文章编号：2095-2163(2020)06-0171-07 中图分类号：TP399 文献标志码：A

一种面向交易关键点的渐近学习算法

郭业清[1]，鞠顺达[2]
（1 天讯瑞达通信技术有限公司，广州 510623；2 深圳大学 大数据系统计算技术国家工程实验室，广东 深圳 518052）

摘 要：在金融时间序列的关键点进行交易是获得高收益低风险的有效方法。本文从给定时间序列中切分出上下文子序列并自动检测子序列中的交易关键点；设计了渐近学习算法 AL，学习子序列中交易关键点的渐近收敛性并做出交易决策。通过交易算法验证，在不同的市场行情下，收益率、最大回撤率等指标上算法 AL 要优于现有算法。

关键词：渐近学习算法；金融时间序列；股票交易；LSTM 神经网络

An Asymptotic Learning Algorithm for Key Trading Points

GUO Yeqing[1], JU Shunda[2]
(1 Tianxun Ruida Communication Technology Co., Ltd., Guangzhou 510623, China;
2 National Engineering Labrador for Big Data System Computing Technology, Shenzhen Guangdong 518052, China)

【Abstract】 Trading at the key points of financial time series is an effective way to obtain high returns at low risk. Firstly, a partition algorithm is designed to segment subsequences in the context from the given time series and automatically detect trading key points. Secondly, the asymptotic learning algorithm AL is arranged to learn the asymptotical convergence and in the subsequence and make trading decisions. Through the trading algorithm, it is verified that the algorithm AL is superior to the existing algorithms in terms of such indicators as RoR and MDD in various market situations.

【Key words】 asymptotic learning algorithm; financial time series; stock trading; LSTM

0 引 言

金融时间序列具有很大的波动性和噪声，增加了高频交易操作中的预测难度和风险。在关键交易点进行准确的交易操作是获得高收益同时降低风险的有效手段[1-4]。对于关键交易点的定义有不同的方式，如 Tang 等人将金融时间序列的拐点做为交易关键点，提出集成 PLR-WSVM 算法预测拐点，并在拐点进行相应“买入”和“卖出”操作，来最大化收益[5]。而 Tsai 等人利用一分钟的日内数据，提出了一种适用于指数期货市场交易的及时开盘突破 TORB 策略。他们将价格突破“阻力位”和“支撑位”的位置作为买入和卖出的关键点，发现 TORB 策略性能优于 TRB 策略[6]。

此外，其他学者还增加了对“持有”操作的研究。基于交易行为的预测[7-9]，为股票算法交易提供了一种端到端的解决方案。如 João Nadkarni 等人将主成分分析法 PCA 与增强拓扑神经演化算法 NEAT 相结合，确定股票市场中的最佳交易点，提出的算法性能表现都优于买入-持有策略[7]。但 Nadkarni 等人并没有在熊市数据集上验证所提出算法的有效性。Sezer 等人提出了一种根据技术指标分析，基于神经网络的股票价格预测与交易系统。利用最常用的技术分析指标 RSI、MACD 等将金融时间序列数据转换为“买入”、“卖出”、“持有”三种交易信号[8]。由于“买入”和“卖出”信号较少，“持有”信号占绝大多数，导致了数据不平衡问题的出现。

深度学习和强化学习技术也在常见交易算法中被使用，如 Jeong 和 Kim 使用深度 Q 学习算法来促进金融交易决策。深度强化 Q 学习通过三层全连接神经网络实现，以预测买入、卖出、持有的股票数量[9]。通过对交易份额的预测，文献[9]中提出的算法，其最大利润比固定股票数量交易算法高出 12 倍。Jinke 等人将 actor-critic 强化学习算法用于交易行为的学习和预测上。该算法中 actor 网络和 critic 网络共享同一个 LSTM 网络，用于更好的产生

基金项目：国家自然科学基金(61972261)。

作者简介：郭业清(1970-)，男，硕士，工程师，主要研究方向：时间序列算法与分布式计算架构；鞠顺达(1995-)，男，硕士研究生，主要研究方向：机器学习与时间序列分析。

通讯作者：鞠顺达 Email：guoyeqing@ tisson.cn

收稿日期：2020-01-17

Q 值和策略逻辑,进一步用于 actor-critic 的决策[10]。此类算法虽通过对交易动作(持股比率或交易概率)做出预测,但忽略了对交易关键点研究。

传统的投资组合选择算法[11-12]也可以被看做是基于交易行为预测的算法。其使用包含多个资产的比率向量作为算法的预测输出,需要预测每项资产的份额,类似于算法[9-10]。上述基于交易行为的预测算法都采取了每一天都进行交易决策的方式进行操盘,具有较高的交易频率,不仅增加了交易手续费且降低了算法抗风险能力。

本文认为,交易关键点只有在特定的上下文子序列中才有意义,定义为子序列中的最佳买入和最佳卖出点。因此,本文设计了上下文子序列分割算法,划分出相应的子序列,构造训练数据集。基于 LSTM,提出面向交易关键点的渐近学习算法 AL。针对序列中关键交易信号数据不平衡问题,提出使用交易关键点附近数据,进行连续预测未来交易关键点统计信号的方法,通过统计信号的收敛性做出交易决策。从而提高了算法预测的准确性和鲁棒性。

1 上下文子序列

为了从市场序列 $X = \{x_i \mid x_i = (o,c,h,l,a)_i^{\mathrm{T}}\}_{i=1}^{n}$ 中划分出上下文子序列,设计辅助序列 R:

$$R = \beta rsi(c_{j=1}^{n}) + (1-\beta) rsi(a_{j=1}^{n}), \quad (1)$$

其中,$\beta \in [0,1]$ 为输入价格序列 $P = \{c_{j=1}^{n}\}$ 和交易量序列 $A = \{a_{j=1}^{n}\}$ 的调整权重。设置权重 β 可以从两个序列中提取出不同的特征组成辅助序列 R,rsi 函数为 *RSI* 指标的计算方法[13]。

1.1 上下文子序列定义

根据辅助序列 R 划分所得的上下文子序列,根据交易操作方向不同分为 RB(Rounding Bottom)子序列和 RT(Rounding Top)子序列。以 RB 子序列为例,将第 i 个 RB 子序列 X^i 展示如图 1 所示:通过图中实线所示,辅助序列 R 在虚线所示的价格时间序列 P 上确定 D 点、L 点和 U 点。a 和 b 分别为辅助序列 R 的上下门限,ε_D 为 a 点的松弛。

第 i 个子序列 X^i 由 3 个关键点 D 点、L 点和 U 点来定义。3 个关键点由包含价格和时间的二维坐标表示,指数的上下门限记为 a 和 b。

D 点表示趋势下降阶段的起点,D 点坐标为 (p_D,t_D)。D 点所处区间为 $[t_1,t_2]$,其中:

$$t_1 = \min\{t \mid R_t \geqslant a, t \in [t_L', t_U']\}, \quad (2)$$

$$t_2 = \min\{t \mid R_t \leqslant a - \varepsilon_D, t_1 < t < t_D + T_B\}, \quad (3)$$

式(2)中,$[t_L', t_U']$ 为前一个 RB 子序列的 L' 点和 U' 点所确定的时间区间。在寻找第一个 RB 子序列时,初始设 $t_L' = 0, t_U' = ¥$。T_B 为 RB 序列的时间窗口大小,R 小于 $a - \varepsilon_D$ 的第一个点。

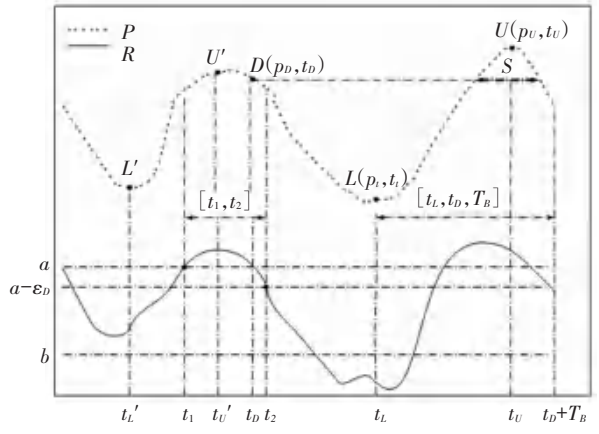


图 1 RB 子序列 X^i

Fig. 1 RB subsequence X^i

$$p_D = \underset{t \in [t_1,t_2]}{\operatorname{argmin}}\{e^{\lambda(t_2-t)} p_t\}, \lambda \in [0,1], \quad (4)$$

其中,$e^{\lambda(t_2-t)}$ 为价格权重,λ 为常量。L 点是趋势下降阶段的终点,坐标为 (p_L,t_L):

$$p_L = \min\{p_t \mid R_t < b, t \in [t_D, t_D + T_B]\}. \quad (5)$$

在 $[t_D, t_D + T_B]$ 区间内,满足 $R_t < b$ 的价格集合中,取最小价格为 p_L,t_L 为对应时间。U 点坐标为 (p_U,t_U),是 RB 子序列的终点,U 点价格应上升到 p_D 之上。S 是 $[t_L, t_D + T_B]$ 区间上价格大于 p_D 的样本集合。S 的表达式为:

$$S = \{t \mid p_t \geqslant p_D, R_t > a, t_L < t < t_D + T_B\}, \quad (6)$$

取 U 点为集合 S 中的价格序列最高点。

$$p_U = \max\{p_t \mid t \in S\}, \quad (7)$$

p_U 为集合 S 中价格最大值,t_U 为 p_U 对应的时间。

确定 RB 子序列后,由图 1 所示,相邻 RB 子序列的 L' 点,与 L 点间可看作 RT 子序列。

1.2 上下文子序列的生成算法

由 1.1 节可知,要确定一个完整的 RB/RT 子序列,首先需要确定 D 点位置,再根据 D 点确定 L 和 U。据此设计上下文子序列生成如算法 1 所示。

算法 1 上下文子序列生成算法

输入:价格时间序列 P 和交易量时间序列 A,辅助序列 R 上下门限 a 和 b,指标的松弛 ε_D,子序列时间窗口 T_B,指数加权调节参数 λ

输出:队列 $Q = \{\langle (p_D,t_D), (p_L,t_L), (p_R,t_R)\rangle_k\}_{k=1}^{n}$

实现步骤如下:

(1)根据式(1)由序列 P 和序列 A 生成 R 时间序列。

(2)初始化 $t_L' = 0, t_U' = ¥$

(3) for r_i in $R[t_L', t_U']$:

(4) if $r_i > a$: $t_1 = i$

(5)for r_j in $R[t_1, t_1 + T_B]$:

(6) if $r_j < a - \varepsilon_D$: $t_2 = j$

(7)for r in $R[t_1, t_2]$: 代入(4)，求得 (p_D, t_D)，将 D 点 (p_D, t_D) 加入队列 Q。

(8)for r in $R[t_D, t_D + T_B]$: 确定 $R < b$ 的区间，代入式(5)，确定 (p_L, t_L)，将 L 点坐标 (p_L, t_L) 加入队列 Q。

(9)for r in $R[t_L, t_D + T_B]$: 确定价格大于 p_D 的样本集合 S。

(10)for s in S: 根据式(7)，求得 (p_U, t_U)，将 U 点 (p_U, t_U) 加入队列 Q。

(11)由[D, U]确定 RB 子序列，相邻[L', L]确定 RT 子序列。

(12)for r in R: 更新 $t_L' = t_L, t_U' = t_U$，转到行(3)，寻找下一个 D, L 和 U。

算法首先提取价格时间序列和交易量时间序列的特征组成辅助序列 R，如式(1)所示，第 3~第 8 行通过辅助序列 R 确定 D 点所处区间。第 9~第 10 行确定 D 点，并加入加入队列 Q，第 11~第 17 行依次根据 D 点确定 L 点和 U 点。最后更新 $t_L' = t_L, t_U' = t_U$，返回行 3，继续寻找下一个 D 点，L 点和 U 点。

2　渐近学习模型

2.1　关键点预测目标函数

给定上下文子序列 $X^i \subset X, i = 1, \ldots, n$，$\dim(X^i) = N \times 5, N \geqslant 1$。对每个块 X^i 分割成 m 个子块 $Y_j^i \subset X^i, j = 1, \ldots, m, \dim(Y_j^i) = M \times 5, M \geqslant s$，$M \bmod s = 0$。其中 $s \geqslant 1$ 为预先设置的常量。如图 2 所示。

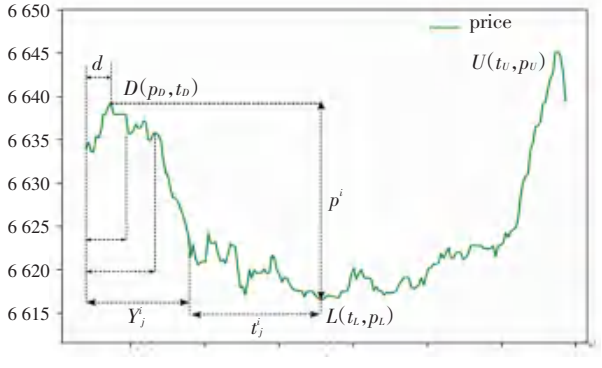


图 2　子序列X^i 中子块Y_j^i

Fig. 2　The time points *D* and *L* in subsequence X^i

设计神经网络函数 $f: R^{5\times M} \rightarrow R^3$ 为：

$$(\hat{\gamma}_j^i, \hat{\rho}_j^i, \hat{\eta}_j^i)^T = f(Y_j^i; \theta). \tag{8}$$

其中，θ 为神经网络优化优化参数，$\hat{\gamma}_j^i$ 是对真值 $\gamma^i = (t_L^i - t_D^i)/W_T$ 的估计，t_L^i、t_D^i 分别表示 Y_j^i 所属 X^i 子序列的 L 点和 D 点对应的时间。$W_T = \max\{t_L^k - t_D^k\}_{k=1}^n$，表示 n 个 X^i 中的 L 点和 D 点的最大时间差。$\hat{\rho}_j^i$ 是对真值 ρ^i 的估计，其中 $\rho^i = 1 - p_L/p_D$。$\hat{\eta}_j^i$ 是对真值 $\eta_j^i = a/a_D^i$，$a = Y_{j+1}^i[M][5]$ 的估计。

神经网络 $f(Y_j^i; \theta)$ 的训练目标：

$$\min_\theta \frac{1}{N}\sum_{i=1}^{N}\frac{1}{m}\sum_{j=1}^{m}\frac{t_L^i - t_D^i}{t_L^i - t_j^i}[(\hat{\gamma}_j^i - \gamma^i)^2 + (\hat{\rho}_j^i - \rho^i)^2 + (\hat{\eta}_j^i - \eta_j^i)^2]. \tag{9}$$

每次输入的起点 D 点是第 d 条记录开始。对于同一个 X^i 的不同次输入 $Y_j^i, j \in [1, m]$，γ^i 和 ρ^i 为固定值，设置惩罚项 $1/(t_L^i - t_j^i)$，当 Y_j^i 记录数越大时，惩罚项越大。

2.2　关键点学习神经网络

基于辅助序列 R，由生成算法分割出 n 个不同上下文子序列，每个上下文子序列 X^i 中子块 Y_j^i 是由包含 o, c, h, l 的 4 维价格向量和交易量组成的 5 维向量。LSTM 神经网络模型采用序列模型，Batch Normalization 层用来保持数据输出的标准化。然后是 LSTM 层，其次是 Dropout 层，以特定的概率随机舍弃一些神经元来防止过拟合。Dense 层用于输出最终结果，输出一个三维向量，即 $(\gamma, \eta, \rho)^T$，如图 3 所示。

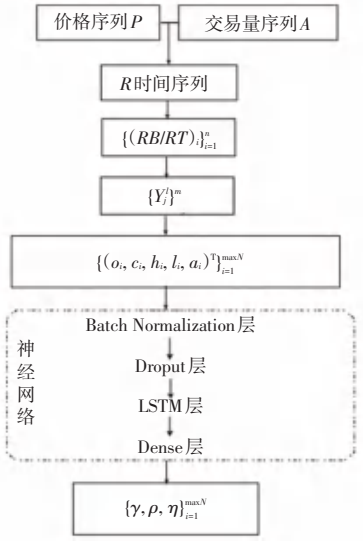


图 3　神经网络架构

Fig. 3　Neural network architecture

2.3 渐近学习算法 AL

将金融时间序列数据输入到 2.1 节中训练好的神经网络 $f(Y_j^i;\theta)$ 中连续预测,并对交易关键点的收敛属性自动进行统计分析,得到交易决策依据,如算法 2 所示。

算法 2 渐近学习算法

输入:时间序列数据 $\{x_k\}_{k=1}^{¥}$, 辅助序列 R, 交易关键点预测神经网络 $f(Y_j^i;\theta)$,R 序列上门限 a, 记录窗口大小 s, 确定 η,ρ,γ 收敛记录数 b 以及收敛阈值 $\varepsilon_\eta,\varepsilon_\rho,\varepsilon_\gamma$。

输出:交易点信息 $\{(k,x_k)\}_k$

实现步骤如下:

(1) $Y=\varphi$, $B=\varphi$, $Q=\varphi$, $sc=0$

(2) for $k=1$ to infinity:

(3) if $R_k >= a$:

(4) sc += 1, $Y=\varphi$, $B=\varphi$, $Q=\varphi$

(5) continue

(6) If $sc > 0$:

(7) $sc=0$, Y += x_k

(8) Let $\dim(Y)=M\times 5$

(9) If M *mod* s == 0:

(10) $(\hat{\gamma},\hat{\rho},\hat{\eta})=f(Y;\theta)$, B += $(\hat{\gamma},\hat{\rho},\hat{\eta})$

(11) Let $L=length(B)$

(12) $(\bar{\gamma},\bar{\rho},\bar{\eta})=\frac{1}{L}\sum_i^L B[i]$, Q += $(\bar{\gamma},\bar{\rho},\bar{\eta})$

(13) Let $L=length(Q)$

(14) If $L < b$: continue

(15) $(\gamma_\varepsilon,\rho_\varepsilon,\eta_\varepsilon)=\sup\{|Q[p]-Q[q]|: p, q\in[L-b+1,L]\}$

(16) If $(\gamma_\varepsilon,\rho_\varepsilon,\eta_\varepsilon) > \varepsilon_\gamma,\varepsilon_\rho,\varepsilon_\eta$: continue

(17)输出交易关键点信息: (k,x_k)

首先根据输入时间序列 $\{x_k\}_{k=1}^{¥}$, 设定状态标志 sc,利用 sc 状态和 R 序列来确定上下文子序列开始标志,如行(3)~行(7)所示。M 为 Y 中记录数,每条记录包含 (o,c,h,l,a) 这 5 个维度的信息,当 M 为 s 整数倍时,将原始数据序列 Y 输入训练好的神经网络函数 $f(Y_j^i;\theta)$ 中去,得到信息的预测序列 B,如行(9)、(10)所示。同时每次计算出预测值序列的平均值,得到平均值序列 Q, 如(11)、(12)所示。当序列 Q 长度大于确定 η,ρ,γ 收敛记录数 b, 计算 Q 中所有元素距离的上确界,当关键交易点相关信息上确界值 $(\gamma_\varepsilon,\rho_\varepsilon,\eta_\varepsilon)$ 小于收敛阈值 $\varepsilon_\eta,\varepsilon_\rho,\varepsilon_\gamma$ 时,输出交易关键点信息,如行(13)~(17)所示。

3 实验结果与分析

本节主要介绍了实验结果评价指标以及训练与测试数据集来源,并按照 3.3 设置各对比算法参数。

3.1 评估指标

本文使用以下 3 种指标来评价各种算法的表现: 回报率(RoR)、夏普率(SR) 和最大回撤率(MDD)。 回报率 RoR: $=\frac{v_f-v_i}{v_i}\times 100$,$v_f$ 是最终资产值, v_i 是初始资产值。夏普率 SR: $=\frac{\mathsf{E}r_i-r_f}{\sigma r_i}\times\sqrt{252}$,$\mathsf{E}r_i$ 是投资的期望回报, r_f 为无风险回报率, σr_i 是 r_i 的标准差。初始值 r_f 设为 0.02。最大回撤率为: MDD: $=\max_{\tau\in(0,n)}\left\{\max_{t\in(0,\tau)}\frac{v_t-v_\tau}{v_\tau}\right\}$。

3.2 数据来源

测试数据如表 1 所示,每条数据分别包含开盘价、收盘价、最高价、最低价、交易量,五个特性的信息。如图 4 中“UBAH”曲线所示,数据集 WMT,AAPL 市场行情上涨,为“牛市”行情。NASDAQ,AMD 数据集市场行情下跌,为“熊市”行情,最后 MSFT,BIDU 数据集行情下跌,为“盘整”行情。因此这六个数据集包含了不同的市场趋势,可以更好地测试各个算法的性能表现。

表 1 数据集

Tab. 1 Summary of dataSets

数据集	日期	记录数	市场行情	来源
D_1	[2012.01.03,2018.12.31]	1760	牛市	AAPL
D_2	[2001.01.02,2004.12.31]	1004	盘整	MSFT
D_3	[2006.01.03,2015.12.31]	2517	熊市	AMD
D_4	[1999.12.31,2007.12.31]	2011	熊市	NASDAQ
D_5	[2013.12.31,2018.12.31]	1259	盘整	BIDU
D_6	[2007.01.03,2014.12.31]	2014	牛市	WMT

3.3 算法参数

AL 算法与其他对比算法的配置如下:

· UBAH 算法为市场策略,即为“买入-持有”策略,反应当前真实市场行情。

· AL:渐近学习算法,具体神经网络参数为:LSTM 层神经元数目为 247,Dense 层神经元数目为 3,学习率为 0.01,迭代系数为 500。

· TORB:即时开盘区间突破算法,以每分钟交易量平均值 PMMV 和每分钟回报方差 PMVR 作为参数[6]。

· PG:梯度策略算法[14]是一种基于智能体的

强化学习算法，旨在模拟专业的建议策略，参数设置：训练回合数为 20，每回合训练步数为 200。

· ONS：使用 Agarwal 的在线牛顿步骤[11]，并将参数设置为 $\eta = 0, \beta = 1, \gamma = 1/8$。

· CFR－OGD：在线梯度下降的组合回归策略[12]，具体参数设置为：$\varepsilon = 10, \omega = 5$。

· PLR－WSVM：一种结合分段线性表示和支持向量机的拐点预测交易算法，买入和卖出的 $RSI_{24}(t)$ 分别设置为 40 和 60，$\alpha_{PLR} = 0.05, \beta_{PLR} = 5$，$nw_{op} = 1, nw_{tp} = 1$，其他参数按论文[5]设置。

3.4　结果分析

针对表 1 中的数据集，分别使用 3.3 小节中提到的算法并设置相关参数进行测试，实验结果如表 2 所示，最佳指标采用加粗显示。

在上述 3 个评价指标中，AL 算法在收益率和夏普率指标在除 D_4 上都达到了最佳。AL 算法与在各数据集上性能表现第二名算法相比，分别将收益率指标提高了［83%，214%］，将夏普率指标提高了［71%，996%］。最大回撤率指标方面，在 D_5 数据集上，传统组合投资算法 ONS 获得最优值，在剩余数据集上 AL 算法均获得最佳。

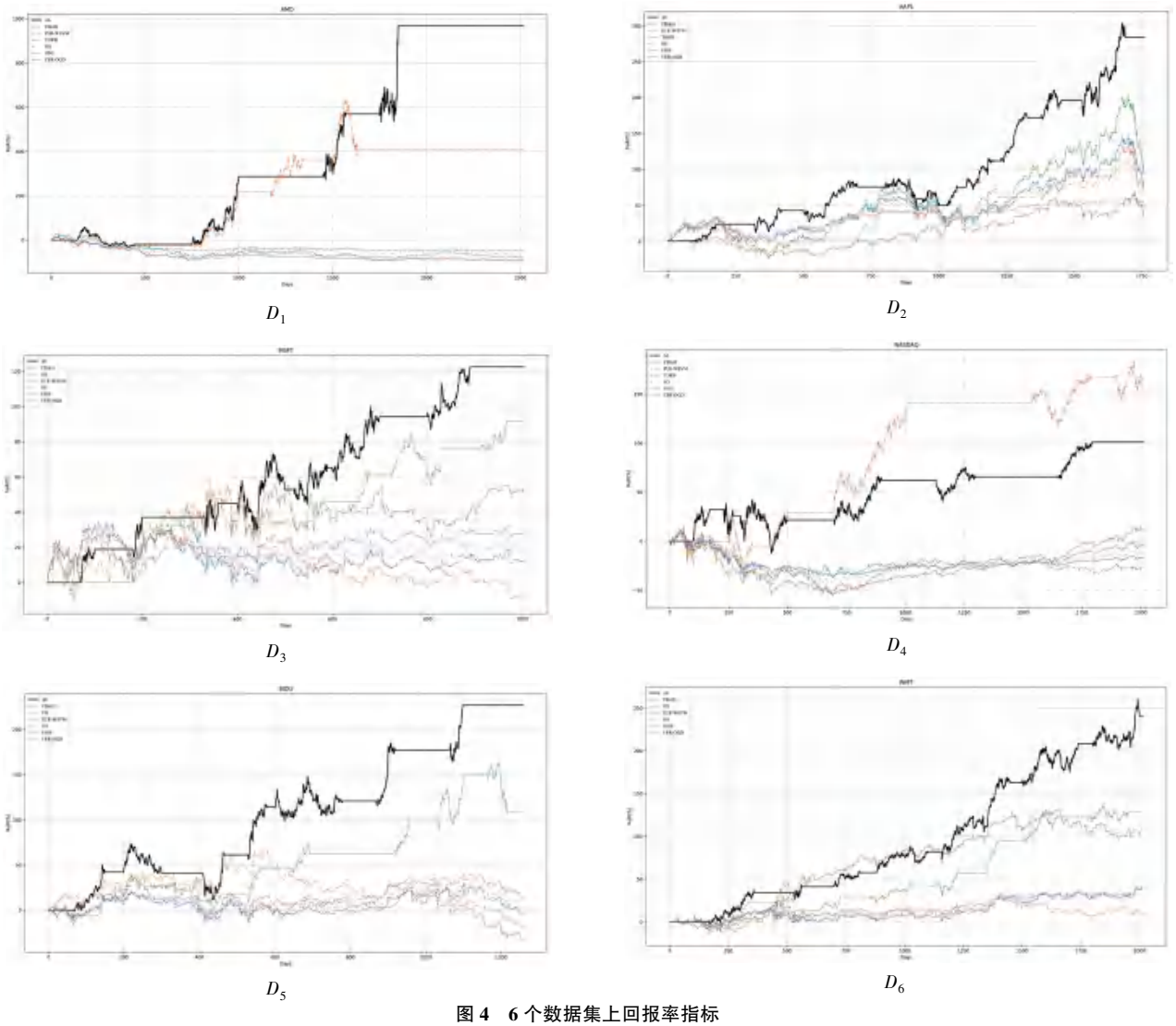
在图 4 中，对比 AL 和其余各算法的 RoR 曲线可知，AL 算法交易频率较低，仅保证必要的交易动作，只在关键的买卖点出进行“买入”和“卖出”操作。相比与交易操作模式相近的 PLR－WSVM 算法，渐近学习算法 AL 交易频率更低，抗风险能力更高，而且渐近学习算法 AL 相对于 PLR－WSVM 交易操作更准确，PLR－WSVM 容易出现买入不准确的情况。例如在 D_1 数据集上，PLR－WSVM 都在第 250 天以及 700 天左右时，都出现了过早买入的情况。

对于其它算法，在 D_1 数据集上，前 1700 天内，TORB 算法收益比较接近 AL 算法，但是在 1700 天后，AL 算法通过鲁棒性的关键点预测比较好的规避了市场行情骤的降风险，如 UBAH 曲线所示。TORB 算法并没有抵抗该风险的能力，所以最终收益出现了较大幅度的损失。在 D_6 数据集上，可以发现前 1300 天时，CFR－OGD 算法可以保持和 AL 算法相同的盈利能力，但是随后 AL 算法盈利能力较大的领先 CFR－OGD 算法。因为从 UBAH 曲线可以看出随后市场为“盘整”行情，AL 算法仍能准确预测出关键买卖点，保持平稳收益，而 CFR－OGD 算法出现收益下滑。

表 2　实验结果

Tab. 2　Summary of experimental results

数据集	指标	AL	UBAH	PG	TORB	EG	ONS	CFR－OGD	PLR－WSVM
D_1	SR	**1.101 490 6**	0.495 799 3	0.647 455 8	0.502 369 1	0.523 952 6	0.569 079 7	0.258 803 3	0.441 291
	RoR(%)	**284.576 44**	84.253 3	54.689 255	106.123 34	74.014 7	54.498 1	43.350 2	81.673 14
	MDD	**0.192 615 9**	0.293 105 1	0.205 214 4	0.364 507 7	0.248 777 9	0.193 015 7	0.417 064 7	0.376 517
D_2	SR	**0.925 577 5**	0.136 532 9	−0.060 326	−1.430 202	0.206 701 6	0.350 481 3	0.460 538 7	0.858 782
	RoR(%)	**122.444 54**	11.602 3	−9	−2.076 39	17.717 3	27.035 7	51.696 7	91.625 38
	MDD	**0.178 128 5**	0.263 550 9	0.427 673	0.196 735 4	0.221 515 4	0.184 199 1	0.207 118 6	0.234 583
D_3	SR	**0.828 296 6**	−0.710 896	−0.750 164	−0.241 728	−0.229 134	−0.116 995	−0.513 328	0.607 556
	RoR(%)	**967.527 72**	−45.571	−23.307 6	−78	−54.628 2	−85.817 1	−93.076 8	408.611 8
	MDD	**0.574 872 2**	0.543 355 6	0.575 943 3	0.868 217 1	0.754 863 4	0.929 441 1	0.953 036 7	0.440 335
D_4	SR	0.435 485 3	−0.344 858	0.052 876 5	−0.205 673	−0.180 143	−0.018 296	0.087 868 3	**0.597 208**
	RoR(%)	100.986 07	−17.411 2	17.111 788	−26	−12.336	−4.914 2	11.703	**167.916 3**
	MDD	**0.384 925 6**	0.431 513 4	0.401 738 7	0.495 652 2	0.497 179	0.599 442 6	0.590 019 6	0.426 823
D_5	SR	**1.080 541 4**	−0.086 373	−0.124 695	−0.002 723	−0.016 79	0.098 509	−0.261 962	0.669 11
	RoR(%)	**226.840 39**	−5.419 4	−17	−13.582	1.231 5	12.027 2	−30.955 2	108.356 1
	MDD	0.357 126 5	0.275 489 2	0.514 970 1	0.468 310 6	0.262 658 2	**0.241 152 9**	0.514 654 2	0.285 179
D_6	SR	**1.075 077 8**	0.264 521 5	−0.002 387	−0.216 33	0.270 705 1	0.278 679	0.584 266 9	0.629 084
	RoR(%)	**240.600 66**	40.304 9	8	11.471 1	39.705 7	38.455	107.610 6	128.211 1
	MDD	**0.107 695**	0.151 282 2	0.210 526 3	0.113 678 5	0.134 896 2	0.108 092	0.152 535 8	0.202 543



D_1 D_2 D_3 D_4 D_5 D_6

图 4 6 个数据集上回报率指标

Fig. 4 The dynamic RoRs of senven algorithms on the six datasets

4 结束语

针对交易关键点预测问题,本文提出渐近学习算法 AL。通过辅助序列 R 设计上下文子序列生成算法,在子序列中,通过 LSTM 神经网络训练学习关键交易点的统计收敛性,并做出相应的交易决策。AL 算法仅在上下文子序列中的关键交易点进行交易,降低了预测交易的频率,从而降低手续费和交易风险。通过多次预测的收敛结果来确定交易关键点的相关信息,提高了算法的鲁棒性。实验结果表明,AL 算法相比传统基于对比的传统算法,减少了操盘次数,降低了风险,提高了交易收益。后续可以进一步研究新的交易关键点的收敛学习算法,比如根据收敛方式的不同继续设计不同的收敛算法,研究关键点的渐近属性与交易操作点的统计相关性。

参考文献

[1] CHANG P C, WU J L, LIN J J. A Takagi-Sugeno fuzzy model combined with a support vector regression for stock trading forecasting[J]. Applied Soft Computing, 2016, 38: 831-842.

[2] ALIMORADI M R, KASHAN A H. A league championship algorithm equipped with network structure and backward Q-learning for extracting stock trading rules [J]. Applied Soft Computing, 2018, 68: 478-493.

[3] BAO D, YANG Z. Intelligent stock trading system by turning point confirming and probabilistic reasoning[J]. Expert Systems with Applications, 2008, 34(1): 620-627.

[4] CHANG P C, LIAO T W, LIN J J, et al. A dynamic threshold decision system for stock trading signal detection[J]. Applied Soft Computing, 2011, 11(5): 3998-4010.

[5] TANG H, DONG P, SHI Y. A new approach of integrating piecewise linear representation and weighted support vector machine for forecasting stock turning points[J]. Applied Soft Computing, 2019, 78: 685-696.

[6] TSAI Y C, WU M E, SYU J H, et al. Assessing the Profitability of Timely Opening Range Breakout on Index Futures Markets[J]. IEEE Access, 2019, 7: 32061-32071.

(下转第 179 页)