NOVA IMS Information Management School

MDSAA

Master Degree Program in

Data Science and Advanced Analytics

Enhancing Buy and Hold Strategy with Deep Reinforcement Learning Ensemble for Drawdown Risk Mitigation in Cryptocurrency Markets

Stanislav Slesarev

Master Thesis

presented as partial requirement for obtaining a Master's Degree in Data Science and Advanced Analytics

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

Enhancing Buy and Hold Strategy with Deep Reinforcement Learning Ensemble for Drawdown Risk Mitigation in Cryptocurrency Markets

by

Stanislav Slesarev

Master Thesis presented as partial requirement for obtaining the Master's degree in Data Science and Advanced Analytics, with a specialization in Data Science

Supervised by

Iann James Scott, PhD, Nova Information Management School

Nuno Tiago Falcão Alpalhão, MsC, Nova Information Management School

July, 2024

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration. I further declare that I have fully acknowledged the Rules of Conduct and Code of Honor from the NOVA Information Management School.

Lisbon, 15 July 2024

DEDICATION

To my mother, who supported me all my life and helped me to become the better version of myself. To every professor and supervisor, who helped me to understand better all the complex topics along the way. To every fellow student who shared the knowledge with me, I have enormous gratitude to all of you mentioned above.

I hope the world will have its deserved peaceful times soon.

ACKNOWLEDGEMENTS

First, I want to express deep gratitude for my supervisor Nuno Tiago Falcão Alpalhão, who was unbelievably kind and supportive during almost the whole year. I feel like this journey could have been much harder without your help. Even though the thesis required a lot of research, I believe your availability for consultation played the crucial part in this journey and helped my progress every time I got stuck.

Secondly, I'm thankful to Professor Ian Scott for accepting my thesis idea and the reviews you've made, which are appreciated, and I believe led to better quality of work.

Finally, every student I worked shoulder to shoulder with on any project, I can not be more grateful to you, namely: Samuel Santos, Janaina, Ines, Adriana, Sabeen, Rafael, Henrique, Vasco, Aidar. I was inspired by your dedication and intelligence, keep going with the same passion.

ABSTRACT

It is known that the average retail investor finds the financial markets quite complex, unpredictable and volatile, which can lead an investor to emotional mistakes and significant financial losses. Buy and Hold investment strategy is known to be the best for retail investors, but it is susceptible to severe drawdowns in the phases of declining market. To address this issue, Buy and Hold can be enhanced by some risk management approach (e.g. trend following strategy). Another way to control risk can be the application of Machine Learning in Finance, namely Deep Reinforcement Learning, which proved to be successful in multiple fields including e-commerce, energy sector, gaming etc. This thesis is dedicated to researching the application of Reinforcement Learning in the form of an ensemble of Deep Reinforcement Learning Agents to mitigate drawdown risks and prevent investors from significant financial losses. The ensemble of agents is trained on past cryptocurrency market data, namely Bitcoin market, while a part of that data is used to evaluate how well the ensemble is generalized and to measure its performance. The findings of this study show the ensemble of trading agents can succeed at risk control and reduction compared to Buy and Hold passive investment strategy but still lacks the ability to achieve the same returns.

KEYWORDS

Reinforcement Learning; Q-Learning; Double Dueling Deep Q Network; Ensemble of DQN Agents; Bitcoin; Trading; Risk management;

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

RL	Reinforcement Learning
MDP	Markov Decision Process
DRL	Deep Reinforcement Learning
DNN	Deep Neural Network
DQN	Deep Q Network
DDQN	Double Deep Q Network
DDDQN	Double Deep Dueling Q Network
MADDQN	Multi-agent Double Deep Q Network
DADE-DQN	Dual Action and Dual Environment Deep Q-Network

1. INTRODUCTION

Financial markets have a complex, uncertain, and dynamic nature, making it challenging to trade and predict their future development (Shavandi & Khedmati, 2022). However, behavioral finance suggests that market participants may not be completely rational or fully informed regarding their actions (Felizardo et al., 2022). The passive trading strategy "Buy and Hold" is known for producing a great expected return in bull markets (Hilliard & Hilliard, 2018) and therefore is widely adopted as a simple basic strategy for retail investors and as a baseline benchmark strategy (Théate & Ernst, 2021). Recently, Deep Learning methods and Deep Q-Network Algorithm (Shavandi & Khedmati, 2022) demonstrated the effectiveness for analysis and trading in stock and crypto markets.

Even though the "Buy and Hold" strategy performs well in growing markets, this strategy is still at risk of severe drawdowns (Grudniewicz & Ślepaczuk, 2023), because of the strong linear correlation with prices of assets. Despite recent advancements in Machine Learning, there is a large variety of studies on the integration of Deep Reinforcement Learning (DRL) approaches for solving trading problems. By exploring Deep Q-Network algorithms in trading strategies, it should be possible to teach agents to opt out of the market in times of uncertainty and apply safer strategies with good returns and low risk (Carta et al., 2021).

To solve this trading problem an ensemble of Deep Reinforcement Learning (DRL) agents will be trained and tested on the historical trading data. The DRL ensemble is supposed to learn an actuation policy, where the correct action is the action that gives our autonomous agents the maximum asset price return while minimizing drawdown risk.

Active trading strategies using Deep Q-Network algorithms are expected to adapt effectively to changing market conditions, resulting in better mitigation of drawdown risks, while demonstrating an equal or better performance compared to the benchmark strategy "Buy and Hold."

2. LITERATURE REVIEW

Managing risks in the markets has been a complex problem since the market's inception. Multiple theories and approaches emerged intending to explain and forecast the market's behavior and to improve risk management.

One of the first attempts to analyze markets, Dow Theory, was developed by Charles Dow in the 19th Century (Yamani, 2023). Dow Theory says the market is in an upward trend if one of its averages (e.g., industrials or transportation) advances above a previous important high and is accompanied or followed by a similar advance in another average.

Modern Portfolio Theory was developed by Harry Markowitz in his paper "Portfolio Selection" 1952, which introduced diversification as an instrument to maximize returns for a certain level of risk (Shadabfar & Cheng, 2020), which can be viewed as an advancement in the risk control approach. The diversification technique allows to combine higher risk assets with lower risk assets therefore offering a mixed portfolio complied with certain risk appetite.

Another related concept is the Capital Asset Pricing Model, developed and published by William Sharpe and John Lintner in 1964 (Boďa & Kanderová, 2014). The Capital Asset Pricing Model provides a framework for determining the expected return on an investment based on its risk relative to the overall market. It incorporates concepts like the risk-free rate, market risk premium, and beta.

The efficient market hypothesis (EMH) was introduced by Eugene Fama in his PhD work "Random Walks in Stock Market Prices". Later on, the EMH was enhanced with three different forms of efficiency – weak, semi-strong, and strong. The strongly efficient market was defined as "a market with a great number of rational, profit-maximizers actively competing, with each trying to predict future market values of individual securities, and where current important information is almost freely available to all participants." (Ţiţan, 2015)

A key component of the weak form of EMH is the Random Walk Theory (RWT), which was popularized by Burton Malkiel in his book "Random Walk Down Wall Street" 1973, which popularized the idea of prices moving randomly and thus the evolution of prices cannot be predicted.

Benoit Mandelbrot, on his turn, criticized RWT and observed that stock prices exhibit long-term dependence and are better modeled by fractal geometry, where investors should consider the risks associated with extreme Black Swan events, which "can be defined as an unexpected year-on-year drop in revenue between 30%–90%" (Christie et al., 2024). These ideas were influential in developing chaos theory in finance.

On the contrary to RWT, the Behavioral Finance approach challenges statements of the EMH. Behavioral finance has roots in the work of psychologists and economists in the 1970s and 1980s. The term itself gained widespread use in the 1990s. Amos Tversky and Daniel Kahneman (Silva et al., 2023) made significant contributions with their prospect theory, challenging the traditional assumptions of rational behavior in finance. Behavioral finance

incorporates psychological factors into market analysis and recognized investors are not always rational and may be subject to cognitive biases and emotions.

Algorithmic trading emerged in the US equities market in the 1990s. There are studies arguing that the speed advantage of algorithmic traders over humans — specifically, their ability to react more quickly to public information — should have a positive effect on the informativeness of prices (Chaboud et al., 2014).

Algorithmic trading can fall into two categories - rule-based and ML-based approaches. Rule-based algorithmic trading allows computers to execute trades using predetermined policies based on traditional trading methods, mathematical models, or strategies invented by humans. On the contrary, Machine Learning (ML) algorithmic trading allows computers to be trained on historical data and invent their policies (Shavandi & Khedmati, 2022).

Supervised ML algorithms became the first attempt to apply ML to financial markets and allowed researchers to take advantage of human expert knowledge to build complex models capable of trading. Deep learning techniques have been excessively used in the financial stock trading market (AbdelKawy et al., 2021), however, due to the markets' noise, uncertainty, and volatility the accuracy of their predictive models might not be good enough for trading in the unpredictable environment (Shavandi & Khedmati, 2022). Moreover, excellent predictive accuracy is not enough to operate in an environment with unseen data, because trading strategy should include facets like risk management and quick response to unexpected events.

Therefore, the next advance in algorithmic trading with Machine Learning was an application of Reinforcement Learning (RL) algorithms not only for predicting the prices but also to actively manage risks in the automated mode. In October 2015, AlphaGo first beat professional Go player (Chao et al., 2018). This achievement sparked the interest in ways to apply Reinforcement learning algorithms in finance (Ye & Schuller, 2023).

However, even before AlphaGo's success, there were important research attempts in the field of Reinforcement learning applications to solve trading problems. Recurrent Reinforcement Learning, an adaptive policy search algorithm that can learn an investment strategy online, was proved to be a working solution by outperforming the Buy & Hold baseline strategy in the SP500 market (Moody & Saffell, 2001). The study by Moody & Lucar Saffell also highlighted the important advantage of policy learning approaches over model-free off-policy RL algorithm called Q-learning (Chakole et al., 2021).

Another study proves Recurrent Reinforcement Learning can perform well in the foreign currency exchange market (Forex) and earn constant gains while avoiding significant drawdowns (Dempster & Leemans, 2006).

One of the novel trading models with reinforcement learning techniques was proposed by (Y. Hu et al., 2015) – a hybrid long-term and short-term evolutionary trend-following algorithm (eTrend) that combines trend-following investment strategies with the eXtended Classifier Systems (XCS). XCS has Genetical Algorithms and Reinforcement Learning embedded, which allows to invent automatic trading rules. Experimental results showed that this approach can bring excessive returns with a high Sortino ratio, which "considers only the downside volatility

as the unpleasant part of price fluctuation" (Chen et al., 2014), which is considered to be a good result.

Another study injected the Deep Learning approach into the Reinforcement Learning framework to achieve trading signals generation without the usage of technical indicators (Deng et al., 2017). This approach showed the effectiveness of cooperation between Deep Learning and Reinforcement Learning concepts, where summarization of market conditions was performed by the Deep Learning model and then the Reinforcement Learning module analyzed market representations and performed trading actions.

Another study (Théate & Ernst, 2021) showed the advantages of Deep Reinforcement Learning application to solve trading problems. The Trading Deep Q-Network algorithm utilized both trend-following and mean-reversion techniques and showed performance slightly better than benchmark strategies on different stock markets.

Another research implemented a Recurrent Reinforcement Learning agent by combining online transfer learning and policy gradient reinforcement learning to achieve an impressive 350% ROI for perpetual futures contract trading on the Bitmex exchange (Borrageiro et al., 2022). It is important to note that profit from funding was one of the rewards for the agent, so the agent captured 71% ROI only by collecting funding.

A recent study (Felizardo et al., 2022) researched an application of Deep learning in the form of ResNET-LSTM to solve financial trading problems. The research also compared the performance of ResNET-LSTM to the performance of Deep Q-Network (DQN) (Jeong & Kim, 2019) on cryptocurrency markets including Bitcoin and Ethereum. The ResNET-LSTM actor had an impressive performance overall, while DQN was significantly underperforming in conditions without transaction costs. When transaction costs were present DQN started to be competitive, which proves Reinforcement learning is a valid approach for solving financial trading challenges.

Another advancement in this field is the Dual Action and Dual Environment Deep Q-Network model (Huang et al., 2023) as an extension of the Deep Q-network approach. DADE-DQN enhances the common trend-following strategy with an impressive cumulative return of 79.43% on Korea Composite Stock Price Index data, while maintaining the Sortino ratio at 2.21, which proves this approach to be a good-performing model with good risk mitigation for automated trading of indexes.

Another study introduced Multi-agent Double Deep Q Network (MADDQN) architecture, which showed satisfactory results in the experiment on the mixed data of three major US market indexes (Huang et al., 2024). MADDQN consists of several independent trading agents trained using Double DQN and the final agent, which receives all decisions of individual trading agents as input and combines them to perform trading decisions. MADDQN showed an average cumulative return of 23.08% and performed better than baseline strategies.

Another step forward was taken through the development of a multi-agent trading system (Shavandi & Khedmati, 2022). The idea was to create a multi-agent Deep Reinforcement Learning framework, where every agent is an expert for a specific timeframe, and to verify if

the ensemble of agents performs better than a single agent using the data of the EUR/USD Forex market. The ensemble consisting of three trading agents, where each one specializes in long-term, mid-term, and short-term timeframes and is trained via a value-based DQN algorithm showed better performance compared to both single DQN agents specialized in the specific timeframe and the common benchmark trading strategies. The ensemble showed an average cumulative return of 56.4%, while the Buy and Hold baseline strategy only gained 5.4% over the same period from 2012 to 2021. Moreover, the ensemble managed to keep risks low by achieving the maximum drawdown of only -11.89%, while the Buy and Hold strategy had a loss of –21% during the same period.

Considering the above studies, we can assume Reinforcement Learning in the form of multi-agent ensemble Deep Q-Network agents can be a good model to mitigate drawdown risks and capture the upward trends in the cryptocurrency market. Another important detail is the ability to train and test Reinforcement Learning agents on 2017-2023 market data because the crypto market can be considered mature in this period.

3. METHODOLOGY

The end goal of the Methodology section is to describe the architecture of the ensemble of DQN agents, so let's explore the different RL algorithms to understand the interrelation between Markov Decision Processes, policy-based Reinforcement Learning, Q-Learning, and Deep Q-Network architecture.

3.1. Reinforcement Learning

The concept of Reinforcement learning (RL) was invented iteratively over decades by various prominent scientists, where Harry Klopf (1994), Andrew Barto and Richard Sutton (1997) are considered the most important contributors. Reinforcement learning is one of the three Machine Learning paradigms alongside Supervised Learning and Unsupervised Learning. Reinforcement Learning gathers multiple different algorithm variations under one term and the classification of algorithms is illustrated in Fig. 3.1. The RL algorithms can be divided into two categories – value-based and policy-based. In the value-based category, we are interested in the Q-learning algorithm, which makes it possible to learn the action-value function. In turn, the Deep Q-Network algorithm can be considered as an enhancement of Q-learning and introduces a way to approximate the action-value function with Deep Neural Networks (Qiu et al., 2023).

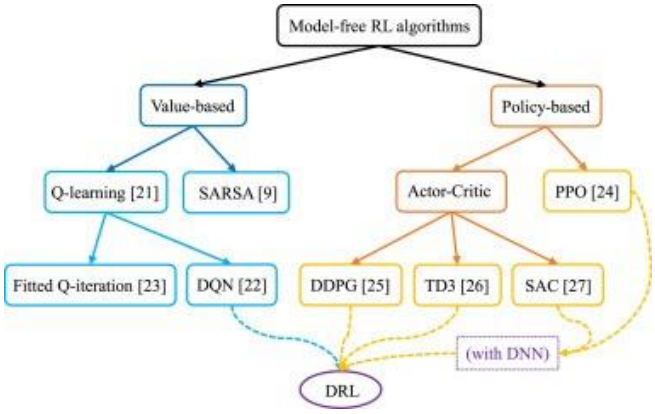

Fig. 3.1 – Classifications of single-agent reinforcement learning

algorithms (Qiu et al., 2023)

3.1.1. Markov Decision Process

Reinforcement Learning problems (Fig. 3.2) are typically formulated as Markov Decision Process (MDP), where MDP provides the mathematical framework to approach those problems.

The key components of MDP are formally described as <S, A, T, P, R> tuple (Adawadkar & Kulkarni, 2022), where:

- S represents a set of all possible states for an agent to be in
- A represents a set of possible actions an agent can take
- T represents the set of time steps, where a decision needs to be made
- P represents transition probabilities from state s to s' with action a : $P_a(s,\ s') = \Pr(S_{t+1} = s' \mid S_t = s,\ A_t = a|)$
- R represents the reward $R_a(s,\ s')$ for a s OBJ $s's'$

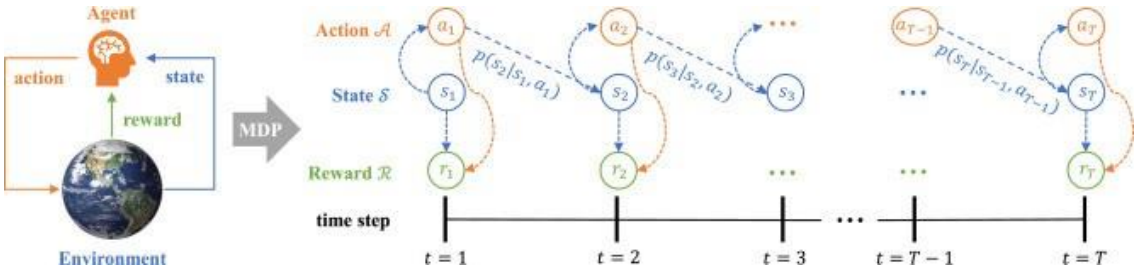

Fig. 3.2 – Agent–environment interaction and the Markov Decision Process (Qiu et al., 2023)

3.1.2. Reinforcement learning framework

The reinforcement learning framework enhances MDP by adding the following key components to learn through experience how to achieve specific tasks (Domingo Colomer et al., 2020):

- Environment
- Agent

The environment is everything the agent interacts with. It can be represented by the set of time steps, where the decision should be made at each step.

In turn, the agent can consist of multiple components, each of which is a function:

- Policies
- Value functions, Q function
- Models

- State update

The agent is a decision-maker, who follows specific policy π and transitions from state π to OBJ. At each given time step t ,s an agent will interpret the environment and will choose the best possible action according to its current policy π to maximize the reward OBJ. Action selection is based on a policy π, a learning agent aims to find the optimal policy π^*, which defines the probability of actions selection, so the sum of rewards is maximized.π in order to maximize the reward r . Action selection is based on a policy π, thus the aim of learning agent is to find the optimal policy π^*, which defines the probability of actions selection, so the sum of rewards is maximized.

The common example of model-free value-based policy learning algorithm is Q-learning (Fig. 3.3). In Q-learning, the Q function of the agent can be represented as a Q-table. Q-table is a data structure storing the Q-value for every possible state-action pair. Q-Learning iteratively updates the Q-table, whose goal is to approximate the Q-value function Q*(s, a). All elements in the table are initialized as zeroes and then are updated one element at a time (Qiu et al., 2023). In its turn, the Q-value represents the combination of instantaneous reward and possible future reward of the next states resulting from current action, assuming probabilistic system evolution with Markov Decision Processes (Zaparoli Cunha et al., 2023).

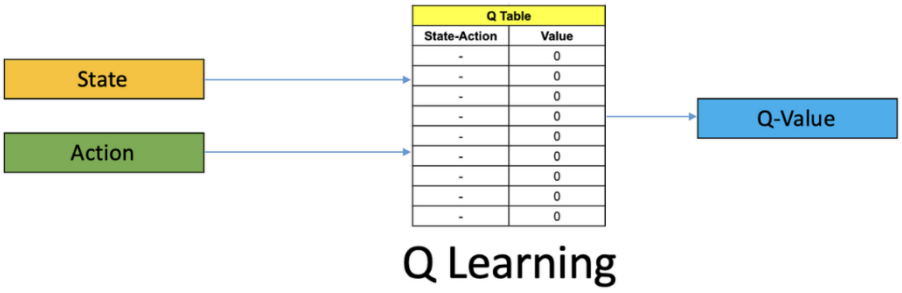

Fig. 3.3 – Q-learning Framework[1]

[1] https://becominghuman.ai/train-your-first-rl-agent-a-step-by-step-guide-353bced83722

3.2. Deep reinforcement learning

Deep Reinforcement Learning (DRL) is the advancement of RL, which combines Reinforcement Learning and Deep Learning approaches. DRL leverages Deep Neural Networks (Fig. 3.4) to improve the handling of complex relationships between the state, actions, and rewards. DRL agents proved to be good performers in various tasks including playing video games (Mnih et al., 2015) and controlling robots.

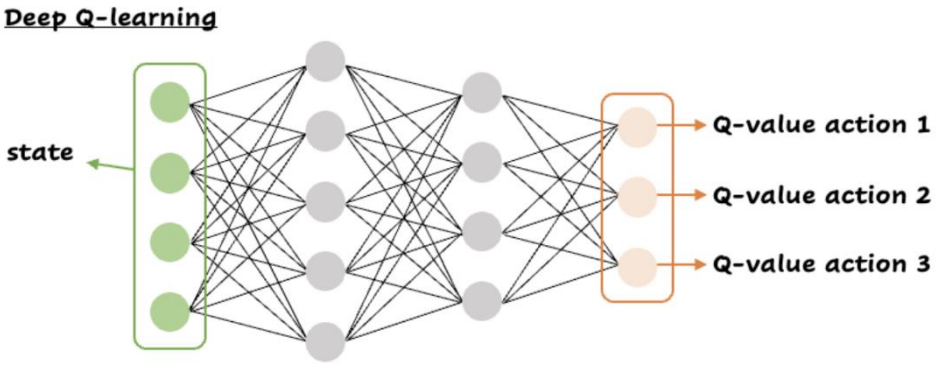

Fig. 3.4 – Deep Neural Network example[2]

An important advantage of the DQNs is Experience Replay, which helps to overcome the correlation between consecutive samples in traditional Q-learning by storing past experiences in the replay memory buffer. Additionally, the Deep Q-Network is optimized towards a frozen target network that is periodically updated with the latest weights at every step, which makes training more stable by preventing short-term oscillations from a moving target (Mnih et al., 2015).

3.2.1. Design of the single DQN agent

In this case, the DQN Agent is a Sequential model with a plain stack of layers where each layer has exactly one input tensor and one output tensor (Fig. 3.5). The DQN Agent will have the following stack of layers (Carta et al., 2021):

- One flatten layer consisting of 68 neurons processing multiple timeframe data
- One fully-connected layer with 35 neurons and LeakyRelu activation
- One fully-connected layer with 3 neurons representing actions and Linear activation

[2] https://towardsdatascience.com/techniques-to-improve-the-performance-of-a-dqn-agent-29da8a7a0a7e

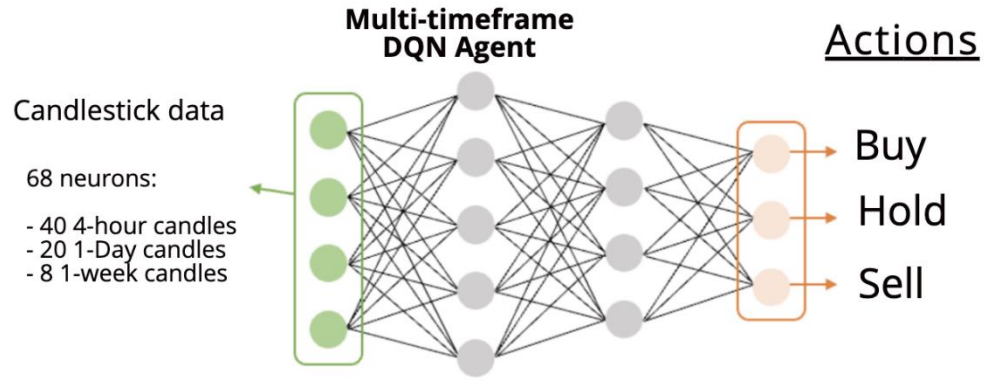

Fig. 3.5 – DNN's architecture of the DQN Agent for multi-timeframe trading[3]

Considering studies of Double DQN by (Van Hasselt et al., 2016) and Dueling DQN by (Wang et al., 2016), Double Dueling DQN should be used to improve the performance of DQN agent. The standard DQN struggles with overestimation issues, while Double DQN mitigates the overestimation bias found in the original DQN approach and enhances learning stability and policy performance. In addition, Dueling DQN introduces a novel network architecture that more accurately estimates state values and 2 advantage values. By explicitly distinguishing between state values and action advantages in the network structure, Dueling DQN provides a more granular approach to Q-value estimation, proving more efficient and effective for numerous decision-making tasks (G. Hu, 2023).

The final architecture of the Deep Reinforcement Learning framework can be seen in Fig. 3.6.

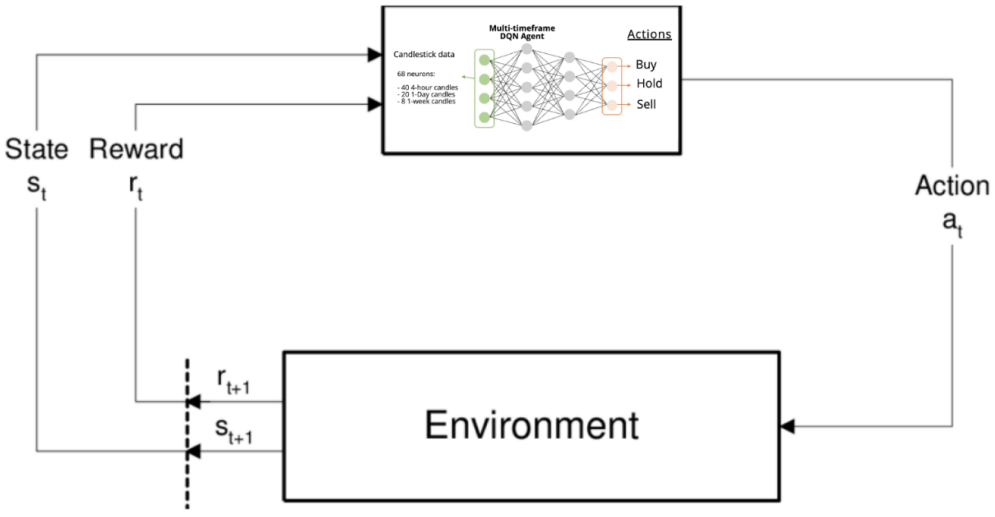

Fig. 3.6 – Deep Reinforcement Learning framework with DQN Agent

[3] https://towardsdatascience.com/techniques-to-improve-the-performance-of-a-dqn-agent-29da8a7a0a7e

3.3. The ensemble of DQN agents

The further improve decision-making and achieve better results in risk management, we propose to aggregate multiple DQN agents into one ensemble. The resulting ensemble will have agents trained at different epochs; therefore, we expect every agent to have different experiences due to the randomness of the training process (Carta et al., 2021). Every epoch contains all decisions taken by the agent during its training process and is stored in a file (Fig. 3.7), which allows an application of the dynamic agreement threshold.

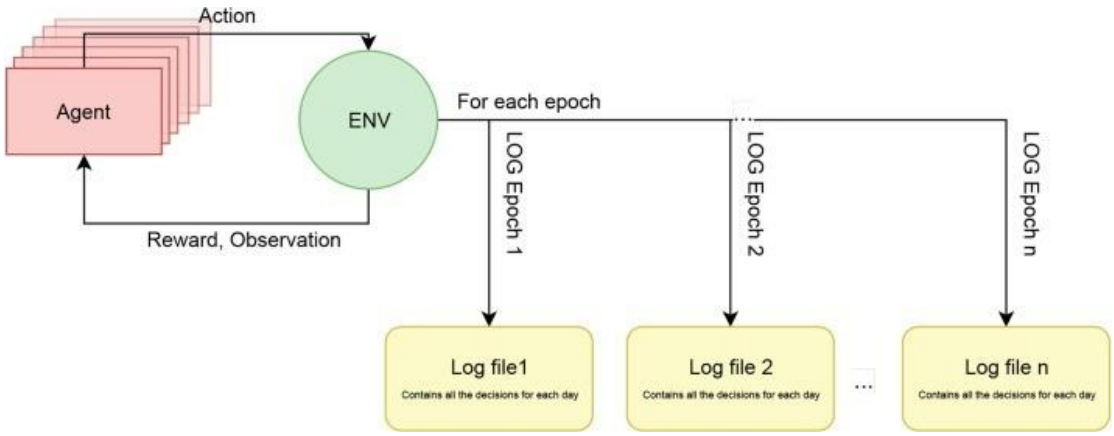

Fig. 3.7 - A proposed ensemble of deep reinforcement learning agents (Carta et al., 2021)

Considering the exponential growth of the cryptocurrency market, we decided to train Only-Long Agent, which is configured to either take Long actions or opt-out from any action. The approach described allows us to control the percentage of agreement threshold and to experiment with assembling ensembles with a different agreement threshold, while the agents are trained only once and stay the same between differently assembled ensembles (Fig. 3.8).

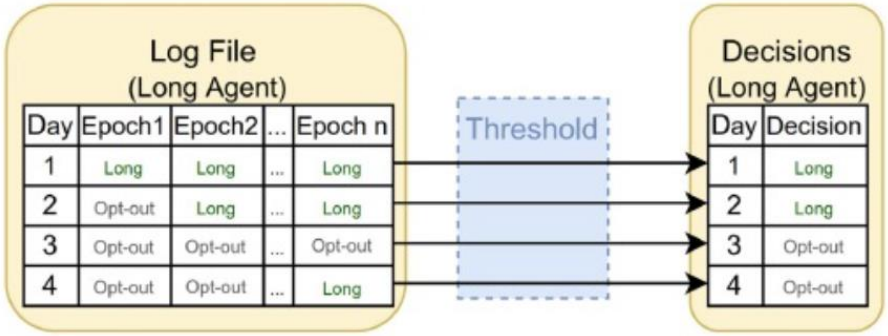

Fig. 3.8 - The agreement threshold allows to control of the resulting decision of the whole ensemble for a specific time step (Carta et al., 2021)

3.4. Dataset

To get cryptocurrency trading data, the Binance API was used to fetch cryptocurrency prices for a specific period in the form of candlesticks. That data will be used to assess the performance of our Deep Reinforcement Learning agent. Experiments will be conducted using data from September 2017 to February 2024 for the major cryptocurrency – Bitcoin. The dataset will be a list of candles, where each candle is represented by the following data: [timestamp, open_price, high_price, low_price, close_price, volume]. In simple terms, every candle is a list of numbers, representing the important trading metrics of the assets on a specific timeframe.

The dataset will have three different timeframes to help trading agents navigate better in complex trading environment – 1 Week, 1 Day, and 4 Hour candlesticks.

The sample of the weekly dataset can be visualized as the following weekly candlesticks chart (Fig. 3.7)

Fig. 3.7 – Weekly candlesticks chart on TradingView, data from Binance[4]

[4] https://www.tradingview.com/symbols/BTCUSDT/

The most crucial dataset is the daily Bitcoin timeframe, and its structure can be seen in Table 3.1:

Date	Time	Open	High	Low	Close
08/17/2017	00:00	4261.48	4485.39	4200.74	4285.08
08/18/2017	00:00	4285.08	4371.52	3938.77	4108.37
08/19/2017	00:00	4108.37	4184.69	3850	4139.98
08/20/2017	00:00	4120.98	4211.08	4032.62	4086.29
08/21/2017	00:00	4069.13	4119.62	3911.79	4016
08/22/2017	00:00	4016	4104.82	3400	4040

Table 3.1 – A sample of daily Bitcoin candlesticks dataset

3.5. Experimental procedure

The proposed approach is to apply Deep Reinforcement Learning framework with a Deep Q-Network Agent available to process multiple timeframes of trading data to learn from trading data and achieve two goals:

- Beat benchmark strategy (Buy and Hold)
- Minimize risks of having significant drawdowns (measured by Sharpe ratio)

To train and validate the agent it's needed to split the data and apply walk-forward validation (Oyewola et al., 2022). The most common approach for Deep Learning training is cross-validation and it is widely used for model assessment and selection. However, classic cross-validation is not suitable for time-series data, because cross-validation expects every observation to be independent and evenly distributed. In the case of cross-validation usage, models can train on the future data, which is an incorrect approach. The walk-forward validation allows to train and validate the agent on all data and does not expose future data during the training and validation phases (Fig. 3.8). In walk-forward validation, instead of splitting data randomly, it is divided into consecutive intervals. Then earlier intervals can be used for model training, while the later intervals are used for testing (Beniwal et al., 2023).

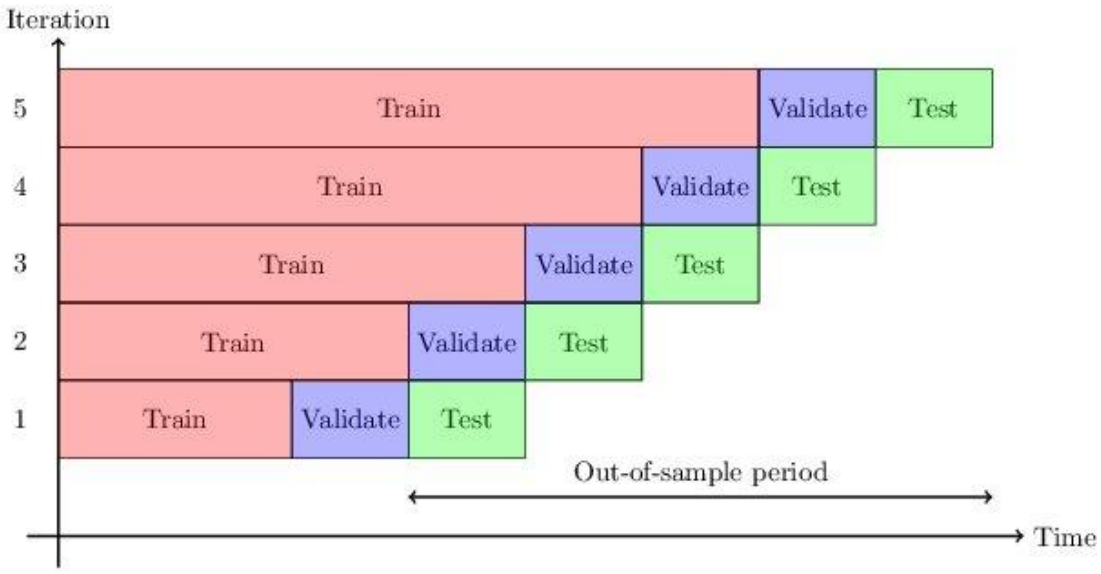

Fig. 3.8 – Forward validation (Chalkidis & Savani, n.d.)

Considering our dataset has 2400 days (about 6 and a half years), it is possible to dedicate 1300 days (about 3 years) for training, 130 days (about 4 months) for the validation dataset, and 130 days (about 4 months) for the test dataset. The split with this proportion allows us to have 4 walks for an agent to learn from the data (Table 3.2).

	Walk 1	Walk 2	Walk 3	Walk 4	Walk 5	Walk 6	Walk 7
Train	0-1300 days	130-1430 days	260-1560 days	390-1690 days	520-1830 days	650-1960 days	780-2090 days
Validation	1301-1430 days	1431-1560 days	1561-1690 days	1691-1830 days	1831-1960 days	1961-2090 days	2091-2220 days
Test	1431-1560 days	1561-1690 days	1691-1830 days	1831-1960 days	1961-2090 days	2091-2220 days	2221-2350 days

Table 3.2 – Forward validation example for daily Bitcoin dataset

3.5. Evaluation metrics

In this section, we describe the evaluation metrics measured to evaluate the performance of DQN trading agent.

3.5.1. Accuracy

The accuracy metric gives information about the number of actions correctly classified compared to the total number of them. Considering there are 2400+ trading days and the

agent took 1200 decisions to open long positions, if only 600 decisions end up as a profitable action, accuracy will be equal to 50%. The accuracy formula is:

$$Accuracy(X') = \frac{X^{(+)}}{|X'|} \text{ (1)}$$

where $|X'|$ is trading days when long or short actions were performed and $X^{(+)}$ stands for the number of trading days with correctly classified price action. As a practical example, consider the daily open at 60,000$ per Bitcoin and the daily close at 63000$, which results in 5% price growth over the day. If the trading agent opened the long position on that day, we could consider the agent predicting the price action on that day correctly.

3.5.2. Coverage

The coverage metric is important to give us an insight into the agent's behavior because it visualizes the tendency of the agent to be over- or under-trading. The coverage metric reports the percentage of times the trading agent took an action (opened Long or Short position) in the total number of days the agent was present in the market.

3.5.3. Cumulative reward

The reward metric shows the cumulative Return on investment percentage (ROI %) of the opened positions in the market. It represents a sum of the reward, which can be negative or positive, for every opened position divided by the sum of the capital involved.

3.5.4. Maximum Drawdown

A drawdown is a metric used to measure the risk of the trading strategies and is calculated like the following:

$$MaxDrawdown_t = \min_{0 \le p \le t} \left(\sum_{p=0}^{t} Reward_p \right) \text{ (2)}$$

Where $Reward_p$ is the cumulative reward at the time *p.* Trading strategy can be considered successful in case it manages to keep Maximum drawdown relatively small compared to its returns. In other words, this metric shows the ability of the trading agent to avoid significant downside risks.*p.* Trading strategy can be considered successful in case it manages to keep Maximum drawdown relatively small compared to its returns. In other words, this metric shows the ability of the trading agent to avoid significant downside risks.

4. EMPIRICAL STUDY

The results were obtained by training the ensemble of agents for 100 epochs and then evaluating the assembled ensemble on the test data with different agreement thresholds applied. Every agent in the ensemble was trained to perform only long or opt-out actions. The final training metrics for all walks executed on the Bitcoin dataset are presented in Fig. 4.1, and Fig. 4.2.

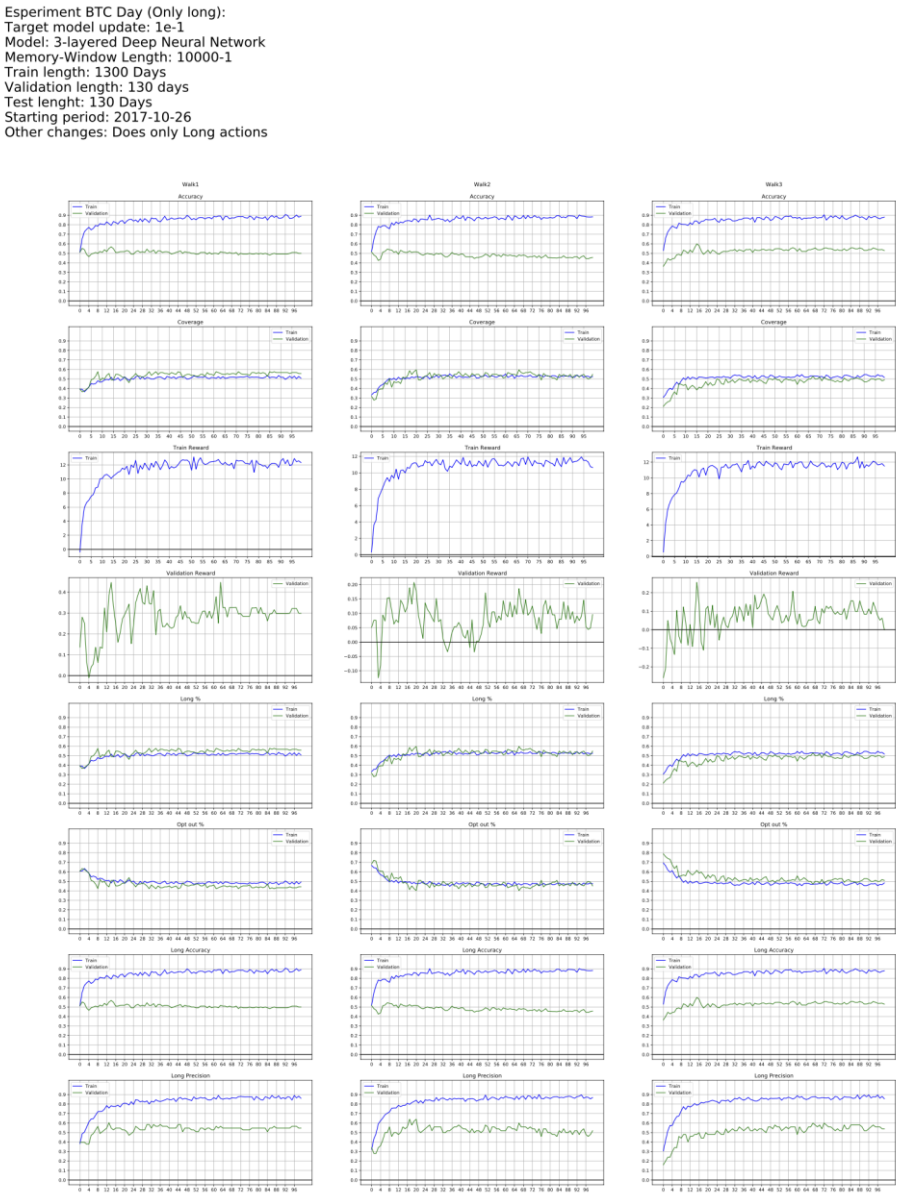

Fig. 4.1 – The metrics for walks #1-3 executed during the training of the ensemble of DQN agents

The accuracy metric chart shows that the model is converging well during the training period but still has a low level of accuracy in the validation environment. The accuracy metric improved rapidly during the first 15 training epochs and then finished in the range between

80% and 90%. The validation accuracy oscillated around 50%, which proves the agent might not be generalized and perform worse than expected in an unknown environment after training. This should not be a huge problem, because the price action of the validation dataset is usually significantly different from the training dataset because of market volatility.

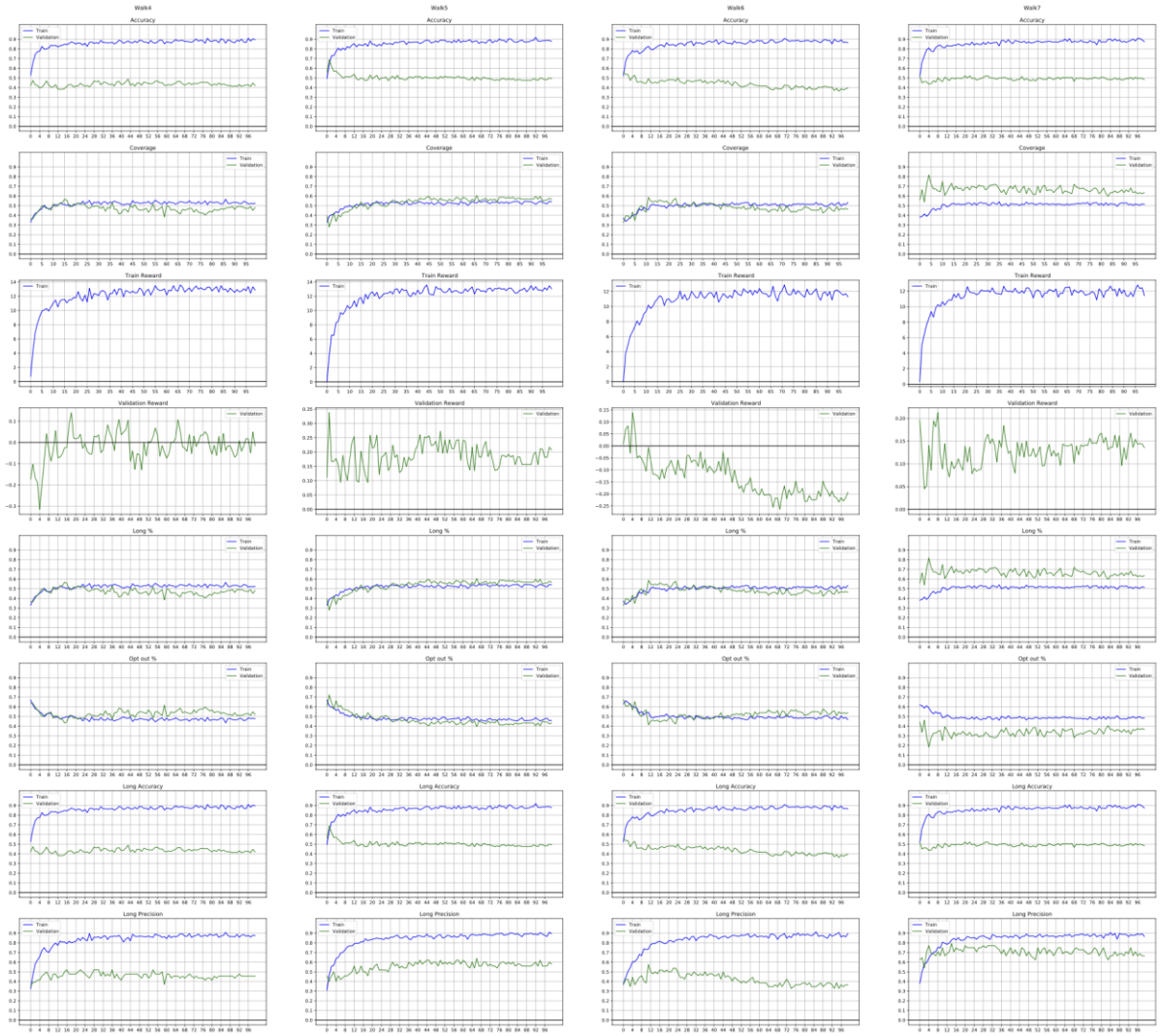

Fig. 4.2 – The metrics for the walks #4-7 executed during the training of the ensemble of DQN agents

The coverage metric on average between all walks went from 40% to 50% during the first 10 training epochs and then it oscillated around 50% for the rest of the training period. In turn, the coverage metric for the validation dataset ranged between 40% and 70% depending on the walk, which might indicate the ensemble behaves more actively in case the price action of the validation dataset is more like the training dataset.

The train reward metric rapidly increased during the first 15 epochs for all walks and then stalled for the remaining epochs, which indicates ensemble learns well how to perform in the training environment. On the opposite side, the validation reward metric was mostly ranging for all walks, which indicates the ensemble did not generalize well and performed suboptimal actions in that environment.

The metrics Long% and Hold% are mirroring each other, Long% metric increases during the first 15 epochs and then stabilizes at 50%, while Opt out% metric slightly decreases. It illustrates the ability of the agent to learn the proper actions in the environment and do more appropriate actions.

Let's consider two similar ensembles, where the only difference is the agreement threshold. Full ensemble (E100), where threshold agreement is 100%, shows slightly better performance in the test environment (Fig. 4.3).

FULL ENSEMBLE

Valid

Iteration	Reward%	#Wins	#Losses	Dollars	Coverage	Accuracy
0	0.05	10	10	2437.33	0.15	0.5
1	0.06	4	5	2752.08	0.07	0.44
2	0.01	3	2	644.21	0.04	0.6
3	-0.25	2	8	-4656.41	0.08	0.2
4	0.11	7	5	2447.34	0.09	0.58
5	-0.08	7	7	-2249.43	0.11	0.5
6	0.03	12	8	1096.75	0.15	0.6
sum	0.01	45	45	2471.87	0.1	0.5

Test

Iteration	Reward%	#Wins	#Losses	Dollars	Coverage	Accuracy
0	0.11	4	3	4522.96	0.05	0.57
1	-0.17	5	7	-5079.13	0.09	0.42
2	-0.05	4	6	-1098.99	0.08	0.4
3	-0.06	10	7	-1358.47	0.13	0.59
4	0.03	9	4	897.37	0.1	0.69
5	0.06	11	5	1989.4	0.12	0.69
6	0.05	9	12	2823.14	0.16	0.43
sum	0.01	52	44	2696.28	0.11	0.54

Fig. 4.3 – Full ensemble of only long agents' statistics for validation and test datasets

The ensemble, which has a 90% agreement threshold (E90), does surprisingly well in the validation environment, however, performs even worse than E100 in the test environment (Fig. 4.4).

90% ENSEMBLE

Valid

Iteration	Reward%	#Wins	#Losses	Dollars	Coverage	Accuracy
0	0.19	23	21	9477.85	0.34	0.52
1	0.1	16	15	4181.69	0.24	0.52
2	0.07	16	12	1985.87	0.22	0.57
3	0.05	14	16	992.98	0.23	0.47
4	0.08	20	21	1861.66	0.32	0.49
5	-0.22	12	21	-5588.36	0.25	0.36
6	0.14	25	26	4638.82	0.39	0.49
sum	0.08	126	132	17550.51	0.28	0.49

Test

Iteration	Reward%	#Wins	#Losses	Dollars	Coverage	Accuracy
0	-0.1	14	15	-4483.28	0.22	0.48
1	-0.41	20	21	-12007.77	0.32	0.49
2	0.09	17	17	1676.21	0.26	0.5
3	0.06	24	22	1378.63	0.35	0.52
4	-0.07	17	14	-1874.5	0.24	0.55
5	0.03	19	18	902.7	0.28	0.51
6	0.27	24	20	15683.26	0.34	0.55
sum	0.01	135	127	1275.25	0.29	0.52

Fig. 4.4 – 90% ensemble of only long agents' statistics for validation and test datasets

The ensemble, having an 80% threshold level, showed slightly better performance in the validation environment, and even worse performance in the test environment, which resulted in a negative reward summary (Fig. 4.5).

80% ENSEMBLE

Valid

Iteration	Reward%	#Wins	#Losses	Dollars	Coverage	Accuracy
0	0.19	25	24	9522.5	0.38	0.51
1	0.06	21	23	2680.28	0.34	0.48
2	0.12	22	18	3520.35	0.31	0.55
3	0.11	16	19	2097.73	0.27	0.46
4	0.07	22	26	1721.08	0.37	0.46
5	-0.22	15	24	-5566.21	0.3	0.38
6	0.12	30	30	4070.08	0.46	0.5
sum	0.08	151	164	18045.81	0.35	0.48

Test

Iteration	Reward%	#Wins	#Losses	Dollars	Coverage	Accuracy
0	-0.16	17	21	-6833.22	0.29	0.45
1	-0.58	24	28	-17564.04	0.4	0.46
2	0.13	19	21	2450.53	0.31	0.47
3	0.09	26	25	2069.58	0.39	0.51
4	-0.15	18	18	-3857.56	0.28	0.5
5	0.01	24	24	463.79	0.37	0.5
6	0.21	26	24	12820.86	0.38	0.52
sum	-0.04	154	161	-10450.06	0.35	0.49

Fig. 4.5 – 80% ensemble of only long agents' statistics for validation and test datasets

5. RESULTS AND DISCUSSION

In the above experiment, we trained the Double Dueling Deep Q Network (DD-DQN) agents and created various ensembles with different agreement thresholds. In general, all ensembles showed better performance in the validation environment and significantly worse performance in the test environment. Even though the reward summary in the validation environment was positive, it was significantly worse compared to the Buy and Hold baseline strategy.

One can notice the proposed DD-DQN ensemble is not learning properly to benefit from massive Bitcoin market uptrends. At the same time, the DD-DQN ensemble managed to avoid severe drawdowns, which can be considered a noteworthy achievement.

The possible things to improve in future work could be the enhancement of DD-DQN architecture, like changing the number of layers in DQN, training every agent for every timeframe separately, and increasing the number of candles the agent receives as input. Another way to improve the results can be enhancements of the environment, like the introduction of a stop-loss mechanism, which might improve cumulative reward and help to beat the baseline Buy and Hold strategy.

Another possible improvement can be the analysis of the agent's actions in specific conditions and training the agent to classify different market phases (namely uptrend, downtrend, and ranging markets) and adjust its actions accordingly.

One more improvement could be adding some synthetic trading data because 7 years of trading data might not be enough to properly train the agent.

6. CONCLUSIONS AND FUTURE WORKS

The thesis explored the design and training of the DDDQN agents' ensemble, which demonstrated good risk management skills but is not yet good at capturing and benefiting from the uptrend periods of the markets.

The thesis presents the following opportunities for future work:

- DNN architecture: Deep Neural Network with 68 neurons input layer might not be the optimal design to perform well in volatile market environments, one can experiment by adding more input neurons, thus allowing a trading agent to have more information about the environment before taking the next action
- Improvement of rewards: currently we only give the reward to an agent in the form of profit or loss for a particular action, while the reward can also include feedback on maximum drawdown and maximum profit during the training period
- Adding indicators data: in the current experiment we supplied only the raw data to train the agent; one can try to improve the agent's performance by calculating and supplying indicator data like Relative Strength Index, Moving Averages, etc in addition to raw candle data

BIBLIOGRAPHICAL REFERENCES

AbdelKawy, R., Abdelmoez, W. M., & Shoukry, A. (2021). A synchronous deep reinforcement learning model for automated multi-stock trading. *Progress in Artificial Intelligence*, *10*(1), 83–97. https://doi.org/10.1007/s13748-020-00225-z

Adawadkar, A. M. K., & Kulkarni, N. (2022). Cyber-security and reinforcement learning — A brief survey. *Engineering Applications of Artificial Intelligence*, *114*, 105116. https://doi.org/10.1016/J.ENGAPPAI.2022.105116

Barto, A. G., & Sutton, R. S. (1997). Reinforcement Learning in Artificial Intelligence. *Advances in Psychology*, *121*(C), 358–386. https://doi.org/10.1016/S0166-4115(97)80105-7

Beniwal, M., Singh, A., & Kumar, N. (2023). Forecasting long-term stock prices of global indices: A forward-validating Genetic Algorithm optimization approach for Support Vector Regression. *Applied Soft Computing*, *145*, 110566. https://doi.org/10.1016/J.ASOC.2023.110566

Boďa, M., & Kanderová, M. (2014). Linearity of the Sharpe-Lintner Version of the Capital Asset Pricing Model. *Procedia - Social and Behavioral Sciences*, *110*, 1136–1147. https://doi.org/10.1016/J.SBSPRO.2013.12.960

Borrageiro, G., Firoozye, N., & Barucca, P. (2022). The Recurrent Reinforcement Learning Crypto Agent. *IEEE Access*, *10*, 38590–38599. https://doi.org/10.1109/ACCESS.2022.3166599

Carta, S., Ferreira, A., Podda, A. S., Reforgiato Recupero, D., & Sanna, A. (2021). Multi-DQN: An ensemble of Deep Q-learning agents for stock market forecasting. *Expert Systems with Applications*, *164*, 113820. https://doi.org/10.1016/J.ESWA.2020.113820

Chaboud, A. P., Chiquoine, B., Hjalmarsson, E., & Vega, C. (2014). Rise of the machines: Algorithmic trading in the foreign exchange market. *Journal of Finance*, *69*(5), 2045–2084. https://doi.org/10.1111/jofi.12186

Chakole, J. B., Kolhe, M. S., Mahapurush, G. D., Yadav, A., & Kurhekar, M. P. (2021). A Q-learning agent for automated trading in equity stock markets. *Expert Systems with Applications*, *163*, 113761. https://doi.org/10.1016/J.ESWA.2020.113761

Chalkidis, N., & Savani, R. (n.d.). *Trading via Selective Classification; Trading via Selective Classification*. https://www.researchgate.net/publication/355730698

Chao, X., Kou, G., Li, T., & Peng, Y. (2018). Jie Ke versus AlphaGo: A ranking approach using decision making method for large-scale data with incomplete information. *European Journal of Operational Research*, *265*(1), 239–247. https://doi.org/10.1016/J.EJOR.2017.07.030

Chen, H. H., Yang, C. B., & Peng, Y. H. (2014). The trading on the mutual funds by gene expression programming with Sortino ratio. *Applied Soft Computing*, *15*, 219–230. https://doi.org/10.1016/J.ASOC.2013.09.011

Christie, N., Jankensgård, H., & Marinelli, N. (2024). The Black Swan problem: The role of capital, liquidity and operating flexibility. *International Review of Financial Analysis*, *91*, 103024. https://doi.org/10.1016/J.IRFA.2023.103024

Dempster, M. A. H., & Leemans, V. (2006). An automated FX trading system using adaptive reinforcement learning. *Expert Systems with Applications*, *30*(3), 543–552. https://doi.org/10.1016/j.eswa.2005.10.012

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. https://doi.org/10.1109/TNNLS.2016.2522401

Domingo Colomer, L., Skotiniotis, M., & Muñoz-Tapia, R. (2020). Reinforcement learning for optimal error correction of toric codes. *Physics Letters A*, *384*(17), 126353. https://doi.org/10.1016/J.PHYSLETA.2020.126353

Felizardo, L. K., Lima Paiva, F. C., de Vita Graves, C., Matsumoto, E. Y., Costa, A. H. R., Del-Moral-Hernandez, E., & Brandimarte, P. (2022). Outperforming algorithmic trading reinforcement learning systems: A supervised approach to the cryptocurrency market. *Expert Systems with Applications*, *202*. https://doi.org/10.1016/j.eswa.2022.117259

Grudniewicz, J., & Ślepaczuk, R. (2023). Application of machine learning in algorithmic investment strategies on global stock markets. *Research in International Business and Finance*, *66*. https://doi.org/10.1016/j.ribaf.2023.102052

Hilliard, J. E., & Hilliard, J. (2018). Rebalancing versus buy and hold: theory, simulation and empirical analysis. *Review of Quantitative Finance and Accounting*, *50*(1), 1–32. https://doi.org/10.1007/s11156-017-0621-5

Hu, G. (2023). *Advancing Algorithmic Trading: A Multi-Technique Enhancement of Deep Q-Network Models*. http://arxiv.org/abs/2311.05743

Hu, Y., Feng, B., Zhang, X., Ngai, E. W. T., & Liu, M. (2015). Stock trading rule discovery with an evolutionary trend following model. *Expert Systems with Applications*, *42*(1), 212–222. https://doi.org/10.1016/j.eswa.2014.07.059

Huang, Y., Lu, X., Zhou, C., & Song, Y. (2023). DADE-DQN: Dual Action and Dual Environment Deep Q-Network for Enhancing Stock Trading Strategy. *Mathematics*, *11*(17), 3626. https://doi.org/10.3390/math11173626

Huang, Y., Zhou, C., Cui, K., & Lu, X. (2024). A multi-agent reinforcement learning framework for optimizing financial trading strategies based on TimesNet. *Expert Systems with Applications*, *237*. https://doi.org/10.1016/j.eswa.2023.121502

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. https://doi.org/10.1016/J.ESWA.2018.09.036

Klopf, A. H. (1994). Drive-Reinforcement Learning and Hierarchical Networks of Control Systems as Models of Nervous System Function. *IFAC Proceedings Volumes*, *27*(1), 511–515. https://doi.org/10.1016/S1474-6670(17)46320-8

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., & Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, *518*(7540), 529–533. https://doi.org/10.1038/nature14236

Moody, J., & Saffell, M. (2001). Learning to Trade via Direct Reinforcement. In *IEEE TRANSACTIONS ON NEURAL NETWORKS* (Vol. 12, Issue 4).

Oyewola, D. O., Dada, E. G., & Ndunagu, J. N. (2022). A novel hybrid walk-forward ensemble optimization for time series cryptocurrency prediction. *Heliyon*, *8*(11), e11862. https://doi.org/10.1016/J.HELIYON.2022.E11862

Qiu, D., Wang, Y., Hua, W., & Strbac, G. (2023). Reinforcement learning for electric vehicle applications in power systems:A critical review. *Renewable and Sustainable Energy Reviews*, *173*, 113052. https://doi.org/10.1016/J.RSER.2022.113052

Shadabfar, M., & Cheng, L. (2020). Probabilistic approach for optimal portfolio selection using a hybrid Monte Carlo simulation and Markowitz model. *Alexandria Engineering Journal*, *59*(5), 3381–3393. https://doi.org/10.1016/J.AEJ.2020.05.006

Shavandi, A., & Khedmati, M. (2022). A multi-agent deep reinforcement learning framework for algorithmic trading in financial markets. *Expert Systems with Applications*, *208*. https://doi.org/10.1016/j.eswa.2022.118124

Silva, E. M., Moreira, R. de L., & Bortolon, P. M. (2023). Mental Accounting and decision making: a systematic literature review. *Journal of Behavioral and Experimental Economics*, *107*, 102092. https://doi.org/10.1016/J.SOCEC.2023.102092

Théate, T., & Ernst, D. (2021). An application of deep reinforcement learning to algorithmic trading. *Expert Systems with Applications*, *173*. https://doi.org/10.1016/j.eswa.2021.114632

Ţiţan, A. G. (2015). The Efficient Market Hypothesis: Review of Specialized Literature and Empirical Research. *Procedia Economics and Finance*, *32*, 442–449. https://doi.org/10.1016/s2212-5671(15)01416-1

Van Hasselt, H., Guez, A., & Silver, D. (2016). Deep reinforcement learning with double Q-Learning. *30th AAAI Conference on Artificial Intelligence, AAAI 2016*, 2094–2100. https://doi.org/10.1609/aaai.v30i1.10295

Wang, Z., Schaul, T., Hessel, M., & Lanctot, M. (2016). *Dueling Network Architectures for Deep Reinforcement Learning Hado van Hasselt*. https://www.youtube.com/playlist?list=

Yamani, E. (2023). Return–volume nexus in financial markets: A survey of research. *Research in International Business and Finance*, *65*, 101910. https://doi.org/10.1016/J.RIBAF.2023.101910

Ye, Z. J., & Schuller, B. W. (2023). Human-aligned trading by imitative multi-loss reinforcement learning. *Expert Systems with Applications*, *234*, 120939. https://doi.org/10.1016/J.ESWA.2023.120939

Zaparoli Cunha, B., Droz, C., Zine, A. M., Foulard, S., & Ichchou, M. (2023). A review of machine learning methods applied to structural dynamics and vibroacoustic. *Mechanical Systems and Signal Processing*, *200*, 110535. https://doi.org/10.1016/J.YMSSP.2023.110535

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa