School of Informatics

Informatics Research Review
Using deep reinforcement learning for automated low-frequency quantitative stock trading

[redacted]

January 2021

Abstract

Algorithmic stock trading has become a staple in today's financial market, the majority of trades being now fully automated. Deep Reinforcement Learning (DRL) agents proved to be to a force to be reckon with in many complex games like Chess and Go. We can look at the stock market historical price series and movements as a complex imperfect information environment in which we try to maximize return - profit and minimize risk. This paper reviews the progress made so far with deep reinforcement learning in the subdomain of AI in finance, more precisely, automated low-frequency quantitative stock trading. Many of the reviewed studies had only proof-of-concept ideals with experiments conducted in unrealistic settings and no real-time trading applications. For the majority of the works, despite all showing statistically significant improvements in performance compared to established baseline strategies, no decent profitability level was obtained. Furthermore, there is a lack of experimental testing in real-time, online trading platforms and a lack of meaningful comparisons between agents built on diferent types of DRL or human traders. We conclude that DRL in stock trading has showed huge applicability potential rivaling professional traders under strong assumptions, but the research is still in the very early stages of development.

Date: Friday 22nd January, 2021

Supervisor: [redacted]

1 Introduction

Stock trading is the buying and selling of a company's shares within the financial market. The goal is to optimize the return on the investment of the capital exploiting the volatility of the market through repetitive buy and sell orders. Profit is generated when one buys at a lower price than it sells on afterwards. Among the key challenges in the finance world, the increasing complexity and the dynamical property of the stock markets are the most notorious. Stiff trading strategies designed by experts in the field often fail to achieve profitable returns in all market conditions [1]. To address this challenge, algortihmic trading strategies with DRL are proposed.

Although automated low-frequency quantitative stock trading may seem like a fancy choice of words, it means something really simple. Automated refers to the fact that we are targeting algorithmic solutions, with very little to no human intervention. Quantitative stock trading refers to the fact that we want the trading agent developed to be able to take its decisions based on historical data, on quantitative analysis, that is based on statistical indicators and heuristics to dechiper patterns in the market and take appropiate actions [1]. Finally, low-frequency trading assures that the solutions that we seek resemble agents which can trade like a human being from 1 minute to a few days' timeframe. This choice is natural and does not narrow the literature down that much, as most high-frequency trading systems act at the level of pico-second or milli-seconds and the studies in this area are usually owned by high-profile companies which try to keep their research hidden. The hardware for such studies is also really limited and represents a reason for such a shortage of studies in high-frequency trading because it does not require only a few very good GPUs but also really small latency devices to connect to live datacenters that contain active stock prices.

For the scope of this paper, there are further limitations that should be implied by the title and the abstract but nevertheless, they will be mentioned here. Our attention is focused on studies building an end-to-end trading agent using DRL as a core learning paradigm. We won't discuss papers debating on forecasting stock prices or the direction of the market with deep learning. Although it is clear that one can develop a good trading agent if such a system would be successful, usually, stock prices are very volatile and can't be predicted in such ways [2] leaving the job to a risk management system to do the work. From the same reason, we won't mention papers that try to combine meta-heuristic algorithms (like Genetic Algorithms - GA [3] or Particle Swarm Optimization - PSO [4]) with Reinforcement learning (RL) or multi-agent solutions to this problem. We also rule out the portfolio management part [5] because it is entirely a different system that should be taken into consideration separately when constructing a trading engine [6, 7]. We focus on the actual agent - learner that makes the trades, the allocation of funds is not a priority to review in this paper.

It is no secret that the main theme of this paper is the applicability of AI in finance, further narrowed down by multiple research questions. As we will see, most of the work in this area is fairly new. The literature can answer the question quite well, as several benchmarks have been a staple for quite some time in algorithmic trading, like yearly return rate on a certain timeframe or cumulative wealth [7] or Sharpe ratio [8]. An issue that is usually problematic here relates to performance comparisons that can be inconclusive. Obviously, some papers test their proposed approaches on different environments and different timelines and may use a different metric for assessing performance, but a common ground can be found if similar settings are used. Moreover, it is not futile to assume that many studies from the literature are kept private in this area, as it is a way to generate a lot of profit, therefore, state-of-the-art methods can hardly be called out; that's why we are going to divide this review paper according to the main idea of the methods used to solve the problem: critic-only RL, actor-only RL and actor-critic RL approaches, which is actually a well-defined categorization of RL in trading [9].

There is an obvious question that this paper tries to answer, that being: *can we make a success-*

ful trading agent which plays on the same timeframes as a regular human trader using the RL paradigm making use of deep neural networks? We explore solutions presenting multiple papers in the current literature that propose approaches to the problem and actual concrete agents that we will be able to assess in order to define the progression on this field of study. We conclude that DRL has huge potential in being applying in algorithmic trading from minute to daily timeframes, several prototype systems being developed showing great success in particular markets. A secondary question that arises naturally from this topic is: *can DRL lead to a super-human agent that can beat the market and outperform professional human traders?* The motivation for considering such a question is that, recently, it was showed that RL systems can outperform experts in conducting optimal control policies [10], [11] or can outperform even the best human individuals in complex games [12]. We shall also try to respond to this question to some extent; we provide a partial answer supported by particular studies [13] where there have been DRL agents developed for real-time trading or for realistic back-testing [14] that can, in theory, allow for such a comparison. We infer that DRL systems can compete with professional traders with respect to the risk-adjusted return rates on short (15-minute) or long (daily) timeframes in particular hand-picked markets, although further research on this needs to be done in order to decipher the true extent of DRL power in this environment.

2 Background

In this section, we provide the necessary background for the reader to get a grasp of some basic concepts we shall recall throughout this review paper.

2.1 Reinforcement Learning

Reinforcement learning [15] is considered to be the third paradigm of learning, alongside supervised and unsupervised learning. In contrast with the other two, reinforcement learning is used to maximize the expected reward when in an environment usually modelled as a Markov Decision Process (MDP). To be more precise, having as input a state representation of the environment and a pool of actions that can be made, we reach the next state with a certain numerical reward of applying a picked action. In order to approximate the expected reward from a state (or the **value** of a state), an action-value function Q can be used. This takes as input a state and a possible action and outputs the expected Q value [16] - the reward that we can expect from that state forward - the cumulative future rewards. In the literature, the action-value function is referred to as the **critic**, one might say it's because it "critiques" the action made in a state with a numerical value. However, RL can directly optimize the policy that the agent needs to take in order to maximize reward. If we view the policy as a probability distribution over the whole action space, then policy gradient (PG) methods [17] can be applied to optimize that distribution so that the agent chooses the action with the highest probability, amongst all possible actions, that gives the highest reward. Judging by the fact that this method directly controls to policy, in the literature, it is referred as the **actor**. RL employs these two approaches in the literature to form 3 main techniques: critic-only, actor-only, actor-critic learning. In the critic-only, we use only the action-value function to make decisions. We can take a greedy approach - always choosing the action that gives the max possible future reward, or we can choose to explore more. Note that the Q function can be approximated in many ways, but the most popular and successful approach is with a neural network (Deep Q Network - DQN [10] and its extensions [18]). In the actor-only approach, we model only the probability distribution over the state - the direct behaviour of the agent. In actor-critic RL, the actor outputs an action (according to a probability distribution) given a state and the critic (we can look at this as a feedback that "critiques" that action) outputs a Q value of applying the chosen action in that given state. Popular, state-of-the-art algorithms

like Advantage Actor Critic (A2C) [19, 20], Deep Deterministic Policy Gradient (DDPG) [21, 22], Proximal Policy Optimization (PPO) [23, 24] all follow this scheme.

2.2 Technical analysis in stock trading: stock market terminologies

The generally adopted stock market data is the sequence at regular intervals of time such as the price open, close, high, low, and volume [1]. It's a difficult task, even for Deep Neural Networks (DNN), to get a solid grasp on this data in this form as it holds a great level of noise, plus the non-stationary trait. Technical indicators were introduced to summarize markets' behaviour and make it easier to understand the patterns in it. They are heuristics or mathematical calculations based on historical price, volume that can provide (fig. 1) bullish signals (buy) or bearish signals (sell) all by themselves (although not accurate all the time). The most popular technical indicators that will be mentioned in the review as well are: Moving Average Convergence Divergence (MACD) [25], Relative Strength Index (RSI) [26], Average Directional Index (ADX) [27], Commodity Channel Index (CCI) [28], On-Balance Volume (OBV) [29] and moving average and exponential moving average [30]. There are many more indicators and the encyclopedia [31] explains nicely each one of them.

The challenge that comes with applying technical indicators in algorithmic trading is the fact that there are so many of them and the majority do not work in specific situations - providing false signals. Moreover, many are correlated with each other - such dependencies need to be taken down as it can affect the behaviour of a learning algorithm because of the overwhelming obscure feature space. Solutions have been employed through deep learning - performing dimensionality reduction (through an autoencoder [33]), feature selection and extraction (through Convolutional Neural Networks - CNN [34] or Long Short Term Memory Networks - LSTM [35]), hence eliminating some noise in the raw market data.

Figure 1: Indicator showing buy and sell signals. Source: iqoption [32]

These indicators alone can be a repertoire for a human trader and combinations between them can represent by themselves winning strategies for some stocks. That's why there is actually a great amount of trading agents based on expert systems rule-based with great success using only technical analysis. There are also papers that present simple algorithms that are based only on data mining and the use of one indicator that can achieve over 32% annual return [36].

3 Literature Review

In this section we review literature works regarding deep reinforcement learning in trading. There are key points we follow in this review, tied to the principal challenges that reinforcement learning faces when dealing with financial trading: *data quality and availability* (high quality data might not be free or it might be limited for certain timeframes), *the partial observability of the environment* (there will always be a degree of un-observability in the financial market [14]) and *the exploration/exploitation dilemma* [15] (a great deal of exploration is usually required for a successful RL agent but this is not feasible in financial trading since random exploration would inevitably generate huge amount of transaction costs and loss of capital).

3.1 Critic-only Deep Reinforcement Learning

Overview

Before getting to review the studies in this category, it is important to highlight the main limitations that such an approach suffers from. The critic method (represented mainly by the DQN and its improved variations) is the most published method in this area of research [19] (compared to the others: actor-only and actor-critic). The main limitation is that a DQN aims to solve a discrete action space problem; so, in order to transform it to a continuous space (this can be needed as we will see in some reviewed papers), certain tricks are to be applied. Moreover, a DQN works well for discrete state representations, however, the prices of all the stocks are continuous-valued entities. Applying this to multiple stocks and assets is also a huge limitation as the state and action space grows exponentially [22]. Moreover, the way that a reward function is defined in this context is crucial; critic-only approaches being very sensible to the reward signals it receives from the environment [1].

Related Works

Application of Deep Reinforcement Learning on Automated Stock Trading [37]. Inspired by the success of state-of-the-art Deep Q Networks in tackling games like Atari [38] and imperfect information games like Poker [39, 40], this paper tries to apply the same idea to develop a trading agent considering the trading environment as a game in which we try to maximize reward signaled by profit. A variant of the DQN learning system is tested by the authors: Deep Recurrent Q- Network (DRQN). The architecture is simple - they use a recurrent neural network at the basis of DQN which can process temporal sequences better and consider a second target network to further stabilize the process. The authors use S&P500 ETF price history dataset to train and test the agent against baseline benchmarks: buy and hold strategy – which is used by human long-term investors and a random action-selected DQN trader. The data for training and testing is obtained through Yahoo Finance and it contains 19 years of daily closing prices, the first 5 being used for training and the rest for testing the agent. Therefore, a state representation is defined with a sequence of adjusted close price data over a sliding window of 20 days. Being a pure critic approach, at any time, an agent has a discrete, finite set of actions - it has to decide between 3 options: buy a share, sell a share or do nothing. The reward is computed as the difference between the next day's adjusted close and the current day's adjusted close price, depending on which action was made.

As we can see in fig. 2, the best agent, based on the DRQN system, is successful and outperforms the baseline models which gives hope for algorithmic trading. The annual expected return from these systems is around 22-23%, which is not bad, but it is far off from some of the other reviewed agents in this paper that can get over 60% annual return [14].

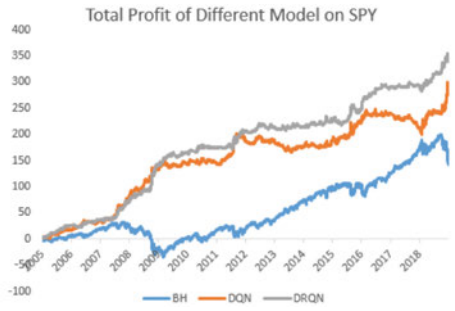

Figure 2: Profit Curves of BH, DQN and DRQN on the SPY Test Dataset [37]

However, there are clear limitations on the study of this paper. First, only one stock is considered as a benchmark. That is not necessarily wrong but it does question the generality of the proposed approach to a larger portfolio, even from our own experiments doing algorithmic trading, some stocks can be very successful and some can lose all the allocated budget. Secondly, the reward function is really weak, it does not address risk as the other reviewed paper do, therefore it might be scary to let this model run on live data. A quick fix would be to use the Sharpe ratio as a reward measurement. Third, the agent is very constrained in what it can do - buying only one stock at a time is very limiting, more actions could be added to allow for

more shares to be bought or sold. The feature representation is far from optimal, raw price data contains a lot of noise - technical analysis features should have been used to mitigate that. In terms of training/testing, it is concerning to us that the training data is much smaller in size and it might not be representative compared to the testing one - the 2006-2018 interval of time contains critical market states like the 2008 financial crisis in which the agent probably won't know how to handle. Furthermore, very basic opponents have been chosen for comparison purposes, even though the buy and hold strategy is objectively good and can be really profitable in real life, being actually the main strategy for investors, the paper lacks a direct comparison with other trading agents from the literature on the same stock. Nevertheless, we would categorize this paper as a solid pilot experiment into this wild field that can provide a proof-of-concept for the utility of deep reinforcement learning in trading.

A strongly related work with the above is the study **Financial Trading as a Game: A Deep Reinforcement Learning Approach** [14] that also employs a DRQN-based agent. However, there are key differences: it uses 16 features in total - raw price data (open, high, low, close, volume (OHLCV)) plus indicators; a substantially small replay memory is used during training; an action augmentation technique is also introduced to address the well-known problem of random exploration in RL. The authors say that this method allows for greedy policies to be used over the course of learning and show strong empirical performance compared to the canonical ϵ-greedy exploration. Moreover, they sample a longer sequence as input for the recurrent neural network. Limitations still exist: the reward function (defined here as the portfolio log returns) is better as it normalizes the returns but still does not specifically address risk and the action space size is still very restricted (3 options).

The algorithm is applied on forex (FX) market from 2012 to 2017, 12 currency pairs, 15-minute timeframe with realistic transaction costs. The results are decent, the algorithm obtaining even 60% annual return on some currencies, averaging about 10% with all the results being positive, which is encouraging further research in this area. It is true that the authors haven't tested this algorithm in stock markets, only on FX, which would rule this paper out of scope for our review. However, considering the fact that this paper presents a better version of the previously-discussed algorithm, it is not wrong to assume it may be applied with a decent amount of success to the stock market as well. It's interesting here to also address the research question regarding the comparison with professional human traders. The environment for backtesting happens to have realistic setting in this paper and yet, the best result is a 60% annual return, which is high. Nevertheless, it is impossible to just go with the numbers because we can't really know what an average professional trader or day-trader scores as profit rate in a year [41], but we do know that 60% a year in profits is much more than 7% (the usual buy and hold annual return rate for a stable growing stock index like S&P 500) and would quickly snowball if continuous reinvesting of funds is done and the algorithm is stable, which is the main goal of traders and investors. Therefore, from this perspective, we can assume that, if consistent in trading, the algorithmic solution might be on par with human traders that make a living out of this.

There are a few studies [1, 19] that employ critic-only deep reinforcement learning methods to compare with the other ones. In **Adaptive stock trading strategies with deep reinforcement learning methods** [1], Gated Recurrent Units (GRUs) are introduced as a deep learning method to extract features from the stock market automatically for its Deep Reinforcement Q-Learning system (GDQN). The motivation for introducing this is the fact that stock market movements cannot reveal the patterns or features behind the dynamic states of the market. In order to achieve that, the authors present an architecture with an update gate and reset gate function for the neural network. The reset gate acts like a filter for the previous stock informationi, keeping only a portion of it, and the update gate decides whether the hidden state will be updated with the current information or not. This is a learnt way to tackle feature engineering for this problem, similar approaches with autoencoder architectures have been tried to achieve the same thing. To note, Dropout [42] is used with GRU in the Q-network.

Another contribution of this paper is the use of a new reward function that better handles risk (something we repeatedly stated that it's missing before): the Sortino ratio (SR) [43]. The authors argue that this reward function would better fair in a high-volatility market because it measures negative volatility - it only factors in the negative deviation of a trading strategy's returns from the mean. As state representation, the usual OHLCV is used plus some popular technical indicators like MACD, MA, EMA, OBV. The action space is again restricted to 1 share per transaction, 3 options being available: buy, sell or hold. Among the reviewed papers so far, this has the most promising setup; the daily stock data used, stretched from early 2008 to the end of 2018, the first 8 years being used for training; it consisted of 15 US, UK and Chinese stocks, each one of them tested individually. The best result of the GDQN system was recorded on a Chinese stock: 171% return and 1.79 SR. A SR of 3.49 and a return of 96.6% on another Chinese stock were also recorded. Just 2 out of 15 stocks were negative in profits. Moreover, the system wins the comparison with the baseline (Turtle strategy [44]) and it is showed to achieve more stable returns than a state-of-the-art direct reinforcement learning trading strategy [13]. GDQN is directly compared with the actor-critic Gated Deterministic Policy Gradient (GDPG) also tested in this paper that we will review in the following sections. It is showed that GDPG is more stable than GDQN and provides slightly better results. Our criticism towards this study remains, like in the other ones, the limitation in what the agent can do - with only 1 share per transaction at max; in addition, transaction costs were not added in the equation, it would be interesting to see how the return rates would change if they would be factored in.

We will very briefly mention **DRL for Trading** [19] as well, though the experiments were conducted only on futures. The model is very similar with [37], implementing a LSTM based Q-network, but making use of some indicators (RSI and MACD) to add to the feature space. Regarding the reward function, it takes into consideration risks adddressing the volatility of the market: scale up trade positions while volatility is low and do the opposite, otherwise. The authors show that the DQN system outperform baseline models (buy and hold, Sign(R) [45, 46], MACD signal [25]) and deliver profits even under heavy transaction costs. Moreover, it records more profits than actor-only algorithm (PG) and the A2C algorithm.

As we mentioned in the first paragraph of this section, the critic-only approach is the most published one in this field of financial trading; we had to remove some (older) studies [47, 48, 49] because they were out of the scope of this review paper - they weren't using deep Q-learning, only Q-learning with simple artificial neural networks (ANNs) or doing the feature extraction beforehand through other methods and not deep learning.

3.2 Actor-only Deep Reinforcement Learning

Overview

With actor-only methods, because a policy is directly learnt, the action space can be generalized to be **continuous** - this is the main advantage of this approach: it learns a direct mapping (not necessarily discrete) of what to do in a particular state. A general concern we have discovered with this actor-only approach is the longer time to train that it takes to learn optimal policies, as also highlighted in [19]. This happens because, in trading, learning needs a lot of samples to successfully train as otherwise, individual bad actions will be considered *good* as long as the total rewards are great. However, this statement might be disputed [9], as in some cases faster convergence of the learning process is possible.

Related Works

Deep Direct Reinforcement Learning for Financial Signal Representation and Trading [13]. We briefly mentioned this study in the previous section (when comparing to the GDQN) as a state-of-the-art direct reinforcement learning trading strategy. This paper from 2017 specifically tries to answer the question of beating experienced human traders with a deep reinforcement

learning based algorithmic-trader. This study is special because the authors claim that (at the time) it represented the first use of Deep Learning for **real-time** financial trading. Their proposed approach is a novel one. They remodel the structure of a typical Recurrent Deep Neural Network for simultaneous environment sensing and recurrent decision making that can work in an online environment. Like in the first paper reviewed, they use a Recurrent Neural Network for the RL part; however, they use Deep Learning for feature extraction combined with fuzzy learning concepts [50, 51] to reduce the uncertainty of the input data. They use fuzzy representations [52] (assign fuzzy linguist values to the input data) as part of the data processing step; after that, they use the deep learning for feature learning on the processed data and this is fed to the RL agent to trade.

What is important here to note is the fact that the proposed solution does not use technical indicators, it utilizes the aforementioned system for raw data processing and feature learning to make educated guesses through the RL system to maximize profit and that is really impressive. We personally thought that technical indicators would greatly improve a system's performance if we aim for the correct amount of un-correlated ones because of a simple reason: it adds more insight to the dataset and more features that can unravel the hidden patterns leading to a successful trading strategy. This does not mean that the performance using only raw data and clever data processing would beat current state-of-the-art models that make use of technical analysis but perhaps a comparison between their proposed system using and not using technical indicators would have been great to add. The experiments part is solid from a comparison point of view between established strategies in the literature and the proposed approaches; however, it lacks the testing on a diverse portfolio: only 3 contracts are chosen (2 commodities – sugar and silver and one stock-index future IF). It is actually addressed in the future work section that this framework should be adapted in the future to account for portfolio management. Nevertheless, their experiments on both commodities and future contracts show that the proposed Direct Deep Reinforcement (DDR) system and its fuzzy extension (FDDR) generate significantly more total profit and are much robust to different market conditions than Buy and Hold, sparse coding-inspired optimal training (SCOT) [53] and DDR system without fuzzy representation for data processing. Moreover, on Sharpe ratio, the proposes systems recorded values over 9, which is incredibly high. The authors also tested the algorithms on stock indexes - like S&P 500, the stock data stretched from 1990 to 2015, the first 8 years being used for training and the transaction cost is set to 0.1% of the index value. It is showed that the agents generate profits, but it's not that profitable over the whole 18 years of simulated testing period.

One serious drawback of this study is the consistency of answering the main question that is opened and addressed in it. There is no comparison to human trading experts' performance in the experiments section, nor any discussion about this in the conclusion. Of course, the total profits displayed in the experiments do not come close to what a successful professional day trader or trading expert would achieve (because it does not come close to an actual annual wage), but this does not negate the premise. This is because the authors fail to mention the actual starting capital used in the experiments; therefore, we can only rely on the Sharpe ratio results for a conclusion. As we have mentioned before, the proposed approaches highlight remarkable high values with this metric: in the range 9-20. We need to be careful in calling this super-human performance, as investors usually consider a Sharpe ratio over 1 as good and over 3 as excellent [54], but the paper does not mention if it uses the annualized Sharpe ratio or not. For more intuition on this, a ratio of 0.2-0.3 is in line with the broader market [55]. There is still a lot of skepticism behind these claims though, as the agents were tested on particular futures and commodities; one needs a broader portfolio to claim super-human success even though, at first, the results seem to be pointing in that direction. Nevertheless, the proposed agents are showed to generate a positive result in the long run which is an accomplishment by itself as there are some research studies [56] arguing that many individual investors hold undiversified portfolios and trade actively, speculatively and to their own detriment. So, if we switch our research question

to address and average trader, DRL approaches would probably be superior.

Quantitative Trading on Stock Market Based on Deep Reinforcement Learning [57] is another study that explores a deep actor-only approach to quantitative trading. Its main contributions are a throughout analysis of the advantage of choosing a deep neural network (LSTM) compared to a fully connected one and the impact of some combinations of the technical indicators on the performance on the daily-data Chinese markets. This paper does prove that a deep approach is better and picking the right technical indicators does make a difference if we care about the return. The pure-performance results are mixed, the authors show that the proposed method can make decent profit in some stocks but it can also have oscillatory behaviour in others. Overall, not over the top results and setup.

We have omitted impressive works like **Enhancing Time Series Momentum Strategies Using DNN** [46] in this section as it focuses more on the portfolio management part, plus it uses offline batch gradient ascent methods to directly optimize the objective function (maximizing the reward and minimizing the risk, Sharpe Ratio) which is fundamentally different from the standard actor-only RL where a distribution needs to be learnt to arrive at the final policy [19].

3.3 Actor-critic Deep Reinforcement Learning

Overview

The third type of RL, actor-critic framework aims to simultaneously train two models at a time: the actor that learns how to make the agent respond in a given state and the critic - measuring how good the chosen action really was. This approach proved to be one of the most successful ones in the literature, state-of-the-art actor-critic DRL algorithms like PPO (in an actor-critic setting like it was presented in the original paper) or A2C solving complex environments [23]. However, in spite of this, the cateogry seems still unexplored; at least as of 2019 [19], it stayed among the least studied methods in DRL for trading and as of current day, only a few new studies in this area have been published. Something that actor-critic algorithms like PPO do better than previous mentioned RL types is that it addresses well-known problems of applying RL to complex environments. One of them is that the training data that is generated during learning is itself dependent on the current policy so that means the data distributions over observations and rewards are constantly changing as the agent learns which is a major cause of instability. RL also suffers from a very high sensitivity to initialization or hyper-parameters: if the policy suffers a massive change (due to a high learning rate, for example) the agent can be pushed to a region of the search space where it collects the next batch of data over a very poor policy causing it to maybe never recover again.

Related Works

Deep Reinforcement Learning for Automated Stock Trading: An Ensemble Strategy [58]. This paper proposed an ensemble system (fig. 4) using DRL to further better the results in the literature with multiple stock trading experiments. They introduce an ensemble deep model that has 3 state-of-the-art DRL algorithms at core: PPO, A2C, DDPG. The authors state that the ensemble strategy makes the trading more robust and reliable to different market situations and can maximize return subject to risk constraints. The experiments are based on the Dow Jones portfolio. The features used to train the ensemble model are the available balance, adjusted close price, shares already owned plus some technical indicators: MACD, RSI, CCI and ADX. The choice for the action space is solid as well: $\{-k, ..., -1, 0, 1, ..., k\}$, where k and $-k$ presents the number of shares we can buy and sell, so in total there are 2k+1 possibility for one stock. This is later easily normalized to the continuous $[-1, 1]$ action space for the main policy algorithms to work. It seems the authors do a better job at portfolio management problem with this approach - not being limited to buy only 1 share when in a great spot (like we saw in some of the previously reviewed papers). However, all the features from all the 30 stocks are merged, the aim being to

also choose what stock to buy or sell – this is another part of portfolio management which is different and maybe should have been treated separately to the main trading algorithm.

Figure 3: Cumulative return curves on different strategies; US 30 index. Initital capital: $1m. [58]

The motivation behind choosing an ensemble is that each trading agent is sensitive to different type of trends. One agent performs well in a bullish trend but acts bad in a bearish trend. Another agent is more adjusted to a volatile market. This paper also discusses the impact of market crash or extreme instability to a trading agent. They employ a *financial turbulence index* that measures extreme asset price movements, using it together with a fixed threshold so that the agent sells everything and stops making any trades if the markets seem to crash or be notoriously unstable. This seems like a good idea; however, one can argue that it's these periods of extreme volatility that a very good trading agent can exploit to make huge profits. The results of the experiments can be seen in fig. 3. It seems the PPO algorithm still managed to provide slightly higher cumulative reward (at about 15% annually); however, the Sharpe ratio which is the risk-adjusted reward is higher in case of the ensemble (which has 13% return annually): 1.30 compared to 1.10. The proposed approach also beats the Dow Jones Industrial average index (DIJA) which is a benchmark for this portfolio and the min-variance portfolio allocation. The return is much greater than what you can expect through a buy and hold strategy (around 7%) and the algorithm seems stable enough during extreme market conditions (the March 2020 stock market crash).

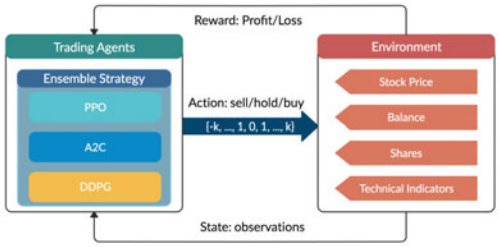

Figure 4: Overview of the ensemble DRL strategy [58]

A strongly related paper with [58] is **Practical DRL Approach for Stock Trading** [22]. This paper only investigates the DDPG algorithm (from the previously reviewed ensemble) under the same conditions and with the same evaluation metrics but without any technical indicators as input. There are other minor differences: historical daily prices from 2009 to 2018 to train the agent and test the performance; data from early 2009 to late 2014 are used for training, and the data from 2015 is used for validation; we test our agent's performance on trading data, which is from early 2016 to late 2018. The results are decent: 25.87% annualized return on this US 30 (Dow Jones) index, more than 15.93% obtained through the Min-Variance method and 16.40% the industrial average. The comparisons on Sharpe ratios only are also favorable (1.79 against 1.45 and 1.27) which makes it more robust than the others in balancing risk and return.

Stock Trading Bot Using Deep Reinforcement Learning [59] is an especially interesting work because of the fact that it combines a DRL approach with sentiment analysis [60] (external information from news outlets) and proves that the proposed approach can learn the tricks of stock trading. Such a study is definitely important because many people are skeptical of the effectiveness of a pure quantitative learning algorithm in trading. Although the most important part of their system is the RL agent, this extra component that tries to predict future movements

in the markets through sentiment analysis on financial news is an extra feature to the DRL model. The proposed approach is fairly simple at core: they use DDPG for the reinforcement learning agent and a recurrent convolutional neural network (RCNN) for classification of news sentiment. The experiments involved market data on a daily timeframe for one stock, so really not on a large scale, but let's not forget that the aim for this paper is more on the side of a proof of concept. The supervised learning system for sentiment analysis (RCNN) needed to be trained beforehand - on 96k samples; it achieved 96.88% accuracy on test set which had 31.5k samples. However, information about the prior of the dataset used is missing, there should be a mention about the distribution to fully claim that such an accuracy is indeed good. Even though the experiments don't show an overwhelming increase in profits, the authors do prove that the agent does learn basic strategies: it does buy and sell continuously, it prefers to hold when there is a continuous decrease in price and it always maintains a higher value than the stagnant stock value.

Lastly, there is the GDPG system [1] we briefly mentioned in the critic-only DRL section as a reference for comparison. The authors introduce GDPG as an actor-critic strategy that combines the Q-Network from the GDQN system with a policy network. It is showed (table 1) that this achieves more stable risk-adjusted returns and outperforms the baseline Turtle trading strategy. To put things into perspective, the turtle trading system is a well-known trend following strategy that was originally taught by Richard Dennis back in 1979 - being the first attempt to autonomous trading. It was really successful back then and some researchers choose to use it nowadays as a valid benchmark system. We do criticize this choice though, because actual market conditions are totally different than the ones 40 years ago, such a system might not be that relevant now.

	GDQN		GDPG		Turtle	
Symbol	**SR**	**R(%)**	**SR**	**R(%)**	**SR**	**R(%)**
AAPL	1.02	77.7	1.30	82.0	1.49	69.5
GE	-0.13	-10.8	-0.22	-6.39	-0.64	-17.0
AXP	0.39	20.0	0.51	24.3	0.67	25.6
CSCO	0.31	20.6	0.57	13.6	0.12	-1.41
IBM	0.07	4.63	0.05	2.55	-0.29	-11.7

Table 1: GDQN, GDPG, Turtle strategy in some U.S. stocks over a 3-year period (2016-2019) [1]

4 Summary & Conclusion

Deep Reinforcement Learning is a growing field of interest in the financial world. All the papers that were discussed here had their own experimental settings, assumptions and constrains. This makes it hard to compare one against another especially when the scope of the study is different between them, a direct comparison is not feasible in this scenario. Nevertheless, comparisons between different types of DRL methods were possible [37, 14, 19, 13] in the context of same study but also, limited, in the context of different studies as we saw with [1] and [13]. We have observed that DRL can be a powerful tool in quantitative low-frequency trading, a few studies with more realistic setups [14, 1, 13] obtaining over 20% annual return with risk-aware metrics also being used. We have come to the conclusion that such agents can rival human traders in specific markets, but further research should be conducted in this area. Future possible directions of research include:

- More extensive experiments on live-trading platforms rather than backtesting or very limited real-time trading.
- Direct comparisons between DRL trading agents with human traders. For example, having a professional day trader run against an algorithmic DRL agent over the same market and interval of time in a controller experiment to observe which one would get higher returns.
- More comparisons among state-of-the-art DRL approaches under similar conditions and data sources.
- More research into the behaviour of DRL agents under critical market conditions (stock market crashes).

References

[1] Xing Wu, Haolei Chen, Jianjia Wang, Luigi Troiano, Vincenzo Loia, and Hamido Fujita. Adaptive stock trading strategies with deep reinforcement learning methods. *Information Sciences*, 2020.

[2] Mojtaba Nabipour, Pooyan Nayyeri, Hamed Jabani, Amir Mosavi, E Salwana, et al. Deep learning for stock market prediction. *Entropy*, 22(8):840, 2020.

[3] David E Goldberg. *Genetic algorithms*. Pearson Education India, 2006.

[4] James Kennedy and Russell Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95-International Conference on Neural Networks*, volume 4, pages 1942–1948. IEEE, 1995.

[5] Robert G Cooper, Scott J Edgett, and Elko J Kleinschmidt. Portfolio management. *Pegasus, New York*, 2001.

[6] Farzan Soleymani and Eric Paquet. Financial portfolio optimization with online deep reinforcement learning and restricted stacked autoencoder-deepbreath. *Expert Systems with Applications*, page 113456, 2020.

[7] Jingyuan Wang, Yang Zhang, Ke Tang, Junjie Wu, and Zhang Xiong. Alphastock: A buying-winners-and-selling-losers investment strategy using interpretable deep reinforcement attention networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1900–1908, 2019.

[8] William F Sharpe. The sharpe ratio. *Journal of portfolio management*, 21(1):49–58, 1994.

[9] Thomas G Fischer. Reinforcement learning in financial markets-a survey. Technical report, FAU Discussion Papers in Economics, 2018.

[10] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. *nature*, 518(7540):529–533, 2015.

[11] Yong-Duan Song, Qi Song, and Wen-Chuan Cai. Fault-tolerant adaptive control of high-speed trains under traction/braking failures: A virtual parameter-based approach. *IEEE Transactions on Intelligent Transportation Systems*, 15(2):737–748, 2013.

[12] David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[13] Yue Deng, Feng Bao, Youyong Kong, Zhiquan Ren, and Qionghai Dai. Deep direct reinforcement learning for financial signal representation and trading. *IEEE transactions on neural networks and learning systems*, 28(3):653–664, 2016.

[14] Chien Yi Huang. Financial trading as a game: A deep reinforcement learning approach. *arXiv preprint arXiv:1807.02787*, 2018.

[15] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[16] C. J. Watkins and Dayan P. Q-learning. *Machine Learning*, 1992.

[17] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. 2014.

[18] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

[19] Zihao Zhang, Stefan Zohren, and Stephen Roberts. Deep reinforcement learning for trading. *The Journal of Financial Data Science*, 2(2):25–40, 2020.

[20] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.

[21] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[22] Zhuoran Xiong, Xiao-Yang Liu, Shan Zhong, Hongyang Yang, and Anwar Walid. Practical deep reinforcement learning approach for stock trading. *arXiv preprint arXiv:1811.07522*, 2018.

[23] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[24] Zhipeng Liang, Hao Chen, Junhao Zhu, Kangkang Jiang, and Yanran Li. Adversarial deep reinforcement learning in portfolio management. *arXiv preprint arXiv:1808.09940*, 2018.

[25] Jamil Baz, Nicolas Granger, Campbell R Harvey, Nicolas Le Roux, and Sandy Rattray. Dissecting investment strategies in the cross section and time series. *Available at SSRN 2695101*, 2015.

[26] J Welles Wilder. *New concepts in technical trading systems*. Trend Research, 1978.

[27] Ikhlaas Gurrib et al. Performance of the average directional index as a market timing tool for the most actively traded usd based currency pairs. *Banks and Bank Systems*, 13(3):58–70, 2018.

[28] Mansoor Maitah, Petr Prochazka, Michal Cermak, and Karel Šrédl. Commodity channel index: Evaluation of trading rule of agricultural commodities. *International Journal of Economics and Financial Issues*, 6(1):176–178, 2016.

[29] William Wai Him Tsang, Terence Tai Leung Chong, et al. Profitability of the on-balance volume indicator. *Economics Bulletin*, 29(3):2424–2431, 2009.

[30] J Stuart Hunter. The exponentially weighted moving average. *Journal of quality technology*, 18(4):203–210, 1986.

[31] Robert W Colby and Thomas A Meyers. *The encyclopedia of technical market indicators*. Dow Jones-Irwin Homewood, IL, 1988.

[32] Iqoption trading website. `https://eu.iqoption.com/en`.

[33] Wei Wang, Yan Huang, Yizhou Wang, and Liang Wang. Generalized autoencoder: A neural network framework for dimensionality reduction. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 490–497, 2014.

[34] Yann LeCun, Yoshua Bengio, et al. Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995, 1995.

[35] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[36] K Senthamarai Kannan, P Sailapathi Sekar, M Mohamed Sathik, and P Arumugam. Financial stock market forecast using data mining techniques. In *Proceedings of the International Multiconference of Engineers and computer scientists*, volume 1, 2010.

[37] Lin Chen and Qiang Gao. Application of deep reinforcement learning on automated stock trading. In *2019 IEEE 10th International Conference on Software Engineering and Service Science (ICSESS)*, pages 29–33. IEEE, 2019.

[38] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin Riedmiller. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

[39] Johannes Heinrich and David Silver. Deep reinforcement learning from self-play in imperfect-information games. *arXiv preprint arXiv:1603.01121*, 2016.

[40] T.V. Pricope. A view on deep reinforcement learning in imperfect information games. *Studia Universitatis Babeș-Bolyai Informatica*, 65(2):31–49, 2020.

[41] Investopedia webstie average return over one year for professsional traders. `https://www.investopedia.com/articles/active-trading/053115/average-rate-return-day-traders.asp`.

[42] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[43] Vivek Mohan, Jai Govind Singh, and Weerakorn Ongsakul. Sortino ratio based portfolio optimization considering evs and renewable energy in microgrid power market. *IEEE Transactions on Sustainable Energy*, 8(1):219–229, 2016.

[44] Curtis Faith. The original turtle trading rules, 2003.

[45] Tobias J Moskowitz, Yao Hua Ooi, and Lasse Heje Pedersen. Time series momentum. *Journal of financial economics*, 104(2):228–250, 2012.

[46] Bryan Lim, Stefan Zohren, and Stephen Roberts. Enhancing time-series momentum strategies using deep neural networks. *The Journal of Financial Data Science*, 1(4):19–38, 2019.

[47] Zhiyong Tan, Chai Quek, and Philip YK Cheng. Stock trading with cycles: A financial application of anfis and reinforcement learning. *Expert Systems with Applications*, 38(5):4741–4755, 2011.

[48] Francesco Bertoluzzo and Marco Corazza. Testing different reinforcement learning configurations for financial trading: Introduction and applications. *Procedia Economics and Finance*, 3:68–77, 2012.

[49] Gordon Ritter. Machine learning for trading. *Available at SSRN 3015609*, 2017.

[50] George J Klir. Fuzzy sets. *Uncentainty and Information*, 1988.

[51] Nikhil R Pal and James C Bezdek. Measuring fuzzy uncertainty. *IEEE Transactions on Fuzzy Systems*, 2(2):107–118, 1994.

[52] Chin-Teng Lin, C. S. George Lee, et al. Neural-network-based fuzzy logic control and decision system. *IEEE Transactions on computers*, 40(12):1320–1336, 1991.

[53] Yue Deng, Youyong Kong, Feng Bao, and Qionghai Dai. Sparse coding-inspired optimal trading system for hft industry. *IEEE Transactions on Industrial Informatics*, 11(2):467–475, 2015.

[54] Investopedia webstie sharpe ration. `https://www.investopedia.com/ask/answers/010815/what-good-sharpe-ratio.asp`.

[55] Daytrading webstie sharpe ration. `https://www.daytrading.com/sharpe-ratio`.

[56] Brad M Barber and Terrance Odean. The behavior of individual investors. In *Handbook of the Economics of Finance*, volume 2, pages 1533–1570. Elsevier, 2013.

[57] WU Jia, WANG Chen, Lidong Xiong, and SUN Hongyong. Quantitative trading on stock market based on deep reinforcement learning. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8. IEEE, 2019.

[58] Hongyang Yang, Xiao-Yang Liu, Shan Zhong, and Anwar Walid. Deep reinforcement learning for automated stock trading: An ensemble strategy. *Available at SSRN*, 2020.

[59] Akhil Raj Azhikodan, Anvitha GK Bhat, and Mamatha V Jadhav. Stock trading bot using deep reinforcement learning. In *Innovations in Computer Science and Engineering*, pages 41–49. Springer, 2019.

[60] Rudy Prabowo and Mike Thelwall. Sentiment analysis: A combined approach. *Journal of Informetrics*, 3(2):143–157, 2009.