# Combining Technical Trading Rules:

## *A Recurrent Reinforcement Learning Approach*

**Yufeng TAO**
Department of Statistics and Actuarial Science, HKU
Advisor: Dr. Philip L.H. Yu

Laidlaw Scholars
Undergraduate Research and Leadership Programme

香 港 大 學
THE UNIVERSITY OF HONG KONG

## Abstract

This study proposes a Rule-based Recurrent Reinforcement Learning (RbRRL) strategy for financial trading to bridge the power of data-driven machine intelligence and the insight from human interpretable industry practice. Technical trading rules from different classes are combined with regularized weighting approximated by counterfactual returns and neuron activation. Although the conventional recurrent reinforcement learning outperforms RbRRL in terms of cumulative capital gain, the latter shows lower volatility as well as higher interpretability.

## Introduction

Technical indicators have been used in the capital market to identify trading opportunities for more than a century. Nevertheless, a single trading rule may not be sufficient to predict the stock price trend accurately. Dr. Philip Yu and his team[1] proposed a Performance-Based Reward strategy (PRS) that combines the two most popular classes of technical trading rules: moving average (MA) and trading range break-out (TRB). Each component rule is assigned a starting weight, and a reward/penalty mechanism based on its recent profit is proposed to update the weighting over time.

However, this closely resembles the rationale of reinforcement learning: through trial and error to maximize cumulative rewards. Recurrent reinforcement learning (RRL) was first introduced by Moody, Wu, Liao, and Saffell [2] where a trading system was developed focusing on a single asset. Most recently, Almahdi and Yang [3] introduced a more generalized recurrent reinforcement learning approach on portfolio management. This study intends to reframe the PRS by RRL, which we later model as Rule-based Recurrent Reinforcement Learning (RbRRL), in the hope of capturing both the power of data-driven trading automation and the insight from human-designed industry practice.

## Methodology

### Component trading rules

MA is the mean of stock prices over a moving window of n days $\bar{p} = \frac{1}{n}\Sigma_{i=t-n+1}^{t} p_i$ where $t$ is the current trading day and $p_i$ is the close stock price on day $i$. In MA rules, there are two averages (long-period and short-period averages) over two moving windows of $m$ days and $n$ days, respectively, where $m > n$. Consider a trading day $t$, a MA rule initiates buy (sell) signal if the short-period moving average is above (below) the long-period moving average.

TRB calculates the highest close price $H = \max(p_{t-1}, p_{t-2}, \ldots, p_{t-n})$ and the lowest close price $L = \min(p_{t-1}, p_{t-2}, \ldots, p_{t-n})$ over a fixed $n$ days interval. The highest and lowest price form a running channel (trading range) for each day's stock price and the trading signals are invoked by the stock price's breakout from the channel. Suppose the close price of trading day $t$ is $p_t$, a buy signal is generated when $p_t > H$ and a sell signal is generated when $p_t < L$.

Figure 1 and 2 demonstrate the two technical indicators of stock Amazon (NASDAQ: AMZN) over 4000 trading days. We can see that SMA (5-day vs 150-day) signals much less frequently compared to TRB (20-day), representing different decision making rationales which we aim to combine.

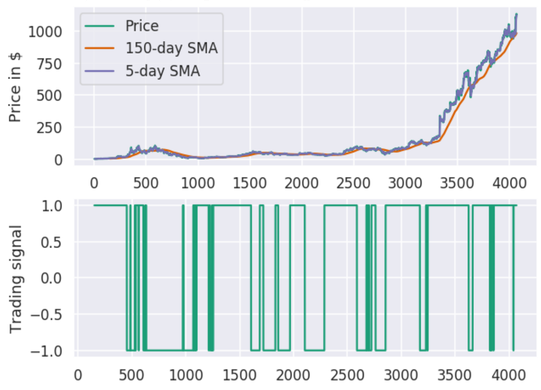


**Figure 1:** Simple Moving Average

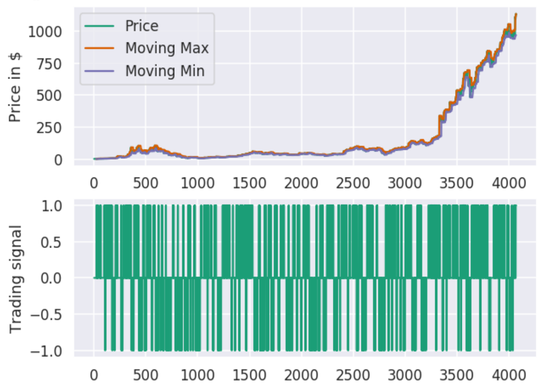


**Figure 2:** Trading Range Break-out

### Combined signal generation

#### *Counterfactual rule returns*

We measure the capital gain $r_{t,i}$ on each trading day $t$ compared to the day before if we executed the trading order following exactly the signals from each individual rule $i$.

$$r_{t,i} = \mu \cdot [S_{t-1,i} \cdot (p_t - p_{t-1}) - \delta |S_{t,i} - S_{t-1,i}|]$$

where $\mu$ is the maximum possible number of shares per transaction, $\delta$ denotes the transaction cost rate and $S_{t,i} \in \{long, neutral, short\} = \{1, 0, -1\}$ is the trader function of rule $i$.

#### *Signal weighting approximation*

The trader function in RRL resembles a neuron activated with the hyperbolic tangent function yielding outputs between -1 and 1. In connection to integrating multiple indicator signals, RbRRL assigns each component rule $i$ a weight $\omega_{t,i}$ at time $t$.

$$\omega_{t,i} = softmax[\langle \mathbf{v}, \mathbf{x}_{t,i} \rangle + b + u\omega_{t-1,i}]$$

$\langle \cdot, \cdot \rangle$ is the inner product, $\mathbf{x}_{t,i} = [r_{t-m,i}, \ldots, r_{t-1,i}] \in \mathbb{R}^m$ defines the feature vector of the current market condition at time $t$ for rule $i$, and $(\mathbf{v}, b)$ are the feature regression coefficients. We adopt the recent $m$ rule returns as the feature vector, so as to extract information of each rule by quantifying the counterfactual capital gains in the case that individual rules were used for trading. In addition to the features, another term $u\omega_{t-1,i}$, where $\omega_{t-1,i}$ stands for the rule weighting at time $t-1$, is also added into the regression to take the latest weighting into consideration. This term is supposed to discourage the agent from frequently changing the trading positions and, hence, to avoid heavy transaction costs.

#### *Regularization*

Simple linear combination of component signals $\Sigma_i \omega_{t,i} S_{t,i}$ like the one in PRS might allow many under-performing rules dominate the few truly useful rules due to adding many yet small weights. To address the possible noise, we only select the best rule (that is assigned the highest weighting) each time within each rule class and combine them across different classes.

$$S_t = \begin{cases} 1 & \text{if } \alpha \Sigma_{k \in C} \omega_t^k S_t^k + (1-\alpha) S_{t-1} > l \\ -1 & \text{if } \alpha \Sigma_{k \in C} \omega_t^k S_t^k + (1-\alpha) S_{t-1} < s \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where $\omega_t^k$ and $S_t^k$ represents the weight and signal of the best rule within class $k$ at time $t$, respectively. In particular, $\omega_t^k$ has to be re-balanced through $softmax$ of the original $\omega_{t,i}$. A new decay parameter $\alpha \in [0, 1]$ is introduced to incorporate the previous trading decision for cost-reducing purpose. Two threshold parameters $l \in [0, 1)$ and $s \in (-1, 0]$ are utilized to discretize the signal and filter out possible noises.

### Optimization

- Parameters: $\{\Theta, \alpha, l, s\}$, where $\Theta = \{\mathbf{v}, b, u\}$
- Objective (reward) function: Sharpe ratio, where $R_t = \mu \cdot [S_{t-1} \cdot (p_t - p_{t-1}) - \delta |S_t - S_{t-1}|]$ is the actual stock return traded based on the RbRRL combined signal. It essentially identifies investment strategies with less volatile profit.

$$Sharpe_T = \frac{Average(R_t)}{Standard\ deviation(R_t)} \qquad \text{for } t \in \{1, 2, \ldots, T\}$$

- Algorithms experimented: particle swarm optimization, gradient descent with automatic differentiation, random optimization (direct search) with clipped boundaries.

## Results

Comparing the automated trading decisions and returns side by side shows that overall RRL still outperforms RbRRL and the buy-and-hold strategy. The discretized action policy by RbRRL in 6 could miss out on plausible trading opportunities of different lot sizes. The reasonable performance in the training set considering transaction costs turns out to be overfitting as suggested by the poor testing set outcome 8. However, figure 3 and 4 display that despite its higher overall gain, RRL brings greater volatility on a daily basis.

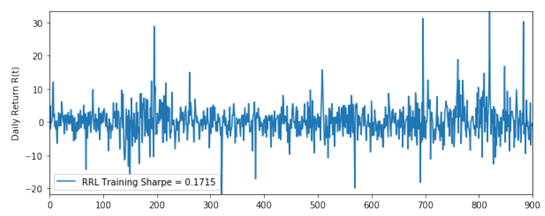


**Figure 3:** RRL Daily Return

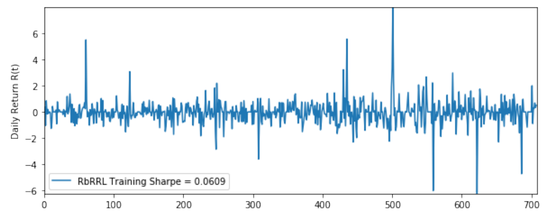


**Figure 4:** RbRRL Daily Return

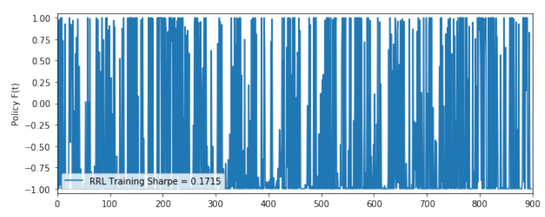


**Figure 5:** RRL Trading Policy

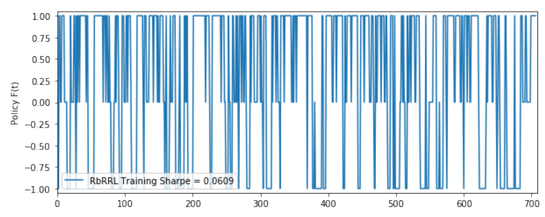


**Figure 6:** RbRRL Trading Policy

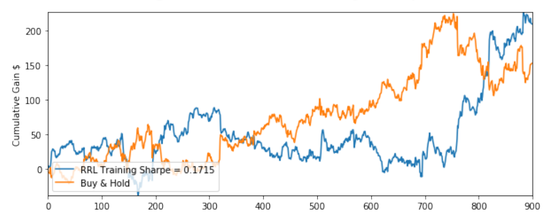


**Figure 7:** RRL Cumulative Gain

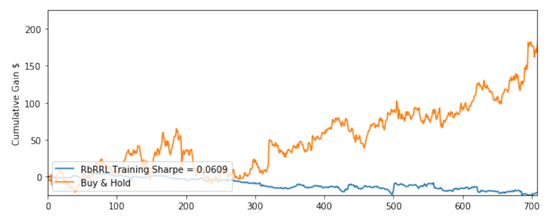


**Figure 8:** RbRRL Cumulative Gain

## Discussion and future work

It is not a full-fledged implementation because the model was only run on a single stock AMZN over a fixed period and parameters together with optimization could be better tuned. Financial markets are not necessarily stationary and thus the selected training period may not correctly reflect the value of both models.

If the RbRRL is to be extended to a portfolio, we could first try the following trading logic: each stock is traded independently with equal initial portfolio weights and no mutual funding is allowed as that in [1]. Furthermore, stock selection (e.g. by low correlation), risk diversification and trading frequency control could be considered.

Since RRL is in fact a one-layer neural network [4], deep learning techniques specifically recurrent neural network could be worth exploring. Nonetheless, one caveat is that efforts to increase the model interpretability such as this study trying to bridge machine intelligence and human insight should not be underestimated in practice.

## Acknowledgement

Special thanks to Dr. Philip L.H. Yu who provided insightful ideas and patient guidance.

### *GitHub Link to Code*

`https://github.com/AndyYFTao/RbRRL-strategy-for-financial-trading/blob/master/RbRRL.ipynb`

## References

[1] Wang, F. , Philip, L. , Cheung, D. W. *Combining technical trading rules using particle swarm optimization*. Expert Systems with applications, 41 (6), 3016–3026, 2014.

[2] Moody, J. , Wu, L. , Liao, Y. , Saffell, M. *Performance functions and reinforcement learning for trading systems and portfolios*. Science, 17 (February 1997), 441–470, 1998.

[3] S. Almahdi, S. Y. Yang. *A constrained portfolio trading system using particle swarm algorithm and recurrent reinforcement learning*. Expert Systems With Applications, 130:145–156, 2019.

[4] Y. Deng, F. Bao, Y. Kong, Z. Ren and Q. Dai. *Deep Direct Reinforcement Learning for Financial Signal Representation and Trading*. IEEE Transactions on Neural Networks and Learning Systems vol. 28, no. 3, pp. 653-664, March 2017.