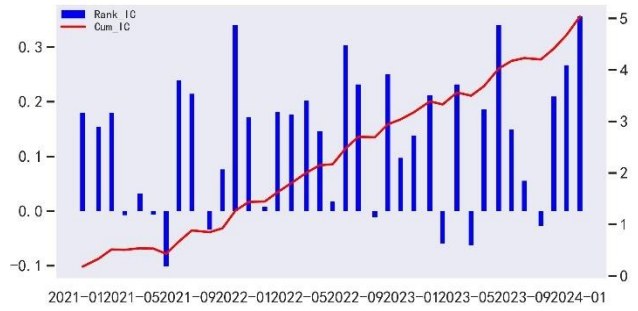
下面是各合成方法测试结果的可视化：

图 18：RRL 合成月度因子分层回测结果



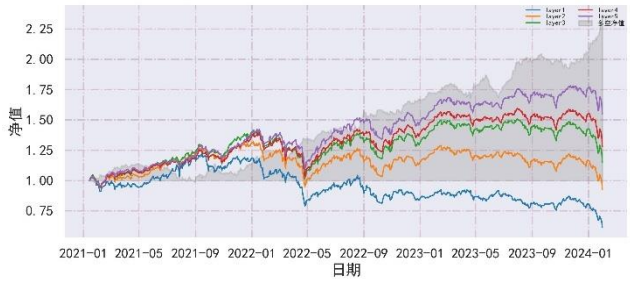
数据来源：东北证券，ricequant

图 19：RRL 合成月度因子 Rank IC



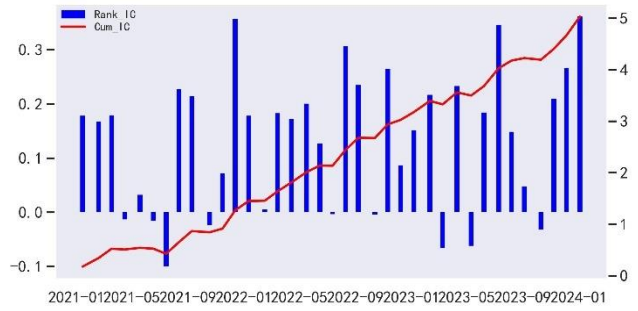
数据来源：东北证券，ricequant

图 20：等权合成月度因子分层回测结果



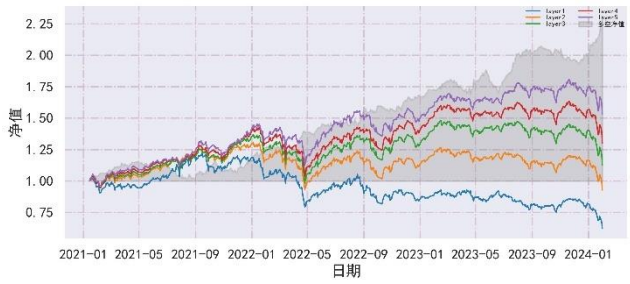
数据来源：东北证券，ricequant

图 21：等权合成月度因子 Rank IC



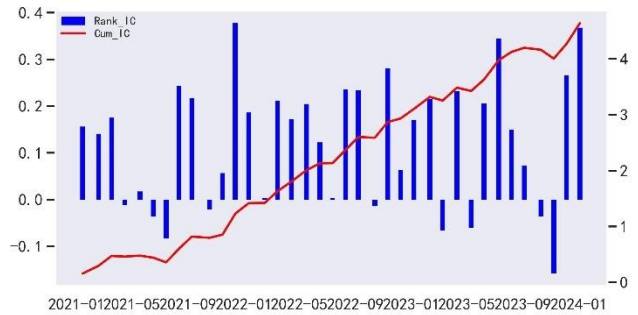
数据来源：东北证券，ricequant

图 22：IC 加权合成月度因子分层回测结果



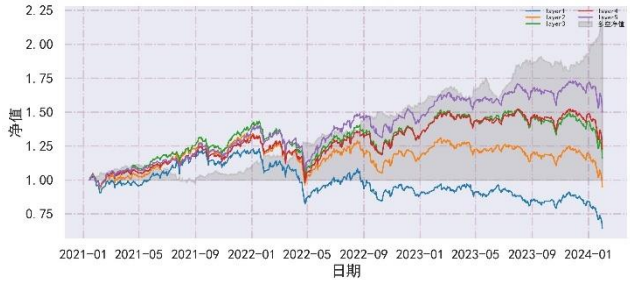
数据来源：东北证券，ricequant

图 23：IC 加权合成月度因子 Rank IC



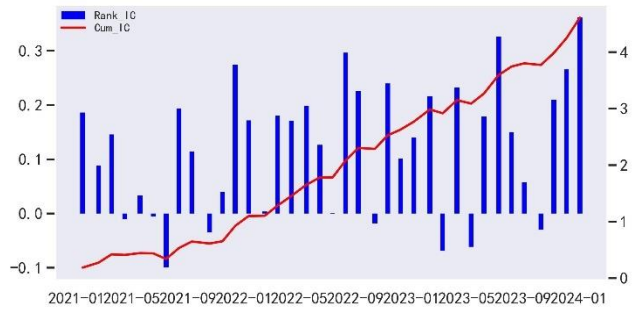
数据来源：东北证券，ricequant

图 24：ICIR 加权合成月度因子分层回测结果



数据来源：东北证券，ricequant

图 25：ICIR 加权合成月度因子 Rank IC

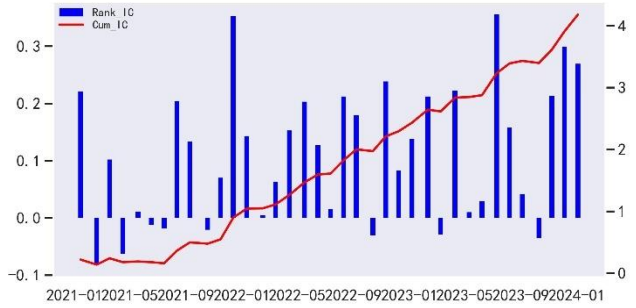


数据来源：东北证券，ricequant

图 26：最大化预期 IC 月度因子分层回测结果

数据来源：东北证券，ricequant

图 27：最大化预期 IC 月度因子 Rank IC



数据来源：东北证券，ricequant

图 28：各合成方法净值对比

数据来源：东北证券，ricequant

从测试结果来看，首先合成后因子表现强于细分因子，说明 Logsig-Alpha 系列因子中各因子含有的 alpha 信息有所不同，融合其交互信息对因子的表现有增强作用。另一方面，各合成方法中，最大化预期 IC 方法表现较弱，这种方法依赖于对 IC 的准确预测，直接使用过去时段 IC 均值作为预期虽简便但准确性较低；IC 加权与 ICIR 加权表现一般，说明传统的基于单期指标的合成方法考虑不够全面，尤其是对于因子动量不显著的情形而言；等权这种一般性的方法表现较优；基于 RRL 的因子合成方法以等权合成方法为基准，综合考虑了多因子多期多指标来给出线性权重，这类方法对信息的利用更加充分，取得最优的合成表现。

4. 总结

强化学习是人工智能领域的重要研究方向，随着 AlphaGo 的成功以及在大模型上的应用，强化学习广泛走进了人们的视野。从最初的 Markov 决策过程建模到经典的强化学习算法，如 Monte-Carlo 方法、时间差分算法等 value-based 方法，这些算法推动了强化学习早期的发展。而后 policy-based 方法的出现以及与深度学习的结合，进一步推进了迈向人工智能前沿的进程，从策略梯度算法开始到 Actor-Critic 架构，这包括 A2C、A3C、PPO 以及确定性策略中的 DDPG、TD3 等高效的算法。

强化学习在量化领域同样有着广泛的应用，包括组合优化、算法交易以及衍生品对冲等方面。本报告着眼于强化学习在因子合成方面的应用，使用简单高效的循环强化学习对基于日内量价序列的 Logsig-Alpha 系列因子进行合成。

循环强化学习 RRL 是一类直接强化学习算法，它直接对策略进行建模，通过一个可微的目标函数优化整个模型，并且可以建模连续的动作与状态，相比于其他经典算法更加灵活。这里的 RRL 模型包含两个模块，信息融合模块以及策略网络模块，前者使用 RNN 融合当前信息与历史信息作为状态，后者基于状态选择动作。

基于 RRL 的因子合成方法中，每个时间点的信息为各因子的多个 Rank IC 以及 ICIR 指标，该点状态定义为当前信息与历史信息的融合；基于状态选择的动作为因子的权重向量；目标函数基于合成因子的整体表现和与基准的偏差来设计。计算目标函数利用梯度上升算法优化参数。

从测试结果来看，基于 RRL 的合成因子表现显著优于细分因子。另外，相比于传统的因子合成方法，如等权、IC 加权、ICIR 加权以及最大化预期 IC，其表现更为优异。RRL 因子合成的优势在于以下几个方面：首先，相比于传统仅考虑单期单因子单指标的方法，基于 RRL 的因子合成方法同时考虑了全部因子多个 Rank IC 和 ICIR 指标，且包含历史数据，也就是说其依据指标的时间序列与期限结构来得到权重，信息输入更加全面。另外，其目标函数可以灵活设定，可以根据不同的偏好选择优化目标，具有一定的可扩展性。最后，RRL 因子合成模型作为一个优化问题解的逼近，架构简单高效，参数量较少，无需额外生成数据序列。

5. 参考文献

[1] Sutton, R. S. Learning to Predict by the Methods of Temporal Differences. Machine Learning 3, 9–44. (1988).

[2] Watkins, C. & Dayan, P. Technical Note: Q-Learning. Machine Learning 8, 279–292 (May 1992).

[3] Rummery, G. & Niranjan, M. On-Line Q-Learning Using Connectionist Systems. Technical Report CUED/F-INFENG/TR 166 (Nov. 1994).

[4] Mnih, V. et al. Human-level control through deep reinforcement learning. Nature 518, 529–33 (Feb. 2015).

[5] Van Hasselt, H., Guez, A. & Silver, D. Deep Reinforcement Learning with Double Q-learning 2015. arXiv: 1509.06461 [cs.LG].

[6] Sutton, R. S., McAllester, D., Singh, S. & Mansour, Y. Policy Gradient Methods for Reinforcement Learning with Function Approximation in Advances in Neural Information Processing Systems (eds Solla, S., Leen, T. & Müller, K.) 12 (MIT Press, 1999).

[7] Williams, R. J. Simple Statistical Gradient-Following Algorithms for Connectionist Reinforcement Learning. Machine Learning 8, 229–256. (2004).

[8] Mnih, V. et al. Asynchronous Methods for Deep Reinforcement Learning (Feb. 2016).

[9] Schulman, J., Wolski, F., Dhariwal, P., Radford, A. & Klimov, O. Proximal Policy Optimization Algorithms (2017).

[10] Lillicrap, T. et al. Continuous control with deep reinforcement learning (Sept. 2015).

[11] Fujimoto, S., van Hoof, H. & Meger, D. Addressing Function Approximation Error in Actor-Critic Methods (2018).

[12] Moody, J. & Saffell, M. Learning to trade via direct reinforcement. IEEE Transactions on Neural Networks 12, 875–889 (2001).

[13] Deng, Y., Bao, F., Kong, Y., Ren, Z. & Dai, Q. Deep Direct Reinforcement Learning for Financial Signal Representation and Trading. IEEE Transactions on Neural Networks and Learning Systems 28, 653–664 (2017).

[14] Buehler, H. et al. Deep Hedging: Hedging Derivatives Under Generic Market Frictions Using Reinforcement Learning. SSRN Electronic Journal (Jan. 2019).

[15] Moody, J. E. & Wu, L. Optimization of trading systems and portfolios. Proceedings of the IEEE/IAFE 1997 Computational Intelligence for Financial Engineering (CIFEr), 300–307. (1997).

[16] Jiang, Z., Xu, D. & Liang, J. A Deep Reinforcement Learning Framework for the Financial Portfolio Management Problem (June 2017).

[17] Du, X. & Zhai, J. Algorithm Trading using Q-Learning and Recurrent Reinforcement Learning.

[18] Lu, D. W. Agent Inspired Trading Using Recurrent Reinforcement Learning and LSTM Neural Networks 2017. arXiv: 1707.07338 [q-fin.CP].

[19] Ning, B., Lin, F. H. T. & Jaimungal, S. Double Deep Q-Learning for Optimal Execution 2020. arXiv: 1812.06600 [q-fin.TR].

[20] Halperin, I. QLBS: Q-Learner in the Black-Scholes(-Merton) Worlds 2019. arXiv: 1712.04609 [q-fin.CP].

[21] Bühler, H., Gonon, L., Teichmann, J. & Wood, B. Deep Hedging 2018. arXiv: 1802.03042 [q-fin.CP].

[22] Horvath, B., Teichmann, J. & Zuric, Z. Deep Hedging under Rough Volatility 2021. arXiv: 2102.01962 [q-fin.CP].

6. 风险提示

分析基于历史数据与模型，存在模型失效风险，历史数据回测结果不代表未来表现。

研究团队简介：

王琦：帝国理工学院数学与金融荣誉硕士，南开大学统计学学士。2021 年加入东北证券上海证券研究咨询分公司任金融工程首席分析师，研究方向为金融工程。曾任职于兴业财富资产管理有限公司，任 FOF 投资经理。

贾英：伦敦大学学院金融数学荣誉硕士，厦门大学数学与应用数学本科。2022 年加入东北证券，研究方向为金融工程、因子选股。现任东北证券金融工程组研究助理。。

张栋梁：复旦大学金融硕士，南京大学金融学本科。2022 年加入东北证券，研究方向为因子选股，现任东北证券上海证券研究咨询分公司金融工程研究助理。

王国鑫：伦敦国王学院金融数学荣誉硕士，武汉大学金融学本科。2023 年加入东北证券，研究方向为量化择时、行业轮动，现任东北证券上海证券研究咨询分公司金融工程组研究人员。

江雨航：加州大学洛杉矶分校金融工程硕士，南开大学理学/经济学学士。2023 年加入东北证券，研究方向为量化固收策略，现任东北证券上海证券研究咨询分公司金融工程组研究人员。

田靖航：北京大学金融硕士，上海财经大学经济学学士。2023 年加入东北证券，研究方向为基金研究，现任东北证券上海证券研究咨询分公司金融工程组研究人员。

刘昱亨：北京大学计算机硕士，北京航空航天大学工学学士。2023 年加入东北证券，研究方向为机器学习与衍生品量化研究，现任东北证券上海证券研究咨询分公司金融工程组研究人员。

分析师声明

作者具有中国证券业协会授予的证券投资咨询执业资格，并在中国证券业协会注册登记为证券分析师。本报告遵循合规、客观、专业、审慎的制作原则，所采用数据、资料的来源合法合规，文字阐述反映了作者的真实观点，报告结论未受任何第三方的授意或影响，特此声明。

投资评级说明

股票投资评级说明	买入	未来 6 个月内，股价涨幅超越市场基准 15%以上。	投资评级中所涉及的市场基准： A 股市场以沪深 300 指数为市场基准，新三板市场以三板成指（针对协议转让标的）或三板做市指数（针对做市转让标的）为市场基准；香港市场以摩根士丹利中国指数为市场基准；美国市场以纳斯达克综合指数或标普 500 指数为市场基准。
	增持	未来 6 个月内，股价涨幅超越市场基准 5%至 15%之间。	
	中性	未来 6 个月内，股价涨幅介于市场基准-5%至 5%之间。	
	减持	未来 6 个月内，股价涨幅落后市场基准 5%至 15%之间。	
	卖出	未来 6 个月内，股价涨幅落后市场基准 15%以上。	
行业投资评级说明	优于大势	未来 6 个月内，行业指数的收益超越市场基准。	
	同步大势	未来 6 个月内，行业指数的收益与市场基准持平。	
	落后大势	未来 6 个月内，行业指数的收益落后于市场基准。	

重要声明

本报告由东北证券股份有限公司（以下称“本公司”）制作并仅向本公司客户发布，本公司不会因任何机构或个人接收到本报告而视其为本公司的当然客户。

本公司具有中国证监会核准的证券投资咨询业务资格。

本报告中的信息均来源于公开资料，本公司对这些信息的准确性和完整性不作任何保证。报告中的内容和意见仅反映本公司于发布本报告当日的判断，不保证所包含的内容和意见不发生变化。

本报告仅供参考，并不构成对所述证券买卖的出价或征价。在任何情况下，本报告中的信息或所表述的意见均不构成对任何人的证券买卖建议。本公司及其雇员不承诺投资者一定获利，不与投资者分享投资收益，在任何情况下，我公司及其雇员对任何人使用本报告及其内容所引发的任何直接或间接损失概不负责。

本公司或其关联机构可能会持有本报告中涉及到的公司所发行的证券头寸并进行交易，并在法律许可的情况下不进行披露；可能为这些公司提供或争取提供投资银行业务、财务顾问等相关服务。

若本公司客户（以下称“该客户”）向第三方发送本报告，则由该客户独自为此发送行为负责。提醒通过此途径获得本报告的投资者注意，本公司不对通过此种途径获得本报告所引起的任何损失承担任何责任。

东北证券股份有限公司

网址：http://www.nesc.cn 电话：95360,400-600-0686 研究所公众号：dbzqyanjiusuo

地址	邮编
中国吉林省长春市生态大街 6666 号	130119
中国北京市西城区锦什坊街 28 号恒奥中心 D 座	100033
中国上海市浦东新区杨高南路 799 号	200127
中国深圳市福田区福中三路 1006 号诺德中心 34D	518038
中国广东省广州市天河区冼村街道黄埔大道西 122 号之二星辉中心 15 楼	510630