National College of Ireland

REINFORCEMENT LEARNING FOR ALGORITHMIC TRADING ON FINANCIAL MARKETS

MSc Research Project
Data Analytics

Claudia Maria Suciu
x15032001

School of Computing
National College of Ireland

Supervisor: Cristina Hava Muntean

National College of Ireland
Project Submission Sheet – 2015/2016
School of Computing

Student Name:	Claudia Maria Suciu
Student ID:	x15032001
Programme:	Data Analytics
Year:	2016
Module:	MSc Research Project
Lecturer:	Cristina Hava Muntean
Submission Due Date:	11/12/2017
Project Title:	REINFORCEMENT LEARNING FOR ALGORITHMIC TRADING ON FINANCIAL MARKETS
Word Count:	XXX

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are encouraged to use the Harvard Referencing Standard supplied by the Library. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action. Students may be required to undergo a viva (oral examination) if there is suspicion about the validity of their submitted work.

Signature:	
Date:	11th December 2017

PLEASE READ THE FOLLOWING INSTRUCTIONS:

1. Please attach a completed copy of this sheet to each project (including multiple copies).
2. **You must ensure that you retain a HARD COPY of ALL projects**, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer. Please do not bind projects or place in covers unless specifically requested.
3. Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

REINFORCEMENT LEARNING FOR ALGORITHMIC TRADING ON FINANCIAL MARKETS

Claudia Maria Suciu
x15032001
MSc Research Project in Data Analytics

11th December 2017

Abstract

Can algorithmic trading learn to trade efficiently by itself, given risk and preference parameters are provided? Financial markets pose increasing challenges in speed and lower fees making difficult trades in the traditional manner with a satisfactory return. Self-learning systems can provide a viable option as standalone trader in the current challenging finance trading environment and big data flow. Wide adoption of algorithmic trading has rooted out some of the market inefficiencies that represent profit opportunities for speculators.Uncorrelated portfolios and market making had left little choice to traditional trader, but to look for a continuous automation and optimization. Recurrent neural networks have still major contributions to make for the stability of the financial market and exploitation of its inefficiencies.

1 Introduction

Financial Markets Financial markets gain more and more interest in research efforts due to the size of the market and the wide influence on economies involved. As well as the effect and magnitude of such effect, the financial markets research can be quantified and observed, through the financial reward they can have. Wide adoption of technology on financial market exchanges had significant impact on the tools and techniques used. Increased computing power and internet speed has made possible automation of trading activities which used to take place on the exchanges floors, removing some of the shortcomings of emotional involvement, excitement or human error. However, the technology did not bring all positive. Technology's speed is currently outing the physical agents out of the exchange floors, due to the increased trading speed and information analysis capabilities beyond human reaction capabilities. In such circumstances, the trading algorithms are required to become more aware of the market changes and smarter, capable of learning from experiences, like humans they tend to replace. Research and development in machine learning techniques in the recent years has made possible implementation of artificial intelligence in machines and robotics, with the help of increased computing capabilities required by such complex algorithms. Machine learning techniques applied

in automated systems such as humanoids among others, mimics human behaviour and learning abilities at a basic level, allowing increased performance and information processing at a better speed than humans, while eliminating human error. For financial markets, automation of trading does not eliminate just human error but translates in more profitable decisions given the volume of information and factors that influence the volatility and profitability of any financial asset at any given time.
Financial markets efficient or inefficient? It is known that in order to achieve the maximum profitability, the trades must take place in a wise manner with the minimum trading cost. In exploiting the price movement on financial markets, one cannot be oblivious of the mechanics and underlying factors that influence the price fluctuations. Seen as the equilibrium point between ask price and bide price, supply and demand, features prices fluctuates with the buy or sell interest. Buyer's interest rising will pull the price level up, while a general tendency of selling would determine the price to collapse. While to blame for the price fluctuation can be anything from general economical development, media, speculation, psychology, these factors have posed significant barriers in mathematical representation and accountability of their effect in financial research. (Kirkpatrick II and Dahlquist; 2010)

A question in finance has been asked again and again and has been considered by algorithmic trading as well. Can one predict the price movement and incorporate such knowledge in machine assisted trading? Can one identify the price trend and trading opportunity to incorporate that knowledge in the self learning trading system as a human would? So many factors can be influential, how can they be reproduced or accounted for?

For simplification, Charles H. Dow and William Peter Hamilton have considered the price only, the underlying factors being reflected in the price established on the market at a certain place and time. Developed further, Eugene F. Fama stated in 1970, the Efficient Market Hypothesis (EMH). According to Fama "prices fully reflect all available information" on financial markets. (Malkiel and Fama; 1970) E(Fama; 1991) (Fama; 1995) However, research in finance has been aimed at confirming or disproving the EMH, and one taking an interest in finance cannot ignore the veracity of this theory. If it is to be true, the multitude of players and independent buy or sell decisions, the size of market makes the price highly improbable to be predicted accurately and such behaviour has been even named "random walk" or random. In his publication from 1995, Fama gives more details to his theory, classifying the factors to influence the price in three categories, "weak". "semi-strong" and "strong". In the weak category are interest and dividends which can pose a certain predictability for the technical analyst, given the seasonality and the historical volume. However, "semi-strong" and "strong" factors, are highly unpredictable. In this category are stated, media and "inside information". (Malkiel and Fama; 1970) E(Fama; 1991) (Fama; 1995)
In practice, financial chartists have been able to predict so called "weak" factors, by studying even just visually the seasonality of the price and volume traded and trade given a price trend has been identified to be favourable or numerical predictions from historical data to identify the trend and opportunity window.

Short term trends are difficult to exploit due to the information advantage dissipation and slippage attributed to poor quality data or infrastructure. Medium and long term price trends are targeted as result.(Bandy; 2015)

Motivation Algorithmic trading has gained a significant attention from the financial institutions due to the tight regulation and trading speed facilitated by higher computing capabilities employed. As such, higher speed of reaction has become a competitive advantage and financial industry has moved quickly towards its adoption to stay profitable on the market. (Kim; 2010)

The information flowing from a multitude of channels with considerable influence on financial market dynamics and higher trading costs has contributed significantly to technology adoption. Although more and more companies trust their trading strategies to be executed by algorithms, the decisions regarding trading parameters and risk levels are still decided by brokers. (Kim; 2010)

Decimalization policy change from 2001 introduced by Securities and Exchange Commission has diminished trading margins of some financial industry forcing towards a wider automation of trading process to remain competitive. Apparition of Financial Information Exchange Protocol provides real time financial information exchange capabilities to software trading systems on the financial markets, allowing small investors and retail traders enter the market along established players, due to the smaller costs associated. (Kim; 2010)
'A proposed algorithmic trade should give you a visual representation of the impact cost and volatility. Post-trade data reports can theoretically guide clients with quickly available data regarding how effi- ciently trades have been executed. Measuring performance is crucial but often gets difficult and complex with customized order flow.' (Kim; 2010)

Disadvantages of algorithmic trading can come from the consistent trading strategy which could be- come predictable. Given increasing adoption of algorithmic trading adoption, a wide adoption would determine a trading pattern established, reducing the randomness brought by non-consistency in trading provided by the human brokers. (Kim; 2010)

Challenges when trading on financial markets. Choosing for implementation the best performing strategies for certain assets during the backtesting could produce an unprofitable algorithm when trad- ing on real market conditions, due to that the algorithm performs the best for the chosen assets in a determined period, and so, any change in market dynamics would deem the trading strategy to be inefficient, similar effect advocated in machine learning called overfitting. (Davey; 2014)

Frequent change in the trading strategies can make a trading strategy inefficient because it doesn't allow time to benefit of the dynamics of market, many traders experiencing change of strategy in the downward trend in the face of panic emotions when losses occur without the benefit of balance rewarded by the price rising trend. (Davey; 2014)

Diversification of portfolios traded have been advised to take advantage of the price evolution in different industries, avoiding correlation between the features chosen. In such circumstances, a strong correlation would determine all the assets follow the same losing trend more often determining a quick loss of capital. Diversification, would allow for a temporary balance between the assets to maintain an acceptable level of funds, in unfavourable trading conditions. However, during recession or mass panic, a portfolio, regardless of diversification level, has been noted that it tends to follow the same general trend. (Davey; 2014)

Financial market follows the principles of a free market and establishes automatically a balance between supply and demand for its financial products. A multitude of factors can influence greatly the fragile balance of virtually infinite number of players involved with major consequences on all economies involved. (Alexander; 2009)

Improvement of investment strategies and advancement in the continuous application of machine learning techniques for financial market could give a better response toolbox for banking systems to overcome the consequences of 2008 bubble crisis, which brought major losses for private and public sectors, with immediate effect on economies. A better risk management strategy would minimize greatly the consequences of human emotion taking lead in financial decisions, that could bring collapse due to the mass panic that leads to. (Alvarez Diaz; 2010)

Predictions on the movement of financial market has kept researchers interest due to the deep implications on the economies taking part and an anticipated reward associated with speculation of risk and oscillating movements associated to their transactions. (Schumaker and Chen; 2009; Alvarez Diaz; 2010)

Although major efforts have been made to understand the factors that influence the financial market evolution, it has been concluded in much of the research work, it is highly unpredictable and in practice it has not been identified all the factors at play, many prediction models having a small or modest financial return. (Schumaker and Chen; 2009; Alvarez Diaz; 2010) Often has been claimed that financial markets follow a random pattern. (Alvarez Diaz; 2010)

A few factors have determined a wider adoption of algorithmic trading and increasing interest from research to develop viable solutions in developing automated systems. One of such factors is the speed of trading on the exchange floors, which brings more challenges for human traders to make profitable decisions, often in terms of fraction of seconds with no error and delays that could prove to be disastrous for investment organisations they represent. Algorithms developed have the final scope to maximise the profit and minimize the risk associated with speculative side of financial trading.

Objectives Given a trader finds himself in an investment position, what would be the most profitable decision? The purpose of this project is to investigate the applicability of deep learning in computational finance; more specific development of a trading algorithm capable of learning to trade on financial markets by itself or with minimal human parameter tuning capabilities similar to learning from practice experienced by human financial traders. The software application will strive to use Recurrent Neural Networks (RNN) for the mining model taking into account the financial market theories, technical indicators as metrics for profitability and optimization practices for investment portfolio.

Experimental Challenges Although not new as a research area, reinforcement learning for finance can pose challenges in setting up an experimental environment and building the self learning model that can integrate harmoniously with text mining to deliver an efficient and profitable automated trading system. Some open source software applications available for such financial engineering tasks are still in developing stage, and setting up the testing environment could pose challenges regarding installation and optimization of different applications required to deliver the computing platform for the experiment.

Another challenge could pose a correct translation of financial model to machine learning model, a precise transposition of financial risk and factors in the learning model to resemble real dynamics of financial market.

The algorithm developed is required to take in consideration data noise and non stationary behaviour present in financial time series.

Tests on limited historical data could learn a seasonal trading model, which performs specifically well on that period, but could be deemed less profitable in live trading. To overcome such shortcomings, the recommendation from literature is to use historical data with a long-time span, allowing reinforcement learning algorithm to learn from financial markets uptrends and downtrends spanning for many years.

Contribution represents the self learning prototype learning model proposed to test and to evaluate an algorithmic trading system. The model would represent a prove of concept allowing room for further improvements, however gives a starting point in developing further an algorithmic trading system for investment business implementation. Although the focus is on machine learning field, the research crossroads with other fields such as computational finance and algorithmic trading had lead to the need for an extensive literature research in these domain. While not exhaustive, the effort will be concentrated to describing and emphasizing the main points where they converge and share interests. On a cautious note, often the resources surveyed have appeared contradictory and inconsistent, however, no effort has been spared to appreciate and extract the relevant information in an unbiased manner to serve the purpose of this research endeavour.

Outline of the Contents The research report is organized in 6 sections. Following *Introduction*, *Related work* provides background and literature survey on the main concepts related to financial market algorithmic trading research trends, reinforcement learning and text mining for finance market. In the third section, *methodology* develops the approach towards the research question and the machine learning model adopted to conduct the analysis. Further, *Implementation*, details on techniques and framework that underlie the proposed solution are given. *Evaluation* section discuss the performance and efficiency of the solution in terms of profitability. *Conclusion and future work* propose potential improvements for the algorithm if live trading needs to be implemented, outlining the results achieved as result of model implementation in an experimental setting. Finalizing the report, conclusions are drawn regarding the results obtained, implications and future research opportunities are identified.

2 Related work

Branch of Machine Learning, Reinforcement Learning (RL) has found applications in engineering, computer science, neuroscience, psychology, economics and mathematics. Like human learning, reinforcement learning, it can be called a semi supervised learning, having only partial information about its performance, where time plays an important role in knowledge accumulation. (Szepesvari; 2010)

Efforts to implement the reinforcement learning in algorithmic trading has been noted in the past decades, with different approaches regarding the evaluation function. J.

Moody and M. Saffell (1999- 2001), have been researching the use of direct reinforcement learning in portfolio optimization with the focus on investment policies, allowing for adjustments of accepted risk levels and effects of the trading transaction fees.(Moody and Saffell; 2001)

Investment decision making is represented as a stochastic process and the direct reinforcement learning refers to its direct interaction with the financial market, eliminating the need to forecast the price movement beforehand.(Moody and Saffell; 2001)

In reinforcement learning, mapping between the states of the environment (financial market in our case) and decision or action to take such as buy, sell or even no action, given a singular state has been described as "policy" (Sutton and Barto; 1998) Policy approaches in literature has been noted as a "value function" or direct space "policy search". (Moody and Saffell; 2001) J. Moody and M. Saffell are contrasting the direct "policy search", direct reinforcement learning or so called "evolutionary" (Sutton and Barto; 1998) with the frequent approach using "value function".

Direct policy search is proven to be more efficient than value function approaches such as TD-learning or Q-learning, and problem representation poses less challenges in the multidimensional data set. However, there has been no details provided regarding the "exploitation vs. exploration" trade off and the learning episodes used.Performance indicators used to assert the efficiency of the trading decisions are cumulative profit, sharpe ratio and deviation ratio.

Performance indicators used to assert the efficiency of the trading decisions are cumulative profit, sharpe ratio and deviation ratio.

M. A. Dempster and V. Leemans (2004), are developing an foreign exchange algorithm using "adaptive reinforcement learning", starting from the previous efforts of J. Moody and M. Saffell , but based on "recurrent reinforcement learning". While similar research are building only the learning algorithm, Dempster and Leemans are building a trading system incorporating the learning algorithm as well as risk management and optimization capabilities, allowing for the trading administrator to adjust the trading parameters according to its risk preferences. (Dempster and Leemans; 2006)

The learning algorithm used is a model based on neural network having a single layer of neurons. Given most algorithms are based on machine learning techniques, often exploration vs. exploitation comes at a very high cost and in practice often algorithms have been deemed to be unusable in practice due to the high transaction charges. (Dempster and Leemans; 2006)

R. P. Schumaker and H. Chen have experimented prediction models incorporating text mining. In their research they used noun phrases and Trademark names as potential predictors and identifier of future price trend. Using keyword collections or 'bag of words' techniques with different weights, the method has established correlation between these keywords and the price of shares they referred to. The prediction method has been noted to be modest. (Schumaker and Chen; 2009)

(Cumming et al.; 2015) investigate as well reinforcement learning for foreign exchange market using the least-square temporal difference (TD) learning. Given the profitability as a performance metric, the concept proved to demonstrate modest results.

(Du et al.; 2016) base it's research in comparing the "direct policy search" with "value search" method in the context of portfolio risk optimization problem. "Value

search" method used is based on batch gradient ascent with back-propagation, a maximization algorithm of utility function while accounting for the loss over the batch training of artificial neural network (ANN). Advantages of the framework noted are avoidance of "Bellman's curse of dimensionality", better suitability to noisy data and less computational requirements.

(Deng et al.; 2017) explore methods such as deep learning (DL) and direct reinforcement learning (DRL) in the context of a high frequency trading system. Complex neural networks (NN) are implemented in both the DL and DRL components. Fuzzy learning is used to reduce the noise and uncertainty characteristic to financial data in the DL configuration and assignment of fuzzy membership is achieved trough direct learning.

3 Trading Algorithm

Requirements Since our goal is to create a self thought Quant, the algorithm needs to comply with a minimal set of requirements to fulfil its declared purpose. As a human trader, application is required to download and process data, identify the trading window and decide when to implement a trading strategy, backtest the strategy and generate orders, updating the strategy performance and allow evaluation. Given the limited resources and business goal, it is sought profit maximization, while explore new profit opportunities, minimizing the trading costs and allow risk adjustment or monitoring in line with the trader's risk preference. Traditional trading algorithms would include definition of the variables and data set, trading strategy or trading rules that apply to specific financial indicators and metrics to evaluate the performance.

Proposed artefact will keep some of the classic elements and structure, but had been developed as composed of two sections: the **Decision Maker** algorithm or model, as it is referred to in literature and the **Backtester**.

3.1 Portofolio selection

Researcher selects the stocks to test trading strategies found by the algorithm after a fundamental financial analysis on a very large data set incorporating 3195 stock indexes from WIKI data set available for free to Quandl members. To test the efficiency of the trading rules inferred by the machine learning model, researcher selects stocks that would have poor performance as investments and good performance. The performance is asserted using fundamental financial indicators such as: daily price change, daily cumulative return, annual return, volatility and Sharpe ratio.

3.2 Self Learner and Decision maker - (RL) framework

RL system is assuming learning capabilities and decision control over its actions to achieve an established goal, a cumulative function, an overall reward, given the actions have positive or negative rewards in the form of profit or loss in our case (Szepesvari; 2010),(Silver; 2016).

Based on the general description of the logic behind the learning, it does not seem relevant the area application of reinforcement learning, as the scenario approximates the human learning process applied every day, where humans are required to solve general

problems that can have multiple solutions over time. (Cumming et al.; 2015)Indeed research is conducted in several domains, implementing RL to produce artificial intelligent systems,and as such, we attempt to map the RL problem to finance domain.

As a learner, RL system, receives prices in the form or time series from its environment, here the financial market, daily prices, and attempts to learn patterns in data and good trading strategies, mimicking human learning process from experience . Having observed market's trend, current prices and the profit or loss received when a buy/sell decision had been made, it decides on the next trade. Good trading strategies at the opportune moment are rewarded with profit or loss which actually represents environment's feedback to agent's actions. Policies are the sequential decisions made to achieve the established goal. The policies can take in consideration user's parameters with regard to how many decisions to be invested in exploring new opportunities or just maximizing profit. Finding the optimal policy for exploration/exploitation problem is a challenge to be addressed by any RL implementation. R. S. Sutton and R. A. Barto are pointing out the evaluative characteristic of reinforcement learning, as opposite to instructive found in the supervised machine learning models. The evaluation is pointed out as tool to optimize the efficiency of the decision made and the quality of actions taken as result of such optimization. Pure evaluative approaches, where the action's reward is giving details about how efficient the decisions are in accumulating the overall reward, but not indication of how good or bad. Sutton and Barto (1998)

Our Reinforcement Learning implementation aim to estimate/predict the next state and the best action to take in the next state that can give the highest reward.

The system has its value function calculated by a recurrent neural network (RNN).

RNN model had been chosen due to its capability to handle sequential data and infer relationship and patterns for the inputs.It's goal is to infer trading rules in our algorithm.

We make the assumption that the price data satisfies the Markov property, where each price point is influenced only by its preceding value. Mathematically, the probability formula describes the relation between consecutive states with x state variables as follows: $p\{\ x(t) \mid x(t-1), ..., x(1)\} = p\{\ x(t) \mid x(t-1)\}$ (3.1) (LazyProgrammer; 2016)

The model takes input a T time length sequence of D dimension vectors, where D dimension vector represents the observations of the state space. The number of dimensions is given by the number of variables that describe the environment's state. While one can imagine state space as external to agent, in reality, the state space can take any variable that is independent of the agent's decision, in the sense that the agent cannot alter.

Current state s at time step t, s_t is defined by the variables $x_1, x_2, ..., x_n$. RNN takes as input state's variables $x_1, x_2, ..., x_n$ to predict the next state $s^{'}$. With the predicted or estimated state $s^{'}$, described by the variables $y_1, y_2, ..., y_n$, the algorithm will decide which action from the available action space can potentially bring the most reward in the next state s_{t+1}. At the time step t+1, when the environment is providing the s_{t+1} values and the reward as response to the action a_t taken at time t, the RNN compares the real state with the estimated, adjusting the parameters of the neural network to improve the quality of its predictions. Value function will be expressed as a sigmoid function of $f : (s_t, x_t) \longmapsto s_{t+1}$ (3.2) where t denotes intermediate time steps, whereas T denotes the end of an episode, if the input data has discrete character.

RL mapping to finance

The RL algorithm behaving like an *agent* and financial market representing the *environment* 'interact' at each of the time steps t = 0,1,2,...,k and so on taking in consideration

daily price data. At each time step increment t_t, t_{t+1}, ...,t_k, the agent receives information about the market's *state* $s_t \in S$ where S represents a collection of possible states (price of stock at time t or even indicators or other variables to describe the environment's dynamics).

As such, the algorithm decides at time t upon an *action* $a_t \in A(s_t)$, where $A(s_t)$ is a series of valid actions such as *buy, sell* or *hold*, if all of these actions are available to the state s_t.

One time step later, after the RL algorithm action had been translated in trade orders, the market returns it's feedback in the form of a positive or negative reward (trade's profit or loss) $r_t \in R$, where R_t is a series of numeric rewards received every time step the RL system takes an action $r_t, r_{t+1}, r_{t+2}, ..., r_{t+k}$. The cumulative return at the end of the trading period would become $G_t = R_{t+1} + R_{t+2} + ... + R_T$ (3.3) where T is the terminal time step of an episodic period.(Sutton and Barto; 2017)
At any given time step increment, RL agent is confronted with a decisional process to choose the next action for the next state from the available actions for that state. At a very basic level, a special case of RL solution is a N armed bandit problem, where the agent is placed in the situation to choose an arm. As some actions at different time steps can have different rewards, the agent is required to map the state space and estimate the rewards would get if would take a specific action, in other words to estimate trade rules or trading *policy* π_t calculated as a value function of state-action pair values at time step t. $\pi_t(s, a)$ is the probability of taking an action $a_t = a$ when a state $s_t = s$ is received.(Sutton and Barto; 2017)

For a limited amount of states and actions, the state-action values can be stored even as a 'table', which brought the name of estimated value function methodology as 'Q-Learning'. Once the feedback from the trading action is returned, the neural network will compare the predicted reward to the real reward received and improves for the next state the prediction and chooses an action with better chances to perform well.

Discounted reward
Profit or loss is calculated with regard to the buying and selling price, and as such, the reward is based on cumulative function of rewards and we can notate with G_t to determine the actual value of reward. Additionally, the reward today is not equally significant to a trader weeks later, so a discount parameter $\gamma, 0 \leq \gamma \geq 1$ is introduced to differentiate the immediate rewards to the rewards k steps away (for $\gamma = 0$ pure rewards is sought, no exploration) and the cumulative reward formula above becomes: $G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} ... + \Sigma_{k=0}^{\infty} \gamma^k R_{t+k+1}$ (3.4)(Sutton and Barto; 2017) where t is the time sequence of k steps in a finite process $t < T$ with T the final time step.

The system progress to next state s_{t+1} and the cycle repeats itself number of times equal to the length of time steps.

Markov property. Markov Decision Process
The Markov property for the environment's states is assumed in the state-action function mapping, due to the fact that it infers a numeric limit to the states and rewards instances. For simplification, definition of value function is formulated for present time step t+1 and previous step t, allowing one step (1 day for daily granularity data, for example) prediction as a hint to the agent decision maker.(Sutton and Barto; 2017)

The trading decision takes the form of a finite Markov Decision Process (MDP). Probability of the agent moving from current state s in the next state s' depends on the

certainty that the current state s had occurred and the reward r had been awarded for the action a taken. Mathematically can be written as:
$p(s', r|s, a) = Pr\{S_{t+1} = s', R_{t+1} = r|S_t = s, A_t = a$ (3.5) (McClure; 2017)

Epsilon. Exploration versus exploitation
In the context of defined goal of achieving the maximum profit from trading activities, the policy π would choose the actions which would bring the most profit *exploiting* only a particular policy. A blunt implementation of business goal such as maximizing the profit would choose in *a greedy* manner the actions with the maximum rewards. However, a such approach could become disadvantageous to the business goal if the RL system does not explore other states/actions pairs, which could represent missed opportunities. And so implementations have to take in consideration an optimal solution to the "*exploitation\exploration dilemma*". (Sutton and Barto; 1998),(White; 2012).

3.3 Backtester

The Backtester class is called to translate the signal into orders at the training and testing stage, assuming the role of interpreter between the reinforcement learning agent and the financial market and supply the Model with reward/revenue feedback. In a traditional architecture, Backtester would instantiate variables for all the parameters required to obtain, store and use stock price data, order and develop trading rules. Due to the special structure of our algorithm, Backtester bears the role of an interpreter of RNN model to the outside world of financial market, translating the trading signals in concrete actions such as *buy, sell or hold*. Backtester class takes the input trade signals and trade with the given price data. We run the Backtester class twice, for cross validation with n=2 folds, one fold for the train stage and one fold for test. The same data set used by the RNN model to produce trading signals at training stage is used to translate environment's rewards to the trading signals. Similarly, at test stage. The reward feedback from the Backtester is feeding back to the RNN algorithm to update the real values of the pairs value-action in the Q-table, comparing to the predicted values by the neural network and update the prediction weights accordingly, as the RNN proceeds to the next time step t_{t+1} and the next state s_{t+1}.

4 Machine Learning Model - RNN architecture

An RNN contains multiple layers of recurrent units or cores. RNN is essentially an artificial neural network (ANN) with hidden layers which are connecting back to themselves (backpropagation?). RNN can use different types of core layers, each core architecture implementing the information analysis and decision control with diffrent approaches.(LazyProgrammer; 2016)

The model stacks linearly multiple neuron layers, all connected to each other in a many to many topology. For a fully connected neural network the structure can be expressed mathematically as $y = \sigma(Ax)$ (4.1) (McClure; 2017)

Here x is the input variable or in our case a sequence of variables describing the state and y is the output or neural network, the estimation with regard to the next state. Weights are calculated as Ax with a function σ which can be implemented using a sigmoid,

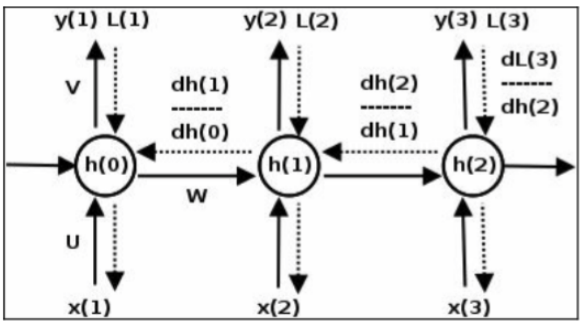

Figure 1: figure
3 layer RNN (Gulli and Pal; 2017)

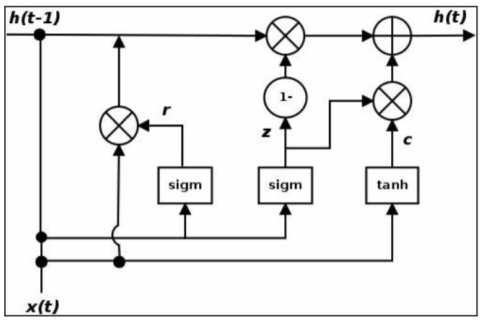

Figure 4.2: figure
GRU From:(Gulli and Pal; 2017)

tangent or relu activation function. Taking the previous time step in consideration, the formula becomes:

$$y = \sigma(B\ddot{y}_{t-1} + Ax_t) \text{ (McClure; 2017)} \tag{4.2}$$

GRU core Introduced in 2014 (Cho et al.; 2014), Gated Recurrent Unit (GRU) had been said to be a less complex, with less parameters to tune and comparable learning performance. We are aiming at investigating if this holds true for our analysis purposes. The GRU RNN had been said to be resistant to the vanishing gradient problem, while having a less complex architecture and requiring less computing power.(Gulli and Pal; 2017) As it can be observed in the figure 3.2 for the GRU core architecture, between the hidden layers h(t-1) and h(t), there are 2 gates gate z "update" and gate r "reset". z gate defines the amount of experience memory to be propagated trough neural network, while reset gate would filter the old and new information input towards the next hidden layer. The formulas to describe the weights in the neural network for the gates are:

$$z = \sigma(W_z\ddot{h}_{t-1} + U_z x_t) \tag{4.3}$$

$$r = \sigma(W_r\ddot{h}_{t-1} + U_t x_t) \tag{4.4}$$

$$c = tanh(W_c(\ddot{h}_{t-1} \otimes r) + U_c x_t) \tag{4.5}$$

$$h_t = (z \otimes c) \oplus \ddot{((}1 - z) \otimes h_{t-1}) \tag{4.6}$$

5 Implementation

The journey to implement and evaluate a self trading algorithm had to take in consideration recommendation from the software development, machine learning and financial technical analysis fields.

5.1 Data

Data collection The data used has been acquired from Quandl and covers daily prices for all stocks traded between January 2000 and September 2017. The data set contains multiple columns, however to download had been chosen the *open, low high and close* prices. The columns are necessary to calculate the indicators to describe and represent the market's trend or the market's *state*.

Assumptions about data Financial data comes as a time series, where the values represent the price evolution in time.The price and volume reflect the stock's value at a certain point in time and incorporate past evolution and it's ability to constitute a potential opportunity for profit to its owner.

Price data presents evolution trends that can generate trading opportunity and these trends can be identified and exploited. Short term trends can be exploited only having access to smaller granularity data and appropriate infrastructure to allow information transfer within the persistence of such trends (orders sent fast enough to Exchange or broker to avoid slippage and information dissipation). (Bandy; 2007) Price data it is assumed in our model to fulfil Markov property. With the resources and the available free data, medium and long term trend are targeted by the reinforcement learning (RL) algorithm.

Given established goal is to mimic a human trader,researcher assumes financial market has inefficiencies to different degrees and there are medium to long term price trends strong enough to constitute opportunity to achieve profit.

5.2 Language choices

Python programming language had been used for the implementation easiness, its support for different programming paradigms and open source character, which allows a better community support in development process. Along with the Python itself, multiple libraries are customized to a multitude of research domains such as finance and artificial intelligence, the main interest with regard to this project. The libraries used have special sectioning in the configuration manual to make a note of the installation procedure and call to these libraries.

5.3 RNN structure

The recurrent neural network is a sequential model that stacks multiple RNN cells. The layers used are two Gated Recurrent Units, with 128 hidden neurons. The Dense core layer is using a uniform initializer which has the goal to provide tensors with uniform distribution. Two Dropout layers with a drop out rate of 0.5 would prevent neural network from over fitting. Activation function is a softmax. For the compiling state,Stochastic Gradient Descent optimizer is used to overcome slowing rate of learning and as a loss function, mean square error measures the efficiency of the model.

6 Analysis

Analysis goals At a first run, the algorithm seems to achieve a modest profit. However it is known that any machine learning application results are influenced by the input data. For a human trader, choosing a good performing stock to trade on is as important as choosing when to buy and sell. As such, the researcher had questioned whether the profit of the algorithm it is influenced by the stocks trend that was chosen to trade. It is questioned whether a stock that exhibits a growth trend and increase of price has influence over the profit achieved and in what measure. It is questioned whether the algorithm it is capable of learning a profitable strategy for a stock exhibiting a down trend or a falling sale value and it could achieve profit, in spite the odds. It is questioned whether the proposed trading algorithm has ability to achieve profit, whether small or

big, regardless of the chosen stock. The goal is to investigate correlation between the return of the asset and return achieved with the trading algorithm.

Researcher questions whether the algorithm will be able to compensate for the bad performance of the stock with further optimization and testing. A sound trading algorithm would not be safe to implement in production , if it does not allow a screening of the available stocks and the risk associated. The data set or chosen stock that is fed in deep learning model cannot be arbitrary.

Analysis methodology Analysis follows the SEMMA guidelines with respect to the knowledge mining methodology and was implemented in stages such as sampling the data, exploration of the data set and visualization, modifications to missing data and NAN values, data preparation to serve as input to the RNN model and assessment of the accuracy achieved by the leaning model.

Sampling Cross validation method is implemented to avoid over fitting the RNN model on the regression task of predicting the value-action function. For cross validation, data set is divided in 2 lots, a data set destined for training and a data set for testing. Data set for training takes all daily price data for a 15 years window, in the range of 2000-2015. To test the trading performance of the algorithm, the period 2016-up to November 2017 is reserved.

Exploration For a small numbers of stocks, chosen as result to portfolio pre analysis and selection, visual exploration of time series aimed to assert normality and distribution shape. Mean, mode and data length are observed. Plots can be visualized in figure.

Modifications The NAN's or missing values attributed to closed for business Exchange, Holiday or stocks not trading for particular lengths of time had been replaced with 0 values. *Scaling* the input features for the RL model is a requirement in order to achieve a quicker convergence to optimization problem. Data set had been scaled as a normal distribution with mean 0 and and single standard deviation unit.

Evaluation

RNN Model metrics

Since we intend to predict the values describing the next state, to infer a suitable trading decision, the neural network task is a numeric prediction task. To evaluate numeric prediction accuracy, a common metric is mean-square error or *MSE*. Mean squared error is used to measure the error neural network performance in estimating the values for the next state price trend. The MSE measures the differences between the observed state values as Y and estimated values as $\hat{Y}$, $MSE = 1/n\Sigma_{i=1}^{n}(Y_i - \hat{Y}_i)^2$.

Financial analysis metrics

Simple daily percentage change

Simple daily percentage change is an indicator to calculate the stock's price p change from a time step t-1 to time step t (or trading day). The formula used is $r_t = (p_t/p_{t-1}) - 1$ (6.1) (Heydt; 2015)

Simple daily cumulative returns
Daily rate of return, returns the portfolio value at regular time intervals. Given the daily percentage change r_t, researcher is taking in consideration as well the values from the previous time steps $i_i = (1 + r_t) * i_{t-1}$, where $i_0 = 1$ (6.2) (Heydt; 2015)
Daily returns and annual returns have been calculated, to quantify each stocks gain or loss in the daily and annual basis.
Mean of returns for each stock for a period of n observations (2000-2017) is calculated using the return for period i with a formula $R_{mean} = \frac{\Sigma_{i=1}^{n} * R_i}{n}$ (6.3) (Heydt; 2015)
Profit and Loss
Profit and loss balance is calculated to account for the return achieved owning an instrument and the cost of purchase.
Sharpe ratio
Sharpe Ratio is used to measure the return of a financial instrument with respect to the risk associated with owning or trading.

Analysis Stock selection Daily and annual cumulative returns have been calculated for over 3000 stock indexes. For 2016 and 2017, researcher had chosen minimum and maximum annual return. On the plots, the stocks exhibit high variance in price. The stocks did not follow a normal distribution.

Performance of the application in financial trading terms is measured with Sharpe ratio, profit and loss at the end of the trading period.

Trades analysis Monte Carlo method simulation has the goal to asses the system's robustness and validate the trading strategy or the policy in the self algorithm. Seeding random numbers at the decision time allows for extra exploration and reproducibility.

Trading with UNXL, the lowest annual return in 2017, revealed that the model has ability to learn profitable trends but only for a short period, however the overall result is not significantly negative. The closing price spiked in 2013, however closing price followed a down trend and it lowered to less than 5$ value. The trading algorithm used some of its original balance, and the profit and loss measure is closing at a value of -10$ loss.

ZINC index, with the lowest cumulative annual return in 2016, with a down trend of 2 years, while previously registered 3 years of rising prices. The histogram plot shows a slightly right tail fat approximately bell shape, however with a mean of 11 $ price. The algorithm learns the unusual trend of the index, and even manages to achieve a small profit at the end of the period of 154$. Given the 2 stocks that have an annual return of approximately -9% the RNN learned a minimal trading pattern to ensure very little losses and some profit, given at each trade the algorithm had to chance to buy or sell a constant amount of 100 shares. Visualizations of the trades can be observed in figure bellow.

7 Conclusion and Future Work

Seasonality of the stocks could make trading strategies ineffective.
The proposed algorithm is implementing trading for a single stock, but further development would be required to trade a portfolio with n stocks. Portfolio optimization can be achieved developing a strategy to select dynamically the most profitable stocks to trade, given the budget with no borrowing constraint and the market conditions. At any point in

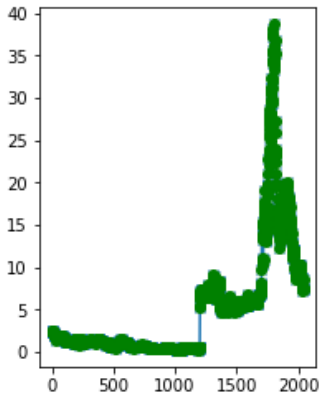

Figure 6.1: figure
UNXL trades

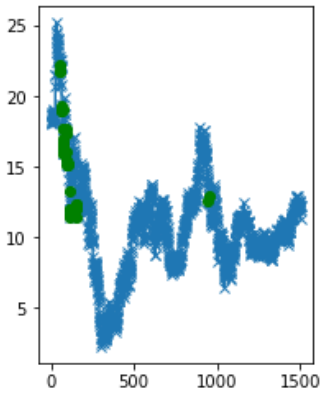

Figure 6.2: figure
ZINC trades

time, the stocks evolution can change. Stocks that used to give the opportunity of a high return on trade might stop to do so. Dynamic portfolio selection periodically (monthly, weekly) would provide the opportunity to explore new stocks, adding or removing the portfolio elements stocks that do well periodically, or in other words, explore stocks that present market inefficiencies (such as undersold or overbought).

For a policy strategy, given an initial capital, as the agent is achieving profit and experience, the agent also gains the freedom to explore more, otherwise it is bound to maximize the profit. A such strategy is implemented annealing the epsilon as the algorithm learns. Further work can be done to implement schedule on trading time. The system would require to alerts a human investor for an opportunity or a stop loss limit if no supervision required. The trading system would be required as well to account for the challenges of real trading market, such as trade charges, slippage.

Comparing a short-term strategy with a very short holding period, small annual return, but very high Sharpe ratio, to a long-term strategy with a long holding period, high annual return, but lower Sharpe ratio, it is still preferable to choose the short-term strategy even if the goal is a long-term growth. Doing so is barring tax and avoid over-borrowing to sustain the financing of the trade when short term losses occur.

The proposed algorithm is fast enough to learn price data with less than daily granularity; it could potentially generate trading signals every 1 to 2 hours interval, given the situation where the minute data granularity is feed in the system. Recurrent neural network had been chosen due to ability to handle sequential data, large amounts of states , achieve better performance due to back propagation. Recurrent neural networks are the reason artificial intelligence field had made major progress lately.

Computing power can be a downside factor in implementing a deep learning model with a complex environment state mapping. Distributed version would provide better use of the resources available. Although RNN networks have achieved remarkable results in sequential prediction, there still many problems we haven't accounted for in our algorithm. Because we performed testing on already available data, there is a lookahead bias when z-score is calculated. Trading test had not taken into account transaction costs, bid/ask spread.

The algorithm is far from ready to be production implemented and would require further research and modelling to be able to perform self trades without human supervision. As well as that, it has been said, the traders pray to make a profit on the market's

inefficiency. We have assumed there are inefficiencies in the market to exploit. In the hypothetical case of all traders being efficient and automatized, such inefficiencies would cease to exist, and so, much of the profit made from trading activities for the sole purpose of speculation would disappear.

References

Alexander, C. (2009). *Market risk analysis, value at risk models*, Vol. 4, John Wiley & Sons.

Alvarez Diaz, M. (2010). Speculative strategies in the foreign exchange market based on genetic programming predictions, *Applied Financial Economics* **20**(6): 465–476.

Bandy, H. B. (2007). Quantitative trading systems.

Bandy, H. B. (2015). *Quantitative Technical Analysis: An integrated approach to trading system development and trading management*, Blue Owl Press, Incorporated.

Berenji, H. R. (1992). A reinforcement learning—based architecture for fuzzy logic control, *International Journal of Approximate Reasoning* **6**(2): 267 – 292.
URL: *http://www.sciencedirect.com/science/article/pii/0888613X9290020Z*

Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation, *arXiv preprint arXiv:1406.1078* .

Cumming, J., Alrajeh, D. and Dickens, L. (2015). *An Investigation into the Use of Reinforcement Learning Techniques within the Algorithmic Trading Domain*.

Davey, K. (2014). *Building Algorithmic Trading Systems: A Trader's Journey from Data Mining to Monte Carlo Simulation to Live Trading*, John Wiley & Sons.

Dempster, M. A. and Leemans, V. (2006). An automated fx trading system using adaptive reinforcement learning, *Expert Systems with Applications* **30**(3): 543–552.

Deng, Y., Bao, F., Kong, Y., Ren, Z. and 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.

Du, X., Zhai, J. and Lv, K. (2016). Algorithm trading using q-learning and recurrent reinforcement learning, *positions* **1**: 1.

Fama, E. F. (1991). Efficient capital markets: Ii, *The journal of finance* **46**(5): 1575–1617.

Fama, E. F. (1995). Random walks in stock market prices, *Financial analysts journal* **51**(1): 75–80.

Gulli, A. and Pal, S. (2017). *Deep Learning with Keras*, Packt Publishing.

Heydt, M. (2015). *Mastering pandas for finance*, Packt Publishing Ltd.

Kim, K. (2010). *Electronic and algorithmic trading technology: the complete guide*, Academic Press.

Kirkpatrick II, C. D. and Dahlquist, J. A. (2010). *Technical analysis: the complete resource for financial market technicians*, FT press.

Kramer, O. (2016). *Machine Learning for Evolution Strategies*, Vol. 20, Springer.

LazyProgrammer (2016). *Deep Learning:Recurrent Neural Networks in Python:LSTM, GRU and more RNN machine learning architectures in Python and Theano*, Amazon Digital Services LLS.

Malkiel, B. G. and Fama, E. F. (1970). Efficient capital markets: A review of theory and empirical work, *The journal of Finance* **25**(2): 383–417.

McClure, N. (2017). *TensorFlow Machine Learning Cookbook*, Packt Publishing.
URL: *https://books.google.ie/books?id=LVQoDwAAQBAJ*

Moody, J. and Saffell, M. (2001). Learning to trade via direct reinforcement, *IEEE transactions on neural Networks* **12**(4): 875–889.

Schumaker, R. P. and Chen, H. (2009). Textual analysis of stock market prediction using breaking financial news: The azfin text system, *ACM Transactions on Information Systems (TOIS)* **27**(2): 12.

Silver, D. (2016). Introduction to reinforcement learning, [Online] Available at: http://www.cs.ucl.ac.uk/staff/D.Silver/web/Teaching_files/intro_RL.pdf. [Accessed 21 February 2017].

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement learning: An introduction*, Vol. 1, MIT press Cambridge.

Sutton, R. S. and Barto, A. G. (2017). *Reinforcement Learning: An Introduction*, Vol. draft, MIT press Cambridge.

Szepesvari, C. (2010). *Algorithms for Reinforcement Learning (Synthesis Lectures on Artificial Intelligence and Machine Learning)*, *Morgan and Claypool* .

White, J. (2012). *Bandit algorithms for website optimization*, " O'Reilly Media, Inc.".