A Deep Reinforcement Learning Approach for Automated Cryptocurrency Trading

Giorgio Lucarelli, Matteo Borrotti

A Deep Reinforcement Learning approach for automated criptocurrency trading

Giorgio Lucarelli[1] and Matteo Borrotti[1,2,*]

[1]Università degli Studi di Milano-Bicocca, Milan, Italy
[2]Istituto di Matematica Applicata e Tecnologie Informatiche, IMATI-CNR, Milan, Italy
*email: matteo.borrotti@unimib.it

Abstract. Nowadays, Artificial Intelligence (AI) is changing our daily life in many application fields. Automatic trading has inspired a large number of field experts and scientists in developing innovative techniques and deploying cutting-edge technologies to trade different markets. In this context, cryptocurrency has given new interest in the application of AI techniques for predicting the future price of a financial asset. In this work Deep Reinforcement Learning is applied to trade bitcoin. More precisely, Double and Dueling Double Deep Q-learning Networks are compared over a period of almost four years. Two reward functions are also tested: Sharpe ratio and profit reward functions. The Double Deep Q-learning trading system based on Sharpe ratio reward function demonstrated to be the most profitable approach for trading bitcoin.

Keywords: Deep Reinforcement Learning, Double Deep Q-learning Networks, Dueling Architecture, Cryptocurrency, Automatic Trading.

1 Introduction

Nowadays, Artificial Intelligence (AI) is reshaping our daily life. AI is the study and design of intelligent agents where an agent is a system that perceives its environment and takes actions in order to maximize its chances of success. AI excels at interpreting signals and real-time analytic which underpin many different applications. For instance, AI is changing the way medical science was perceived just few years ago. Autonomous machines play an increasingly important role in surgery, improving patient outcomes and reducing expensive hospital stay time. Elsewhere, computer vision are improving diagnostic technologies and making them more accessible, while predictive algorithms are facilitating more rapid drug discovery. A less noble application is related to the financial sector, where AI is used to build automatic trading systems which are poised to foster a new financial technology transformation. Furthermore, the arrival of cryptocurrency has given new interest in the application of AI techniques for predicting the future price of a financial asset (*i.e.* Bitcoin).

In this context, Reinforcement Learning (RL) [6] [21] has demonstrated the potential to transform how classical trading systems work. RL is an autonomous,

self-teaching system that essentially learns by trial and error. It performs actions with the aim to maximize rewards and achieve the best outcomes [2] [17].

In this work, we investigate the performance of two different trading systems based on deep RL approaches: Double Deep Q-Network (D-DQN) [16] and Dueling Double Deep Q-Network (DD-DQN) [24]. The two trading systems are compared with a Deep Q-Network (DQN) [16].

The article is structured as follows: Section 2 provides a definition of cryptocurrency and bitcoin. Section 3 gives a short description of Reinforcement Learning. Section 4 introduces and describes the proposed Q-learning trading system. Main results are reported in Section 5 and Section 6 concludes the work.

1.1 Related Work

Deep Learning (DL) and Reinforcement Learning (RL) are viable approaches for market making. In recent years, the use of DL and RL is increased a lot demonstrating the powerful of these techniques.

McNally, S. et al. (2018) [15] applied different Machine Learning (ML) techniques on bitcoin cryptocurrency. More precisely, they compared Recurrent Neural Network (RNN) and Long Short Term Memory (LSTM) network against a more classical approach such as AutoRegressive Integrated Moving Average (ARIMA) model. RNN and LSTM outperformed ARIMA in a traditional classification setting.

Patel, Y. (2018) [19] proposed a multi-agent approach that operates at two different levels: (i) minute level (macro-agent) and (ii) order book level (micro-agent). The macro-agent is based on a Double Q-learning network composed by a Multi-Layer Perceptron (MLP) and the micro-agent is realized with a Dueling Double Q-learning network with reward function based on volume weighted average bitcoin price. The multi-agent did not outperfom the simple macro-agent but it obtained better results with respect to a uniform Buy and Hold and Momentum Investing techniques in terms of cumulative profits.

Previous works were applied only to bitcoin movements, Bu, S.-J. et al (2018) [5] tested a hybrid approach (Boltzmann machine and Double Q-learning network) against LSTM, MLP, Convolutional Neural Network (CNN) over eigth cryptocurrencies. They used the ratio between total value after investement and initial value as evaluation score. The hybrid approach demonstrated to be more profitable than competitors but more risky and unstable.

Alessandretti, L. and coauthors (2018) [1] and Jiang, Z. et al. (2017) [10] applied Artificial Intellingent (AI) approaches on portfolio management. In [1] the authors applied a gradient boosting decision tree (*i.e.* XGBoost) and LSTM network on a cryptocurrency portfolio. Performance were evaluated considering Sharpe ratio [20] and geometric mean return. All proposed strategies produced profit over the entire test period. Jiang, Z. et al. (2017) [10] applied a deterministic policy gradient using a direct reward function (average logarithmic return) for solving the portfolio management problem. The approach demonstrated to outperfom classical management techniques except against a Passive Aggressive Mean Reversion technique in terms of cumulative return.

In this work, the proposed trading systems based on deep RL approaches differ from previous techniques for the use of a reward function based on Sharpe ratio. Furthermore, Double Q-learning and Dueling Double Q-learning networks are used as agents that interact with the financial market in a macro level.

2 Cryptocurrency and Bitcoin

A cryptocurrency can be seen as a digital or virtual currency that works as a medium of exchange. In few words, it is a set of limited entries in a database that no one can change unless specific conditions are fulfilled. Bitcoin is one of the most established and discussed cryptocurrency available today. Since its origination in 2009, bitcoin has received the stature of a digital commodity and its value is considered comparable to traditional currencies [7]. The exchanges of bitcoin are verified for secure transaction by network nodes which use cryptographic techniques. They are recorded in a public distributed ledger called block chain which records bitcoin transactions [7].

Considering a specific time interval, bitcoin price information is represented by *candlesticks*, or *Open-High-Low-Close* (OHLC) chart. A candle consists of four measurements for an asset during a period: the opening price at the start of the period, the highest and lowest price within the period, and the closing price at the end of the period. The opening and closing part of a candle is usually charted as a box and the highest and lowest prices as the "wicks" above and below. Candles themselves trivially aggregate into larger candles. For instance, a 1 hour candle is easily derived by aggregating 60 candles of 1 minute.

2.1 Automated trading

Automated trading can be seen as an automated decision-making procedure. Usually, automated trading procedures aim at predicting whether a possible positive return will be realized in the near future. The automated trading procedure should define whether to buy or sell the asset under consideration or hold the current position.

At time step t, the automated trading procedure will then act based on the decision rule defined in Eq. 1.

$$Action = \begin{cases} Buy,\ if\ E[p_{t+h}] > p_t, \\ Hold,\ if\ E[p_{t+h}] = p_t, \\ Sell,\ if\ E[p_{t+h}] < p_t. \end{cases} \tag{1}$$

Given the price of an asset at time t, p_t, the automated trading procedure buys if the expected price at time $t + h$, $E[p_{t+h}]$, is greater than p_t and sell if $E[p_{t+h}]$ is lower than p_t, otherwise it does not do any action (*hold*). h is some positive number of time steps in the future [3].

Throughout this work we make use of two common market orders: *long* and *short*. Long trades are the classic method of buying with the intention of profiting

from a rising market, *i.e.* $E[p_{t+h}] > p_t$. Short trades are used with the intention of profiting from a falling market, *i.e.* $E[p_{t+h}] < p_t$. Other two orders are commonly used in defining trading strategies: *stop-loss* and *take-profit*. Both of them are used to buy or sell an asset when it reaches a particular price. Stop-loss is used to reduce a possible loss. Take-profit is used to guarantee a possible gain.

3 Reinforcement learning

Reinforcement learning (RL) can be seen as the formalization of an optimal policy capable of ensuring the maximization of the expected cumulative profit of an agent [6]. In the course of this section, we keep close to the description as given in [9] [12] [18].

The agent interacts with the environment by executing actions and receiving observations and rewards. At each time step t, which ranges over a set of discrete time intervals, the agent select an action a from a set of legal actions $\mathcal{A}$ at state $s_t \in \mathcal{S}$, where $\mathcal{S}$ is the set of possible states. Action selection is based on a policy, π. The policy is a description of the behaviour of the agent and tells the agent which actions should be selected for each possible state. As a result of each action, the agent receives a scalar reward $r_t \in \mathcal{R}$, and observes next state $s_{t+1} \in \mathcal{S}$. The *transition probability* of each possible next state s_{t+1} is defined as $P(s_{t+1}|s_t, a_t)$, with $s_{t+1}, s_t \in \mathcal{S}$ and $a_t \in \mathcal{A}$. Similarly, the *reward probability* of each possible reward r_t is defined as $P(r_t|s_t, a_t)$ where $s_t \in \mathcal{S}$, $a_t \in \mathcal{A}$. Hence, the expected scalar reward, r_t, received by executing action a in current state s is calculated based on $E_{P(r_t|s_t,a_t)}(r_t|s_t = s, a_t = a)$. This framework can be seen as a finite Markov Decision Process (MDP).

The aim of the learning agent is to learn an optimal policy π^*, which defines the probability of selecting action a in state s, so that the sum of the discounted rewards over time is maximized. The expected discounted return R at time t is defined as follows:

$$R_t = E[r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \dots] = E[\sum_{k=0}^{\infty} \gamma^k r_{t+k}], \tag{2}$$

where $E[.]$ is the expectation with respect to the reward distribution and $0 < \gamma < 1$ is called the discount factor. At this point a Q-value function, $Q^\pi(s, a)$, can be defined as follows:

$$Q^\pi(s, a) = E_\pi[r_t|s_t = s, a_t = a] = E_\pi[\sum_{k=0}^{\infty} r_{t+k}|s_t = s, a_t = a]. \tag{3}$$

The Q-value, $A_\pi(s, a)$, for an agent is the expected return achievable by starting from state $s \in \mathcal{S}$ and performing action $a \in \mathcal{A}$ following policy π. Eq. 3 satisfies a recursive property, so that an iterative update procedure can be used

for the estimation of Q-value function:

$$\begin{aligned} Q^{\pi}_{i+1}(s,a) &= E_{\pi}[r_t + \gamma \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s, a_t = a] = \\ &= E_{\pi}[r_t + \gamma Q^{\pi}_i(s_{t+1} = s', a_{t+1} = a') | s_t = s, a_t = a], \end{aligned} \tag{4}$$

for all $s, s' \in \mathcal{S}$ and $a, a' \in \mathcal{A}$.

Reinforcement learning agent aims at finding the policy which achieves the greatest outcome. Hence, it must learn an optimal policy π^* with the expected value greater than or equal to all other policies, and leading to an optimal Q-value $Q^*(s,a)$. In particular, the iterative update procedure for estimating the optimal Q-value function ca be defined as in Eq. 5.

$$Q_{i+1}(s,a) = E_{\pi}[r_t + \gamma \max_{a'} Q_i(s', a') | s, a]. \tag{5}$$

The iteration procedure converges to the optimal Q-value, Q^*, as $i \to \infty$ and is called *value iteration algorithm*. One of the most popular value-based algorithms is the *Q-learning* algorithm [25]. The basic version of Q-learning algorithm makes use of the Bellman equation for the Q-value function [4] whose unique solution is $Q^*(s,a)$:

$$Q^*(s,a) = (\mathcal{B}Q^*)(s,a), \tag{6}$$

where $\mathcal{B}$ is the Bellman operator mapping any function $K : S \times A \to R$ into another function $S \times A \to R$ and is defined as follows:

$$(\mathcal{B}K)(s,a) = \sum_{s' \in S} T(s,a,s')[R(s,a,s') + \gamma \max_{a' \in A} K(s', a')], \tag{7}$$

where T is the function for calculating the transaction value to go from s to s' given action a. One general proof of convergence to the optimal value function is available [25] under the conditions that: (i) the state-action pairs are represented discretely, and (ii) all actions are repeatedly sampled in all states (which ensures sufficient exploration, hence not requiring access to the transition model).

In that context, a parametric value function $Q(s,a;\theta)$ is needed, where θ refers to some parameters that define the Q-values. Different Q-networks are available in literature:

- **Deep Q-Networks** (DQNs): DQNs were introduced by Mnih et al. (2015) [16]. DQNs stabilize the training of action value function approximation with deep neural networks, in particular Convolutionary Neural Networks (CNNs) [6], using experience replay [13] and target network.
- **Double Deep Q-Networks** (D-DQNs): D-DQN improved DQN avoiding over-estimation. In D-DQN a greedy policy is evaluated in accordance with a online network and a target network is used to estimate its value.
- **Dueling Double Deep Q-Networks** (DD-DQNs): DD-DQN [24] is based on a dueling network architecture to estimate value function $V(s)$ and the

associated advantage function $A(s,a) = Q(s,a) - V(s)$, and then combine them in order to estimate $Q(s,a)$. In DD-DQN, a CNN layer is followed by two streams of fully connected (FC) layers, used to estimate the value function and the advantage function separately; then the two streams are combined to estimate the action value function.

4 Q-learning Trading System

The proposed Q-learning trading system is based on (i) D-DQN and (ii) DD-DQN. In both cases, an agent interacts with the financial market. Given a certain state of the financial market, the agent defines the type of the action a (buy, hold, sell) to do on a bitcoin unit. If a bitcoin is acquired, it is then added to a wallet. A stop-loss ($sl = -5\%$) and a take-profit ($tp = +12\%$) are also applied to the wallet. For instance, if the wallet loses more than a threshold (*i.e.* -5%), all open positions are closed.

The exploration-exploitation *dilemma* is of fundamental importance for deep RL techniques as well as for the proposed Q-learning trading system. Exploitation concerns information about the environment (*i.e.* transition and reward functions) while exploitation is about maximizing the expected return given the current knowledge. For this reason, the agent can take a random action with probability, ϵ, and follows the policy that is believed to be optimal with probability, $1-\epsilon$ (ϵ-greedy technique). In the proposed trading system, an $\epsilon_{initial} = 1$ is selected for the first observations ($n_{obs} = 300$) and then is set to a new value $\epsilon_{new} = 0.12$. For a more realistic study, a trade transition cost equal to 0.3% is applied both for long and short actions.

The Q-learning trading system rewards the agent with two possible functions: (i) Sharpe ratio [20], $s_{p_t} = \frac{(\varrho_{p_t} - \varrho_f)}{\sigma_{p_t}}$, where ϱ_{p_t} is the return of the portfolio or merely the return of the asset, ϱ_f is the risk-free rate ($\varrho_f = 0$ in our work), σ_{p_t} is the standard deviation of portfolio's return and (ii) a simple profit function, $g_{profit} = (p_t - p_{t-1})$ (*i.e.* nominal return), where p_t is the asset price at time t and p_{t-1} the asset price at time $t-1$. More precisely, in the first case the trading strategy at time t is:

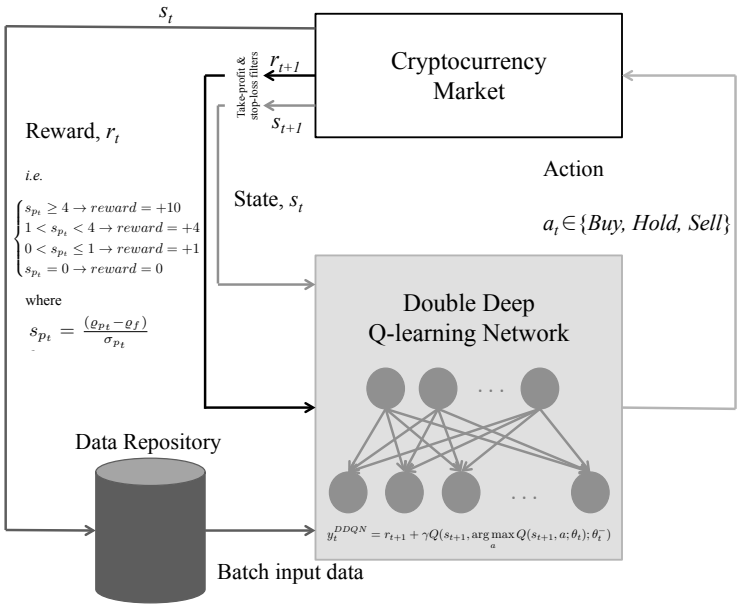
$$\begin{cases} s_{p_t} \geq 4 \rightarrow reward = +10 \\ 1 < s_{p_t} < 4 \rightarrow reward = +4 \\ 0 < s_{p_t} \leq 1 \rightarrow reward = +1 \\ s_{p_t} = 0 \rightarrow reward = 0 \\ 0 < s_{p_t} \leq -1 \rightarrow reward = -1 \\ -1 < s_{p_t} < -4 \rightarrow reward = -4 \\ s_{p_t} \leq -4 \rightarrow reward = -10 \end{cases} \tag{8}$$

In the second case, the trading strategy at time t is:

$$\begin{cases} g_{profit} > 0 \rightarrow reward = 1 \\ g_{profit} = 0 \rightarrow reward = 0 \\ g_{profit} < 0 \rightarrow reward = -1 \end{cases} \tag{9}$$

Fig. 1 shows the Q-learning trading system based on a Double Deep Q-learning Network with Sharpe ratio reward function.

Fig. 1. Double Deep Q-learning trading system with Sharpe reward function.



The Q-learning trading system based on the D-DQN is composed by 2 CNN layers with 120 neurons each. In the case of DD-DQN, 2 CNN layers with 120 neurons each are followed by two streams of FC layers: the first with 60 neurons dedicated to estimate the value function and the second with 60 neurons to estimate the advantage function. In both cases, the number of epochs is set to 40 as well as the batch size. For weight optimization, the ADAM algorithm [11] is applied. The loss function is the Mean Squared Error, $MSE = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n}$. The activation function is set as the Leaky Rectified Linear Units (Leaky ReLU) function [14]. The discount factor, γ, is set to 0.98 in both D-DQN and DD-DQN.

A similar setting is also used to implement the trading strategies based on DQN.

5 Experimental Data and Results

5.1 Bitcoin historical data

The proposed Q-learning trading systems are tested on bitcoin historical data. Data can be found on the well-known *Kaggle* (www.kaggle.com) platform[1]. We considered bitcoin price in USD dollars from the 1^{st} December 2014 to the 27^{th} June 2018, sampled at 1 minute interval. For each observation, time stamp, OHLC (Open, High, Low, Close) values, volume in bitcoin, volume in USD dollars, and weighted bitcoin price are collected. The dataset is composed by roughly 2 million rows and 8 variables. Based on the time stamp, data is hourly aggregated obtaining a final dataset with more than 30.000 observations and the same number of variables.

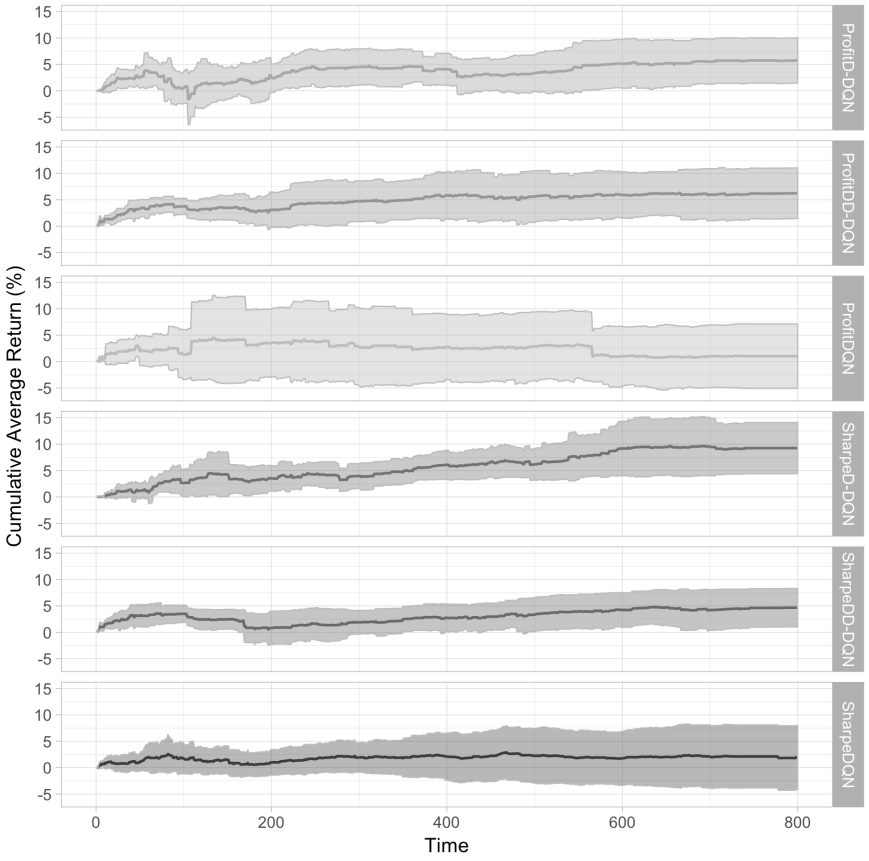


Fig. 2. Average percentage returns over the 10 trading periods, *i.e.* different combinations of start and end dates for the trading activity.

[1] *www.kaggle.com/mczielinski/bitcoin-historical-data*

5.2 Results

The Q-learning trading system is tested with four different settings based on:

1- Double Deep Q-Network with a profit reward function (*Profit*D-DQN);
2- Double Deep Q-Network with Sharpe ratio reward function (*Sharpe*D-DQN);
3- Dueling Double Deep Q-Network with a profit reward function (*Profit*DD-DQN);
4- Dueling Double Deep Q-Network with Sharpe ratio reward function (*Sharpe*DD-DQN);

The four settings are compared with a Deep Q-Network with profit reward function (*Profit*DQN) and a Deep Q-Network with Sharpe ratio reward function (*Sharpe*DQN).

Table 1. Average performance over the 10 trading periods.

Trading System	Avg. Return (%)	Max. Return (%)	Min. Return (%)	St. Dev.
*Profit*D-DQN	3.74	21.31	-10.74	4.87
*Profit*DD-DQN	4.85	17.34	-8.49	5.10
*Profit*DQN	2.32	22.59	-17.97	7.93
*Sharpe*D-DQN	5.81	26.14	-5.64	5.26
*Sharpe*DD-DQN	3.04	13.03	-8.49	3.81
*Sharpe*DQN	1.83	15.80	-9.29	5.46

Test 1. All Q-learning trading system settings are compared sampling 10 different periods of size 4.000. For each period, 80% is dedicated for training purpose and 20% for testing the performance.

In Fig. 2, the cumulative average return (%) over the 10 test sets is reported. 95% confidence intervals around the mean are also included. DD-DQN and D-DQN trading systems clearly outperform the simpler DQN system. In average the best cumulative return (%) is reached by the *Sharpe*D-DQN.

Table 1 summarizes main statistical indicators. The trading systems based on DD-DQN and D-DQN reaches higher cumulative average return (%). In fact, the *Profit*DQN and *Sharpe*DQN obtain the worst results over all the test periods. Furthermore, DQN has the highest standard deviation demonstrating high instability. *Sharpe*D-DQN has the highest average return (5.81%) over all the test period. It reaches a maximum value of return percentage equal to 26.14% and a minimum value equal to -5.64%. The DD-DQN and D-DQN trading systems based on the profit reward function have comparable results.

From this preliminary analysis, the *Sharpe*D-DQN has demonstrated to be the best Q-learning trading system.

Test 2. Given the previous results, the *Sharpe*D-DQN is tested on the entire period (from the 1^{st} December 2014 to the 27^{th} June 2018).

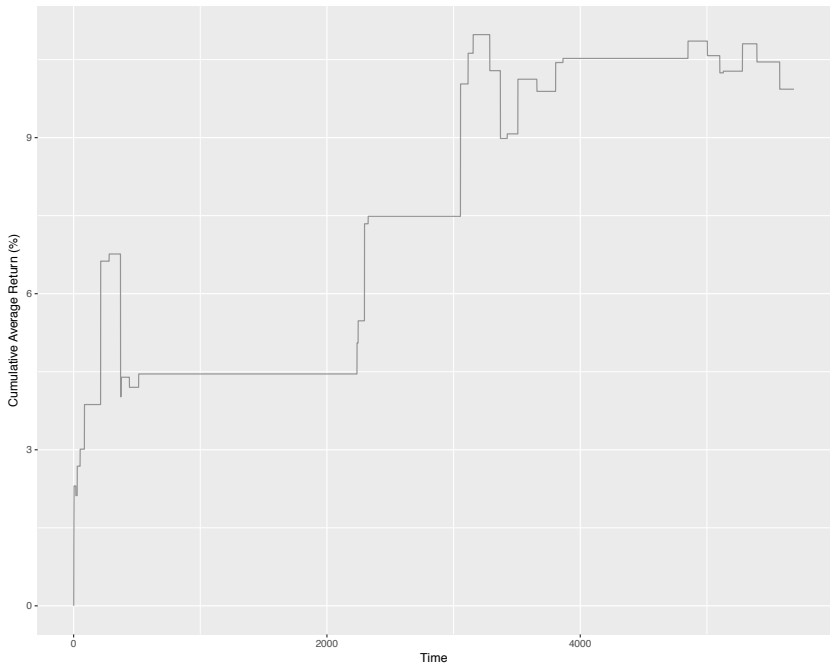


Fig. 3. *Sharpe*D-DQN performance over the entire period.

Observations from 1^{st} December 2014 to 1^{st} November 2017 are used by *Sharpe*D-DQN system to learn how to trade the cryptocurrency. After that period, *Sharpe*D-DQN system has acted as an autonomous algorithmic trading system (from 2^{st} November 2017 to 26^{th} June 2018). It had an average percentage return (%) of almost 8% with a standard deviation 2.77. In Fig. 3, the cumulative percentage return over the entire period is shown.

6 Conclusions and Future Work

In this work, the performance of different trading systems based on Deep Reinforcement Learning were tested on hourly cryptocurrency (*i.e.* bitcoin) prices. The trading systems were based on Double and Dueling Double Deep Q-learning Networks. Furthermore, the previous trading systems were compared with a simpler Deep Q-learning Network. Each of them were tested with two different reward functions. The first function was based on the Sharpe ratio, a measure of the risk-adjusted return on an investment, and the second function was related to profit. Then, six different Q-learning trading system settings where tested on

bitcoin data from the 1^{st} December 2014 to the 27^{th} June 2018. Performance were evaluated in terms of percentage returns.

All systems produced positive return (in average) for a set of shorter trading periods (different combinations of start and end dates for the trading activity). The trading systems based on Double Q-learning and Sharpe ratio reward function (*Sharpe*D-DQN) achieved larger return values. *Sharpe*D-DQN was also tested over the entire considered period producing a positive percentage return value (average percentage return 8%).

It is important to stress that this work has some limitations. First, a broader set of performance indicators should be used to compare the different approaches. Second, the proposed Deep Reinforcement Learning techniques should be compared with recent AI approaches for a more accurate comparison study. Third, a parameter optimization should be done to improve the performance of the learning techniques. Given that, the presented methods were able to generate positive returns on all conducted tests. Extending the current analysis by considering these elements is a direction for future work.

A different yet promising approach is to study the impact of social media on bitcoin and other cryptocurrency fluctuation prices and incorporating news and public opinion into the Deep Reinforcement Learning approach. In addition, uncertainty estimations should be investigated since uncertainty is essential for efficient reinforcement learning.

Lastly, the proposed approaches can be extended for anomaly detection. Following the work of Du, M. et al. (2017) [8], Q-learning approaches can be used to build a framework for online log anomaly detection and diagnosis. Such an approach could be a critical step towards building a secure and trustworthy anomaly detection system.

References

1. Alessandretti, L., ElBahrawy, A., Aiello, L. M., Baronchetti, A.: Anticipating cryptocurrency prices using machine learning. Complexity, Vol. 2018, pp. 1-16 (2018).
2. Almahdi, S., Yang, S.Y.: An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown. Expert Systems with Applications, Vol. 87, pp. 267-279 (2017).
3. Bach, W.G., Kasper L.N.: On machine learning based cryptocurrency trading. Aalborg University, Denmark (2018).
4. Bellman, R.E., Dreyfus, S.E.: Applied dynamic programming. RAND Corporation, Santa Monica CA (1962).
5. Bu, S.-J., Cho, S.-B.: Learning optimal Q-function using deep Boltzmann machine for reliable trading of cryptocurrency, In: Proceedings of the 19^{th} International Conference on Intelligent Data Engineering and Automated Learning (IDEAL'18), pp. 468-480. Madrid SP (2018).
6. Buduma, N.: Fundamentals of deep learning: designing next-generation artificial intelligence algorithms. O'Reilly Media, Sebastopol CA (2017).
7. Cheeda, S.R., Singh, A.K., Singh, P.S., Bhole, A.S.: Automated trading of cryptocurrency using twitter sentimental analysis. International Journal of Computer Science and Engineering, Vol. 6, pp. 209-214 (2018).

8. Du, M., Li, F., Zheng, G., Srikumar, V.: DeepLog: anomaly detection and diagnosis from system Logs through Deep Learning. In: Proceeding of the Conference on Computer and Communications Security (CCS'2017), pp. 1285-1298. Dallas TX (2017).
9. François-Lavet, V., Henderson, P., Islam, R., Bellemare, M.G., Pineau, J.: An introduction to deep reinforcement learning. Foundation and Trends in Machine Learning, Vol. 11, pp. 219-354 (2018).
10. Jiang, Z., Liang, J.: Cryptocurrency portfolio management with deep reinforcement learning. In: Proceedings of the Intelligent Systems Conference (IntelliSys'17), pp. 905-913 (2017).
11. Kingma D.P., Ba, J.: Adam: A method for stochastic optimization. In: Proceedings of the 3^{rd} International Conference of Learning Representations (ICLR'15), pp. 1-15. San Diego CA (2015).
12. Li, Y.: Deep reinforcement learning. arXiv, pp. 1-150 (2018).
13. Lin, L.-J.: Programming robots using reinforcement learning and teaching. In: Proceedings of the Ninth National Conference on Artificial Intelligence (AAAI'91), pp. 791-786. AAAI Press, Anaheim CA (1991).
14. Maas, A.L., Hannun, A.Y., Ng, A.Y.: Rectifier nonlinearities improve neural network acoustic models. In: Proceedings of the $30^{t}h$ International Conference on Machine Learning, pp. 1-6. Atlanta GA (2013).
15. McNally, S., Roche, J., Caton, S.: Predicting the price of bitcoin using Machine Learning. In: Proceedings of the $26^{t}h$ Euromicro International Conference on Parallel, Distributed and Network-based Processing (PDP'18), pp. 339-343, IEEE, Cambridge UK (2018).
16. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M.: Playing Atari with deep reinforcement learning. arXiv, pp. 1-9 (2013).
17. Moody, J., Saffell, M.: Learning to trade via direct reinforcement. IEEE Transactions on Neural Networks. Vol. 12, pp. 875-889 (2001).
18. Mousavi, S.S., Schukat, M., Howley, E.: Deep reinforcement learning: an overview. arXiv, pp. 1-17 (2018).
19. Patel, Y.: Optimizing market making using multi-agent reinforcement learning. arXiv, pp. 1-10 (2018).
20. Sharpe, W.f.: The Sharpe ration. The Journal of Portfolio Management, Vol. 21, pp. 49-58 (1994).
21. Sutton, R.S., Barto, A.G.: An introduction to reinforcement learning. MIT Press, Cambridge MA (2015)
22. 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 (NIPS'99), pp. 1057-1063. MIT Press, Cambridge MA (2000)
23. Thrun, S., Schwartz, A.: Issues in using function approximation for reinforcement learning. In: Proceedings of the Fourth Connectionist Models Summer School, pp. 1-9. Lawrence Erlbaum Publisher, Hillsdale NJ (1993).
24. Wang, Z., Schaul, T., Hessel, M., van Hasselt, H., Lanctot, M., de Freitas, N.: Dueling network architectures for deep reinforcement learning. In: Proceedings of the 33^{rd} International Conference on Machine Learning (ICML'16), pp. 1-9. PLMR, New York NY (2016).
25. Watkins, C.J.C.H., Dayan, P.: Q-learning, Machine Learning. Vol.8, pp. 279-292 (1992).