DOI: 10.48011/asba.v2i1.1108

Image Representation of Time Series for Reinforcement Learning Trading Agent

Guinther K. da Costa[1], Leandro dos S. Coelho[1,2] Roberto Z. Freire[1]

1 - Pontifical Catholic University of Parana (PUCPR), Industrial and Systems Engineering Graduate Program (PPGEPS) Rua Imaculada Conceição, 1155, 80215-901 Curitiba, PR – Brazil (guintherk14@gmail.com; roberto.freire@pucpr.br)
2 - Department of Electrical Engineering (DEE/PPGEE), Federal University of Parana (UFPR), Polytechnic Center, 81531-980 Curitiba, PR – Brazil (leandro.coelho@pucpr.br)

Abstract: The availability of diverse data has increased the demand for expertise in algorithmic trading strategies. Reinforcement learning has shown interesting applicability in a wide range of tasks, especially in some challenging problems as trading, where slow model convergence, inference speed, and reduced model accuracy appear as barriers in this type of application. In this paper, we propose the transformation of time series into images considering a transfer learning based on a semi-supervised model with deep Q learning agents, where labels were generated by an evolutionary algorithm to improve both training speed and performance measures.

Resumo: A disponibilidade de dados diversos aumentou a demanda por conhecimento em estratégias de trading algorítmico. O aprendizado por reforço mostrou aplicabilidade interessante em uma ampla gama de tarefas, especialmente em alguns problemas desafiadores como negociação, onde a convergência lenta de modelos, velocidade de inferência e precisão reduzida do modelo aparecem como barreiras nesse para esse tipo de abordagem em ambientes reais. Neste artigo, propõe-se a transformação de séries temporais em imagens considerando o uso de transferência de aprendizado baseado em um modelo semi-supervisionado com agentes de aprendizado profundo Q por reforço, onde os rótulos foram gerados por um algoritmo evolutivo para melhorar a velocidade do treinamento e as medidas de desempenho.

Keywords: stock trading; algorithmic trading; financial market; machine learning, reinforcement learning; time series forecasting.

Palavras-chaves: negociação de ações; negociação algorítmica; mercado financeiro; aprendizado de máquina, aprendizado por reforço; previsão de séries temporais.

1. INTRODUCTION

Algorithmic trading in stocks is attracting the attention of specialists in machine learning, as financial area researchers and market practitioners are considering recent advances for automatic or supported decisions. The problem in decision-making is to learn feature representation from non-stationary and noisy financial time series.

Machine Learning (ML) techniques are being used for market trading. Gerlein et al. (2016) conducted analyses on the role of simple machine learning models to achieve profitable trading through a series of trading simulations in the FOREX Market. The researchers discussed how a combination of attributes in addition to technical indicators as predictors are used to enhance the classification capabilities. Xiaodong et al. (2016) presented the design and architecture of a trading signal mining platform that employs extreme learning machine (ELM) to make stock price prediction, based on news (text) and quantitative data concurrently.

Vora et al. (2017) presented a review of some machine learning approaches (classification and regression) to make stock predictions and a comparison of programming languages for the applications, and Weng et al. (2017) used three models (decision trees, neural networks and support vector machines) to show that diversifying the knowledge base by combining quantitative and disparate sources data can help improve the performance of financial expert systems.

Alessandretti et al. (2018) used two gradient boosting decision trees (GBM) and one long and short-term memory networks (LSTM) to make daily forecasts of 1,681 cryptocurrency prices and results find that all of the three models perform better than a baseline ‘simple moving average’. Macchiarulo (2018) made a comparison on whether machine learning or technical analysis best predicts the stock Market using 20 years of stock market data and concludes at the 99% confidence level, machine learning outperformed compared to all but the buy and hold technical analysis method in the up-market period, but underperformed at the low-market period.

With the objective of outperforming baselines models in the context of trading, Zhang et al. (2019) used Deep Q-learning Networks (DQN), Mnih et al. (2013) considered the Double Deep Q-Learning DDQN, Hasselt et al. (2015) the Policy Gradients (PG), Sutton et al. (1999) the Advantage Actor-Critic (A2C) Konda and Tsitsiklis (1999).

Pengfei Y. & Xuesong Y. (2020) used financial product price data treated as a one-dimensional series generated by the projection of a chaotic system composed of multiple factors

into the time dimension, and the price series was reconstructed using the time series phase-space reconstruction (PSR) method with an LSTM, and this approach provided higher accuracy than baseline methods used in trading. Lei et al. (2020) proposed a time-driven feature-aware jointly deep reinforcement learning model (TFJ-DRL) that integrates deep learning and reinforcement learning models to improve the financial signal representation learning and action decision-making in algorithmic trading, showing robust results.

Modern portfolio theory (Markowitz, 1952) implies that, given a finite time horizon, an investor chooses actions to maximize some expected utility of final wealth:

$$E[U(W_T)] = E[U(W_0 + \sum_{t=1}^{T} \delta W_T)], \quad (1)$$

where U is the utility function, W_T is the final wealth over a finite horizon T, and δW_T represents the change in wealth. The value of the final wealth measure relies upon sequences of interdependent actions where optimal trading decisions do not decide just immediate trade returns but also affect subsequent future returns, thus, has a long-term dependency. As mentioned in (Merton, 1969), this falls under the framework of optimal control theory (Kirk, 2012), and forms a classical sequential decision-making process. If the investor is risk-neutral, the utility function becomes linear and we only need to maximize the expected cumulative trades returns $\mathrm{E}(\sum_{t=1}^{T} \delta W_T)$ and we observe that the problem fits exactly with the framework of Reinforcement Learning (RL), where the goal is to train a return-maximizing agent in an uncertain and dynamic environment that has much in common with an investor or a trading strategy that interacts with markets. This situation relies on the availability of an environment, which in this case, is the availability of historic trading data itself.

The Deep Reinforcement Learning approach has been successfully applied to game-playing agents, most prominently to the game of Go (Silver et al, 2013), but also complex video games (Mnih et al., 2012), which indicated the ability of these models to deal with constantly changing Environments.

The key contributions of this paper are: i) to consider multidimensional time series as images for reinforcement learning tasks; ii) to assume Differential Evolution (DE) to optimize the parametric rule-based strategy to label data and further train a model with these labels; and iii) to use transfer learning from this supervised trained model to improve training time and convergence in a Deep Q Learning model.

The next section of this paper describes the trading problem details and different strategies. In the sequence, methods and techniques are presented in detail. Section 4 presents the preliminary results of the proposed method. Finally, the conclusion and future works are addressed in Section 5.

2. THE TRADING PROBLEM

The availability of data and accessible Application Programming Interfaces (APIs) have increased the demand for expertise in algorithmic trading strategies. Based on computer programs to automate trading, some common tasks are portfolio management (Chandrinos et al., 2018), where the goal is to optimize the amount of capital allocated in each operation keeping some balance between risk and returns; forecasting (Lim et al., 2019), where the goal is to directly predict the value of an asset in the future and use it to make decisions; and finally, testing and strategies evaluation (Jansen, 2018) where the goal is to make the correct evaluation of a method in a backtest to assure that it will provide reasonable performance in real applications.

The financial crises of 2001 and 2008 have affected how investors approach diversification and risk management, stimulating low-cost passive investment vehicles in the form of exchange-traded funds (ETFs). Amid low yield and low volatility cost-conscious, investors shifted 2 trillion dollars from actively managed mutual funds to passively managed ETFs. According to Jansen (2018), algorithmic trading is applied in a wide range of time scales and distinct data sources.

In High-Frequency Trade (HFT), orders are executed with extremely low latency, in the microsecond range, holding positions for short periods. The goal is to detect and exploit inefficiencies in the market microstructure. Aiming to earn small profits per trade, HFT considers both passive or aggressive strategies. In Day Trading, high volatility is expected, and the time interval of operation varies from minutes to hours, exploiting momentum and sentiment factors to design trading strategies.

In Swing Trade, usually, a hybrid of fundamental and technical (candle shape) analysis is used, and the operations happen in a week frame. Finally, in Position Trade, economic and political scenarios are evaluated, and the positions are held in the range of months. This work focus on operations in the range of hours, thus, Day Trade is the kind of operation it is suited for.

3. METHODS

This section addresses the techniques assumed for the trading problem. The first subsection presents concepts of Reinforcement Learning (RL) and how indicators were converted into images. In the sequence, transfer learning concepts are presented.

3.1 Reinforcement Learning

As presented by Sutton and Barto (1998), RL is a prevalent self-taught learning paradigm that was developed to solve the Markov decision problem (Tesauro, 1994).

The current literature on RL in trading can be categorized into three main methods: critic-only, actor-only, and actor-critic approach (Fischer, 2018). The critic-approach, mainly DQN, is the most published method in this field (Bertoluzzo and Corazza, 2012; Jin and El-Saawy, 2016; Tan et al., 2011; Huang, 2018; Ritter, 2017), where a state-action value function, Q, is constructed to represent how good a particular action is in a state.

Recently, a combination of Reinforcement Learning (RL) and Deep Learning (DL) provided interesting results in a large variety of problems, showing applicability for financial tasks. There is a large variety of models that combine DL and RL, to cite some not previously cited: Asynchronous Advantage

Actor-Critic (A3C) (Mnih et al., 2016), Proximal Policy Optimization Algorithms (PPO) (Schulman et al., 2017), Trust Region Policy Optimization (TRPO) (Schulman et al., 2015), Deep Deterministic Policy Gradient (DDPG) (Lillicrap et al., 2015), Twin Delayed Deep Deterministic Policy Gradient (TD3) (Fujimoto et al., 2018), Soft Actor-Critic (SAC) (Haarnoja et al. 2018), 51-atom agent (C51) (Bellemare et al. 2017), Regression DQN (QR-DQN) (Dabney et al., 2017), and Hindsight Experience Replay (HER) (Andrychowicz et al., 2017). A more visible relation and aspects of those models can be seen in figure 1.

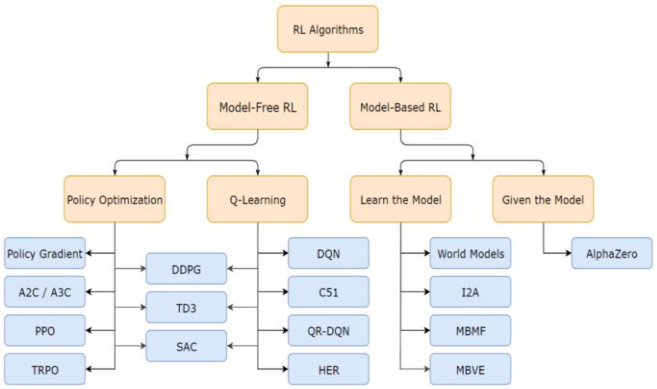

Fig. 1: Map of Reinforcement Learning models.

In this work, we choose to use Deep Q-Learning (DQN), which is an approach that uses Deep Learning to evaluate the quality of a predefined Action, providing information about the environment.

In the context of trading, the usage of time series as images has shown potential in (Cohen et al., 2018), where a classification approach was used. Based on the Elliott Waves theory (Elliott et al., 1994) we propose a combination of three concepts: RL, Evolutionary Algorithms (EA) – Differential Evolution Algorithm proposed by Storn and Price (1997) – and the transformation of time series into images. These concepts were assumed to propose a model for trading. In this way, state, action, reward, and environmental aspects are addressed in the sequence, followed by a model description.

3.1.1 State

The state is the respective situation of the agent in the environment. Usually, four indicators are assumed to describe the state of the model: price, volume, relative strength index (RSI), and moving average convergence divergence (MACD). Price is the mean between low and high from the past period (candle). Volume is the sum of exchanged values given by

$$\begin{aligned} MA[i] &= \sum_{i}^{N} \frac{Y[i]}{N}, \\ MACD &= MA[i] - MA[i+H], \\ SIGNAL &= MA[j] : i < j < i+H. \end{aligned} \tag{2}$$

where MACD is the difference from two moving averages of different periods and the third one with a period among them. From Eq. 1, MA is the moving average, Y is the price time series, N is the window size, and H is the slower MA window size. The RSI estimates the acceleration of the price movement and gives indications about price movement changes of direction, with the concept that price should slow down before changing direction being

$$RSI = 100 - \left[\frac{100}{1 + \frac{\alpha}{\beta}}\right], \tag{3}$$

where α and β represent the average percentual loss and gain, respectively. After obtaining these indicators, the time series of features are transformed into images of 64×64 pixels and concatenated into 4×64×64, as each one of the indicators can be stated as an image channel (Figure 2).

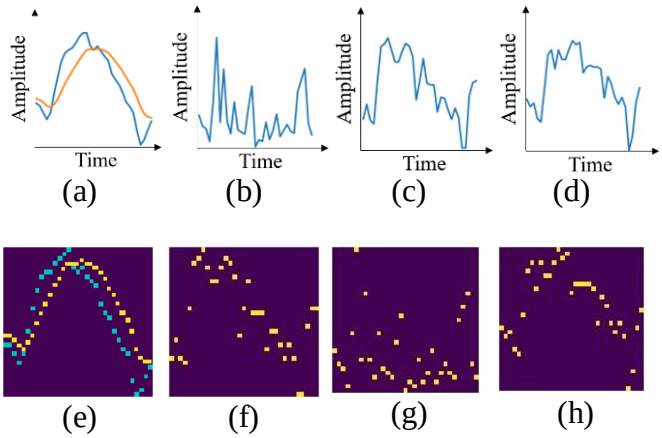

Fig. 2: Conversion of time series to images: (a) and (e) MACD; (b) and (f) RSI; (c) and (g) volume; (d) and (h) price.

3.1.2 Action

The action represents the agent movement at each time-step following some policy to increase rewards. By considering the action space restricted to buy, sell, or hold, it is assumed a predefined size for the order placement. The buy orders turn into a first-in, first-out (FIFO) queue, with a predefined limit of entry orders.

3.1.2 Reward

The reward is the signal which assesses the quality of the agent's actions. Classical trading strategies often use the concept of Alpha (Jansen, 2018). It is important, for any trading strategy, to find optimal enter and exit moments, both maximizing returns. Assuming return as a reward showed no convergence as the model is often encouraged to take no actions as a way to minimize loss, so we applied a weighted evaluation to each action taken. A negative reward is applied if the model chooses to hold (wait). Again, if low weight is applied to this negative reward, the model fast converges into no action to minimize the loss of a bad decision. But if a too high value is used, the quality of actions decay, as there is not enough difference between a random decision and no action. This situation can lead to non-optimal sell time by early decision to avoid the negative reward of waiting. Another aspect considered is the time of operation between buying and selling, as it something that is commonly used to stop real operations. Based on these assumptions, the following rules were stated:

$$\begin{cases} R_{buy} = R - 50 \\ R_{sell} = \begin{cases} R + R_a + R_a * \Delta P + (t_s - t_b) * \Delta P: if\ \Delta P > 0 \\ R - R_a + R_a * \Delta P + (t_s - t_s) * \Delta P: if\ \Delta P < 0 \end{cases}, \\ R_{hold} = R - R_h \end{cases} \quad (4)$$

where R is the cumulated previous reward, R_a is the action reward (equal to 100), R_h is the hold reward (equal to 2), ΔP is the profit, t_b is the buy instant, and t_s is the sell instant. Both R_a and R_h were trial-and-error defined.

3.1.2 Environment

In this study, it was assumed that models' actions do not affect the prices, and no trade fees were considered. The dynamic aspects of the environment are the financial balance and the financial signals (price, volume and indicators) of assets, which have very non-linearity and non-stationary behaviors. For real-world applications, other aspects can be stated as the delay of order placement, no market maker to fill the order, and false triggers made by other agents.

2.3 Model

DQL algorithms estimate the action-value function $Q(s,a)$, which indicates the quality of an action a, giving a State s, for a sequence of actions and observed rewards. At each time-step, the agent selects an action from $A = \{1,\dots,K\}$. The algorithm uses an aprioristic environment model to learn based on experience or estimates the rewards from $Q(s,a)$ without considering a model. Often, the sequence of state actions is too large for a discrete function approximator. In this case, the neural network is used to approximate the $Q(s,a)$ function trough a reward function that estimates the optimal action based on the Bellman's equation:

$$Q^*(s,a) = E(R_{t+1} + \rho Max_{a'} Q(s,a)), \quad (5)$$

where E is the expectation of an R_{t+1} reward plus a ρ discount factor times the $Max\ Q(s,a)$ quality state reward function. The goal of the agent is to find the optimal policy giving the set of actions in the environment. The model in this work was trained using an adaptation of the Deep Q-learning with Experience Replay algorithm from the study presented by Mnih et al. (2013), with a transfer learning approach, which will be discussed in the next section.

3.2 Transfer Learning

Taking into account that RL agents usually take several thousands of episodes to converge, and maybe do not converge at all (Lim et al., 2019), in this study, we propose the use of the evolutionary algorithm DE to optimize one parametric rule-based strategy. which has access to the whole dataset in time series format and is responsible for labeling each t instant as buy, sell, or hold, to maximize the differences of subsequent buys and sells using the following set of rules:

$$\begin{cases} if: \sum_{i}^{N} P_i X_i[t] > P_{i+1} : BUY \\ if: \sum_{j}^{M} P_j X_j[t] > P_{j+1} : SELL \end{cases}, \quad (6)$$

where X_i is the indicator i at time t.

The previous 64 samples of price, volume, MACD, and RSI, before each label (buy, sell or hold) defined by DE are transformed into images and used to train a supervised classification model. The model is a Convolutional Neural Network (CNN) with the same architecture as the one in the RL agent $Q(s,a)$. The proposed model is described in Figure 3, where both the evolutionary and supervised steps are executed previously to the RL (Q approximator) train loop in the environment.

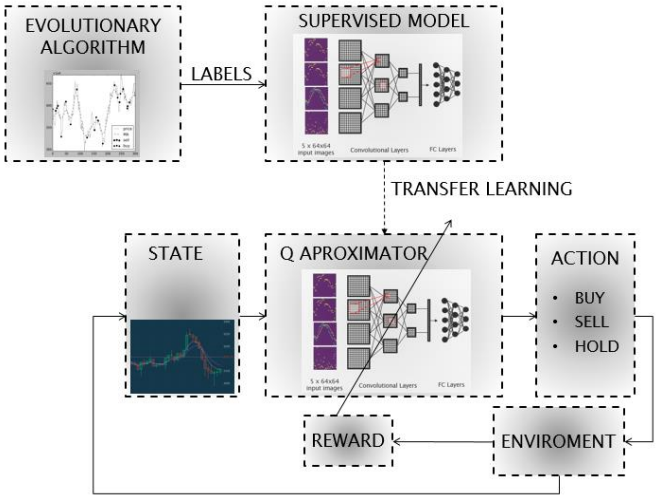

Fig. 3: Hybrid model proposed for trading.

4. EXPERIMENTS

The experiments assumed to validate the proposed model consisted of both training and testing phases. The agent was trained using hourly data from 200 stocks, with 460 days each, from The New York Stock Exchange (NYSE). The parameters of the EA were standardized for all models and were not the subject of study associated with this research. In this way, it was assumed 10 individuals multiplied by the number of parameters, 10 generations, 25% of recombination factor, and mutation factor equal to 0.5, using the best $best2exp$ strategy. For the CNN, 3 convolutional layers were assumed, paddling the image into 64×64, 32×32, and 16×16, with kernel sizes of 9×9, 6×6, 3×3, with 124, 64, and 32 kernels in each layer. At the top, a Multilayer Perceptron with 64, 32, and 8 neurons in each layer was considered. The RL agent was trained using memory replay size equal to 32 steps, with γ equal to 0.95, initial ε equal to 1, and three different decay values (φ).

Finally, 7,800 stocks of NYSE were assumed to test the model considering 2,000 episodes. Assuming that the agent has always an available amount of US$ 100.00 whenever he

decides to buy. The return for each one of the stocks can be verified in Figure 4.

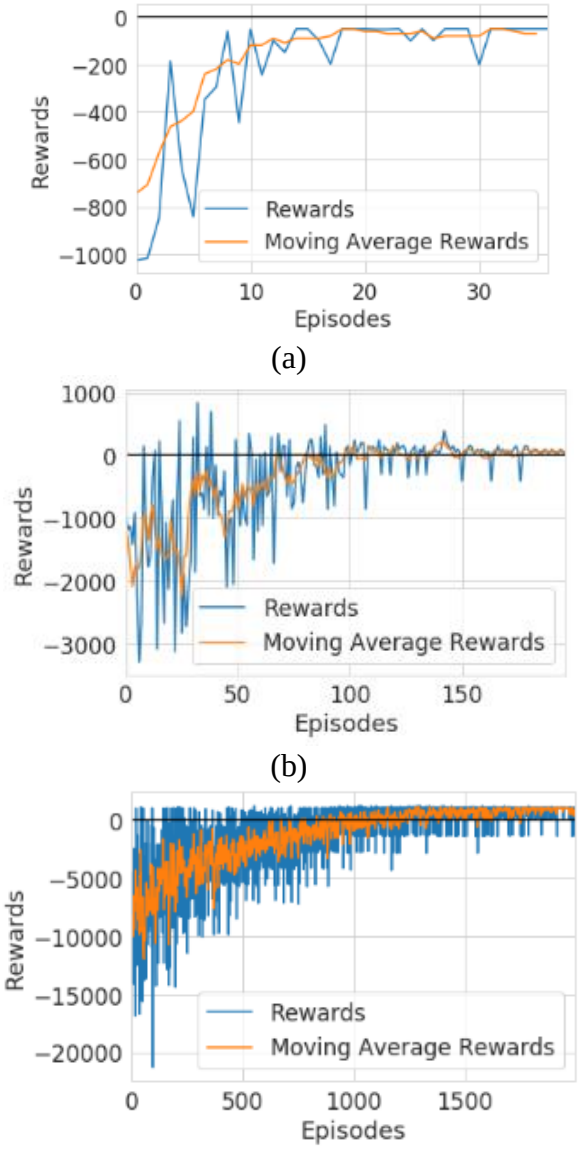

Fig. 4: Results: (a) $\varphi = 0.9991$ in 35 episodes; (b) $\varphi = 0.9996$ in 200 episodes; (c) $\varphi = 0.99996$ in 2,000 episodes.

The final cumulative profit was calculated in comparison to the available US$ 100.00. In 4,126 stocks, the return was negative, in 2,351, the agent took no actions, and in 1,323 it was positive. The maximum return was 330%, and the worst return was −152, as shown in Figure 5.

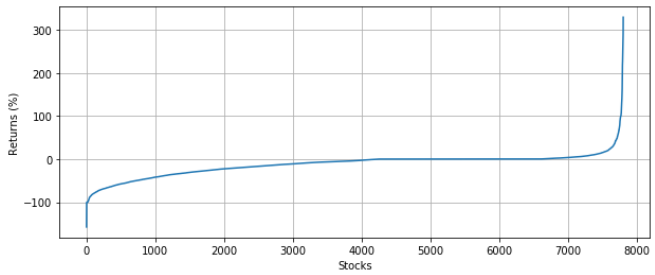

Fig. 5: Agent returns (%) over 7,800 NYSE stocks in backtesting.

ACKNOWLEDGMENT

The authors thank the support from the National Council for Scientific and Technological Development (CNPq – Grant # 304783/2017-0) for funding this research.

REFERENCES

Gerleinac, E.A., McGinnityab, M., Belatrechea. (2016) Evaluating machine learning classification for financial trading: An empirical approach. *Expert Systems with Applications*, Volume 54, 15 July, Pages 193-207.

Xiaodong Li, Haoran Xie, Ran Wang, Yi Cai, Jingjing Cao, Feng Wang, Huaqing Min & Xiaotie Deng. (2016). Empirical analysis: stock market prediction via extreme learning machine. *Neural Computing and Applications*. volume 27, pages 67–78.

Vora D., Singh N., Khalfay N., Soni V. (2017) Stock Prediction using Machine Learning a Review Paper. *International Journal of Computer Applications* 163(5):36-43

Weng B., Mohamed A., Fadel A., Megahedbc M. (2017). Stock market one-day ahead movement prediction using disparate data sources. *Expert Systems with Applications*. Volume 79, 15, Pages 153-163

Alessandretti L., Bahrawy E. A., Maria L. A., Baronchelli A. (2018). Anticipating Cryptocurrency Prices Using Machine Learning. *Complexity Journal*. Volume 2018.

Macchiarulo A. (2018). Predicting and beating the stock market with machine learning and technical analysis, *Journal of Internet Banking and Commerce*.

Zhang Z., Zohren S., Roberts S. (2019). Deep Reinforcement Learning for Trading *arXiv preprint arXiv:1911.10107*.

Deng Y., Bao F., Kong Y., Ren Z., Dai Q. (2016). Deep Direct Reinforcement Learning for Financial Signal Representation and Trading. *IEEE Transactions on neural networks and learning systems*.

Pengfei Y., Xuesong Y. (2020). Stock price prediction based on deep neural networks. *Neural Computing and Applications* volume 32, pages 1609–1628

Lei K., Zhang B., Yu L., Min Y. , Ying S. (2020) Time-driven feature-aware jointly deep reinforcement learning for financial signal representation and algorithmic trading *Expert Systems with Applications*. Volume 140, February.

Mnih V., Kavukcuoglu K., Silver D., Graves A., Antonoglou I., Wierstra D., Riedmiller M. (2013). Playing atari with deep reinforcement learning. *preprint arXiv:1312.5602*.

Silver D., Schrittwieser J., Simonyan K., Antonoglou I., Huang A., Guez A., Hubert T., Baker L., Lai M., Bolton

A., Chen Y., Lillicrap T., Hui F., Sifre L., Driessche G., Graepel T., Hassabis D. (2017). Mastering the game of go without human knowledge. *Nature*, 550:354–359.

Sutton R. S., McAllester D., Singh S., Mansour Y. (1999). Policy Gradient Methods for Reinforcement Learning with Function Approximation, *Advances in Neural Information Processing* (NIPS) 12.

Konda V. R., Tsitsiklis J. N. (1999). Actor-Critic Algorithms, Advances. *Neural Information Processing* (NIPS) 12.

Hasselt H., Guez A., Silver D. (2015). Deep Reinforcement Learning with Double Q-learning. *arXiv preprint arXiv:1509.06461*.

Mnih V., Badia A. P., Mirza M., Graves A., Harley T., Lillicrap T. P., Silver D., Kavukcuoglu K. (2016). Asynchronous Methods for Deep Reinforcement Learning.., *preprint arXiv:1509.06461*.

Schulman J., Wolski F., Dhariwal P., Radford A., Klimov O. (*2017).* Proximal Policy Optimization Algorithms *arXiv preprint 1707.06347*

Schulman J., Levine S., Moritz P., Jordan M. I., Abbeel P. (2015). Trust Region Policy Optimization *arXiv preprint 1602.01783.*

Lillicrap T. P., Hunt J. J., Pritzel A., Heess N., Erez T., Tassa Y., Silver D., Wierstra D. (2015). Continuous control with deep reinforcement learning *arXiv preprint 1509.02971*

Fujimoto S, Hoof H., Meger D. (2018). Addressing Function Approximation Error in Actor-Critic Methods *arXiv preprint 1802.09477*

Haarnoja T., Zhou A., Abbeel P., Levine S. (2018). Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. *arXiv preprint 1801.01290*

Bellemare M. G., Dabney W., Munos R. (2017) A Distributional Perspective on Reinforcement Learning, Actor *arXiv preprint 1707.06887*

Dabney W., Rowland M., Bellemare M. G., Munos R. (2017). Distributional Reinforcement Learning with Quantile Regression *arXiv preprint 1710.10044*

Andrychowicz M., Wolski F., Ray A., Schneider J., Fong R., Welinder P., McGrew B., Tobin J., Abbeel P., Zaremba W. (2017). Hindsight Experience Replay. *31st Conference on Neural Information Processing Systems* (NIPS 2017), Long Beach, CA, USA.

Markowitz H. (1952). Portfolio Selection. *The Journal of Finance*, Vol. 7, No. 1., pp. 77-91.

Merton R. C. (1969). Lifetime Portfolio Selection under Uncertainty: The Continuous-Time Case. *Review of Economics and Statistics* 51(3):247-57.

Kirk E. (2004). Optimal Control Theory: An Introduction. *Courier Corporation*. London.

Almahdi S., Yang S. Y. (2019). A constrained portfolio trading system using particle swarm algorithm and recurrent reinforcement learning. *Expert Systems with Applications*, Elsevier, 130:145-156.

Chandrinos, S. K., Sakkas, G. and Lagaros, N. D. (2018). AIRMS: A risk management tool using machine learning. *Expert Systems with Applications*, Elsevier, 105:34-48.

Lim B., Zohren S., Roberts S. (2019). Enhancing Time Series Momentum Strategies Using Deep Neural Networks. *The Journal of Financial Data Science*, PMR, arXiv preprint arXiv:1904.04912.

Jansen S. (2018). Hands-On Machine Learning for Algorithmic Trading: Design and implement investment strategies based on smart algorithms that learn from data using Python. *Packt Publishing*, Birmingham.

Storn R., Price K. (1997). Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of global optimization*, 11(4):341-359,.

Cohen N., Balch T., Veloso M. Trading via Image Classification, arXiv preprint arXiv:1907.09567v2, 2019.

Intercontinental Exchange, New York Stock Exchange Accessed: 11-August-2019, available at: [https://www.nyse.com/market-data/historical].

Elliott R. N., Prechter R., Jr. (1994). (ed.). R.N. Elliott's Masterworks. Gainesville, GA: *New Classics Library*. pp. 70, 217, 194, 196.

Hasselt H., Guez A., Silver D. (2016). Deep reinforcement learning with double Q-learning. In *Thirtieth AAAI conference on artificial intelligence*.

Arrow K. J. (1971). The theory of risk aversion. *Essays in the theory of risk-bearing*, pages 90–120.

Sutton R. S., Barto A. G. (1998). Reinforcement Learning: An Introduction. Cambridge, MA, USA: MIT Press.

Tesauro G. (1994). "TD-Gammon, a self-teaching backgammon program, achieves master-level play," *Neural Comput.*, vol. 6, no. 2, pp. 215–219.

Fischer T. G. (2018). Reinforcement learning in financial markets - A survey. Technical report, FAU *Discussion Papers in Economics*.

Huang C. Y. (2018). Financial trading as a game: A deep reinforcement learning approach. arXiv preprint arXiv:1807.02787.

Jin O. El-Saawy H. (2016). Portfolio management using reinforcement learning. Technical report, Working paper, Stanford University.

Ritter G. (2017). Machine learning for trading. Working paper, New York University.

Tan Z., Quek C., Cheng P. (2011). Stock trading with cycles: A financial application of ANFIS and reinforcement learning. *Expert Systems with Applications*, 38(5):4741–4755.