[Research Proposal]

FINANCIAL FORECASTING AND PORTFOLIO OPTIMISATION

School of Computer Science & Applied Mathematics
University of the Witwatersrand

Ntsika Mabe
1846377

Supervised by Prof. Terence van Zyl and Dr. Andrew Paskaramoorthy

June 28, 2022

A proposal submitted to the Faculty of Science, University of the Witwatersrand, Johannesburg, in partial fulfilment of the requirements for the degree of Bachelor of Science with Honours

Declaration

I, Ntsika Mabe, hereby declare the contents of this research proposal to be my own work. This proposal is submitted for the degree of Bachelor of Science with Honours in Computer Science at the University of the Witwatersrand. This work has not been submitted to any other university, or for any other degree.

Signed: [illegible]
Date: June 28, 2022

Abstract

It is no exaggeration to say that Modern Portfolio Theory is amongst the most influential inventions in the world of asset management. Together with capital asset pricing, they form the foundation of portfolio management. The goal of this theory is to construct an asset portfolios that maximise the potential return at any given level of risk. Using the investor's expected returns and the historical data of the financial assets, the theory borrows from statistical ideas of mean and variance to form a model in which the maximum potential return can be calculated at every level of risk. It is our goal to extend this framework by using machine learning techniques that can hopefully make better predictions and satisfy investors' needs.

Acknowledgements

I would like to thank my supervisor, Prof. van Zyl, for his guidance and advice.

Contents

Chapter 1

Introduction

Modern Portfolio Theory (MPT) is the bedrock of portfolio optimization and financial forecasting. It introduced a systematic, detailed approach to a field with high speculation and uncertainty. At its core, the theory outlines a method of assembling a portfolio of financial assets in such a way that they always yield the highest level of possible return, for any given level of risk. A portfolio constructed in this way is known as the optimal (or efficient) portfolio. Traditionally, the problem of finding the optimal portfolio has been modeled as a quadratic programming problem, that is easily solved using the mean-variance optimization (MVO) framework Fabozzi *et al.* [2012]. This technique was introduced by Markowitz in 1952 but has since become too crude to model modern-day financial portfolios, where investors have multiple constraints other than volatility. Fabozzi *et al.* [2012]

Since the development of this theory, the begging question is how does one go about constructing such a portfolio? Although the mean-variance framework works well in theory, it has proven difficult to execute in practical terms. This is due mainly to the complexities that exist in the financial markets, as the correlation between different assets is not always well defined. Another obstacle to the formation of the optimal portfolio is the consideration of the many different assets that exist. A glance at the NASDAQ shows that there are over 3000 stocks to choose from. To consider a combinatorial approach to finding the right stocks, in the right proportions, for over 3000 stocks would be computationally suicidal, and quickly intractable. There is a need, therefore, for a method in which one can firstly minimize the universe of asset choices, and secondly optimize a portfolio of these assets in a computationally reasonable manner.

A brief look into the current body of literature points us in the direction of some interesting papers using deep learning techniques to solve this problem. A paper by Zhang *et al.* [2020] suggests using deep learning as a way to optimize the portfolio Sharpe ratio. This is done by creating a model that begins by combining multiple input variables (features) from different assets, and then using an artificial neural network to output portfolio weights that maximize the Sharpe ratio from a single observation. The infinite space of asset choices is minimized by bounding the selection to Exchange-Traded Funds (ETFs). The deep learning approach negates the need for forecasting expected returns, and instead directly optimizes the portfolio weights by updating the parameters of the model. The result of this deep learning method is a model that outperforms many traditional, industry-favored algorithms such as the classical mean-variance optimization, maximum diversification, and stochastic portfolio theory model. In another paper by Sen *et al.* [2021] an LSTM model is built to forecast returns of the top five assets in each of the nine key sectors of the Indian stock market. The authors aim to build nine different portfolios - one for each key sector - and use an LSTM trained on historical data from the 1st of January 2010 to the 31st of December 2020 to optimize these portfolios. The authors of this paper find that the LSTM is well adapted to predicting stock market performances over short periods.

In this paper, we propose a deep learning algorithm that uses a state-of-the-art architecture to directly optimize a portfolio of preselected assets, by bypassing the forecasting step of expected returns, and instead optimizing the Sharpe ratio (the return per unit risk of a portfolio). Moreover, we advance the deep learning model by including gradient boosting, and we discuss whether transformers serve as a better architecture than current LSTM models. We believe that this approach will improve the mean-variance framework, and will advance subsequent works that aim to do the same thing.

To that effect, we set up the remainder of this research proposal as follows: Chapter 2 presents the background to the proposal. In it, we lay the foundations of what is to follow by providing definitions of important terminology used in the proposal. Chapter 3 will discuss the related works and the ideas behind the current body of literature. Chapter 4 will detail the methodology of the system we aim to build. Finally, Chapter 5 will give a breakdown of the timeline of the research and the deliverables at each stage in the next coming months.

Chapter 2

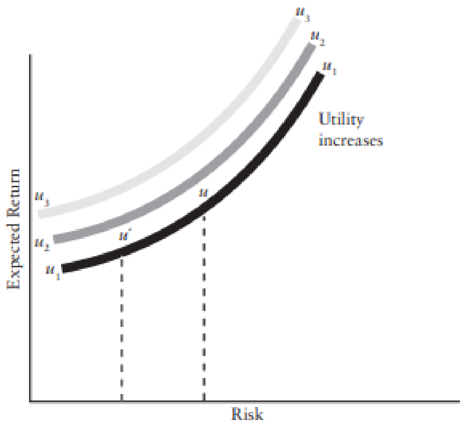
Background and Related Work

2.1 Definitions

Below are the definitions of terminology used in this paper.

Utility Function: A function that assigns a numeric value to all the possible choices that an entity faces.

Indifference Curve: a curve illustrating the different utilities at each level of risk. This can be seen in the figure below.



Efficient portfolio: A portfolio that provides the largest possible expected return for a given level of risk.

Return on a portfolio of assets: The weighted sum of all individual assets in the portfolio.

$$R_p = \sum_{i=1}^{n} w_i R_i. \tag{2.1}$$

Return of a portfolio: this is simply the weighted average of the expected return of each asset in the portfolio.

$$E(Rp) = \sum_{i=1}^{n} wiE(Ri) \tag{2.2}$$

Expected return of a risky asset: This is defined simply as the weighted average of the possible returns of an asset, where the weight is the probability associated with the possible return.

$$E(Ri) = \sum_{j=1}^{k} p_j R_j. \tag{2.3}$$

Variance of a risky asset: the measure of the dispersion of possible rate of return outcomes around the expected return.

$$Var(R_i) = \sum_{n=1}^{N} p_n [r_n - E(R_i)]^2 \tag{2.4}$$

Where R_i is the asset i,
r_j is the return of R_i at time j and
p_j is the probability of return r_j

Standard deviation: the positive square root of variance. This is done to 'take away' the squared units of variance, resulting in a measurement that is more linear.

$$SD(R_i) = \sqrt{Var(R_i)} \tag{2.5}$$

2.2 Related Work

2.2.1 Introduction

The previous section introduced the research area and the limitations present within it. In this section, we discuss in detail some of the subsequent works that aim to overcome these limitations, and how they lay the foundation for the research we are about to undertake. The rest of this chapter is thus laid out as follows. Section 2.2.2 formally introduces Modern Portfolio Theory. In this section we describe its foundations, and how it works in full detail. Section 2.2.3 then discusses the different machine learning approaches that have been used in portfolio theory and financial forecasting. The aim of this section is to get a brief overview on how machine learning as a whole has been used to make advancements in this field. Section 2.2.4 narrows our scope to deep learning as a paradigm for solving financial forecasting problems. Finally, Section 2.2.5 focuses on the use of recurrent neural networks as the architecture of choice for our task.

2.2.2 Modern Portfolio Theory

The theory of portfolio selection was formulated by Markowitz in 1952 and is now the bedrock of portfolio optimization and financial forecasting. It introduced a systematic, detailed approach to a field with high speculation and uncertainty. The theory is often referred to as mean-variance portfolio analysis or mean-variance analysis for short. The basic outline of this theory is to put together a portfolio of risky assets in such a way as to maximize the rate of return for a given level of risk. To discuss exactly how to go about this process, we first define some key concepts. Firstly, we look at how to measure a portfolio's return. Markowitz defines the return on a portfolio of assets as the weighted sum of all individual assets in the portfolio. The expected return of a portfolio is the weighted sum of the expected return of each asset in the portfolio. Naturally, the question that arises is what is the expected return of a risky asset? This is defined simply as the weighted average of the possible returns of an asset, where the weight is the probability associated with the possible return. These key definitions form the first part of what is known as the two-parameter framework for portfolio optimization. They focused on returns, but now we look at the second part of the framework: risk.

To quantify risk, Markowitz used well-defined statistical ideas of variances and covariance. He argued that the level of variance and covariance of a portfolio is indeed the risk of that portfolio and that the greater the variance or covariance, the greater the level of risk. This fully describes the two-parameter framework in terms of returns and associated risk of the portfolio. From this understanding of the rate of returns and risk, we can now look at key pillars of portfolio theory. The first is portfolio feasibility and efficiency. Any portfolio that an investor can design given the set of available assets is referred to as a feasible portfolio. The set containing all possible portfolios is known as a feasible set. A portfolio that has the highest expected return of all feasible portfolios at that level of risk is termed an efficient portfolio. An efficient set, or efficient frontier, is the set of all efficient portfolios (Fabozzi *et al.* [2012]). The goal of modern portfolio theory is to find this efficient frontier - to maximize the expected return of a portfolio at any given risk level. Once the efficient frontier has been marked, the question becomes which of the portfolios in the efficient frontier should an investor hold? The risk appetite of each investor determines the decision for which portfolio an investor should hold. The risk appetite of any investor can be expressed by a utility function, which is a measure of relative satisfaction that an investor derives from different portfolios (Stigler [1950]). Describing the utility function of investors is a problem that has been difficult to solve by economists, and it is reasonable to believe that investors will use subjective means to select a portfolio in the efficient frontier (Stigler [1950]). The problem of finding the efficient frontier, however, is a quadratic programming problem, and we can use steepest descent to find the global minimum of the objective function that maximizes the portfolio weights for the greatest return for a given risk level (qua [2006]). This section over-viewed the mean-variance framework for portfolio optimization. Subsequent sections in this chapter will show how we build open this to create models that perform better in financial forecasting and portfolio optimization.

2.2.3 Different Machine Learning Approaches

This section discusses the different machine learning approaches that exist within the literature concerning portfolio optimization and financial forecasting. Ma *et al.* [2021] use two machine learning models, random forest (RF) and support vector regression (SVR), as well as three deep learning models, LSTM neural network, deep multilayer perceptron (DMLP), and convolutional neural network, to predict return in portfolio creation. Their prediction results are used in the research to advance mean-variance (MV) and omega portfolio optimization models. The results of the experiments reveal that RF outperforms the other models. The results show that the RF+MVF (mean-variance framework) model still outperforms the other MVF models, but the RF+OF (omega framework) model is overtaken by SVR+OF due to its greater turnover rate. For daily trading investment, the authors advocate creating an MVF model with an RF return estimate.

In another paper by Obeidat *et al.* [2018], the authors present an LSTM approach to adaptive asset allocation, extending previous work on neural network training to model causality. The deep learning model discussed in this work consumes historical price data and consumes macroeconomic data and market indicators using PCA. The model then estimates the expected return, volatility, and correlation for the selected assets. The model outputs were then turned into action recommendations using an MV optimization framework augmented with a forward-looking rolling window technique. The results of the deep learning model were compared to multiple classical passive portfolio management approaches, on a dataset with a 7-year long duration. The authors find that the deep learning models outperform the classical passive portfolio management approaches by significantly greater annualized returns. In this section, we discussed the different machine learning approaches that exist within the literature.

2.2.4 Deep Learning

This section discusses deep learning as it relates to the subject of financial forecasting and portfolio optimization. Deep learning is a form of machine learning, in which a model is trained on labeled data to make predictions in the future on previously unseen data. The data that a deep learning model is trained on is often multi-dimensional and complex and is usually difficult or impossible to model using economic or statistical modeling techniques. The design of a deep learning architecture is what allows it to accurately model this complex data and it is this very architecture that we aim to describe in this section. Essentially a machine learning model is some input-output mapping, $Y = F(X)$, where X is some high dimensional data point $(X1, \ldots, Xd)$, that is passed through an activation function F, to produce some output Y.

What makes a machine learning model deep is passing this data point X, whose content describes the features of the data, through multiple layers of abstraction. At each layer, the features of the data point are assigned different weights and the weighted sum of these features is passed through the activation function that transforms the data point in some way, and then propagated to the next layer of the architecture. This continues until the data point reaches the final layer, known as the output layer, where the final activation function will produce an output Y, which is the prediction of the model based on the specific data point.

A paper authored by Zhang *et al.* [2020] suggests using deep learning as a way to optimize the portfolio Sharpe ratio. The deep learning approach negates the need for forecasting expected returns, and instead directly optimizes the portfolio weights by updating the parameters of the model. The result of this is a model that outperforms many traditional, industry-favored algorithms such as the classical mean variance optimization, maximum diversification, and stochastic portfolio theory model.

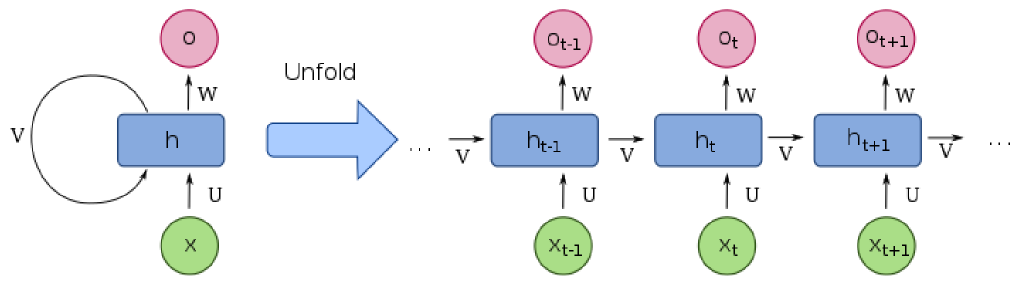
Wang *et al.* [2020] present a deep learning model to investigate an in-depth asset preselection process prior to optimal portfolio creation, ensuring high-quality inputs to the ideal portfolio. The systematic technique presented in this paper can assist in determining which assets should be included in the portfolio as well as the value composition of those assets. A paper by Cao *et al.* [2020] compares different deep learning models, including Residual Networks (ResNet), LSTM, Gated Recurrent Unit (GRU), SelfAttention (SA), Additive Attention (AA), and some combinations of these: AA + GRU, SA + GRU, AA + LSTM, and SA + LSTM. The experimental results show that the AA + GRU outperforms the rest of the methods on the Sharpe ratio and provides promising results for the portfolio optimization problem not only in Vietnam but also in other countries.

There is a space for deep learning models to make several improvements in the literature on financial forecasting. In the following section, we explore the use of the different deep learning approaches that can be applied to this.

2.2.5 Recurrent Neural Networks and Long Short-Term Memory Networks

In this section, we discuss the use of Recurrent Neural Networks, and more specifically LSTM models, as an architecture for solving the financial forecasting problem. Although the concepts of RNNs and LSTMs have already been introduced in this paper, in this section we give a full description of what these are and how they work. We then give the motivation of why we believe RNNs and LSTMs are perhaps the most promising tool in machine learning for financial forecasting and portfolio optimization. We will conclude our discussion with the gradient boosting algorithm, and how it might improve the performance of RNNs.

RNNs belong to the family of artificial neural networks, in which the connections between neurons are stringed together to form a network that can transmit a temporal sequence of information. Essentially, RNNs use sequential data as a means of learning patterns and trends, in a way that ordinary ANNs cannot. This is because RNNs have a special component known as the memory unit - a structure in each node of an RNN that allows it to keep track of current information - and use that information as the input into the next layer of neurons. This structure of the RNN is what allows it to learn data contextually because by "remembering" past information, it can more accurately predict what might come next. The structure of RNNs can be represented as folded or unfolded, as can be seen in the figure below.



Another defining characteristic of RNNs is that they share the same weight parameters within each layer of the network. This is different from traditional ANNs, in which each layer usually has different weight parameters. In terms of training, however, RNNs are akin to ANNs, using backpropagation and gradient descent to update their weights. RNNs specifically use the backpropagation through time (BPTT) algorithm, since the data being learned is sequential. The principles of BPTT are the same as the normal backpropagation algorithm, with the exception that BPTT sums up the error at each time step since the parameters are shared through each layer of the network. There are two common problems that RNNs tend to face due to the BPTT algorithm, these are the vanishing gradients and exploding gradients problem. The vanishing gradient problem occurs when the gradients become smaller and smaller after each update, such that when added to the weights, they do not make a distinguishable change. In essence, the gradients "vanish" as they tend to zero - affecting the RNNs ability to learn as the weights are no longer updated. The exploding gradients problem has the opposite effect - it occurs when the gradients become larger and larger, diverging to a point where there is an overflow error in the machine. A possible solution to both problems is to reduce the complexity of the RNN. By removing some layers in the neural network, it may be possible to stop gradients from becoming too large that they result in overflow errors, yet large enough that they do not tend to zero.

This effectively ends the discussion on traditional RNNs. We now focus our attention on LSTMs, which are a variant of the RNN architecture. LSTMs arise from the need for RNNs to model long-term dependencies more accurately. Consider the following example, in which we aim to predict the italicized term: Alice is lactose intolerant. When ordering dessert, she stays away from getting *ice cream*. From the context of the first sentence (Alice's lactose intolerance), we can anticipate that the second sentence will end with ice cream (or some other dairy product). This, however, would not be obvious for an RNN. It may even be impossible for the RNN to make such a prediction. This is because RNNs are unable to store information for multiple time steps, and are thus unable to fully join the dots from the historic information with what it currently sees. LSTMs overcome this shortcoming by having much more sophisticated neurons in their hidden layer, which allows them to keep track of past information for longer periods. This is possible because LSTMs use what are known as cells in their hidden layer. Each cell has three gates: an input gate, an output gate, and a forget gate. These gates control the passage of data through the hidden layers and advance the information that is deemed the most important for future predictions. Training LSTMs use BPTT in the same way as RNNs, however, they do not suffer from the vanishing and exploding gradient problems of traditional RNNs.

Now that we have laid the groundwork for RNNs and LSTMs, we consider their role in advancing financial forecasting and portfolio optimization. In a paper by Liu [2019], a deep LSTM model is used to forecast the volatility of the S&P 500 and AAPL indexes. The model is tested against SVMs (as the benchmark regression model in ML) and the popular Generalized Autoregressive Conditional Heteroskedasticity (GARCH) model for forecasting risk. The task of creating a model capable of predicting stock market volatility is a worthwhile one for LSTMs because volatility data is usually highly complex and nonlinear, with an extremely high degree of temporal variability. It is therefore no surprise that the LSTM outperforms the GARCH model, especially for a large time interval forecasting. In comparing the performance of the LSTM with the SVM however, the results are not so clear cut. Although the SVM tends to perform well in most cases, it often fails to make accurate predictions when trained on big data (¿ 10 TB). This is where the power of LSTMs shines through. When learning from big raw data, LSTMs can achieve very good predictions for long sequence data.

In another paper by Samarawickrama and Fernando [2017], the authors use RNN architecture to predict daily stock prices in the Sri Lankan Stock Market. The authors selected three companies from the Colombo Stock Exchange, namely, Commercial Bank Plc (COM), Royal Ceramics Limited (RCL), and Jon Keels Holdings (JKH). They then use the data following data: from the last two days of trading, they input the closing prices C(t-1) and C(t-2), the low prices L(t-1) and L(t-2), and the two high prices H(t-1) and H(t-2). The authors feed this data into three RNN architectures, namely Simple Recurrent Neural Network (SRNN), Gated Recurrent Unit (GRU), and an LSTM. These models were all benchmarked against a traditional MLP. After training these models on historical data, the authors compare the results of each architecture by considering the forecasting error. In this comparative analysis, SRNN and LSTM produced lower errors than the standard MLP. The authors also find that GRUs tend to produce higher errors in forecasting.

It is therefore apparent that RNNs are well adapted to the problem of financial forecasting and portfolio optimization. This is due to their ability to learn long-term temporal trends, in a way that is unlike traditional ANNs. Not only are RNNs able to learn long-term sequence data, but LSTMs are also good at handling very large raw data. The main advantage of this fact is that LSTMs can be trained faster, and more accurately, with less computational power than traditional ANNs, or other ML paradigms.

Chapter 3

Research Aims, Hypothesis and Methodology

3.1 Introduction

The previous chapter discussed different machine learning paradigms and their application to financial forecasting and portfolio optimization. At each stage, we have narrowed down on paradigms, and architectures that we believe hold the most promise for this task. After having narrowed down the scope from the entire universe of ML, to RNNs with gradient boosting, the main goal of the proposed research can now be stated. In this chapter, we shall formally state the objectives of this research. We shall also discuss the details of the implementation of this research. The remainder of this chapter is laid out as follows: Section 3.2 presents the research hypotheses and motivates the choice of these hypotheses. Section 3.3 then describes the steps to be taken in order to accept or reject these hypotheses, followed by a short summary of the salient points of this chapter (Section 3.4).

3.2 Research Hypothesis

As discussed in the previous chapter, ANNs have long been used in the formation of optimal portfolios for the task of financial forecasting. The purpose of this research is to determine if adding gradient boosting to an RNN architecture can improve the performance of the RNN, which is already considered amongst the most well-suited architectures in the class of ANNs for this task. The research hypothesis can thus be laid out as follows:

Hypothesis: Adding gradient boosting to an RNN architecture improves the performance of the RNN in the task of optimising a portfolio of assets.

3.3 Methodology

To test the above hypotheses, an experiment must be conducted. This section details the design of such an experiment.

3.3.1 Phase 1: Implementation

Implementation To implement this experiment, we must first source the required data. Since the universe of asset choices is practically infinite, we must first begin by narrowing our potential area of interest. For this problem, we will consider the BRICS market indexes. To further simplify our task, we will only look at the top-performing ETF from the most popular stock exchange in each of the five countries. Once these ETFs have been identified, we will consider the historical data of these ETFs dating back to 2012, to get 10 years' worth of data on which to train our model. Once this data has been sourced, it will be cleaned and split into training, validation, and testing data. Once the data has been split, it will be ready to be used to train three models: a traditional RNN, an LSTM, and an LSTM with gradient boosting applied to it.

3.3.2 Phase 2: Training

Training Once the first phase is complete, we will move on to training. The three models will be built using python libraries such as PyTorch or Keras. After creating these models, we will use the training data to create a baseline model for each. We will then optimize the models through the use of the validation data, as we will perform hyperparameter tuning on the model as a way to increase its performance. To gain computational strength, we will use the Wits Core Cluster to train the model. If there is a need, we will also take advantage of the computational power provided by Google Colab. The models will be trained until convergence, or for a certain number of iterations, whichever comes first. More clarity on this, however, will only be known once the research fully commences.

3.3.3 Phase 3: Testing

After training the model to a satisfactory level, we will implement the testing phase of our research. The testing phase will include running each model on the test data and recording the outcomes of each model. The typical evaluation metrics used in machine learning will be looked at as a performance measure, but we will also look at some metrics from the economics background. The primary metric we will consider is the Sharpe ratio, which will tell us which model results in a portfolio that has the highest level of expected return per unit risk.

Once testing is complete, we will be in a position to either accept or reject our hypothesis. To accept our hypothesis, we require the LSTM with gradient boosting to outperform the standard RNNs. This will be done if the LSTM with gradient boosting can optimize the portfolio's Sharpe ratio, and reach a Sharpe ratio that is greater than that of both the standard RNN and LSTM. If this is not the case, we will have to reject our hypothesis, and discuss why implementing gradient boosting does not necessarily result in greater performance in RNNs.

3.4 Conclusion

This chapter presented the methodology of the proposed research. The research hypotheses were formally stated, as were the steps necessary to accept or reject the hypotheses. The manner in which these steps lead to the acceptance or rejection of the hypotheses was also provided. The next chapter provides a more concrete plan for implementing the above experiment.

Chapter 4

Time Plan

4.1 Introduction

Chapter 3 introduced the research hypothesis and method by which we intend to test it. We now consider the practical execution of this research and present a time plan governing our intended progress. Lastly, we consider possible issues which may arise during the course of this project.

4.2 Deliverables

4.2.1 Phase 1: Implementation

At this stage of the research project, we aim to have cleaned data that has been organised in the form of training, validating, and testing. We also aim to have the traditional RNN, the LSTM, and the LSTM with gradient boosting models ready to be trained in the next phase.

4.2.2 Phase 2: Training

At this stage of the project, we aim to deliver three models that have been trained with the training data prepared in phase 1, and tuned with the validating data also prepared in phase 1. This will give us the models that we will use to test our hypothesis on in the next phase.

4.2.3 Phase 3: Testing

At this final stage of the project, we aim to have the final results of the tests. These results will form the basis of whether we accept or reject the stated hypothesis. At this stage, we will also be able to make claims on the success of the research; whether the aims have been met, and whether we are satisfied with the undertaking.

4.3 Time Plan

Week	Task	Allocated Hours
July 17-24	Collect, prepare and clean the data	3
July 24-31	Initialise the 3 models to be trained	5
August 1-7	Train and tune the simple RNN	10
August 7-14	Train and tune tune LSTM	10
August 14-21	Train and tune LSTM with gradient boosting	10
August 21-28	Test the the simple RNN	5
September 1-7	Test the LSTM	5
September 7-12	Test LSTM with gradient boositng	7
September 12-16	Study break	-
September 26-30	Report write up	12
October 3-7	Hand in 1st draft for feedback	1
October 10-14	Start preparing presentation while waiting for feedback	6
October 17-21	Make changes to report based on feedback, prepare presentation	6
October 24-28	Proof-read report and work on presentation	5
November 1-4	Work on presentation and report touch ups	5
November 7-12	Hand in report	1
November 14-18	Presentation	1

4.4 Potential Issues

A potential issue is the sourcing of the data. In some preliminary attempts, it has already proven to be a challenge to find recent stock price information of some indices in many of the Russian Stock Exchanges. This is of course due to the current war in Ukraine. It is unclear when some of this information will be made available, and as such we can only anticipate that it may never arrive in time for phase 1. This would be a shame, as we would like to compare the models performance in predicting the market behaviour during a time of crisis such as this.

Another potential issue is the rolling blackouts that are quite common in South Africa. Although most of the work will be done at The University of the Witwatersrand, which has access to generators when there is a blackout in the area, there can sometimes be technical issues caused by such blackouts. Computers in the labs having to reboot, or Wits lamp servers having to restart after every load-shedding event occurs, could potentially lead to the work being set back. This is again an issue outside of our control, however we hope it all set backs will be minor and not catastrophic in how they affect our plans.

A potential issue relating to the work, is that the training of the RNN could lead to vanishing or exploding gradient problem. We should anticipate the likelihood of such an event, and put measures in place to stop or limit its effects. To do this, we must understand the implementation of the BPTT algorithm that lies under the hood of the Keras or PyTorch libraries. It is very likely that these libraries make provisions for this, however the onus is on us to make sure that is indeed the case.

Chapter 5

Conclusion

In this paper, we introduced the concept of Modern Portfolio Theory and discussed how it is the bedrock of portfolio optimization and financial forecasting. In so doing, we uncovered that the field of portfolio optimization and financial forecasting has fascinated researchers for years. This is because although MPT offers a framework (the MVO framework specifically) to construct an optimal portfolio, it is often too simple to work in the real world. This has led to multiple iterations of the mean-variance framework, as many others try their hand at the traditional quadratic programming problem.

The early iterations tackled the problem using statistical models such as the ARIMA model, or stochastic models such as the GARCH model, as frameworks that work better in the real world. Although these models certainly do advance the traditional MVO framework, they too suffer from limitations made from the assumptions that make them simple enough to implement in the first place. This has led to the need for further advancements in the field. It has created the need for models that make few to no assumptions about the data, and few to no hypotheses about investor behaviour. Essentially, the field has needed an unbiased framework, and in some way naive to the problem at hand.

This need has necessarily led to the use of machine learning techniques in the attempt to solve this problem. Machine learning at its core is particularly well adapted to the problem of financial forecasting because it makes no assumptions about the data, nor about the investors who are the beneficiaries of such optimal portfolios. Machine learning techniques also have the added benefit of being good at learning large, complex data sets. They strive in identifying patterns in data that are nonlinear, in a way that would be impossible for humans, traditional statistical or economic models ever could.

It is thus no surprise that many different machine learning paradigms have been used to solve this problem. This paper takes a deep dive into the many different machine learning paradigms that have been used, from support vector machines to random forest trees, and artificial neural networks. We have discussed the application ofmany of these paradigms, and how they work in practice. We then identified recurrent neural networks, an architecture belonging to the family of ANNs, as perhaps the most promising technique for solving this probably so far.

This paper discusses the general structure of RNNs, and what exactly makes them promising for this problem, and how we believe they can advance the current body of literature. From there, we state our intended research aim: to advance RNNs in such a way that they can yield even better performance in the construction of the optimal portfolio by incorporating the gradient boosting algorithm.

We hope this research can yield promising results that can advance the literature incrementally forwards, in a similar fashion to every other advanced discussed prior to this.

References

[Cao *et al.* 2020] Hieu K. Cao, Han K. Cao, and Binh T. Nguyen. Delafo: An efficient portfolio optimization using deep neural networks. In Hady W. Lauw, Raymond Chi-Wing Wong, Alexandros Ntoulas, Ee-Peng Lim, See-Kiong Ng, and Sinno Jialin Pan, editors, *Advances in Knowledge Discovery and Data Mining*, pages 623–635, Cham, 2020. Springer International Publishing.

[Fabozzi *et al.* 2012] Frank J Fabozzi, Harry M Markowitz, Petter N Kolm, and Francis Gupta. Mean-variance model for portfolio selection. *Encyclopedia of Financial Models*, 2012.

[Liu 2019] Yang Liu. Novel volatility forecasting using deep learning–long short term memory recurrent neural networks. *Expert Systems with Applications*, 132:99–109, 2019.

[Ma *et al.* 2021] Yilin Ma, Ruizhu Han, and Weizhong Wang. Portfolio optimization with return prediction using deep learning and machine learning. *Expert Systems with Applications*, 165:113973, 2021.

[Obeidat *et al.* 2018] Samer Obeidat, Daniel Shapiro, Mathieu Lemay, Mary Kate MacPherson, and Miodrag Bolic. Adaptive portfolio asset allocation optimization with deep learning. *International Journal on Advances in Intelligent Systems*, 11(1):25–34, 2018.

[Samarawickrama and Fernando 2017] AJP Samarawickrama and TGI Fernando. A recurrent neural network approach in predicting daily stock prices an application to the sri lankan stock market. In *2017 IEEE International Conference on Industrial and Information Systems (ICIIS)*, pages 1–6. IEEE, 2017.

[Sen *et al.* 2021] Jaydip Sen, Abhishek Dutta, and Sidra Mehtab. Stock portfolio optimization using a deep learning lstm model. In *2021 IEEE Mysore Sub Section International Conference (MysuruCon)*, pages 263–271. IEEE, 2021.

[Stigler 1950] George J Stigler. The development of utility theory. i. *Journal of political economy*, 58(4):307–327, 1950.

[Wang *et al.* 2020] Wuyu Wang, Weizi Li, Ning Zhang, and Kecheng Liu. Portfolio formation with preselection using deep learning from long-term financial data. *Expert Systems with Applications*, 143:113042, 2020.

[Zhang *et al.* 2020] Zihao Zhang, Stefan Zohren, and Stephen Roberts. Deep learning for portfolio optimization. *The Journal of Financial Data Science*, 2(4):8–20, 2020.