機器學習第二次報告

陳慧如

Department of Applied Mathmatics
National Chung Hsing University
South District, Taichung City 40227, Taiwan
luluchenyo@smail.nchu.edu.tw

I. Deng, Y., Bao, F., Kong, Y., Ren, Z., & Dai, Q. (2017). Deep Direct Reinforcement Learning for Financial Signal Representation and Trading. IEEE Transactions on Neural Networks and Learning Systems, 28(3), 653-664. doi: 10.1109/tnnls.2016.2522401 Retrieved fromhttps://ieeexplore.ieee.org/document/7407387

本篇的目的為透過歷史股價資料，做出交易決策，並透過極大化總報酬的過程，得到最佳決策函數。模型以歷史股價資料 $\{P_1, P_2, ..., P_t\}$ 當作輸入，輸出為第 t 期的交易決策 $\delta_t(\theta)$，模型分成三個部分：第一部分為 Fuzzy 轉換，先將股價前後期取差，利用 K-means 方法分成上漲趨勢、趨勢不明以及下跌趨勢三種，目的為取得每一期相較於前一期的股價，隸屬於不同趨勢的程度；第二部分利用四層的 DNN 萃取出趨勢的特徵，在這裡會利用 auto-encoder 的手法取得初始化參數；第三部分為一層的 RNN，使得每一期的決策都可以考慮到過去做過的所有決策。

A. 主要模型

在 fuzzy 轉換的部分，輸入為歷史股價資料，輸出為每一個時刻隸屬於三種趨勢的程度。首先會將輸入的歷史股價資料轉換成每一期的價差 $f_t = \{Z_1, Z_2, ..., Z_t\}$：

$$Z_t = P_t - P_{t-1} \tag{1}$$

接著使用 K-MEANS 方法將每一期的價差分成漲、跌以及趨勢平緩三群，得到每一群的 μ_i、σ_i；最後將每一期的價差分別代入不同趨勢的 fuzzy 函數，得到每個價差分別隸屬於三種趨勢的程度：

$$v(f_t) = e^{-(f_t - m_i)}, \quad i = \{1, 2, 3\} \tag{2}$$

DNN（$g(\bullet)$）部分為四層全連接層：

$$a_i^l = \langle w_i^l, o^{l-1} \rangle + b_i^l \tag{3}$$

每層的神經元數為 128 個，輸出為 20 維的特徵 F_t，activation function 皆使用 sigmoid：

$$o_i^l = \frac{1}{1 + e^{-a_i^l}} \tag{4}$$

最後一部分為一層的 RNN：

$$\delta_t(\theta) = tanh[\langle w, F_t \rangle + b + u\delta_{t-1}(\theta)] \tag{5}$$

整個模型架構為 Fig. 1

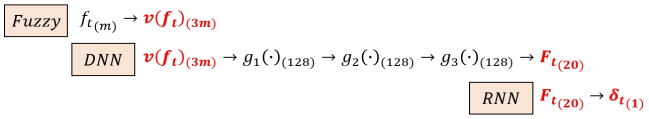

Fig. 1: FDRNN

B. 初始化參數

在 DNN 部分，用 auto-encoder 的手法，先讓輸入層接到 DNN 的第一層（中間層），得到 θ_1，再接到一個模擬輸入層的輸出層（Fig. 2）得到 γ，透過式 (6) 找到可以保留住最多重要特徵的 θ_1，作為第一層的初始化參數：

$$\arg\min_{\theta} \|X_t^{(l)} - h_\gamma(h_\theta(X_t^{(l)})\|_2^2 + \eta\|w^{(l+1)}\|_2^2 \tag{6}$$

接著固定住第一層的初始化參數，從第一層出發，以第二層當作中間層，重複上述過程到找完四層的初始化參數為止。

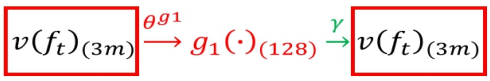

Fig. 2: AE

C. Reward Function

模型輸出 $\delta_t(\theta)$，決定該期持有何種交易部位，$\delta_t(\theta) \in \{long, neutral, short\} = \{1, 0, -1\}$，透過極大化總報酬的過程，得到最佳的決策函數：

$$\delta_t(\theta) = tanh[\langle w, g(v(f_t)) \rangle + b + u\delta_{t-1}(\theta)] \tag{7}$$

$$R_t = \delta_{t-1}(\theta)Z_t + c|\delta_t(\theta) - \delta_{t-1}(\theta)| \tag{8}$$

$$\arg\max_{g(\bullet), w, b, u} \sum_t R_t \tag{9}$$

D. 實作方向

以在 yahoo! finance 取得的 IF（2889 筆）以及台灣 50（2929 筆）股價日資料，各取 2000 筆訓練模型，並利用 2001 筆之後的股價測試模型表現。

II. Jeong, G., & Kim, H. Y. (2019). Improving financial trading decisions using deep Q-learning: Predicting the number of shares, action strategies, and transfer learning. Expert Systems with Applications, 117, 125-138. doi: 10.1016/j.eswa.2018.09.036 Retrieved fromhttps://www.sciencedirect.com/science/article/pii/S0957417418306134

本篇目的為透過歷史股價決定交易動作以及買賣數量，相較於上一篇，本篇的模型可以根據市場情況的不同，決定要投資的數量多寡。使用的模型為 DQN，動作及數量的 Q 值分別有各自的 TdError 做更新：

$$Q_{action}(S_t, a_t) := Q_{action}(S_t, a_t) + \alpha \times [R_t + \gamma Q_{action}(S_{t+1}, a_t) - Q_{action}(S_t, a_t)] \quad (10)$$

$$R_{num}(S_t, a_t) := R_{num}(S_t, a_t) + \alpha \times [R_t + \gamma R_{num}(S_{t+1}, a_t) - R_{num}(S_t, a_t)] \quad (11)$$

$$R_t = num_{t-1}(\theta) \times (1 + a_{t-1}(\theta) \times \frac{P_t - P_{t-1}}{P_{t-1}}) \frac{P_{t-1}}{P_{t-200}} \quad (12)$$

模型以歷史股價資料當作輸入，輸出為交易動作的價值以及數量的比例，決策方面選擇當期 Q 值最大的動作：

$$a_t(\theta) = \arg\max_{a_t(\theta)} Q_{action}(S_t, a_t(\theta)) \quad (13)$$

投資數量為數量比例乘上投資張數的上限，本篇將此上限定義為 10 張：

$$num_t(\theta) = R_{num}(\theta) \times 10$$
$$where\ a_t^*(\theta) = \arg\max_{a_t(\theta)} Q_{action}(S_t, a_t(\theta)) \quad (14)$$

另外，本篇為了解決數據不足造成的 overfitting，利用成分股對股市指標進行 transfer learning。

A. 主要模型

1) NumQ: 第一個模型叫做 NumQ（Fig. 3），以歷史股價資料 $f_t = \{P_1, P_2, ..., P_t\}$ 為輸入，首先會經過三個全連接層，在第四層的時候將交易動作以及交易比例分別獨立出來：

$$fc1 = Relu(W_1 \times f_t + b_1) \quad (15)$$

$$fc2 = Relu(W_2 \times fc1 + b_2) \quad (16)$$

$$fc3 = W_3 \times fc2 + b_3 \quad (17)$$

$$Q_{action}(S_t, a_t) = W_4 \times Relu(fc3) + b_4 \quad (18)$$

$$R_{num}(S_t, a_t) = Softmax[W_4' \times Sigmoid(fc3) + b_4'] \quad (19)$$

2) NumDReg– AD: 第二個模型叫做 NumDReg–AD（Fig. 4），大致上與第一個模型相同，主要差異為第二個模型在第一層之後就將交易動作以及交易比例分別做訓練，以及使用了減少的神經元數以避免 overfitting。

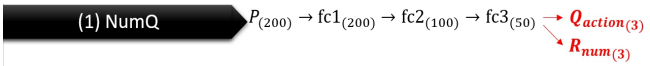

Fig. 3: NumQ

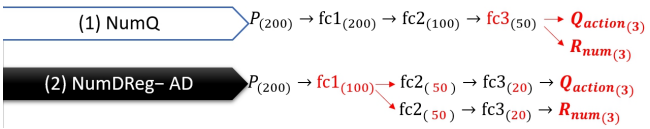

Fig. 4: NumDReg– AD

3) NumDReg– AD3: 由於第二個模型結構較為複雜，將第二個模型分三個步驟訓練，第一步使用第一個模型並以減少的神經元數做 pretrain，第二步將取得的參數固定在交易動作的分支上，而後對交易數量的分支做 pretrain（Fig. 5），第三步以 end-to-end 的方式做訓練。

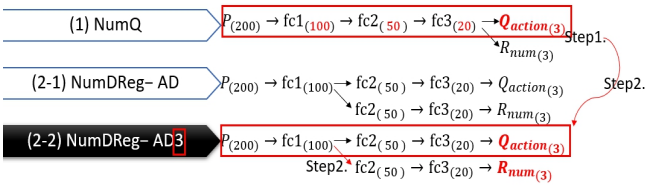

Fig. 5: NumDReg– AD3

4) NumDReg– ID3: 將第二個模型在交易數量部分的輸出由三個改為一個，即為第三個模型（Fig. 6），目的是使得交易數量不受不同的交易動作影響。

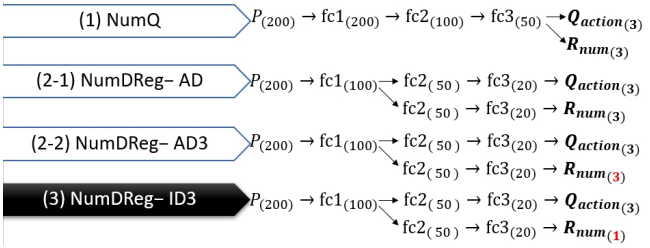

Fig. 6: NumDReg– ID3

B. Transfer Learning

本文利用了成分股 pretrain 模型，以解決股市指標價格數據不足的問題。做法為先將各個成分股與股市指標的相關性計算出來，並依照相關性高低分組，接著將每一組對模型進行 pretrain，選出 total profit 最高的一組作為 transfer learning 使用的 data。為了測試此方法的有效性（以 total profit 為衡量基準），本文使用了四種股市指標（S&P500、KOSPI、HSI 以及 EUROSTOXX50）來進行實驗。

C. Reward Function

本文 [2] 參考了 Wang et al. (2017)[3] 在使用 QLearning 進行交易時所定義的 R_t 以及 $profit_t$，在 reward

function 上考慮了實際報酬以及長期趨勢：

$$R_t = (1 + a_{t-1}(\theta) \times \frac{P_t - P_{t-1}}{P_{t-1}}) \frac{P_{t-1}}{P_{t-200}} \tag{20}$$

$$profit_t = a_{t-1}(\theta) \times \frac{P_t - P_{t-1}}{P_{t-1}} \tag{21}$$

本文為了比較考慮投資數量後的差異，直接在 Wang et al. (2017)[3] 的 reward function 乘上投資數量作為本文的 reward function：

$$R_t = num_{t-1}(\theta) \times (1 + a_{t-1}(\theta) \times \frac{P_t - P_{t-1}}{P_{t-1}}) \frac{P_{t-1}}{P_{t-200}} \tag{22}$$

$$profit_t = num_{t-1}(\theta) \times (a_{t-1}(\theta) \times \frac{P_t - P_{t-1}}{P_{t-1}}) \tag{23}$$

D. 實作方向

以在 yahoo! finance 上取得的 S&P500（23086 筆）、KOPSI（5521 筆）、EUROSTOXX50（8262 筆）以及 HSI（8130 筆）進行實驗。另外，本文 [2] 在 reward function 上的動作意義（action），與參考的 Wang et al. (2017)[3] 所定義的決策意義（position）有些出入，在 reward function 方面，以作者定義的動作來說，買進下一期有上漲的股票會得到 reward 是沒有問題的，但在計算 total profit 的部分，這一期的買進數量乘上下一期的漲跌幅，並不完全等於實際上得到的報酬。故在評估 transfer learning 是否成功以及模型績效時，我把 $profit_t$ 統一以式 (24) 為基準，預計將進行 (1) 決策意義為 position（不考慮數量）、(2) 決策意義為 position（考慮數量）以及 (3) 決策意義為 action（考慮數量）三種組合的比較。

$$profit_t = position_{t-1}(\theta) \times \frac{P_t - P_{t-1}}{P_{t-1}} \tag{24}$$

References

[1] Deng, Y., Bao, F., Kong, Y., Ren, Z., & Dai, Q. (2017). Deep Direct Reinforcement Learning for Financial Signal Representation and Trading. IEEE Transactions on Neural Networks and Learning Systems, 28(3), 653-664. doi: 10.1109/tnnls.2016.2522401 Retrieved fromhttps://ieeexplore.ieee.org/document/7407387

[2] Jeong, G., & Kim, H. Y. (2019). Improving financial trading decisions using deep Q-learning: Predicting the number of shares, action strategies, and transfer learning. Expert Systems with Applications, 117, 125 - 138. doi: 10.1016/ j.eswa. 2018.09.036 Retrieved fromhttps://www.sciencedirect.com/science/article/pii/S0957417418306134

[3] Wang Y., Wang D., Zhang S., Feng Y., Li S., Zhou Q. (2017) *DeepQ − trading* Retrieved from http://cslt.riit.tsinghua.edu.cn/mediawiki/images/5/5f/Dtq.pdf