Summer 2019

Less is more: Beating the market with recurrent reinforcement learning

Louis Kurt Bernhard Steinmeister

LESS IS MORE: BEATING THE MARKET WITH RECURRENT REINFORCEMENT LEARNING

by

LOUIS KURT BERNHARD STEINMEISTER

A THESIS

Presented to the Graduate Faculty of the

MISSOURI UNIVERSITY OF SCIENCE AND TECHNOLOGY

In Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

in

APPLIED MATHEMATICS WITH STATISTICS EMPHASIS

2019

Approved by

V. A. Samaranayake, Advisor
Donald C. Wunsch II, Co-Advisor
Wenqing Hu

PUBLICATION THESIS OPTION

This thesis consists of the following article which has been submitted for publication, or will be submitted for publication as follows:

Paper I: Pages 23-41 have been submitted to NeurIPS Conference.

ABSTRACT

Multiple recurrent reinforcement learners were implemented to make trading decisions based on real and freely available macro-economic data. The learning algorithm and different reinforcement functions (the Differential Sharpe Ratio, Differential Downside Deviation Ratio and Returns) were revised and the performances were compared while transaction costs were taken into account. (This is important for practical implementations even though many publications ignore this consideration.) It was assumed that the traders make long-short decisions in the S&P500 with complementary 3-month treasury bill investments. Leveraged positions in the S&P500 were disallowed. Notably, the Differential Sharpe Ratio and the Differential Downside Deviation Ratio are risk adjusted and are therefore expected to yield more stable and less risky strategies. Interestingly, the Return-traders performed the most consistently. Explanations for these findings were explored. The strong performance of the return-based traders - even based on few and readily available macro-economic time series - showed the power and practical relevance of the simpler algorithm.

ACKNOWLEDGMENTS

The author would like to thank Professor V. A. Samaranayake from the department Mathematics and Statistics and Professor Donald C. Wunsch from the department of Electrical and Computer Engineering for excellent advise given and support during difficult times. He would also like to thank Markus and Walburga Steinmeister as well as Yuan Gui for personal advise and support throughout his academic career. Additionally, the author would like to appreciate Professor Wenqing Hu from the department of Mathematics and Statistics for joining his thesis committee and Professor Markus Pauly from Dortmund University, formerly from Ulm University, for the excellent education he provided the author in Statistics. Special thanks goes to Christof Steinmeister for positively impacting the author's personal and professional development.

TABLE OF CONTENTS

Page

SECTION

PAPER

SECTION

LIST OF ILLUSTRATIONS

Figure Page

SECTION

PAPER I

SECTION

LIST OF TABLES

Table Page

SECTION

PAPER I

SECTION

1. INTRODUCTION

In the following, we are implementing reinforcement learning agents for investment decisions with the objective of comparing the performance of different reinforcement functions. The mathematical formulation of the model is given as a Markov Decision Process, which we shall revise below.

Markov Decision Process

The mathematical formulation of stochastic control problems is often given as a Markov Decision Process, which is defined in Definition 1.1. As we observe in Figure 1.1 the model can be interpreted as an environment and an actor or decision maker. The environment supplies the agent with information regarding the current state. Based on this information, the agent makes a decision with the goal of maximizing rewards or minimizing penalties, which supplied by the environment after an action was taken and can depend on the new state. In Section 2, we will identify and these components in the financial setting and explore different possible options for our model.

The nature of financial data is distinctly different from data in other settings. Therefore, we shall firstly discuss the subsequently arising challenges regarding state and action space. Moreover, it is unclear how to rationally assess the performance of a trading strategy or rather certain investment decisions. This is an issue that finance has tried to answer for centuries, leading to different approaches which we discuss in the second subsection.

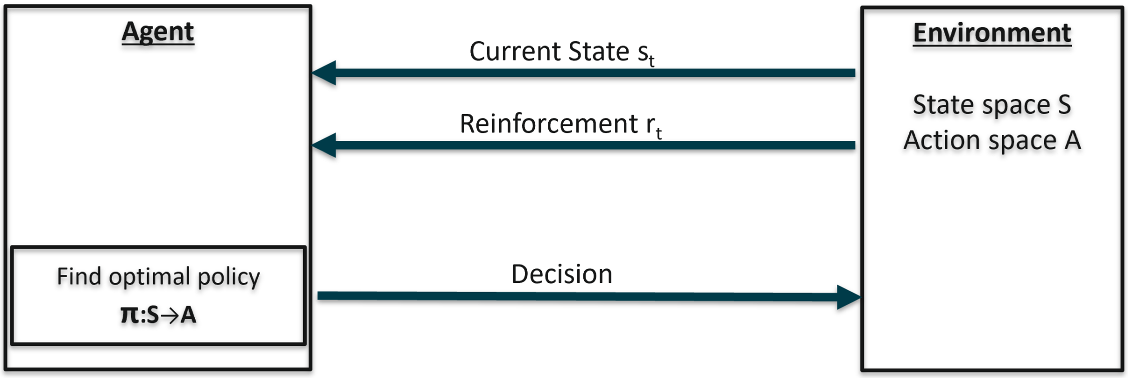


Figure 1.1. Illustration of a Markov Decision Process.

Definition 1.1: Markov Decision Process

A (stationary) Markov decision Process is defined as $(S \times A, D, Q, r, \beta)$, where

- S denotes the state space, endowed with σ-field $\mathcal{S}$
- A denotes the action space, which is endowed with a σ-field $\mathcal{A}$
- $D \subset S \times A$ is measurable w.r.t. $\sigma(\mathcal{S} \cup \mathcal{A})$, which is the smallest σ-field containing both $\mathcal{S}$ and $\mathcal{A}$), and denotes the set of possible state-action pairs.
- Q is the transition kernel such that $B \mapsto Q(B|s, a)$ is a probability measure $\forall B \in \mathcal{S},\ s \in S,\ a \in A$.
- r denotes a measurable function $r : S \times A \mapsto \mathbf{R}$, which denotes the expected reward $r(s, a) = \int_S \text{reward}(s, a, \tilde{a}) Q(d\tilde{a}|s, a)$, if the current state is s and action a is taken.
- β is a constant discount rate.

This definition is illustrated in Figure 1.1. For further reference, the reader is encouraged to see [Bäuerle and Rieder, 2011, Puterman, 2014].

2. DATA STRUCTURE,PRE-PROCESSING AND IMPLICATIONS

Let us consider data from the S&P500 ranging from November of 1960 to November 2018. The observable behavior closely resembles that of a geometric Brownian motion with upwards trend. One way of interpreting a stochastic process is to view it as a random variable on an (infinitely) high dimensional product space, where dependencies can, for instance, be modeled using Markov kernels. This clearly demonstrates one major issue that we encounter in finance: The whole stock market is a single multivariate observation containing all tradable assets. The obvious consequence is that classical statistical approaches relying on multiple observations are rendered impractical (compare Figure 2.1).

Statisticians have, by introducing time series analysis, found a way around this problem. The observed processes remain as before, however, by assuming certain models or structures, inference and prediction (at least in the short term) is possible. One such model is the **A**uto**R**egressive **M**oving **A**verage (ARMA), is given by a (stationary) processes $\{X_t\}_{t\in\mathbb{Z}}$ fulfilling

$$X_t - \phi_1 X_{t-1} - \ldots - \phi_p X_{t-p} = Z_t + \theta_1 Z_{t-1} + \ldots + \theta_q Z_{t-q}$$

where $p, q \in \mathbb{N}$ denote the autoregressive and the moving average order respectively and $\{Z_t\}_{t\in\mathbb{Z}}$ is some white noise process [Brockwell *et al.*, 2002].

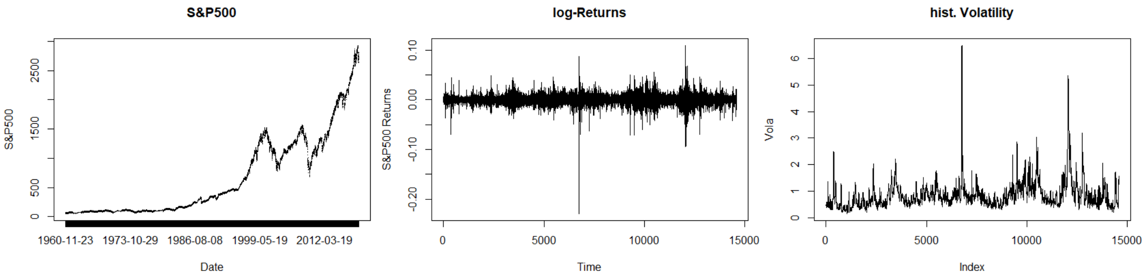


Figure 2.1. From left to right: S&P500 Index from November 1960 to November 2018, log-returns and historical Volatility respectively.

However, there are good reasons for why such a static model might not be fully appropriate. On one hand, if tradable patterns are observed, market participants will quickly start making money from them causing the effect to vanish. On the other hand, governments might choose to change tax codes, regulations, or provide subsidies, which may alter the way markets function. Moreover, the economy is evolving over time as innovations build upon each other and we learn from historical events. This leads to an ever-changing environment and assuming that, despite these effects, core mechanics remain the same is a strong assumption.

Another issue that is difficult to tackle is that many risks in the past were not observed. Assume, for instance, that the United States had lost the cold war. Today we know, that the chances for this were diminishingly low. However, based on the partial knowledge at the time of the conflict, this risk may have seem much more severe. The conclusion is that highly unlikely events with high impact have not occurred or our knowledge of them is so limited that our assessment of past risks and opportunities may be inaccurate [Taleb, 2007]. Reinforcement learning, in contrast, is capable to learn hidden patterns implicitly through trial and error. This feature may prove valuable for learning patterns which might not be easily observable and adopting dynamically to changing environments. In order to implement a reinforcement learning algorithm for investment decisions, we need create a model for the agent (see Section 3.1) and the environment (section 3.2) in which it acts. Before we can come to the implementation and set up a model for analysis, we need to obtain and pre-process the data. For more details on the data selection (i.e. the definition of the state and action space), see Section 3.2.

Data assimilation and pre-processing

Table 2.1 gives a brief summary of the data sources of the inputs used. After the data was obtained and merged in Power Query, the database tool in Microsoft Excel, the economic indicators were lagged by one month to adjust to ensure that we are not relying on future data in our decision making, since only the data of the past month is known when making a decision for the next. Furthermore, missing data was replaced by data from the previous month.

Table 2.1. Summary of input data with data source.

Data Type	**Source**	**Period**	**Interval**	**Use**
S&P500 Price Index (ˆGSPC)	Yahoo Finance	1950-01-01 / 2019-04-01	monthly	log-returns
AAA Corporate Bond Yield	FRED	1919-01-01 / 2019-03-01	monthly	6 month difference
10 year Treasury Rate	FRED	1953-04-01 / 2019-03-01	monthly	Yield Curve
3 month Treasury Bill Rate	FRED	1934-01-01 / 2019-03-01	monthly	Yield Curve, 6 month difference, risk free rate

Data Normalization

Additionally, we need to normalize the inputs since we are employing L1-regularization (weight decay). Because L1-regularization shrinks all weights equally, we want the inputs to be centered around zero and to have similar ranges. At the same time, we do not want to use future information in the process of normalization (since we aim for an online-algorithm as described in Section 3.1b. However, the maximum and minimum depends on future observations. In order for the algorithm to learn from data which has similar ranges as unobserved data in the markets, we estimate the minimum and maximum using a heuristic, which builds on results from Extreme Value Theory. The algorithm goes as follows:

1. Determine the minimum and maximum of the first few observations. The exact number may depend on the history available. Generally, the more data, the more exact the estimate will become. At the same time, we do not want to cut too deep into our training set in order to have a realistic learning environment. We decided to use up to $n = 48$ months of training data for this step and any additional history before that.

2. The definition of a maximum and a minimum trivially entails, that it superseded or went below the previous historical maximum or minimum respectively. Hence, we need to adjust up- and downwards. To achieve this, we compute the 95% quantile of the extreme value distribution of the normal distribution (which is the Gumbel distribution as shown in Theorem 2.1) for both the sample size n, which we used in step 1, and for the size of the data set N. Dividing the later by the former then gives a multiplier which we apply to our computed minimum and maximum.

3. Once, the estimated minimum $\tilde{m}$ and maximum $\tilde{M}$ are obtained, the data is normalized as $Z_n := \frac{X_n - \zeta_{0.5}}{\tilde{M} - \tilde{m}}$, where $\zeta_{0.5}$ is the median of $X = X_1, ..., X_n$.

Theorem 2.1: Extreme Value Distribution of the Normal distribution

Let $X = X_1, ..., X_N \overset{i.i.d}{\sim} N(0, 1)$. Then

$$\frac{M_N - a_N}{b_N} \underset{N \longrightarrow \infty}{\overset{\mathcal{D}}{\longrightarrow}} Z \sim G,$$

where $M_N := \max\{X_1, ..., X_N\}$, G is the Gumbel distribution and $a_N := \frac{1}{N\Phi(b_N)}$ and $b_N := \Phi^{-1}\left(1 - \frac{1}{N}\right)$ with Φ being the cumulative distribution function of the Normal distribution. For proof, see [De Haan, 1976, Herbert and Nagaraja, 2003].

After this transformation, the economic indicators, which will serve as inputs into our model, have a more similar range as seen in Figure 2.2.

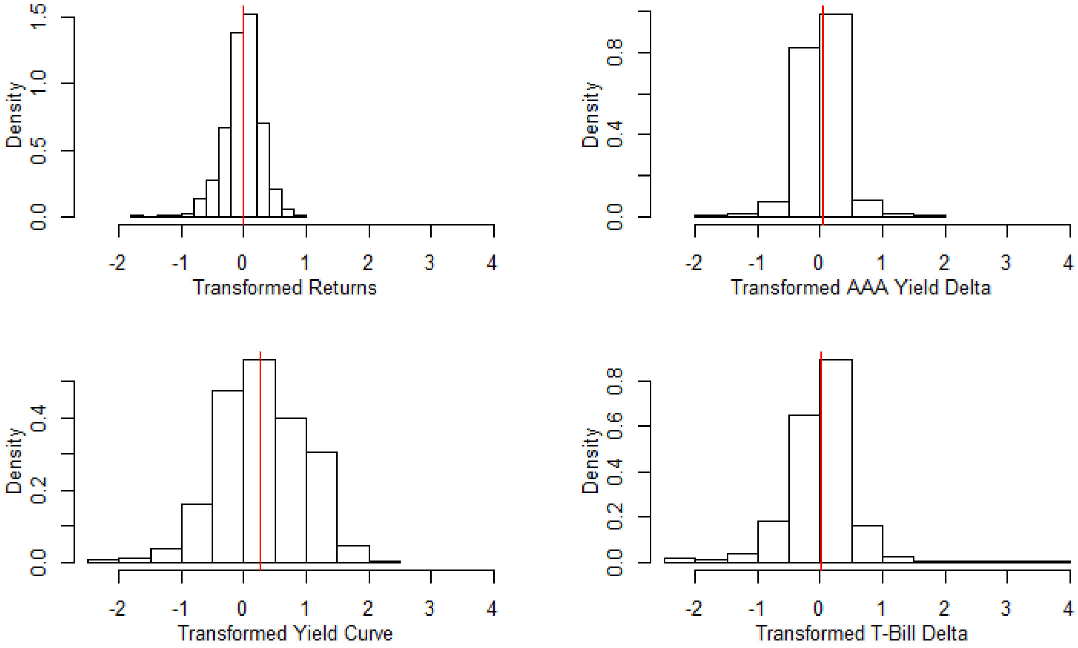


Figure 2.2. Histograms of the economic data. The red line indicates the location of the median.

3. DEFINING THE MODEL AS A MARKOV DECISION PROCESS

As we noted in Definition 1.1, there are multiple aspects to a Markov Decision Process. In this chapter, we will explore how we can translate our problem into a Markov Decision Process and use reinforcement learning to optimize risk adjusted returns. This is a classical control problem with a few twists to it in our special case. Hence, we examine how the our situation fits into the framework of agent (in Section 3.1) and environment (in Section 3.2).

3.1. THE AGENT

Past experiments suggest, that an integrated decision-making module performs best in this scenario [Bengio, 1997][bengio, early moody?]. Works by Moody et al. further indicate that a policy gradient algorithm implementing a recurrent neural network performs well and provides a comprehensive interpretation. Moody et al. compare how traders implementing the Q-learning algorithm, which tries to estimate values for state-action pairs and derives the policy implicitly, performs against a recurrent reinforcement learning (RRL) traders and the market. They find that the RRL traders are trading less frequently and are able to outperform the Q-learning traders and the market. Possible reasons for these findings include

1. The 2-modular approach using distinct prediction and decision-making modules creates a bottleneck. Additional information, f.e. regarding the accuracy of the prediction, is neglected. This causes a shrinkage of the σ-field, which we condition on to make decisions and hence our decision making is negatively impacted.

2. The Q-learning approach may suffer from the "curse of dimensionality". Many of the features, on which we base the decision-making, can be unbounded and continuous. Furthermore, the dimensions quickly explode as we expand to consider further time series. This may demand for a large and deep neural network, which may be slow to train.

3. The Q-learning algorithm is discontinuous with regard to updates. In other words, a small change in the Q-function may cause a radically different policy due to the arg max operator.

4. The RRL algorithm may be faster to adopt to non-stationarities in financial markets, since it does not need to unlearn an outdated Q-function.

This motivates the implementation of an integrated (single module) trading system, which is trained via RRL.

Learning Rule

Depending on how the agent is designed and the reinforcement is received from the environment (see Section 3.4), different learning approaches are required. Both rules, which are presented here are based on gradient descent. Firstly, we explore how feedback for an interval of decision points is handled. Secondly, we proceed by deriving a learning rule for online-feedback, where reinforcement is provided between two successive points of decision making. For the sake of completeness, much of this section will be a reiteration of Section 3 of the submitted paper.

We consider a trader in one asset, who can make buying and (short-)selling decisions. Hence, the desired output at time t of our trading system is $a_t \in [-1, 1]$. Here, a trading decision of $a_t = 1$ would refer to a long position (buying or holding 100% of the available funds in the asset) in the asset, whereas $a_t = -1$ refers to a short position in the same asset (borrowing an equivalent of 100% of the available funds of the asset's shares and selling them in the market to buy them back at a later time - hopefully at a lower price). The algorithm we use has previously been proposed by Moody, *et al.* [Moody and Saffell, 2001, Moody *et al.*, 1999, Moody and Wu, 1997, Moody *et al.*, 1998, Moody and Saffell, 1999]. They suggest a single neuron direct reinforcement learning algorithm using a tanh activation function. This design is beneficial since it allows to easily account for transaction costs and other market frictions such as tax. They demonstrate that their direct reinforcement trader outperforms their trading algorithm based on the Q-learning algorithm. Therefore, we focus our attention to the direct reinforcement trader with the tanh activation, which guarantees the desired output range.

3.1.1. Batch Update. The consideration of transaction costs implies a recurrent structure, since the investment decision a_t depends partly on the previous decision a_{t-1}. We want to optimize a utility U of portfolio returns $(R^{(p)}_{d+1}, R^{(p)}_{d+2}, ...)$ which are generated over time (for optimization, we consider a finite horizon of $T \in \mathbb{N}$ such returns). The trader will be supplied with a moving window of $d \in \mathbb{N}$ past asset returns $(R_{t-d+1}, ..., R_t)^T$ and $h \in \mathbb{N}$ further economic indicators $(E_1, ..., E_h)^T$. Together, these form the vector of economic inputs $I_t := (R_{t-d+1}, ..., R_t, E_1, .., E_h)^T)$. Additionally, the previous asset allocation or investment decision a_t is provided. The policy of the direct reinforcement trader is parameterized by $\theta \in \mathbb{R}^m$. In our simple case ($m = d + h + 1$), this will look as follows:

$$a_{t+1} = F\left(a_t, I_t|\theta\right) = \tanh\left(\theta_1 R_{t-d+1} + ... + \theta_d R_t + \theta_{d+1} E_1 + ... + \theta_{d+h} E_h + \theta_{d+h+1} a_t\right)$$

It should be noted that this is similar to a simple generalized linear model in the covariates $\left([I_t^T, a_t]^T\right)_{t\in\mathbb{N}}$ with response $(a_{t+1})_{t\in\mathbb{N}}$. However, these response variables are unobserved and are chosen in a way that maximizes the utility or reinforcement function. Rather then obtaining a closed form expression as we might in the case of a linear model, we will therefore maximize the target function U via a gradient ascent optimizer as seen in Figure 3.1a (assuming U is differentiable with regard to the set of weights θ almost everywhere). Hence, let us proceed to computing the gradients:

$$\frac{dU_T}{d\theta} = \frac{dU_T\left(R^{(p)}_{d+1}(a_d, a_{d+1}), ..., R^{(p)}_{d+T}(a_{T-1}, a_{d+T})\right)}{d\theta} = \sum_{t=d+1}^{T} \frac{\partial U_T}{\partial R^{(p)}_t}\left[\frac{\partial R^{(p)}_t}{\partial a_t}\frac{da_t}{d\theta} + \frac{\partial R^{(p)}_t}{\partial a_{t-1}}\frac{da_{t-1}}{d\theta}\right],$$

where $\frac{da_t}{d\theta}$ is recursively given by $\frac{da_t}{d\theta} = \frac{\partial a_t}{\partial\theta} + \frac{\partial a_t}{\partial a_{t-1}}\frac{da_{t-1}}{d\theta}$.

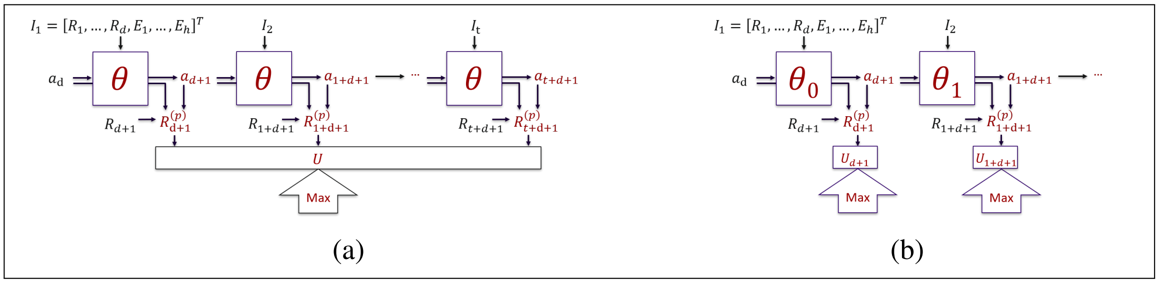


Figure 3.1. Diagram of (a) the batch version and (b) the incremental version of the algorithm.

3.1.2. Incremental Update. The approach as given above has a few weaknesses. Gradient ascent is an iterative optimization algorithm and we would need to run through the whole dataset in order to compute a single iteration. Moreover, due to possible non-stationarity of the environment and to allow for continuous learning and adaptation we are interested in an online system. Furthermore, the time frame for our online trading system should not be bounded by a finite number T. If we can break up U_T as follows

$$U_T\left(R^{(p)}_{d+1}(a_d, a_{d+1}), ..., R^{(p)}_{d+T}(a_{T-1}, a_{d+T})\right) = \sum_{t=d+1}^{T} U^{(t)}_T(R^{(p)}_t) =: \sum_{t=d+1}^{T} U^{(t)}(R^{(p)}_t),$$

then we are able to remove the time constraint and apply incremental updates by maximizing each summand individually. The resulting algorithm is illustrated in Figure 3.1b. We approximate the gradients as follows:

$$\begin{aligned}\frac{dU^{(t)}}{d\theta_t} = \frac{dU^{(t)}\left(R^{(p)}_t(a_{t-1}, a_t)\right)}{d\theta_t} &= \frac{\partial U_t}{\partial R^{(p)}_t}\left[\frac{\partial R^{(p)}_t}{\partial a_t}\frac{da_t}{d\theta_t} + \frac{\partial R^{(p)}_t}{\partial a_{t-1}}\frac{da_{t-1}}{d\theta_t}\right] \\ &\approx \frac{\partial U_t}{\partial R^{(p)}_t}\left[\frac{\partial R^{(p)}_t}{\partial a_t}\frac{da_t}{d\theta_t} + \frac{\partial R^{(p)}_t}{\partial a_{t-1}}\frac{da_{t-1}}{d\theta_{t-1}}\right],\end{aligned}$$

where $\frac{da_t}{d\theta_t} = \frac{\partial a_t}{\partial\theta_t} + \frac{\partial a_t}{\partial a_{t-1}}\frac{da_{t-1}}{d\theta_t} \approx \frac{\partial a_t}{\partial\theta_t} + \frac{\partial a_t}{\partial a_{t-1}}\frac{da_{t-1}}{d\theta_{t-1}}$.

According to [Williams and Zipser, 1989] the approximation error is negligible for small learning rates. Additionally, it allows us to efficiently compute the gradients since we only need to store the total derivative $\frac{da_t}{d\theta_t}$. Note the similarity to backpropagation through time [Werbos *et al.*, 1990]. Now that we have discussed the updating scheme, let us turn to the evaluation of the performance of financial assets and which criterion to optimize.

Furthermore, they suggest multiple candidates for the reinforcement function which we shall explore in the next section.

3.2. THE ENVIRONMENT

As we noted in Section 1, the environment supplies the actor with information about the current state and penalizes or rewards actions. Hence, we discuss which variables to consider and supply to the agent. Furthermore, we review some economic results regarding decision making under risk in Section 3.3 to construct plausible reinforcement signals in Section 3.4.

The State- and Action Spaces

We have a few variables which are likely to impact our investment decision and should therefore be contained in our state space:

- *Past stock prices:* $S_t^{(h)} := (S_t, ..., S_{t-h})^T$

  There is a whole branch of finance called *technical analysis* which is dedicated to analyzing and trading patterns in past stock prices. reinforcement learning could help us discover significant higher order patterns, which are not easily observable, or at least learn. Neglecting this aspect could undercut the performance [Blume *et al.*, 1994, Lo *et al.*, 2000], even though some have argued that technical analysis may be self-fulfilling [Taylor and Allen, 1992].

- *Additional Economic data:* I_t

  including additional economic data such as the yield curve, Commitment of Trader (CoT) Reports, unemployment rates, etc. may help to make better investment decisions.

- *Past Asset Allocation:* F_{t-1}

  If we are interested to potentially implement the trading system, we need to take taxation and transaction costs into account. Hence, past asset allocations will impact allocation decisions in the future.

- *Exploration Parameter:* z_t

  We may want to include a noisy parameter which does not contain any information for purposes of exploration. This allows to explore different decisions if all other inputs are held equal. In the financial setting, however, we already have much noise. Therefore, we will not end up running a single sub-optimal greedy policy and we can neglect this input.

The goal is to arrive at a portfolio. Hence the action space relies on the number of assets we are considering. Generally, we can limit our action to

$$\left(F_t, w_t^f\right) \in [-1, 1]^d \times \mathbb{R}$$

where $d \in \mathbb{N}$ denotes the number of risky assets and $F_t := \left(F_t^1, ..., F_t^d\right)^T$ is a vector containing the respective portfolio weights and w_t^f is the weight of the "risk free" asset. A negative weight corresponds to a short position in that asset and conversely a positive weight represents a long position. We require all weights to sum up to 1:

$$\sum_{k=1}^{d} F_t^{(k)} + w_t^f = 1$$

For now, however, we shall only consider a single risky asset as other authors have done, see [Deng *et al.*, 2017, Du *et al.*, 2016, Moody and Saffell, 2001].

3.3. THE REINFORCEMENT

In order to derive, understand and interpret the reinforcement functions that we present later, we need to study decision making under risk. In order to motivate the economic theory, we introduce the well-known St. Petersburg Paradox in Section 3.3.1. In Section 3.3.2 we give the classical response first formulated by [Bernoulli, 2011]. We then move on to the commonly used model for financial decision making in Section 3.3.3. We proceed by reviewing different reinforcement functions in Section 3.4.

Decision Making under Risk

In economic Theory, there are generally two different approaches to describe and advise decision making in the presence of unsure outcomes. We present these in Sections 3.2 and 3.3 after a short motivation in Section 3.3.1.

3.3.1. The St. Petersburg Paradox. Assume that you are offered the following lottery (A), which you may enter, if your bid is accepted by the counterparty:

i) You start off with an amount of 2$.

ii) A fair coin is tossed.

iii) If the coin shows head, the amount is doubled. Then go to step ii). If the coin shows tail, the game ends and you win the amount.

Now, how much are you willing to bid in order to enter the lottery? You may quickly notice that the expected payoff is infinite:

$$E[A] = \sum_{k=1}^{\infty} \left(\frac{1}{2}\right)^k 2^k = \sum_{k=1}^{\infty} 1 = \infty$$

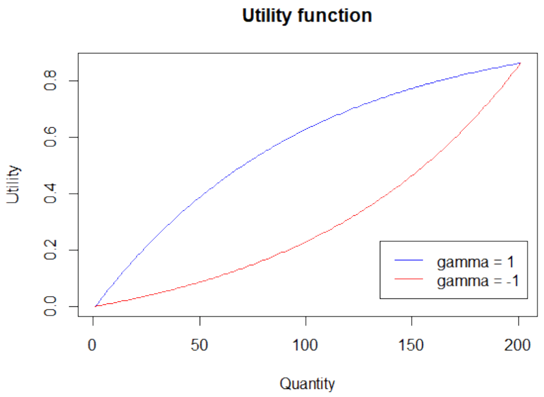


Figure 3.2. Rescaled Utility for the exponential utility function: $u(c) := \frac{1-exp(-\gamma c)}{\gamma}$. Positive γ implies risk aversion, whereas negative γ results in a convex utility function implying risk seeking behavior. And $\gamma = 0$ describes risk neutral preferences.

However, most people would only bid relatively small amounts for this lottery. It may seem like a contradiction but there is an explanation for this behavior: Due to limited resources and minimum consumption required for survival, risk adverse behavior is rational. This behavior can be described by the use of utility functions.

3.3.2. The Utility Function. The utility function is helps to describe rational decision making under risk. Usually, diminishing marginal utility can be observed (compare Figure 3.2). In other words, the higher the quantity of a good, the less useful will an additional unit of that quantity be. This is equivalent to risk aversion and is described by an increasing (more is better) and concave utility function. A popular class of utility functions is the exponential utility function as shown below [Bernoulli, 2011].

However, in finance there is usually a different approach which is applied to assess the performance and make risky decisions:

3.3.3. Mean-Variance Analysis. Harry Markowitz famously suggested considering risky assets in an (expected return, volatility)-plane [Markowitz, 1952]. Let us assume, that we have two risky assets as shown on the left hand side in Figure 3.3, with an expected return and where the risk is given by the volatility of the return. Now, the assumption is

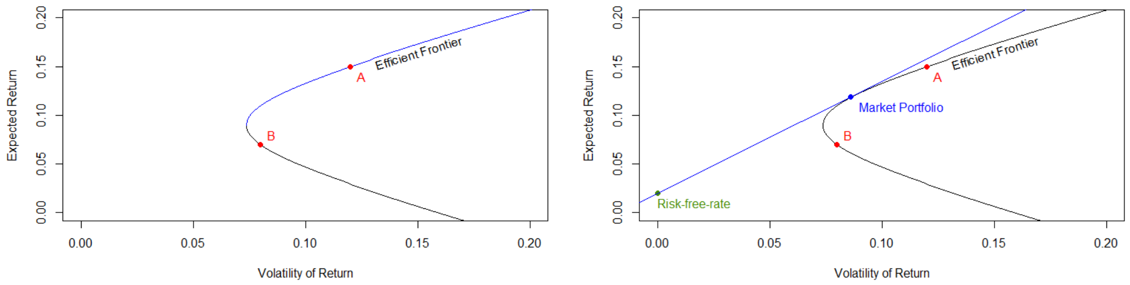


Figure 3.3. On the left hand side, we have two assets, A and B, in the expected return-volatility-plane. The curve represents the possible portfolio combinations of assets. A similar case holds, if the investor has more then two investment opportunities. On the right hand side, we introduce a risk free rate, at which the investor may borrow or deposit. The resulting portfolios can be illustrated like as blue line, which represents the one such line with maximum slope.

that rational investors seek to maximize return for any fixed and achievable amount of risk (volatility). Doing so yields portfolios which lie on the efficient frontier. These, are also called dominant portfolios (compare to Definition 3.1. Introducing a risk free rate produces a new set of dominant portfolios, as observed on the right hand side of Figure 3.3.

Definition 3.1: Dominant Portfolio

A dominant portfolio is a portfolio, which yields a superior return given a lower amount of risk (standard deviation). In Figure 3.3 the dominant portfolios are given in blue.

Of the previously dominant portfolios, only one portfolio remains dominant after introducing the risk free rate, which we shall refer to as the market portfolio. It is characterized by having maximal Sharpe Ratio, which is introduced in Definition 3.2. For any given amount of risk (volatility) one simply deposits or borrows money at the risk free rate and invests in the tangent or market portfolio. Since it has a larger Sharpe Ratio, these mixtures will always produce a more attractive investment than any other portfolio. We observe that this principle holds for any two assets with differing Sharpe Ratios. A rational investor (assuming our assumptions are satisfied) will always prefer to allocate his resources in the portfolio with maximal Sharpe Ratio. Therefore, the Sharpe Ratio can be used to assess risk adjusted performance of investments and to represent the preferences of a rational investor [Sharpe, 1994].

Definition 3.2: Sharpe Ratio

The Sharpe Ratio $SR(A)$ of an asset A is defined as the slope of the line passing through $(0, R_f)$ and $(\sigma(r_A), E[r_A])$, where the net return of the asset A is denoted by r_A and σ denotes the volatility or standard deviation. Hence, the Sharpe Ratio us given by

$$SR(A) = \frac{E[r_A] - r_f}{\sigma(r_A)}.$$

3.4. POSSIBLE REINFORCEMENT FUNCTIONS

After having discussed the state and action spaces, an obvious question remains on which feedback we provide our learning agent. In Section 3.3, we discussed a few potential options. John Moody and various co-authors have discussed different reinforcement functions [Moody and Saffell, 2001, Moody and Wu, 1997, Moody *et al.*, 1998, Moody and Saffell, 1999]. Nevertheless, let us summarize some observations and advantages and disadvantages regarding diverse reinforcement functions.

3.4.1. Return. The most apparent choice would be to use the return or the net profit of an investment decision. We can maximize the total gross return $R_{(T)}$ of the trading system. At time t, we denote the gross return $R_t = 1 + r_t$, with net return r_t (f.e. $r_t = 2\%$). Since the total gross return is the product of the gross returns up until time T:

$$R_{(T)} = \prod_{t=1}^{T} R_t = exp\left(\sum_{t=1}^{T} \tilde{R}_t\right),$$

where $\tilde{R}_t := log(R_t)$ is the log-return. Hence, maximizing $R_{(T)}$ is equivalent to maximizing $\tilde{R}_t,\ \forall t = 1, ..., T$, which is equivalent to maximizing $R_t,\ \forall t = 1, ..., T$ individually.

However, a high risk high return strategy would get rewarded highly. From previous observations in Section 3.3, we have learnt that we need to adjust for risk. Otherwise, we might obtain higher expected returns with the same amount of risk (or lower risk at the same expected return). This could be remedied by using the utility of investment decisions instead. Here, risk is accounted for. However, the utility of a decision also depends on the wealth and risk preference of the observer.This makes formulating a universal reinforcement rather difficult.

3.4.2. Differential Sharpe Ratio. The approach of using the Sharpe Ratio introduced in portfolio theory, is an attractive alternative. It does not depend on personal wealth and risk preferences can be incorperated later - by choosing an according mixture of the strategy (the risky portfolio) and the risk free asset. Nevertheless, if we consider the definition we see that the Sharpe Ratio is the quotient of the expected risk premium and the expected volatility: $SR(A) = \frac{Er_A - r_f}{\sigma(r_A)}$. These are future values that are not easily determined. Usually, the expectation and the variance are estimated from past data. However, it is not clear that past data is a good predictor of future values due to the non-stationarity (at least in a conditional sense) of the data. Furthermore, positive past Sharpe Ratios would significantly impact future estimates. This may lead to a situation, where bad decisions get rewarded, even though we are making decisions that reduce the estimate of the Sharpe Ratio (take an unproportionally high amount of risk to achive the return). This behavior is troublesome. Hence, let us rewrite the estimate of the Sharpe Ration in terms of moment estimates of the returns to obtain incremental changes to the overall Sharpe Ratio:
First Moment or arithmatic mean:

$$A_n := \frac{1}{n}\sum_{t=1}^{n} r_t = A_{n-1} + \frac{1}{n}(r_n - A_{n-1})$$

Second Moment (non-central):

$$B_n := \frac{1}{n}\sum_{t=1}^{n} r_t^2 = B_{n-1} + \frac{1}{n}(r_n^2 - B_{n-1})$$

Now, instead weighing the update term with $1/n$ we can generalize the above terms to:

$$A_n = A_{n-1+}\eta_n(r_n - A_{n-1}),\ B_n = B_{n-1} + \eta_n(r_n^2 - B_{n-1}),$$

where we obtain the previous estimates by setting $\eta_n = 1/n$. Instead, we could also use exponential smoothing to arrive at exponential moving estimates by holding $\eta_n \equiv \eta$ constant as suggested by John Moody and Lizhong Wu[Moody and Wu, 1997]. This might be a good approach, since the impact of old observations decays at an exponential rate and makes the algorithm adopt faster to a changing setting. Also, new observations will have a larger impact on the exponential moving average since the adoption rate η is held constant over time, whereas it would converge to zero with the arithmetic average. This may prove to be an advantage given that we do not need to assume stationarity. Furthermore, it will make a few derivations easier. However, this is a change that might bias and make it inconsistent. This trade-off could be further investigated.

Taylor expansion around $\eta = 0$ yields:

$$SR_n(\eta) = SR_{n-1}(\eta) + \eta\ \frac{d}{d\eta}SR_n(\eta)\bigg|_{\eta=0} + O(\eta^2),$$

where the change in the Sharpe Ratio with regard to r_n is only given in the second term (all higher order terms contained in $O(\eta^2)$ are independent of r_n) and $SR_\eta(n-1)$ is simply the previous Sharpe Ratio. Therefore we define

Definition 3.3: Differential Sharpe Ratio

The Differential Sharpe Ratio, which is motivated above, is defined as

$$DSR_n := \frac{d}{d\eta} SR_n(\eta)\bigg|_{\eta=0}.$$

using some simple calculus, we can show that[Moody and Wu, 1997]

$$DSR_n = \frac{B_{n-1}\Delta A_n - \frac{1}{2}A_{n-1}\Delta B_n}{\left(B_{n-1} - A_{n-1}^2\right)^{3/2}},$$

where $\Delta A_n := r_n - A_{n-1}$ and $\Delta B_n := r_n^2 - B_{n-1}$.

The main drawback of this performance function is, that it penalizes both extraordinarily high and small returns. In fact [Moody and Saffell, 2001] note that the DSR is maximized by $r_n^* = \frac{B_{n-1}}{A_{n-1}}$. This may not be an optimal feature for training. A solution is presented in the next section.

3.4.3. Differential Downside Deviation Ratio. The Downside Deviation Ratio (DDR) is similar to the Sharpe Ratio, which was introduced in Section 1. The difference lies in the definition of Risk. To obtain the Sharpe Ratio, we defined Risk in terms of volatility of net returns. However, the impact of a negative deviation of net returns is much heavier then that of a positive deviation. The case where the expected net return is $Er_t = 2\%$. A negative deviation of -22% requires a positive deviation of 23% just to break even, let alone meet expectation. Hence, it may be rational to perceive of risk asymmetrically. An example is the Downside Deviation, which results in the Downside Deviation Ratio as given in Definition 3.4.

Definition 3.4: Downside Deviation Ratio

The Downside Deviation with an investor specific expectation $h \geq 0$ is defined as

$$DD_n := E^{1/2}\left[\min\left(R_n - h, 0\right)^2\right]$$

and induces the Downside Deviation Ratio:

$$DDR_n := \frac{E[r_n] - r_f}{DD_n}, .$$

Similar as in Section 3.4.2, we can use exponential moving averages

$$\begin{aligned} A_n &:= A_{n-1} + \eta\left(r_n - A_{n-1}\right) \\ D_n^2 &:= D_{n-1}^2 + \eta\left(\min\left(r_n - h, 0\right)^2 - D_{n-1}^2\right) \end{aligned}$$

to approximate the Downside Deviation Ratio. Using the Taylor expansion, we once again arrive at an incremental reinforcement, which is given in definition 3.5.

Definition 3.5: Differential Downside Deviation Ratio

The Differential Downside Deviation Ratio with investor-specific expectation $h \geq 0$ is defined as below.

$$DDDR_n := \frac{d}{d\eta} DDR_n(\eta)\bigg|_{\eta=0} = \begin{cases} \frac{r_n - \frac{1}{2}A_{n-1}}{D_{n-1}^3} & , \; r_n - h > 0 \\ \frac{D_{n-1}^2\left(r_n - \frac{1}{2}A_{n-1}\right) - \frac{1}{2}A_{n-1}r_n^2}{D_{n-1}^3} & , \; r_n - h \leq 0 \end{cases}$$

Now that we have introduced and justified Returns, Differential Sharpe Ratio and the Differential Downside Deviation Ratio, we proceed to compare the performance of the discussed trading system receiving reinforcement from these functions. Afterwards, we discuss some points, which were left out for the publication for brevity.

PAPER

I. LESS IS MORE: BEATING THE MARKET WITH RECURRENT REINFORCEMENT LEARNING

Louis K. Steinmeister

Applied Computational Intelligence Laboratory

Department of Mathematics and Statistics

Missouri University of Science and Technology

Rolla, MO 65409

mail: lks8kd@mst.edu

V. A. Samaranayake

Department of Mathematics and Statistics

Missouri University of Science and Technology

Rolla, MO 65409

mail: vsam@mst.edu

Donald C. Wunsch II

Applied Computational Intelligence Laboratory

Department of Electrical & Computer Engineering

Missouri University of Science and Technology

Rolla, MO

mail: dwunsch@mst.edu

ABSTRACT

Multiple recurrent reinforcement learners were implemented to make trading decisions based on real and freely available macro-economic data. The learning algorithm and different reinforcement functions (the Differential Sharpe Ratio, Differential Downside Deviation Ratio and Returns) were revised and the performances were compared while transaction costs were taken into account. (This is important for practical implementations even though many publications ignore this consideration.) It was assumed that the traders make long-short decisions in the S&P500 with complementary 3-month treasury bill investments. Leveraged positions in the S&P500 were disallowed. Notably, the Differential Sharpe Ratio and the Differential Downside Deviation Ratio are risk adjusted and are therefore expected to yield more stable and less risky strategies. Interestingly, the Return-traders performed the most consistently. Explanations for these findings were explored. The strong performance of the return-based traders - even based on few and readily available macro-economic time series - showed the power and practical relevance of the simpler algorithm.

1. INTRODUCTION

The development of computers and the Internet have drastically changed and enhanced the efficiency of financial markets. Today, numerous Robo-Advisors advertise to provide investors with automated investment strategies tailored to their personal risk profile. At the same time, high frequency traders are replacing market makers and arguably increasing market efficiency and thus reducing bid-ask spreads when beneficial, which translates to a more efficient allocation of liquidity and resources [Carrion, 2013]. In other optimization problems like job scheduling [Zhang and Dietterich, 1995] in operations research, control problems [Lillicrap *et al.*, 2015] or various games such as Atari games [Mnih *et al.*, 2013], Markov Decision Processes and reinforcement learning have demonstrated that they can produce positive results and have often outperformed the previous state of the art.

There is a distinction in reinforcement learning, namely between value-based and policy-based methods. One such policy-based algorithm is policy gradients [Sutton *et al.*, 2000], which we use in conjunction with recurrent Neural Networks (rNNs). These are popularly tested in the area of speech recognition [Graves *et al.*, 2013] but can be applied to any environment with sequential data and were thus employed to solve (dynamic) control problems [Ku and Lee, 1995] and time series prediction [Connor *et al.*, 1994]. Given this background, Moody et al. [Moody and Saffell, 2001] suggested to apply reinforcement learning, using an rNN structure for the policy approximation, to online trading systems. They employed a single tanh-Neuron for the decision making and maximizing a financial criterion, namely the Differential Sharpe Ratio, which they introduced [Moody and Saffell, 2001, Moody *et al.*, 1999, Moody and Wu, 1997, Moody *et al.*, 1998, Moody and Saffell, 1999]. We revise this algorithm in Section 3 and use it to compare how maximizing Returns, the Differential Sharpe Ratio and the Differential Downside Deviation Ratio, as derived in Section 4.2, contrast against each other. In Section 5 we discuss the details of our experiments, including the data used and its sources, the partitioning of the available data into offline training, online validation and online testing sets and the hyperparameters we employed. The results including some possible explanations are covered in Section 6. Also, we compare the findings during the validation phase from Section 6.1 with those from the testing phase in 6.2 and suggest a selection procedure to boost the performance of the trading system. Furthermore, we explore how well the online phase is able to adopt to changes by comparing the results with a model, which was retrained on the training and validation set. But first, let us explore what has been published in this area prior to this work.

2. PRIOR LITERATURE

In the financial setting, Neural Networks have traditionally been applied to stock price prediction [Kim and Han, 2000, Kohara *et al.*, 1997, Saad *et al.*, 1998, White, 1988]. However, in 1997 Yoshua Bengio suggested using an integrated system that directly learns to make investment decisions instead of a modular system consisting of a prediction module and a decision module [Bengio, 1997]. In the same period, Moody et al. introduced the idea of using a recurrent neural network model trained via reinforcement learning and Gradient Decent to directly optimize a financial criterion. In the same paper they suggested optimizing additive returns, an economic utility function (see [Bernoulli, 2011]), and the Differential Sharpe Ratio, which we define later [Moody and Wu, 1997, Moody *et al.*, 1998, Moody and Saffell, 1999]. Later, they proceeded to show impressive performance in a synthetic and real world environments (Forex Trading and S&P500) and that approximating the policy directly using a single tanh-Neuron outperforms their Q-Learning trader. Furthermore, they suggested that alternatively the Downside-Deviation can be utilized as a risk measure [Moody and Saffell, 2001, Moody *et al.*, 1999]. Dempster et al. employed a recurrent reinforcement learning trader in combination with a risk management layer. Hyperparameters were automatically tuned [Dempster and Leemans, 2006]. Deng et al. implemented a variation of Moody's RRL-Trader using different approaches for more efficient signal representation [Deng *et al.*, 2017, 2015]. Almahdi et al. extended the RRL-Trader to a multiple asset setting. They proposed the use of Expected Maximum Drawdown as a risk measure. They emphasized the improvement of statistical properties and its resilience against increased transaction costs [Almahdi and Yang, 2017].

3. NEURAL NETWORK DESIGN AND LEARNING RULE

We consider a trader in one asset, who can make buying and (short-)selling decisions. Hence, the desired output at time t of our trading system is $a_t \in [-1, 1]$. Here, a trading decision of $a_t = 1$ would refer to a long position (buying or holding 100% of the available funds in the asset) in the asset, whereas $a_t = -1$ refers to a short position in the same asset (borrowing an equivalent of 100% of the available funds of the asset's shares and selling them in the market to buy them back at a later time - hopefully at a lower price). The algorithm we use has previously been proposed by Moody, et al [Moody and Saffell, 2001, Moody *et al.*, 1999, Moody and Wu, 1997, Moody *et al.*, 1998, Moody and Saffell, 1999]. They suggest a single neuron direct reinforcement learning algorithm using a tanh activation function. This design is beneficial since it allows to easily account for transaction costs and other market frictions such as tax. They demonstrate that a their direct reinforcement trader outperforms their trading algorithm based on the Q-learning algorithm. Therefore, we focus our attention to the direct reinforcement trader. Moreover, the tanh activation guarantees the desired output range.

3.1. BATCH UPDATE

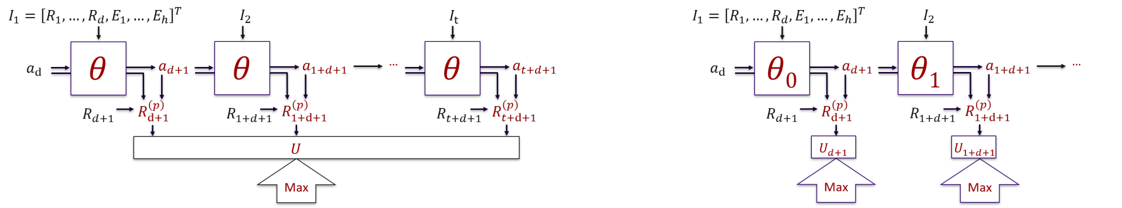
The consideration of transaction costs implies a recurrent structure, since the investment decision a_t depends partly on the previous decision a_{t-1}. We want to optimize a utility U of portfolio returns $(R^{(p)}_{d+1}, R^{(p)}_{d+2}, ...)$ which are generated over time (for optimization, we consider a finite horizon of $T \in \mathbb{N}$ such returns). The trader will be supplied with a moving window of $d \in \mathbb{N}$ past asset returns $(R_{t-d+1}, ..., R_t)^T$ and $h \in \mathbb{N}$ further economic indicators $(E_1, ..., E_h)^T$. Together, these form the vector of economic inputs $I_t := (R_{t-d+1}, ..., R_t, E_1, .., E_h)^T)$. Additionally, the previous asset allocation or investment decision a_t is provided. The policy of the direct reinforcement trader is parameterized by $\theta \in \mathbb{R}^m$. In our simple case ($m = d + h + 1$), this will look as follows:

$$a_{t+1} = F\left(a_t, I_t|\theta\right) = \tanh\left(\theta_1 R_{t-d+1} + ... + \theta_d R_t + \theta_{d+1} E_1 + ... + \theta_{d+h} E_h + \theta_{d+h+1} a_t\right)$$

It should be noted that this is similar to a simple generalized linear model in the covariates $\left([I_t^T, a_t]^T\right)_{t\in\mathbb{N}}$ with response $(a_{t+1})_{t\in\mathbb{N}}$. However, these response variables are unobserved and are chosen in a way that maximizes the utility or reinforcement function. Rather then obtaining a closed form expression as we might in the case of a linear model, we will therefore maximize the target function U via a gradient ascent optimizer as seen in Figure 1a (assuming U is differentiable with regard to the set of weights θ almost everywhere). Hence, let us proceed to computing the gradients:

$$\frac{dU_T}{d\theta} = \frac{dU_T\left(R^{(p)}_{d+1}(a_d, a_{d+1}), ..., R^{(p)}_{d+T}(a_{T-1}, a_{d+T})\right)}{d\theta} = \sum_{t=d+1}^{T} \frac{\partial U_T}{\partial R^{(p)}_t}\left[\frac{\partial R^{(p)}_t}{\partial a_t}\frac{da_t}{d\theta} + \frac{\partial R^{(p)}_t}{\partial a_{t-1}}\frac{da_{t-1}}{d\theta}\right],$$

where $\frac{da_t}{d\theta}$ is recursively given by $\frac{da_t}{d\theta} = \frac{\partial a_t}{\partial \theta} + \frac{\partial a_t}{\partial a_{t-1}}\frac{da_{t-1}}{d\theta}$.



(a) Diagram of the batch version of the algorithm.

(b) Diagram of the incremental version of the algorithm.

Figure 1. Diagram of the a) batch version and b) the incremental version of the algorithm.

3.2. INCREMENTAL UPDATE

The appoach as given above has a few weaknesses. Gradient ascent is an iterative optimization algorithm and we would need to run through the whole dataset in order to compute a single iteration. Moreover, due to possible non-stationarity of the environment and to allow for continous learning and adaptation we are interested in an online system. Furthermore, the timeframe for our online trading system should not be bounded by a finite number T. If we can break up U_T as follows

$$U_T\left(R^{(p)}_{d+1}(a_d, a_{d+1}), ..., R^{(p)}_{d+T}(a_{T-1}, a_{d+T})\right) = \sum_{t=d+1}^{T} U^{(t)}_T(R^{(p)}_t) =: \sum_{t=d+1}^{T} U^{(t)}(R^{(p)}_t),$$

then we are able to remove the time constraint and apply incremental updates by maximizing each summand individually. The resulting algorithm is illustrated in Figure 1b. We approximate the gradients as follows:

$$\begin{aligned}\frac{dU^{(t)}}{d\theta_t} = \frac{dU^{(t)}\left(R_t^{(p)}(a_{t-1}, a_t)\right)}{d\theta_t} &= \frac{\partial U_t}{\partial R_t^{(p)}}\left[\frac{\partial R_t^{(p)}}{\partial a_t}\frac{da_t}{d\theta_t} + \frac{\partial R_t^{(p)}}{\partial a_{t-1}}\frac{da_{t-1}}{d\theta_t}\right] \\ &\approx \frac{\partial U_t}{\partial R_t^{(p)}}\left[\frac{\partial R_t^{(p)}}{\partial a_t}\frac{da_t}{d\theta_t} + \frac{\partial R_t^{(p)}}{\partial a_{t-1}}\frac{da_{t-1}}{d\theta_{t-1}}\right],\end{aligned}$$

where $\frac{da_t}{d\theta_t} = \frac{\partial a_t}{\partial \theta_t} + \frac{\partial a_t}{\partial a_{t-1}}\frac{da_{t-1}}{d\theta_t} \approx \frac{\partial a_t}{\partial \theta_t} + \frac{\partial a_t}{\partial a_{t-1}}\frac{da_{t-1}}{d\theta_{t-1}}$.

According to [Williams and Zipser, 1989] the approximation error is negligible for small learning rates. Additionally, it allows us to efficiently compute the gradients since we only need to store the total derivative $\frac{da_t}{d\theta_t}$. Note the similarity to backpropagation through time [Werbos *et al.*, 1990]. Now that we have discussed the updating scheme, let us turn to the evaluation of the performance of financial assets and which criterion to optimize.

4. FINANCIAL PERFORMANCE EVALUATION AND REINFORCEMENT

The most intuitive assessment of financial performance is the return of a portfolio. Therefore, we formally derive incremental updates utilizing the return of our trading system before proceeding to the derivation of the Differential Sharpe Ratio and the Differential Downside Deviation Ratio.

4.1. RETURN

We can maximize the total gross return $R_{(T)}$ of the trading system. At time t, we denote the gross return $R_t = 1 + r_t$, with net return r_t (f.e. $r_t = 2\%$). Since the total gross return is the product of the gross returns up until time T:

$$R_{(T)} = \prod_{t=1}^{T} R_t = exp\left(\sum_{t=1}^{T} \tilde{R}_t\right),$$

where $\tilde{R}_t := log(R_t)$ is the log-return. Hence, maximizing $R_{(T)}$ is equivalent to maximizing $\tilde{R}_t, \ \forall t = 1, ..., T$, which is equivalent to maximizing $R_t, \ \forall t = 1, ..., T$ individually.

However, this performance function does not take risk into account adequately. It may therefore adopt high return policies that are very risky. For example, it may - under a different model - learn to leverage the market portfolio. Assuming the market portfolio generally follows an upwards trend, the trader will yield a multiple of the positive market risk premium. This, however, comes at the cost of higher β (market risk according to the Capital Asset Pricing Model (CAPM) [Cochrane, 2009]), which means that the trader has not outperformed the market but merely purchased a higher expected return with increased undiversifiable risk.

4.2. SHARPE RATIO AND DOWNSIDE DEVIATION RATIO

The Sharpe Ratio has been suggested by Markowitz [Markowitz, 1952] who lay the foundation for modern portfolio theory. In short, it is assumed that risk is completely represented by the standard deviation or volatility of future returns. Instead of maximizing the returns, we maximize the ratio of expected excess returns $E\left[r_{(T)}\right] - r_f$, with r_f denoting the risk free rate, and volatility σ:

$$SR_{(T)} = \frac{E\left[r_{(T)}\right] - r_f}{\sigma\left(r_{(T)}\right)},$$

which is called the Sharpe Ratio. This quotient can be interpreted as the slope of the line representing mixtures of the risk free asset and the asset in question. Investment opportunities are thereby made comparable to each other, since the same volatility can be reached by either leveraging one or mixing the other asset with the risk free asset. The asset with higher Sharpe Ratio will thus yield a higher expected return. The Downside Deviation Ratio works similarly, only that Downside Deviation (eq. 1) is used as a measure of risk. We will use these measures to assess the performance of the trader's portfolio and to provide reinforcement. Since we do not know the true values of $E\left[r_{(T)}\right]$ and $\sigma\left(r_{(T)}\right)$, we approximate these values using the exponential moving averages

$$\begin{aligned}A_n &:= A_{n-1} + \eta\left(r_n - A_{n-1}\right)\\ B_n &:= B_{n-1} + \eta\left(r_n^2 - B_{n-1}\right).\end{aligned}$$

For a small adaptation rate $\eta > 0$, we obtain the estimator

$$\hat{SR}_n(\eta) := \frac{A_n - r_f}{(B_n - A_n^2)^{\frac{1}{2}}}.$$

Observe, that setting $\eta = \frac{1}{n}$ results in the standard moment estimates and the above estimate for the Sharpe Ratio is asymptotically unbiased. This can be transformed into an incremental performance function by Taylor expansion:

$$\hat{SR}_n(\eta) = SR_{n-1}(\eta) + \eta\,\frac{d}{d\eta}SR_n(\eta)\bigg|_{\eta=0} + O(\eta^2).$$

We therefore maximize

$$DSR_n := \frac{d}{d\eta}SR_n(\eta)\bigg|_{\eta=0} = \frac{B_{n-1}\Delta A_n - \frac{1}{2}A_{n-1}\Delta B_n}{\left(B_{n-1} - A_{n-1}^2\right)^{\frac{3}{2}}}.$$

This term was named the Differential Sharpe Ratio (DSR) by Moody et al. [Moody and Wu, 1997]. The main drawback of this performance function is, that it penalizes both extraordinarily high and small returns. In fact [Moody and Saffell, 2001] note that the DSR is maximized by $R_n^* = \frac{B_{n-1}}{A_{n-1}}$. This may not be an optimal feature for training. In contrast, the Downside-Deviation-Ratio (DDR)

$$DDR_n := \frac{E[r_n] - r_f}{DD_n}, \tag{1}$$

where $DD_n := E^{1/2}\left[\min(R_n - h, 0)^2\right]$ with an investor specific expectation $h \geq 0$, only views the negative deviation from the mean as risk, leaving the agent free to collect arbitrarily large returns. We may again use exponential moving averages

$$\begin{aligned} A_n &:= A_{n-1} + \eta\,(r_n - A_{n-1}) \\ D_n^2 &:= D_{n-1}^2 + \eta\left(\min(r_n - h, 0)^2 - D_{n-1}^2\right) \end{aligned}$$

to arrive at an incremental update

$$DDDR_n := \frac{d}{d\eta} DDR_n(\eta)\bigg|_{\eta=0} = \begin{cases} \frac{r_n - \frac{1}{2}A_{n-1}}{D_{n-1}^3} & , \; r_n - h > 0 \\ \frac{D_{n-1}^2\left(r_n - \frac{1}{2}A_{n-1}\right) - \frac{1}{2}A_{n-1} r_n^2}{D_{n-1}^3} & , \; r_n - h \leq 0 \end{cases}$$

which we may use as reinforcement as mentioned in Section 3.2.

Table 1. Summary of economic data used in the experiment.

Data Type	Source	Period	Interval	Use
S&P500 Price Index (^GSPC)	Yahoo Finance	1950-01-01 / 2019-04-01	monthly	log-returns
AAA Corporate Bond Yield	FRED	1919-01-01 / 2019-03-01	monthly	6 month difference
10 year Treasury Rate	FRED	1953-04-01 / 2019-03-01	monthly	Yield Curve
3 month Treasury Bill Rate	FRED	1934-01-01 / 2019-03-01	monthly	Yield Curve, 6 month difference, risk free rate

Table 2. Summary of data partitions.

Set	Size (months)	Purpose
Offline Training	240	Training the randomly initialized traders independently of each other over muliple epochs.
Online Validation	200	Validation of the trained models. Tuning of the hyperparameters and detection of overfitting.
Online Test	330	Testing and comparing the models on untainted data.

5. EXPERIMENTAL DESIGN

We compare 30 agents maximizing returns, the Differential Sharpe Ratio and the Differential Downside-Deviation-Ratio ($DDDR$). The algorithm was implemented in TensorFlow (version 1.12.0) and run on Google Colaboratory using Tensor Processing Units (TPUs). The analysis of the results was performed using R and Microsoft Excel. Each agent was randomly initialized using a single tanh-Neuron and trained with total of 86 inputs over a total of roughly 66 years of data. The inputs were standardized using estimates of the maxima and minima which were estimated using the first few observations and a heuristic based on extreme value theory. This allows for true online optimization without incorporating knowledge of future data points whatsoever. Furthermore, we used a total of 791 months of data (note that we used a moving time window of 21 months as inputs). Macroeconomic data was lagged by one month to account for reporting. The data set was split into 3 disjunct subsets as seen in Table 2. The algorithm was then trained offline using an ADAM-optimizer for the first 20 years of data over 100 to 500 epochs and then compared while using online Stochastic Gradient Decent for the validation period of 200 months. The remaining data remains untouched for out of sample testing to prevent data snooping [Sullivan *et al.*, 1999]. Transaction costs were fixed to 0.5‰ and the model eights were randomly initialized. We regularized the weights to prevent overfitting and saturation. Contrary to previous works [Moody and Saffell, 2001], we found that L1 regularization performed uniformly best. This seems plausible since it is more likely to produce parameters which equal zero and it can therefore also be used for variable selection [Tibshirani, 1996], which could be beneficial because some of the parameters may be insignificant and should be discarded for out-of-sample application. Hence, using L1-regularization may have similar effects to applying a filter to the inputs. For each objective function, we trained 30 models (traders) to test for consistency. We found that the parameters in Table 3 yielded the most promising results. We chose a small adaptation rate η for the Differential Downside Deviation Ratio since the denominator can vanish if we adapt too quickly to a series of returns, which lie above the expectation h. In contrast, quick adaptation is beneficial for the optimization of the Differential Sharpe Ratio since it becomes maximal for $R_n^* = \frac{B_{n-1}}{A_{n-1}}$ as noted in 4.2. Moment estimates and A_n and B_n, which do not adjust "quick enough" may therefore reduce the likelihood of or slow the convergence to positive results.

Table 3. Summary of parameter values used.

Utility of Trader	**Reg. type**	**Reg. Coeff.**	η	**epochs**
Differential Sharpe Ratio (DSR)	L1	0.01	0.08	100
Differential Downside Deviation Ratio (DDDR)	L1	0.025	0.001	500
Return	L1	0.005	-	500

6. RESULTS AND DISCUSSION

In this section, we will observe and analyze findings made during the validation phase. Afterwards, we will compare these to the results yielded by the test phase. Here we will compare two different approaches. Firstly, we will initialize the weights of the traders with the weights at the end of the validation period. Then we will compare the results with those of the same model using both the training and the online-validation set for offline-training (using the same parameters as established during validation).

6.1. FINDINGS DURING VALIDATION

A surprising result was that the Return-Trader performed most consistently during validation and resulted in the highest terminal value as visible in Figure 2. It is worth pointing out that all 30 Return-Traders also converged to a similar policy, which yields the degenerated box-plot. Both the DSR-Trader and the Return-Trader significantly outperformed the market over this period. This result is especially pleasing due to use of few and freely available macroeconomic data-series.

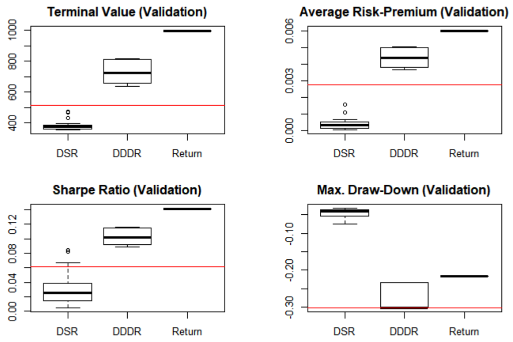
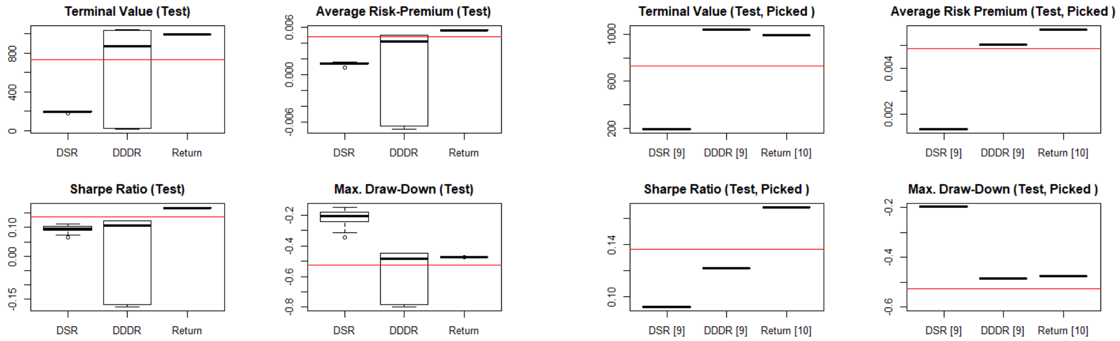


Figure 2. Comparison of the traders utilizing different performance functions. The buy-and-hold portfolio is represented as the red line and serves as a baseline. All portfolios are standardized to 100 at the beginning of the validation phase.



(a) Comparison of the traders utilizing different performance functions. The buy-and-hold portfolio is represented as the red line.

(b) Comparison of the picked traders utilizing different performance functions. The buy-and-hold portfolio is represented as the red line. According to our observation, we select the traders in each category which lie close to the top 10% quantile of Terminal Value during the Validation phase. The number of remaining traders is given on the x-axis.

Figure 3. Comparison of the a) traders utilizing different performance functions and b) the traders picked during validation.

6.2. FINDINGS DURING ONLINE TESTING

We observe that the Return-Traders continued to perform best on average. This can partly be explained by a lower trading frequency of the Return-Traders. Generally, the results of our Testing phase without retraining resemble those found during our Validation phase (compare Figure 3a and Figure 2). Contrasting our findings during validation, it seems that the higher returns of the DDDR-Traders where purchased by more then the proportional amount of standard deviation as the Sharpe Ratio falls below the market's one. Again, the Return-Traders incurred a reduced Maximum Draw Down. We notice however, that the observed statistics of the DDDR-Traders are widely spread out, as seen in Figure 3a. It turns out that they are bimodally distributed, with some traders performing very well and other doing badly. It is noteworthy that the terminal value on the test data of the DDDR-Traders shows a correlation of about 0.92 with those during validation. All traders which performed poorly during validation also performed poorly or mediocre during the test phase. Furthermore, all the DDDR-Traders performing exceptionally during the validation phase also did exceptionally well during testing. Thus, it seems that trader selection during validation could boost the overall performance. The traders utilizing the Differential Sharpe Ratio and Returns seem less effected. This may have to do with a lower range of terminal values. The results of picking the traders that perform close to the top 10% quantile in their category can be seen in Figure 3b.

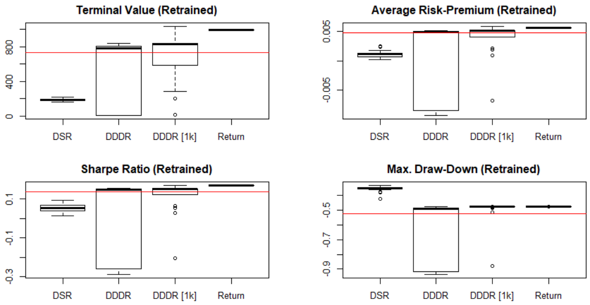


Figure 4. Comparison of the retrained traders utilizing different performance functions. The buy-and-hold portfolio is represented as the red line. DDDR [1k] refers to the DDDR-Traders with a total of 1000 training epochs.

We observed similar findings with the retrained model (now trained on the training and validation set). We conclude that the online optimization of the traders is either well at work or that no significant structural change was found. The Return-Traders benefited the most from the extended training period, whereas the DSR-Traders remained largely unchanged and the DDDR-Trader became less profitable, although further training of another 500 epochs drastically improved profitability as seen in Figure 4.

7. CONCLUSION

We have shown that both the DDDR-Traders, optimizing the Differential Downside Deviation Ratio, and the Return-Traders outperform the S&P500 over a time window of roughly 19 years. These results are enhanced when incorporating knowledge from the Online-Validation phase and selecting the Traders that performed close to the upper 10% quantile in their category. It is noteworthy that little and only freely available macroeconomic data was used to yield these results, which shows the power of the algorithm. Given that the Return-Traders perform the most consistently and also perform relatively well in the risk assessments, we conclude that for this data less is more, and the direct avoidance of negative returns and seeking maximal positive returns does better than incorporating risk. However, there could be multiple explanations for this effect. On one hand, the risk estimates may not have statistical significance in our case and may therefore lead to “confusing” reinforcements for our agents. On the other hand, we may require more inputs to effectively optimize these more complex performance functions. On limited data, there is less risk of overfitting when learning a simpler function.

The stated results should, however, be taken with a grain of salt. It is not assured that the method would perform this well if employed in actual capital markets. Firstly, only past prices can be observed. It merely monitors the price at which the last transaction took place and it is not guaranteed that the system will be able to transact at the closing price. When trading, one must deal with bid-ask spreads which are assumed to be represented in the 0.5% transaction cost. When trading larger positions, this assumption may be problematic as one may incur larger slippage costs. Furthermore, breaking up orders and execution over time exposes oneself to unaccounted market risks. Secondly, the structure of capital markets may have changed. Third, it is not clear that the analyzed employed risk measures actually measure risk adequately. Risk lies in the future but we use metrics which rely on past data in the evaluation. This effects the trading system less than the evaluation presented here, since the reinforcement functions penalize future (incurred) risk. Moreover, the shift to automated trading may already be capturing the alpha which is observable in the past, hence leaving less of a profit opportunity.

REFERENCES

Almahdi, S. and Yang, S. Y., 'An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown,' Expert Systems with Applications, 2017, **87**, pp. 267–279.

Bäuerle, N. and Rieder, U., *Markov decision processes with applications to finance*, Springer Science & Business Media, 2011.

Bengio, Y., 'Using a financial training criterion rather than a prediction criterion,' International Journal of Neural Systems, 1997, **8**(04), pp. 433–443.

Bernoulli, D., 'Exposition of a new theory on the measurement of risk,' in 'The Kelly Capital Growth Investment Criterion: Theory and Practice,' pp. 11–24, World Scientific, 2011.

Blume, L., Easley, D., and O'hara, M., 'Market statistics and technical analysis: The role of volume,' The Journal of Finance, 1994, **49**(1), pp. 153–181.

Brockwell, P. J., Davis, R. A., and Calder, M. V., *Introduction to time series and forecasting*, volume 2, Springer, 2002.

Carrion, A., 'Very fast money: High-frequency trading on the nasdaq,' Journal of Financial Markets, 2013, **16**(4), pp. 680–711.

Cochrane, J. H., *Asset pricing: Revised edition*, Princeton university press, 2009.

Connor, J. T., Martin, R. D., and Atlas, L. E., 'Recurrent neural networks and robust time series prediction,' IEEE transactions on neural networks, 1994, **5**(2), pp. 240–254.

De Haan, L., 'Sample extremes: an elementary introduction,' Technical report, 1976.

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

Deng, Y., Bao, F., Kong, Y., Ren, Z., and Dai, Q., ‘Deep direct reinforcement learning for financial signal representation and trading,’ IEEE transactions on neural networks and learning systems, 2017, **28**(3), pp. 653–664.

Deng, Y., Kong, Y., Bao, F., and Dai, Q., ‘Sparse coding-inspired optimal trading system for hft industry,’ IEEE Transactions on Industrial Informatics, 2015, **11**(2), pp. 467–475.

Du, X., Zhai, J., and Lv, K., ‘Algorithm trading using q-learning and recurrent reinforcement learning,’ positions, 2016, **1**, p. 1.

Graves, A., Mohamed, A.-r., and Hinton, G., ‘Speech recognition with deep recurrent neural networks,’ in ‘2013 IEEE international conference on acoustics, speech and signal processing,’ IEEE, 2013 pp. 6645–6649.

Herbert, A. D. and Nagaraja, H., ‘Order statistics,’ Series in probability and statistics (3rd ed.). Hoboken: Wiley, 2003.

Kim, K.-j. and Han, I., ‘Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index,’ Expert systems with Applications, 2000, **19**(2), pp. 125–132.

Kohara, K., Ishikawa, T., Fukuhara, Y., and Nakamura, Y., ‘Stock price prediction using prior knowledge and neural networks,’ Intelligent Systems in Accounting, Finance & Management, 1997, **6**(1), pp. 11–22.

Ku, C.-C. and Lee, K. Y., ‘Diagonal recurrent neural networks for dynamic systems control,’ IEEE transactions on neural networks, 1995, **6**(1), pp. 144–156.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D., ‘Continuous control with deep reinforcement learning,’ arXiv preprint arXiv:1509.02971, 2015.

Lo, A. W., Mamaysky, H., and Wang, J., ‘Foundations of technical analysis: Computational algorithms, statistical inference, and empirical implementation,’ The journal of finance, 2000, **55**(4), pp. 1705–1765.

Markowitz, H., ‘Portfolio selection,’ The journal of finance, 1952, **7**(1), pp. 77–91.

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

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

Moody, J., Saffell, M., Andrew, W. L., Abu-Mostafa, Y. S., LeBaraon, B., and Weigend, A. S., ‘Minimizing downside risk via stochastic dynamic programming,’ Computational Finance, 1999, pp. 403–415.

Moody, J. and Wu, L., ‘Optimization of trading systems and portfolios,’ in ‘Computational Intelligence for Financial Engineering (CIFEr), 1997., Proceedings of the IEEE/IAFE 1997,’ IEEE, 1997 pp. 300–307.

Moody, J., Wu, L., Liao, Y., and Saffell, M., ‘Performance functions and reinforcement learning for trading systems and portfolios,’ Journal of Forecasting, 1998, **17**(5-6), pp. 441–470.

Moody, J. E. and Saffell, M., ‘Reinforcement learning for trading,’ in ‘Advances in Neural Information Processing Systems,’ 1999 pp. 917–923.

Puterman, M. L., *Markov decision processes: discrete stochastic dynamic programming*, John Wiley & Sons, 2014.

Saad, E. W., Prokhorov, D. V., and Wunsch, D. C., ‘Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks,’ IEEE Transactions on neural networks, 1998, **9**(6), pp. 1456–1470.

Sharpe, W. F., ‘The sharpe ratio,’ Journal of portfolio management, 1994, **21**(1), pp. 49–58.

Sullivan, R., Timmermann, A., and White, H., ‘Data-snooping, technical trading rule performance, and the bootstrap,’ The journal of Finance, 1999, **54**(5), pp. 1647–1691.

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y., ‘Policy gradient methods for reinforcement learning with function approximation,’ in ‘Advances in neural information processing systems,’ 2000 pp. 1057–1063.

Taleb, N. N., ‘Black swans and the domains of statistics,’ The American Statistician, 2007, **61**(3), pp. 198–200.

Taylor, M. P. and Allen, H., ‘The use of technical analysis in the foreign exchange market,’ Journal of international Money and Finance, 1992, **11**(3), pp. 304–314.

Tibshirani, R., ‘Regression shrinkage and selection via the lasso,’ Journal of the Royal Statistical Society: Series B (Methodological), 1996, **58**(1), pp. 267–288.

Werbos, P. J. *et al.*, ‘Backpropagation through time: what it does and how to do it,’ Proceedings of the IEEE, 1990, **78**(10), pp. 1550–1560.

White, H., ‘Economic prediction using neural networks: The case of ibm daily stock returns,’ 1988.

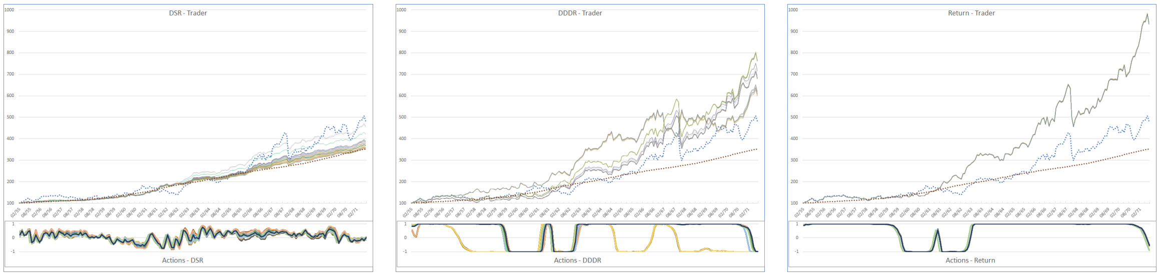
Williams, R. J. and Zipser, D., ‘A learning algorithm for continually running fully recurrent neural networks,’ Neural computation, 1989, **1**(2), pp. 270–280.

Zhang, W. and Dietterich, T. G., ‘A reinforcement learning approach to job-shop scheduling,’ in ‘IJCAI,’ volume 95, Citeseer, 1995 pp. 1114–1120.

SECTION

4. FURTHER OBSERVATIONS

While analyzing the results, we made additional observations, which were not published in Section 6 of the paper. During validation, we observe in Figure 4.1 that the Return traders trade the least, closely followed by the DDDR traders. Further, the general pattern of asset allocations of the Return and the DDDR traders is largely similar. However, the DDDR traders seem to converge to two different optima (the yellow curve contains far more short positions then the others). In comparison, the DSR traders seem to have converged to a common policy, which is by far more noisy and indecisive. These findings are reflected in the overall developments of the portfolios. The return traders do best, closely followed by the DDDR traders, wheras the DSR traders are outperformed by the market and show a similar performance to the t-bill portfolio.



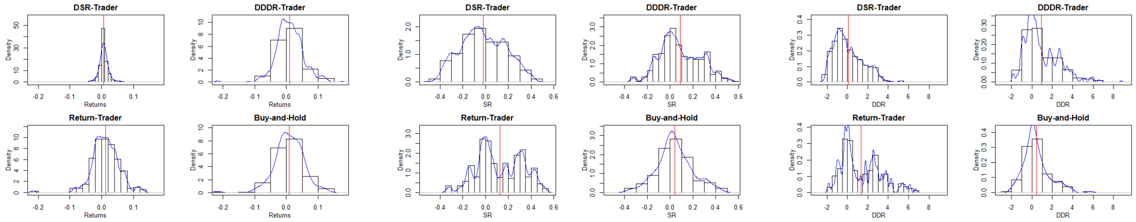
(a) Portfolios and actions of 30 Differential Sharpe Ratio traders.

(b) Portfolios and actions of 30 Differential Downside Deviation Ratio traders.

(c) Portfolios and actions of 30 Return traders.

Figure 4.1. Comparison of the portfolios and positions of 30 recurrent reinforcement learner traders employing (a) the Differential Sharpe Ratio, (b) the Differential Downside Deviation Ratio and (c) returns as reinforcement. In the upper graphs, the blue dotted line represents the market portfolio and the red dotted line represents a portfolio of reinvested t-bills. The faint lines represent a trader each. Below, the asset allocation of the different traders is given. All data is from the validation period.

One could argue, that the box plots (Figure 2 - Figure 4), which presented in the paper, contain limited information regarding the risk involved in investing in these portfolios. A general issue is that all traders were trained on the same data and therefore they are not independent from each other. Given the setup of the experiment, there is little we can do to avoid this problem. Another problem, however, arises from only considering the terminal values and the risk factors and the end of the presented time horizons. This neglects the path of arriving at these values, which might have been more or less volatile across different time frames. Hence, we present histograms of 24-month investments in the different strategies during the validation phase in Figure 4.2. These confirm the observations we made in the paper on basis of the presented box plots (Figure 2).



(a) Distribution of 2 year returns for the different traders.

(b) Distribution of 2 year Sharpe Ratios for the different traders.

(c) Distribution of 2 year Downside Deviation Ratios for the different traders.

Figure 4.2. Histograms giving the distributions of (a) returns, (b) Sharpe Ratio and (c) Downside Deviation Ratio over a random 2-year investment in the presented portfolios during validation. Notably, the observations are not independent. The traders were trained on common data and there is 23-month-dependence since there may be up to a 23-month overlap in observations. The blue curves indicate a kernel density estimate and the vertical red lines give the location of the estimates of the distribution means.

REFERENCES

Almahdi, S. and Yang, S. Y., 'An adaptive portfolio trading system: A risk-return portfolio optimization using recurrent reinforcement learning with expected maximum drawdown,' Expert Systems with Applications, 2017, **87**, pp. 267–279.

Bäuerle, N. and Rieder, U., *Markov decision processes with applications to finance*, Springer Science & Business Media, 2011.

Bengio, Y., 'Using a financial training criterion rather than a prediction criterion,' International Journal of Neural Systems, 1997, **8**(04), pp. 433–443.

Bernoulli, D., 'Exposition of a new theory on the measurement of risk,' in 'The Kelly Capital Growth Investment Criterion: Theory and Practice,' pp. 11–24, World Scientific, 2011.

Blume, L., Easley, D., and O'hara, M., 'Market statistics and technical analysis: The role of volume,' The Journal of Finance, 1994, **49**(1), pp. 153–181.

Brockwell, P. J., Davis, R. A., and Calder, M. V., *Introduction to time series and forecasting*, volume 2, Springer, 2002.

Carrion, A., 'Very fast money: High-frequency trading on the nasdaq,' Journal of Financial Markets, 2013, **16**(4), pp. 680–711.

Cochrane, J. H., *Asset pricing: Revised edition*, Princeton university press, 2009.

Connor, J. T., Martin, R. D., and Atlas, L. E., 'Recurrent neural networks and robust time series prediction,' IEEE transactions on neural networks, 1994, **5**(2), pp. 240–254.

De Haan, L., 'Sample extremes: an elementary introduction,' Technical report, 1976.

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

Deng, Y., Bao, F., Kong, Y., Ren, Z., and Dai, Q., 'Deep direct reinforcement learning for financial signal representation and trading,' IEEE transactions on neural networks and learning systems, 2017, **28**(3), pp. 653–664.

Deng, Y., Kong, Y., Bao, F., and Dai, Q., 'Sparse coding-inspired optimal trading system for hft industry,' IEEE Transactions on Industrial Informatics, 2015, **11**(2), pp. 467–475.

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

Graves, A., Mohamed, A.-r., and Hinton, G., ‘Speech recognition with deep recurrent neural networks,’ in ‘2013 IEEE international conference on acoustics, speech and signal processing,’ IEEE, 2013 pp. 6645–6649.

Herbert, A. D. and Nagaraja, H., ‘Order statistics,’ Series in probability and statistics (3rd ed.). Hoboken: Wiley, 2003.

Kim, K.-j. and Han, I., ‘Genetic algorithms approach to feature discretization in artificial neural networks for the prediction of stock price index,’ Expert systems with Applications, 2000, **19**(2), pp. 125–132.

Kohara, K., Ishikawa, T., Fukuhara, Y., and Nakamura, Y., ‘Stock price prediction using prior knowledge and neural networks,’ Intelligent Systems in Accounting, Finance & Management, 1997, **6**(1), pp. 11–22.

Ku, C.-C. and Lee, K. Y., ‘Diagonal recurrent neural networks for dynamic systems control,’ IEEE transactions on neural networks, 1995, **6**(1), pp. 144–156.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D., ‘Continuous control with deep reinforcement learning,’ arXiv preprint arXiv:1509.02971, 2015.

Lo, A. W., Mamaysky, H., and Wang, J., ‘Foundations of technical analysis: Computational algorithms, statistical inference, and empirical implementation,’ The journal of finance, 2000, **55**(4), pp. 1705–1765.

Markowitz, H., ‘Portfolio selection,’ The journal of finance, 1952, **7**(1), pp. 77–91.

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

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

Moody, J., Saffell, M., Andrew, W. L., Abu-Mostafa, Y. S., LeBaraon, B., and Weigend, A. S., ‘Minimizing downside risk via stochastic dynamic programming,’ Computational Finance, 1999, pp. 403–415.

Moody, J. and Wu, L., ‘Optimization of trading systems and portfolios,’ in ‘Computational Intelligence for Financial Engineering (CIFEr), 1997., Proceedings of the IEEE/IAFE 1997,’ IEEE, 1997 pp. 300–307.

Moody, J., Wu, L., Liao, Y., and Saffell, M., ‘Performance functions and reinforcement learning for trading systems and portfolios,’ Journal of Forecasting, 1998, **17**(5-6), pp. 441–470.

Moody, J. E. and Saffell, M., ‘Reinforcement learning for trading,’ in ‘Advances in Neural Information Processing Systems,’ 1999 pp. 917–923.

Puterman, M. L., *Markov decision processes: discrete stochastic dynamic programming*, John Wiley & Sons, 2014.

Saad, E. W., Prokhorov, D. V., and Wunsch, D. C., ‘Comparative study of stock trend prediction using time delay, recurrent and probabilistic neural networks,’ IEEE Transactions on neural networks, 1998, **9**(6), pp. 1456–1470.

Sharpe, W. F., ‘The sharpe ratio,’ Journal of portfolio management, 1994, **21**(1), pp. 49–58.

Sullivan, R., Timmermann, A., and White, H., ‘Data-snooping, technical trading rule performance, and the bootstrap,’ The journal of Finance, 1999, **54**(5), pp. 1647–1691.

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y., ‘Policy gradient methods for reinforcement learning with function approximation,’ in ‘Advances in neural information processing systems,’ 2000 pp. 1057–1063.

Taleb, N. N., ‘Black swans and the domains of statistics,’ The American Statistician, 2007, **61**(3), pp. 198–200.

Taylor, M. P. and Allen, H., ‘The use of technical analysis in the foreign exchange market,’ Journal of international Money and Finance, 1992, **11**(3), pp. 304–314.

Tibshirani, R., ‘Regression shrinkage and selection via the lasso,’ Journal of the Royal Statistical Society: Series B (Methodological), 1996, **58**(1), pp. 267–288.

Werbos, P. J. *et al.*, ‘Backpropagation through time: what it does and how to do it,’ Proceedings of the IEEE, 1990, **78**(10), pp. 1550–1560.

White, H., ‘Economic prediction using neural networks: The case of ibm daily stock returns,’ 1988.

Williams, R. J. and Zipser, D., ‘A learning algorithm for continually running fully recurrent neural networks,’ Neural computation, 1989, **1**(2), pp. 270–280.

Zhang, W. and Dietterich, T. G., ‘A reinforcement learning approach to job-shop scheduling,’ in ‘IJCAI,’ volume 95, Citeseer, 1995 pp. 1114–1120.

VITA

Louis K. Steinmeister received his B.Sc. in Business Mathematics from the University of Hamburg in 2016. Having emphasized Finance, Stochastic Processes and Statistics, he proceeded with an internship to obtain business experience in Risk Management. Afterwards, he chose to enroll in the M.Sc. Business Mathematics program at Ulm University in 2017 while starting his own consulting business. Through the exchange program offered at Ulm University, he came to Missouri University of Science and Technology, where he was awarded his M.Sc. in Applied Mathematics with Statistics Emphasis in July 2019.