DEGREE PROJECT IN MATHEMATICS,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2020

Portfolio Performance Optimization Using Multivariate Time Series Volatilities Processed With Deep Layering LSTM Neurons and Markowitz

ARON ANDERSSON

SHABNAM MIRKHANI

KTH ROYAL INSTITUTE OF TECHNOLOGY
SCHOOL OF ENGINEERING SCIENCES

Portfolio Performance Optimization Using Multivariate Time Series Volatilities Processed With Deep Layering LSTM Neurons and Markowitz

ARON ANDERSSON

SHABNAM MIRKHANI

Degree Projects in Financial Mathematics (30 ECTS credits)
Master’s Programme in Applied and Computational Mathematics
KTH Royal Institute of Technology year 2020
Supervisor at Erik Penser Bank: Jonas Thulin
Supervisor at KTH: Henrik Hult
Examiner at KTH: Henrik Hult

TRITA-SCI-GRU 2020:091
MAT-E 2020:053

Royal Institute of Technology
School of Engineering Sciences
KTH SCI
SE-100 44 Stockholm, Sweden
URL: www.kth.se/sci

Abstract

The stock market is a non-linear field, but many of the best-known portfolio optimization algorithms are based on linear models. In recent years, the rapid development of machine learning has produced flexible models capable of complex pattern recognition. In this paper, we propose two different methods of portfolio optimization; one based on the development of a multivariate time-dependent neural network, the long short-term memory (LSTM), capable of finding long short-term price trends. The other is the linear Markowitz model, where we add an exponential moving average to the input price data to capture underlying trends.

The input data to our neural network are daily prices, volumes and market indicators such as the volatility index (VIX). The output variables are the prices predicted for each asset the following day, which are then further processed to produce metrics such as expected returns, volatilities and prediction error to design a portfolio allocation that optimizes a custom utility function like the Sharpe Ratio.

The LSTM model produced a portfolio with a return and risk that was close to the actual market conditions for the date in question, but with a high error value, indicating that our LSTM model is insufficient as a sole forecasting tool. However, the ability to predict upward and downward trends was somewhat better than expected and therefore we conclude that multiple neural network can be used as indicators, each responsible for some specific aspect of what is to be analysed, and based on these draw a conclusion from the results. The findings also suggest that the input data should be more thoroughly considered, as the prediction accuracy is enhanced by the choice of variables and the external information used for training.

Portföljprestanda optimering genom multivariata tidsserie volatiliteter processade genom lager av LSTM neuroner och Markowitz

Sammanfattning

Aktiemarknaden är en icke-linjär marknad, men många av de mest kända portföljoptimerings algoritmerna är baserade på linjära modeller. Under de senaste åren har den snabba utvecklingen inom maskininlärning skapat flexibla modeller som kan extrahera information ur komplexa mönster. I det här examensarbetet föreslår vi två sätt att optimera en portfölj, ett där ett neuralt nätverk utvecklas med avseende på multivariata tidsserier och ett annat där vi använder den linjära Markowitz modellen, där vi även lägger ett exponentiellt rörligt medelvärde på prisdatan.

Ingångsdatan till vårt neurala nätverk är de dagliga slutpriserna, volymerna och marknadsindikatorer som t.ex. volatilitetsindexet VIX. Utgångsvariablerna kommer vara de predikterade priserna för nästa dag, som sedan bearbetas ytterligare för att producera mätvärden såsom förväntad avkastning, volatilitet och Sharpe ratio.

LSTM-modellen producerar en portfölj med avkastning och risk som ligger närmre de verkliga marknadsförhållandena, men däremot gav resultatet ett högt felvärde och det visar att vår LSTM-modell är otillräckligt för att använda som ensamt predikteringsverktyg. Med det sagt så fungerade den ändå relativt bra som trendindikator. Vår slutsats är därför att man bör använda flera neurala nätverk som indikatorer, där var och en är ansvarig för någon specifikt aspekt man vill analysera, och baserat på dessa dra en slutsats. Vårt resultat tyder också på att inmatningsdatan bör övervägas mera noggrant, eftersom predikteringsnoggrannheten förbättras av valet av vilka variabler och vad för ytterligare data man använder sig utav.

Acknowledgements

We would like to thank our supervisors Jonas Thulin at the *Erik Penser Bank* and Henrik Hult at *KTH* as well as Jonathan Werner and Elliot Gyllenswärd at the *Erik Penser Bank* for their valuable guidance and for helping us with the data extraction.

Contents

Chapter 1

Introduction

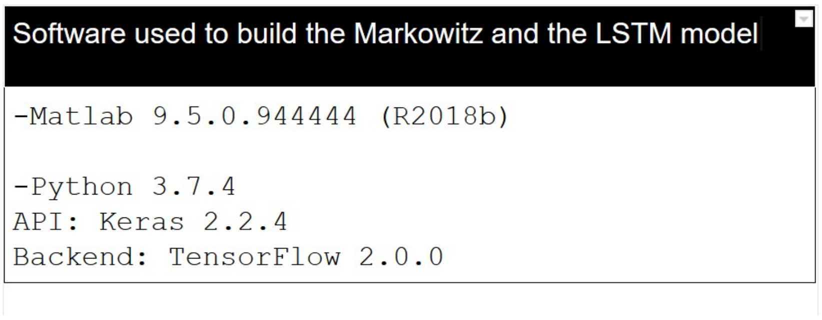
1.1 Background

Modern portfolio theory is a mathematical framework, where the aim is to create a balanced portfolio with the investment capital diversified across different assets in order to design an optimal portfolio with the highest expected return given a certain level of risk.[1]

A portfolio can consist of different asset classes such as bonds, stocks, cash, commodities and rates. It can also contain private investments such as real estate and art. Selecting different assets for a portfolio requires an investor to consider her/his individual risk tolerance and the time horizon of the allocations. One of the standard methods used for portfolio optimization is the Markowitz model, where the idea is to optimize the relationship between the expected return and the risk based on the historical returns of each asset.[2][3]

Forecasting is a complicated issue with many underlying factors, some of which will be detected, while others will not. With the boom in machine learning and its ability to discover these patterns, companies across a broad spectrum of markets have exhibited interest in what benefits they can acquire from machine learning. In this thesis we will use a Recurrent Neural Network (RNN) with Long Short Term Memory (LSTM) designed to remember what happened previously in order to make an accurate decision on the target of interest. Furthermore, the model does not impose any restriction on the input data, making it highly flexible.[4]

This thesis is conducted in collaboration with *Erik Penser Bank*, which is a Swedish investment and wealth consulting firm. The data that are being used are Exchange Traded Funds and indices from 2011-2020. Figure 1.1.1 shows the software that have been used throughout the process. The Markowitz model is applied in *Matlab* and the LSTM model has been conducted in *Keras* with *Tensorflow* as a backend.


Software used to build the Markowitz and the LSTM model

```
-Matlab 9.5.0.944444 (R2018b)

-Python 3.7.4
API: Keras 2.2.4
Backend: TensorFlow 2.0.0
```

Figure 1.1.1: Software used in this thesis

1.2 Research question

The research question is: Can we find patterns in the history of returns by applying a deep neural LSTM model, and will we be able to create a portfolio that outperforms the Markowitz portfolio?

1.3 Aim

The aim of this thesis is to compare two portfolio optimization frameworks. First, the Markowitz model will be applied to construct an optimal portfolio. Once this has been done, the next step is to investigate the possibility of designing a comparable portfolio using a deep LSTM model. The focus will be on constructing a short-term optimal portfolio, i.e., a “portfolio for tomorrow”.

1.4 Delimitation

The Markowitz framework itself has many delimitations that must be considered when comparing it to other frameworks, several of which are presented in Chapter

2. Another delimitation is that the assets are derived from different local markets but taking neither currency depreciation nor trading costs into account.[5] The performance of a neural network is strongly influenced by the amount and relevance of the data with which it is fed.[6]. The dataset we used is fairly small.

Additionally, many consider that the performance of a network in terms of predicting moves and prices is random and when such RNN models have been tested in reality, i.e., on the stock market, they have frequently failed. [7]

1.5 Outline

The first section of Chapter 2 will introduce the Markowitz model and related concepts, such as risk, limitations and types of investment problem. After that, several key concepts, e.g. exponential moving averages and efficient frontiers that contribute to the final portfolio, are presented. Chapter 2 concludes with a general presentation of neural networks and a detailed description of the Recurrent Neural Network (RNN) and the Long Short-Term Memory (LSTM). In Chapter 3, the data are presented in relation to different aspects, followed by the methodology and design of each model. In Chapter 4, the results of both the Markowitz model and the RNN model with LSTM are presented. Finally, in Chapter 5 the Conclusion and Implications for future research are outlined.

Chapter 2

Literature and theory

In this chapter the models and theories found in the literature will be presented. The chapter begins with a presentation of the Markowitz model and related factors, such as expected return, volatility, the efficient frontier, exponential moving average and the max Sharpe ratio. This is followed by information about the background of machine learning and finally our choice of neural network, which is the LSTM-layer recurrent neural network.

2.1 Markowitz model

The Markowitz model was developed by Harry M. Markowitz (1952) and is a portfolio optimization framework, in which the concept of risk measurement by means of average return and variance is used to select the most efficient portfolios from a set of assets. While models such as the Capital Asset Pricing Model (CAPM) developed by William F. Sharpe (1964) show the relationship between the systematic risk and expected return of portfolios by providing a measure of the risk premium, the Markowitz model only gives the relationship between return and the unsystematic risk of individual assets and their interrelationship.[8] Both models constitute the foundation of modern portfolio theory and are widely used for constructing diversified portfolios that maximize the expected return and minimize the risk.[3]

2.1.1 Risk and diversification

Risk is measured as the deviation from historical returns over a given period of time. The total risk of a portfolio can be divided into systematic and unsystematic risk. Systematic risk can have an impact on many assets across the market. Inflation, interest rate changes and recessions are some examples of systematic risk. Unsystematic risk is a micro-level risk that affects one asset or a small group of assets. Poor entrepreneurship is an example of this type of risk. Systematic and unsystematic risks cannot be completely eliminated, but the latter can be reduced by diversification.[9][3] The idea behind diversification is that if there are more than two uncorrelated assets in a portfolio, the total variance will be less than the variance of each individual asset.[10] See Figure 2.1.1 below. Diversification is a fundamental aspect of modern portfolio theory.

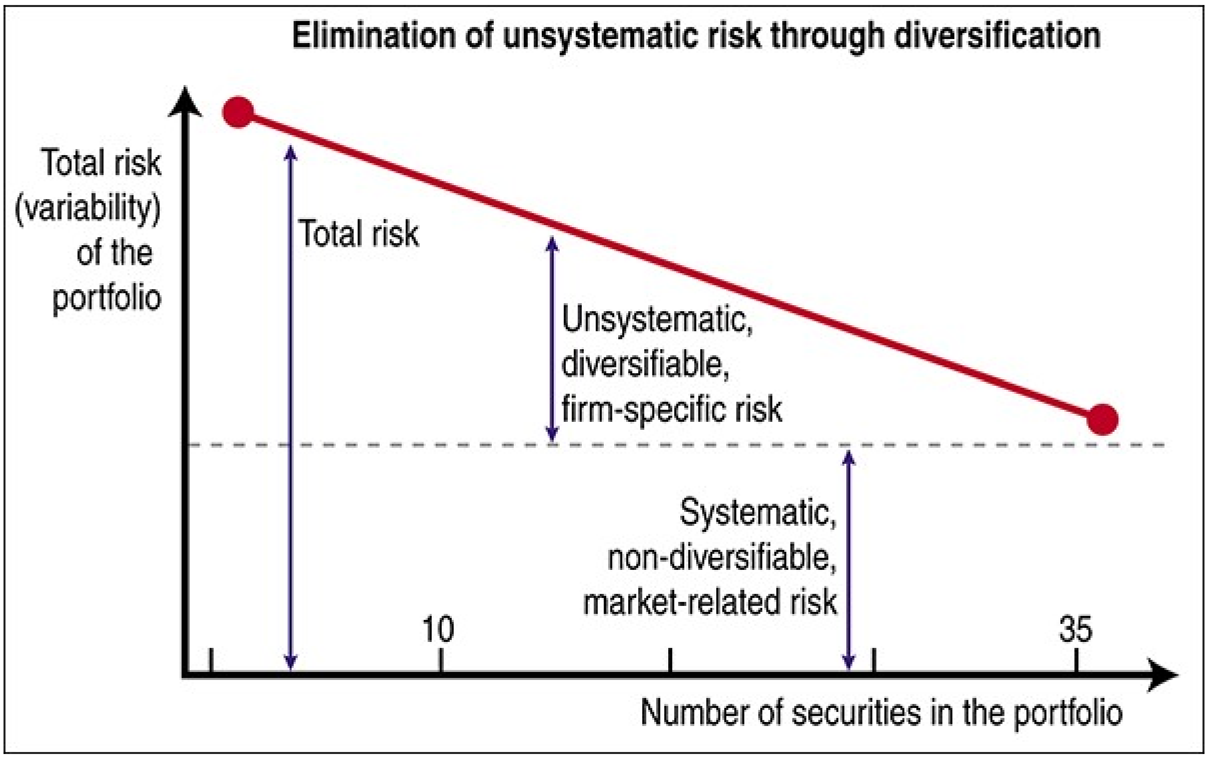

Elimination of unsystematic risk through diversification
Total risk (variability) of the portfolio
Total risk
Unsystematic, diversifiable, firm-specific risk
Systematic, non-diversifiable, market-related risk
10
35
Number of securities in the portfolio

Figure 2.1.1: The Markowitz model only gives the relationship between return and the unsystematic risk of individual assets and their interrelationship [10]

2.1.2 Expected return, variance, covariance matrix and correlation coefficient

The expected return of a portfolio is,

$$E[\mathbf{w^T R} + w_0 R_0] = \mathbf{w^T}\boldsymbol{\mu} + w_0 R_0 \tag{2.1}$$

where $\mathbf{w^T}$ is the transposed weight vector over all the individual assets in the portfolio, R is the return vector of the assets and w_0 and R_0 are the weight of and the return on the risk-free asset.[11]

The risk in a portfolio can be measured by calculating the sum of variances over the individual weighted asset returns. The measure shows how a set of data is spread and how it deviates from the average value. According to [11], the variance of a portfolio can be measured by multiplying the weight vectors by the covariance matrix as follows;

$$\boldsymbol{\sigma^2} = Var(\mathbf{w^T R}) = \mathbf{w^T \Sigma w} \tag{2.2}$$

To derive both (2.1) and (2.2) and properly evaluate a portfolio, one must first calculate the following factors. The sample covariances between the assets return X_j and X_k on N observations are correlated as follows;

$$cov(X_j, X_k) = \frac{1}{N-1}\sum_{i=1}^{N}(X_{ij} - \overline{X}_j)(X_{ik} - \overline{X_k}) \tag{2.3}$$

Where $\overline{X}_j$ and $\overline{X}_k$ are the mean of each asset;

$$\overline{X}_j = \frac{1}{N}\sum_{i=1}^{N} X_{ij} \tag{2.4}$$

The covariance matrix gives the covariances between returns on each pair of assets,

$$\Sigma = cov[X_j, X_k] = \frac{1}{N-1}\sum_{i=1}^{N}(X_{ij} - \overline{X}_j)(X_{ik} - \overline{X_k}) for j, k = 1, ..., n \tag{2.5}$$

where n is the number of assets.[12] The correlation coefficient determines the degree to which the returns on various assets are correlated and is calculated based on returns X_j and X_k on assets j and k,

$$\rho_{X_j,Y_k} = \frac{cov(X_j, X_k)}{\sigma_j \sigma_k} \tag{2.6}$$

where σ_j, σ_k are the standard deviation for assets j and k respectively.[13]
The correlation coefficient will be positive if the returns on the assets are linearly positively correlated, with the value 1 indicating that they are perfectly positively correlated, but negative if the returns on assets are linearly negatively correlated, with the value -1 indicating that they are perfectly negatively correlated. If there is no linear relationship, the correlation coefficient is zero, i.e. the asset returns are completely uncorrelated.[3][14][15] Correlation does not reveal whether x causes y or vice versa, or if the association is caused by another factor. To further analyse the causal relationship, heteroscedasticity can be applied, see section 2.4. Whether risk-averse or a risk-seeker, it is extremely important to analyse the correlation coefficient.

2.1.3 Investment problems

The allocation of assets to efficient portfolios is set differently for various types of investment problem, resulting in portfolios that are suitable for different types of investor. There are three common optimization problems:

- The trade-off problem
- The maximization-of-expectation problem
- The minimization-of-variance problem

These problems include investment with and without a risk-free asset, which will be presented in the following chapter. The expected value and variance of each of these problems can be determined by equations (2.1) and (2.2) above.

The weights in a portfolio can be defined as;

$$w_k = h_k S_0^k \tag{2.7}$$

where k is the asset and S_0^k is the stock price of the $k:th$ asset at $t_0 = 0$. In an investment with a risk-free asset, i.e. when $h_0 > 0$, the weight is;

$$w_0 = h_0 B_0 \tag{2.8}$$

where B_0 is the current price of a risk-free asset. $t_0 = 0$ indicates the current time and $t_1 = 1$ is the time at the end of the investment horizon. V_0 and V_1 denote the initial respectively the future portfolio value at the end of the investment horizon.[11]

$$w_0 + \mathbf{w^T 1} \leq V_0 \tag{2.9}$$

and

$$V_1 = w_0 R_0 + \mathbf{w^T R} \tag{2.10}$$

where $\mathbf{w^T}$ and $\mathbf{R}$ are vectors. If weights are negative, it means we are shorting that asset.

2.1.4 Investment without a risk-free asset

Trade-off problem without a risk-free asset

The trade-off problem can be described as seeking to optimize a portfolio in terms of maximizing the expected value $E[V_1]$ and minimizing the variance $Var(V_1)$, with a restriction on the initial invested value. The problem can be expressed as follows;

$$\text{maximize } \mathbf{w^T}\boldsymbol{\mu} - \frac{c}{2V_0}\mathbf{w^T \Sigma w} \tag{2.11}$$

$$\text{subject to } \mathbf{w^T 1} \leq V_0 \tag{2.12}$$

where $\boldsymbol{\mu}$ denotes the mean of the assets and c is the trade-off parameter. To select the trade-off parameter, one can evaluate different values of c for the solution $\mathbf{w}$ and then choose the c that is most suitable. The solution to (2.11) is given by;

$$\mathbf{w} = \frac{V_0}{c}\mathbf{\Sigma^{-1}}\left(\boldsymbol{\mu} - \frac{\left(\mathbf{1^T \Sigma^{-1}}\boldsymbol{\mu} - c\right)^+}{\mathbf{1^T \Sigma^{-1} 1}}\mathbf{1}\right) \tag{2.13}$$

where $\mathbf{\Sigma^{-1}}$ is the inverse of the covariance matrix of the assets and $x_+ = max(x, 0)$. A possible optimal portfolio that can be generated by the trade-off problem is one in which not all of the initial capital is invested, i.e. “throwing away money”. This can be dealt with when we introduce a risk-free asset.[11]

2.1.5 Investment with a risk-free asset

In theory, the risk-free asset is indeed a zero-risk investment, but in reality there is always a risk, and the closest to risk-free assets are government bonds. In the following problems, the risk-free asset is a zero-coupon bond with maturity at $t_1 = 1$ and face value 1. Its return is set to $R_0 = \frac{1}{B_0}$.

Trade of problem

The trade-off problem for an investment with a risk-free asset is defined as follows;

$$\text{maximize } w_0 R_0 + \mathbf{w}^\mathbf{T}\boldsymbol{\mu} - \frac{c}{2V_0}\mathbf{w}^\mathbf{T}\boldsymbol{\Sigma}\mathbf{w} \tag{2.14}$$

$$\text{subject to } w_0 + \mathbf{w}^\mathbf{T}\mathbf{1} \leq V_0 \tag{2.15}$$

Furthermore, the optimal solution to this is;

$$\mathbf{w} = \frac{V_0}{c}\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - R_0\mathbf{1}) \tag{2.16}$$

and the weight of the risk-free asset is denoted w_0, where

$$w_0 = V_0 - \mathbf{w}^\mathbf{T}\mathbf{1} \tag{2.17}$$

That is, the money in the investment problem solution (2.13) that was thrown away is now invested in the risk-free asset. In section 2.5, one can see the trade-off portfolio with a risk-free asset visualized as the max Sharpe ratio portfolio.

Maximization of expectation

The investment problem that seeks to maximize the expectation on a future portfolio can be constructed with regard to the constraint that the risk has a lower boundary and that the initial value invested must be less or equal to a set value. This is formulated by;

$$\text{maximize } w_0R_0 + \mathbf{w^T}\boldsymbol{\mu} \tag{2.18}$$

$$\text{subject to } \mathbf{w^T}\boldsymbol{\Sigma}\boldsymbol{w} \leq \sigma_0^2V_0^2 \tag{2.19}$$

$$w_0 + \mathbf{w^T}\mathbf{1} \leq V_0 \tag{2.20}$$

The optimal solution to this investment problem is given by;

$$\mathbf{w} = \sigma_0V_0\frac{\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - \boldsymbol{R}_0\mathbf{1})}{\sqrt{(\boldsymbol{\mu} - R_0\mathbf{1})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - R_0\mathbf{1})}} \tag{2.21}$$

Minimization of variance

The third type of investment problem is minimization of variance. Here, the aim is to construct a portfolio with a minimum risk related to the constraint of the expected portfolio value having to be greater than a certain value, and that the invested value into the portfolio must be less than a set value;

$$\text{minimize } \frac{1}{2}\mathbf{w^T}\boldsymbol{\Sigma}\mathbf{w} \tag{2.22}$$

$$\text{subject to } w_0R_0 + \mathbf{w^T}\boldsymbol{\mu} \geq \mu_0V_0 \tag{2.23}$$

$$w_0 + \mathbf{w^T}\mathbf{1} \leq V_0 \tag{2.24}$$

The optimal solution is obtained by;

$$\mathbf{w} = V_0(\mu_0 - R_0)\frac{\Sigma^{-1}(\boldsymbol{\mu} - R_0\mathbf{1})}{(\boldsymbol{\mu} - R_0\mathbf{1})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - R_0\mathbf{1})} \tag{2.25}$$

Based on these weights, the expected value of the portfolio and the variance can be determined by equations (2.1) and (2.2).[11]

2.2 Volatility

The standard deviation on the rate of return on an investment is a measure of the volatility of the investment. This means that volatility is a measure of the variability and degree of fluctuation of an investment over time. The standard way of measuring volatility is the application of historical or implied volatility, where the latter refers to the assessment of future volatility in the market. Volatility forecasting is one of the most important ways to evaluate the risk of a portfolio, which can be seen in the Markowitz portfolio framework. Furthermore, volatility has many characteristics that help to increase the accuracy of the forecasting in relation to forecasting returns.[16]

2.2.1 Historical volatility

Historical volatility measures the standard deviation of past stock price movements over a set period, which depends on whether one seeks short or long-term trends. The first step in calculating historical volatility is to decide the period for which the calculation is to be performed, i.e. n days. The next step is to calculate the day-to-day return for each of the n days,

$$R_n = ln\frac{C_n}{C_{n-1}} \tag{2.26}$$

where n is the number of days, C_n is the closing price for day n and C_{n-1} is the closing price for the previous day. The third step is to calculate the standard deviation, where we first calculate the average return,

$$R_{avg} = \frac{\sum_{i=1}^{n} R_i}{n} \tag{2.27}$$

where R_i is the day-to-day return. By adding the squared deviation from every trading day and dividing it by $n-1$, we obtain the variance of returns,

$$\sigma^2 = \frac{\sum_{i=1}^{n} \left(R_i - R_{avg}\right)^2}{n-1} \tag{2.28}$$

Lastly, the standard deviation is the square root of the variance,

$$\sigma = \sqrt{\frac{\sum_{i=1}^{n} \left(R_i - R_{avg}\right)^2}{n-1}} \tag{2.29}$$

where division by $n - 1$ is performed in order to obtain the sample standard deviation.

This represents the one-day historical volatility. The annualized historical volatility can be measured by taking one-day historical volatility multiplied by the square root of the number of trading days in a year.[17] Volatility only suggests the magnitude of the move of an investment, not the direction. Which leads us into to the next chapter.

2.3 Exponential moving average

By applying an exponential moving average (EMA), greater significance is placed on the most recent data points to yield an assumption about short or long term trends. Given sequence x_t beginning at time $t = 0$, the simplest form of an EMA is;

$$s_0 = x_0 \tag{2.30}$$

$$s_t = \alpha x_t + \left(1 - \alpha\right) s_{t-1}, t > 0 \tag{2.31}$$

where s_t is sequence of outputs, x_t is the most recent observation, α is the smoothing factor and $0 < \alpha > 1$. As an EMA places more weight on recent behaviour, it has a quicker response to the market when, for example, a downward price break-out occurs. A sign of this is when an EMA begins to flatten out before the break-out, indicating that it is time to go short.[18][19][20]

2.4 Heteroscedasticity

Heteroscedasticity occurs when;

$$var\left(y \mid x\right) \neq const. \tag{2.32}$$

In more detail, a predictive model expects the residuals;

$$r_i = y_i - \overline{y} \tag{2.33}$$

to show a randomness over the input data. That is, we want the output data to behave in a homoscedastic manner. It is important to analyse this behaviour, because in the presence of heteroscedasticity the standard errors in the output data are underrepresented in the true value, i.e. showing better accuracy of the model than is actually the case.[21][22] Figure 2.4.1 below illustrates a scatter plot showing heteroscedastic behaviour.

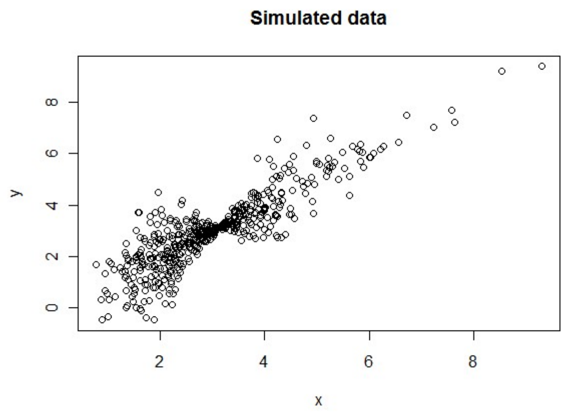


Figure 2.4.1: Here we see clear heteroscedasticity; first a big spread, then a small and then back to a big spread [23]

2.5 Efficient frontier

The efficient frontier is a significant part of modern portfolio theory and was constructed by Harry Markowitz. An efficient frontier can be plotted for given data with different returns, volatilities and correlations, where each point on the frontier is the most optimal portfolio in terms of return and risk. The blue line in Figure 1 is the efficient frontier, on which the optimal portfolios are set. The red dots are the assets that make up the portfolios.[24]

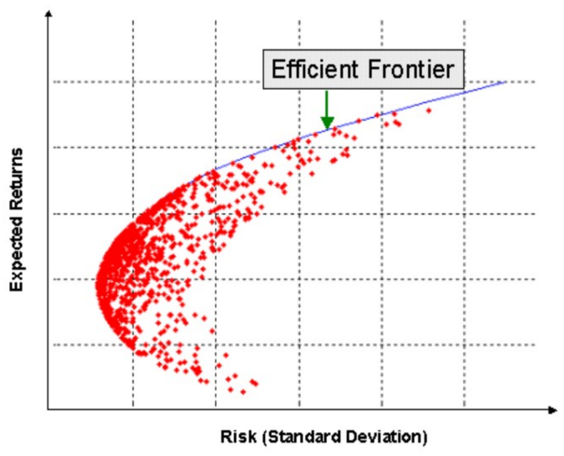


Figure 2.5.1: Efficient frontier of different portfolios. Red dots are assets or portfolios [24]

The max Sharpe ratio is the point on the frontier drawn by a tangent line from origo. A risk-averse investor is placed at the max Sharpe ratio, or below, if a risk-free asset is added. In Figure 2.5.2, the capital allocation line consists of all the optimal portfolios with an added risk-free asset, which is either used to create a leveraged or a lending portfolio, i.e., borrowing or lending at the risk-free rate.[25]

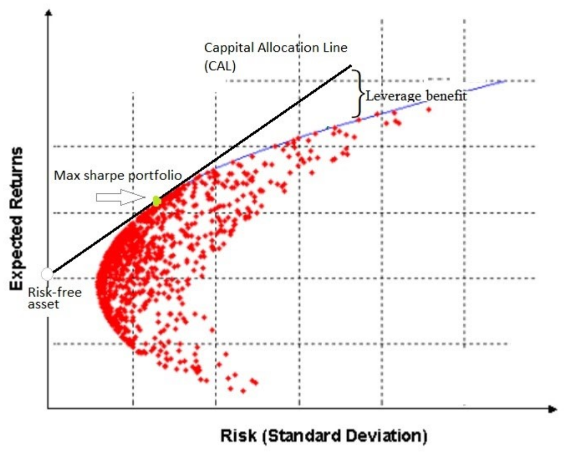


Figure 2.5.2: The new efficient frontier with an added risk- free asset is the black line. Below the max sharpe portfolio, on the black line, is lending portfolios. That is, lending risky assets to be able to buy more risk-free assets for the risk-free rate [24]

The curvature of the frontier is a good illustration of how the risk/reward profile changes with different sets of allocations in a portfolio. Highly risk-tolerant investors tend to invest in portfolios towards the right end of the efficient frontier. Portfolios on the Capital Allocation Line to the right of the max Sharpe portfolio are borrowing at the risk-free rate and portfolios to the right of it are lending at the risk-free rate. One of the assumptions about the efficient frontier is that the asset returns follow a normal distribution, which is rarely the case in the real stock market, where such returns more often follow a heavy-tailed or a leptokurtic distribution. Another assumption is that the efficient frontier is valid in a market with rational and risk-averse investors who do not influence market prices. But having said that, the efficient frontier is still a popular theoretical tool.

2.6 The Pre-processing of data

The preprocessing of data has become increasingly necessary in machine learning when handling large data sets. The quality of the data is fundamental for designing better models and predictions. The data transferred may be inaccurate due to having missing values, an incorrect format or duplication of some data. To ensure high quality it is therefore necessary to preprocess the data.[26]

2.6.1 Methods for pre-processing data

The following are some important preprocessing steps:

Data cleaning
In this step, the data are cleaned by removing or dealing with outliers and missing values, as well as correcting inconsistent and noisy data. When repairing missing values for large datasets, it is convenient to use regression models or a machine learning process. Predictions on missing data can be made by letting the model be set by integrity constraints and fed with intact data sequences.

Data reduction
In this step the volume of data is reduced by removing missing values. Data that have more missing values than a set threshold are removed. Features with a variance smaller than the threshold or a correlation coefficient higher than the threshold are also removed.

Data transformation
As a final step, if necessary, the data are transformed to a data modelling form by

normalization, where they are scaled between 0 and 1 or -1 and 1. Deep learning methods disregard units by setting all features to the same level of magnitude.[26] The following is called the Z-score method and is used to scale the data between 0 and 1,

$$Z = \frac{x - \mu}{\sigma} \tag{2.34}$$

where μ is the mean of the population and σ the standard deviation. These parameters are used as internal objects when scaling and, as feature and labelling data often have different ranges, it is important to set a Z-score for each.

2.7 Sharpe ratio

The Sharpe ratio was developed by William F. Sharpe (1966). It is used to find the risk-adjusted return relative to the risk-free rate, i.e., how much return an investment yields based on the risk involved compared to a risk-free asset.

The ratio can be calculated using the following formula;

$$\text{Sharpe ratio} = \frac{R_p - R_f}{\sigma_p} \tag{2.35}$$

where R_p is the expected return on the portfolio and is assumed to be normally distributed , R_f is the risk free return and σ_p is the standard deviation of the portfolio.[27]
On a scale of 1-3, a Sharpe ratio of 3 or higher is considered excellent and below 1 as suboptimal. If the Sharpe ratio is negative, it indicates that the risk-free return is higher than that of the portfolio, or that the portfolio has a negative return.[28] However, a negative Sharpe ratio can be brought closer to zero by either increasing returns (a good thing) or increasing volatility (a bad thing). Thus, for negative returns, the Sharpe ratio is not a particularly useful analytic tool. Furthermore, the Sharpe ratio treats all volatilities in the same way and therefore penalizes strategies that have upside volatility.[28] Measures such as the Sortino ratio can reduce the limitations of the Sharpe ratio to some extent by only penalizing those returns below a required rate of return, thus making it possible to calculate a portfolio's downside risk-adjusted return.[29]

2.8 Weaknesses

As the Markowitz model for portfolio optimization only considers means and variance, it disregards the fact that many portfolios have different probability distributions that are not adequately addressed by these measures.

If the density distribution is unimodal with a symmetric distribution around the mean, variance as a measure of risk remains relevant. However, a bimodal density distribution that is also symmetric around the mean will perform poorly when it comes to calculating the variance and using the mean as a measure of risk. There are parameters other than the mean and variance that are more suitable for describing the bimodal density distribution which, if ignored when applying the Markowitz model, would lead to a misleading result.[11] To further show how the density distributions can differ completely from each other, but still generate the same mean and variance, see Figure 2.8.1 below where;
$\mu = 1$ and $\sigma^2 = 1$ are the same in both distributions.

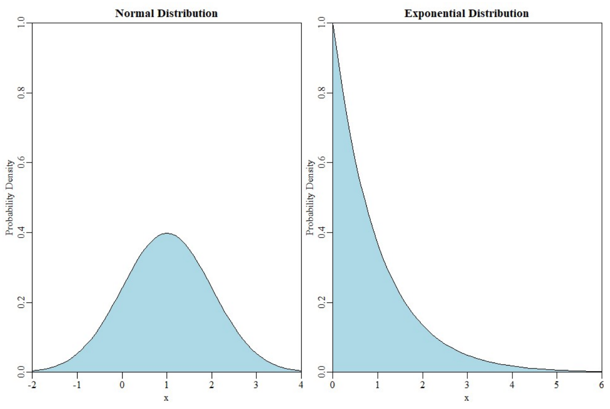


Figure 2.8.1: Both graphs have the same mean and variance [30]

Mean is not an absolute term, e.g. there is no point obtaining a 12% return when inflation is 15%, which gives a return of -3%. Furthermore, the variances do not take into account whether the current trend is upward or downward. Additionally, the covariances are difficult to obtain using empirical data, as one has to consider further analytical studies in order to determine causation.[11] Neural networks can deal with some of these problems.

2.9 Neural network

The neural network is a new field of research where traditional statistical methods are replaced by computational pattern recognition. A neural network is built by training the model by the use of previous data. Once the model is built, it should be able to respond properly to information not previously encountered. There are different types of network depending on the kind of information to be extracted.

In general, without going into detail about each individual neural network, there are three basic major types of learning strategy, comprising supervised learning, unsupervised learning and reinforced learning. In supervised learning the model processes a labelled training set and is adjusted until exhibiting adequate accuracy. In an unsupervised strategy there is no labelled data set and instead the model searches for previously undetected patterns. Reinforced learning is training a model to behave by performing actions and taking account of the result, where it is penalized for negative results and reinforced for positive results.[31][32]

In recent years neural networks have been widely used for forecasting financial information because of their ability to analyse non-linear data. One of the most famous networks in this regard is the Recurrent Neural Network (RNN), which is a model for sequential data that has the potential to learn long-term dependencies.[33]

2.9.1 Recurrent Neural Network (RNN)

Time series models are purely dependent on the idea that past behaviours can be used to predict future behaviour. In traditional neural networks, the input and output are independent of each other. However, for some problems it would be optimal if the model could remember what happened in the previous step. Thus, the RNN was developed.[34]

The architecture of RNN

The RNN converts the independent activations in each layer into dependent ones by providing the same weights and biases to all layers, thus making the hidden layers recurrent. This reduces the complexity of increasing parameters and helps in memorizing each previous output by using these parameters as input to the next hidden layers.[35] See figure (2.9.1) for an illustration. The following will explain the different layers of an RNN in greater detail. The input layer consists of N input neurons. The input to this layer is a sequence of vectors $\{.., \boldsymbol{x_{t-1}}, \boldsymbol{x_t}, \boldsymbol{x_{t+1}}, ...\}$, where $\boldsymbol{x_t} = (\boldsymbol{x_1}, \boldsymbol{x_2}, ...\boldsymbol{x_N})$. The neurons in the input layer are fully connected to the neurons in the next layer. The connections between these two layers are defined by a weight matrix W_{IH} with a size of IM, i.e., input neurons multiplied by the hidden neurons. The hidden layer acts as a "memory" and consists of M neurons for every time step, $\boldsymbol{h_t} = (\boldsymbol{h_1}, \boldsymbol{h_2}, ...\boldsymbol{h_M})$, where each neuron typically has a non-linear activation function $f_H(\cdot)$,

$$\boldsymbol{h_t} = \boldsymbol{f_H}(\boldsymbol{o_t}) \tag{2.36}$$

where

$$\boldsymbol{o_t} = \boldsymbol{W_{IH}}\boldsymbol{x_t} + \boldsymbol{W_{HH}}\boldsymbol{h_{t-1}} + \boldsymbol{b_h} \tag{2.37}$$

where $\boldsymbol{b_h}$ is the bias vector of the hidden neurons. Finally, there are the weighted connections from the hidden layer to the output layer W_{HO}. The output layer has P neurons, is defined as $\boldsymbol{y_t} = (\boldsymbol{y_1}, \boldsymbol{y_2}, ...\boldsymbol{y_P})$ and can be calculated as;

$$\boldsymbol{y_t} = \boldsymbol{f_O}(\boldsymbol{W_{HO}}\boldsymbol{h_t} + \boldsymbol{b_O}) \tag{2.38}$$

where $\boldsymbol{f_O}$ is the non-linear activation function of the output layer and $\boldsymbol{b_O}$ is the bias vector. The weight matrix elements will be adjusted by the training data in such a way that the network will also perform well on unseen data.[36] Figure 2.9.1 below shows the structure of an RNN in great detail.

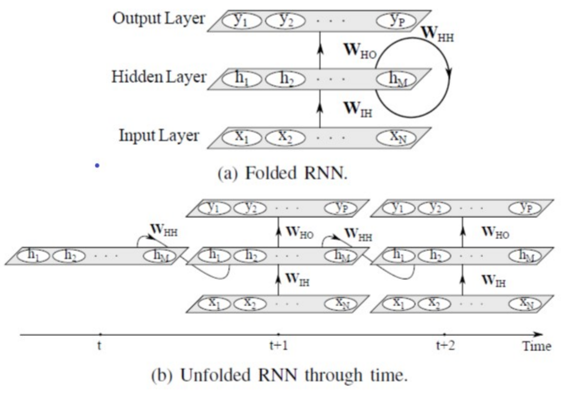


Figure 2.9.1: Structure of RNN over time. a) A folded RNN. b) An unfolded RNN can be seen as a deep neural network, except that the recurrent weights are tied [36]

Optimizer

During the training of the model, the weights are tweaked to minimize the cost function. This is where optimizers come in. Adam is a commonly used optimizer in recurrent networks. One of the benefits is that it optimizes the learning rate by applying a decaying learning rate, which means that the steps become smaller as the optimizer converges to the global minima, which speeds up the process as well as the accuracy.[37] [38]

Activation function and the vanishing gradient problem

We want to adjust the weights and biases connected to each neuron in every layer in such a way that small adjustments only cause small changes in the output. This is where activation functions come into play, as they squeeze high values into small ones. There are several different activation functions depending on the data and the task at hand. The tanh activation function squeezes values into an interval of [-1,1] and is defined as;

$$\boldsymbol{h_t = tanh(W_{HH}h_{t-1} + W_{IH}x_t)} \tag{2.39}$$

[35] The vanishing gradient problem may occur with the tanh function as it has a steep derivative, resulting in a decrease in prediction and efficiency. The problem appears when the model goes through back-propagation, which is when the gradient of the cost function is calculated by the chain rule with respect to the weights and biases of each layer. The problem is usually resolved when the hyperparameters are properly set by fine-tuning.[39][40]

Cost function and overfitting

The performance of an RNN centers around the minimization of the cost function, which is a measure of the level of error in estimating the relationship between input and output data. For a vanilla RNN, the loss function takes the following form;

$$\boldsymbol{L(x_1, ..., x_\tau, y_1, ..., y_\tau, W_{HH}, W_{IH}, W_{HO}}, \mathbf{b}, \mathbf{c}) = \sum_{t=1}^{\tau} l_t \tag{2.40}$$

where **b** and **c** are the bias vectors for the hidden layer and the output layer respectively. x_t and y_t are the training samples and the corresponding labels respectively. l_t is the loss function, where it is common to use the Root Mean Square Error (RMSE);

$$l_t = RMSE = \sqrt{\frac{\Sigma(p_i - y_i)^2}{n}} \tag{2.41}$$

where p_i is the predicted value for the ith observation and y_i is the observed value.[41][42][43] We also need to ensure that the cost function does not overfit the training data, which happens if the weighting is too heavy. This usually occurs if the model is trained with insufficient data. This can be prevented by adding a regularisation term L into the cost function, which acts as a weight decay.[38] For this thesis' purpose we treat loss and cost as synonyms.

2.9.2 Long-short-term memory (LSTM)

An RNN updates its weights for every batch of sequences it is fed, but as the batch size grows, the RNN becomes unable to learn to connect the information from the past to the current state.[35] Long Short-Term Memory (LSTM) is specifically designed to handle these long-term dependencies. The hidden activation function, , is still active, but the following are added to the hidden layer;

- Input gate
- Forget gate
- Output gate

These three gates have independent weights and biases that will be trained to decide how much of the current input to keep, how much information is no longer required and how much of the internal hidden state to send to the output.[44][45]

2.9.3 Stateful and stateless LSTM

All the RNN or LSTM models are stateful in theory. That is, the intention is that the network will remember the whole input, fed as a long sequence. However, as the sequence length of the data increases, the complexity of the network becomes greater. This is the reason for the introduction of batches and allowing the model to update its weights with back-propagation through all the mini sequences in each batch. Networks do not back-propagate through a set of batches.[46]

A stateful LSM learns from the batch that is fed to the network. Once the back-propagation is done and weights updated, the network is fed to the next batch. That is, the weights set in each layer from previous back-propagation are used as initial states for the next batch. [stateful]. However, a stateless LSTM works differently; it resets the weights to initial states for every batch.[46]

In a financial model such as price prediction, the time-series in different batches should be regarded as dependent on each other, which is why a stateful model is desirable.

Chapter 3

Data, methodology and portfolio design

In the first chapter we present the data received from the Erik Penser Bank, how we pre-processed our classic portfolio and our deep model. This is followed by the methodology chosen and how we achieved the optimal classic portfolio and deep portfolio. The two different models and the end result, namely the portfolios, will be denoted “the classic model” and “the deep model” respectively “the classic portfolio” and “the deep portfolio”.

3.1 Data

Table 3.1.1 and table 3.1.2 present our data. The first table consists of Exchange Traded Funds (ETFs) and indices. These are the assets that will be used to construct our optimal classic portfolio and the deep portfolio.

Class of asset	Swedish stocks	Global stocks	Hedge funds	Short rates	Long rates	Commodities
Type	ETF	ETF	ETF	Index	Index	ETF
Market	Sweden(SS)	USA (US)	Switzerland (SW)	Sweden	Sweden	USA (US)
Ticker	XACTOMX	VT	HFCHAS	RXVX	RXBO	GSG
Volume	✓	✓	✓			✓

Table 3.1.1: The assets. Some or all of these assets will constitute a part of the final portfolios. As can be seen, they are traded on different markets and only some have corresponding volumes, which will be used when constructing the deep portfolio.

The data are the daily closing prices in their given local currency. Indices are by nature a benchmark value and used as such. They will therefore not be traded on a market, but treated as an asset that can be owned in a portfolio.

Furthermore, we have market indicators, i.e. data that indicate the state of the market, see table below. These will be used in the deep model to increase the accuracy of the price prediction.

Class of indicator	Market volatility	Currency:	
Ticker	VIX	USDSEK	EURSEK

Table 3.1.2: These three market indicators are in the form of daily data. Professional market practitioners refer to the VIX as a fear index, as it shows the level of volatility in the market.

Time period	2011/05/18 - 2020/01/20
Number of trading days	2259

Table 3.1.3: We deliberately chose data from the period before the impact of the corona pandemic on the financial markets. This will be discussed in greater detail in section 3.1.2 Pre-processing of data for the Deep LSTM portfolio.

3.1.1 Graphs

Figure 3.1.1 below presents the daily price of each asset in its local currency. One can clearly see a drop in some of the assets because of the financial crash resulting from the corona pandemic. Only data from before the crash are taken into account in both models.

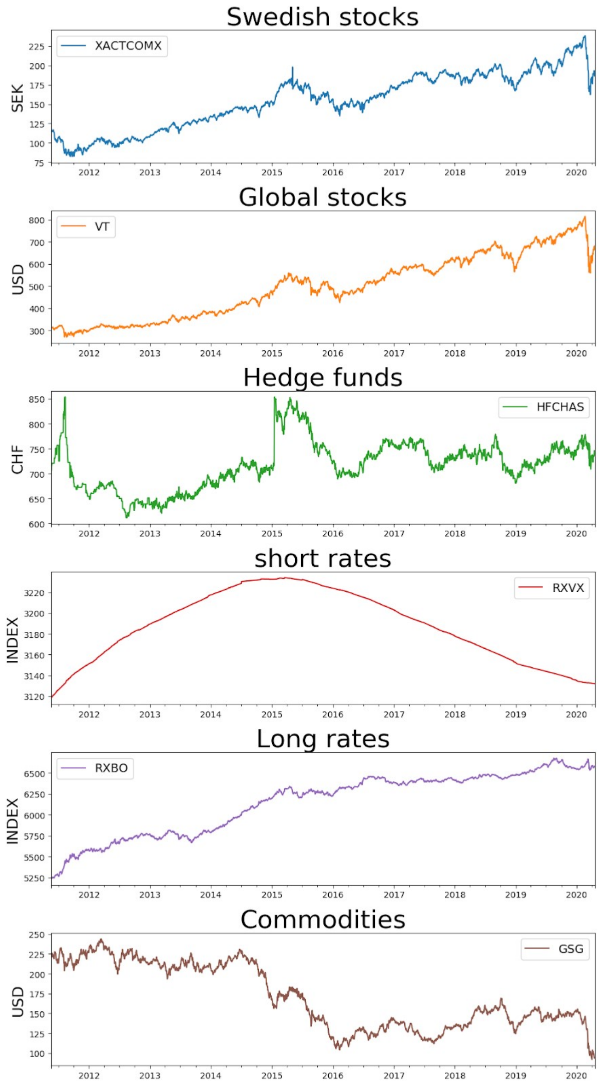


Figure 3.1.1: Subplot of the price development in local currencies from May 2011 to the end of March 2020.

When designing optimal portfolios, we did not consider currency depreciation or appreciation. This has an impact when designing a portfolio with assets from different markets. However, we will assume that all assets are from the same market and that their individual returns consist of comparable units. In this way, we can assign a benchmark value of 1 as a starting price for each asset and then let the assets develop to see how they perform in relation to each other. In Figure 3.1.2 below we present the comparable price developments for 2019. As we can see, the Swedish and global ETFs had the biggest price developments, which has a great impact on the final portfolios.

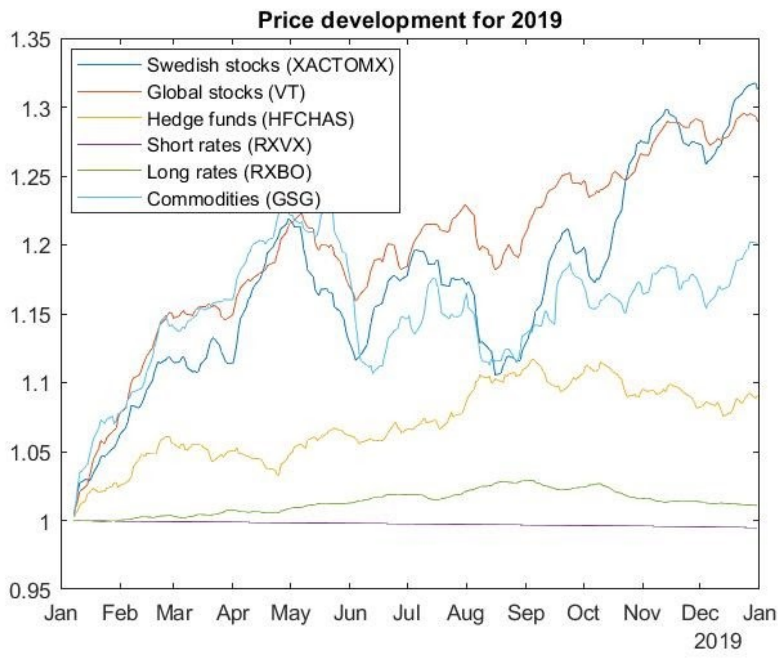


Figure 3.1.2: Development of prices for all assets in 2019. We will use this time period when designing our classic portfolio

We also looked at the volatility of each asset compared to the others, see Figure 3.1.3 below. To obtain a clear view, we kept the 1-year period that we presented in the previous figure.

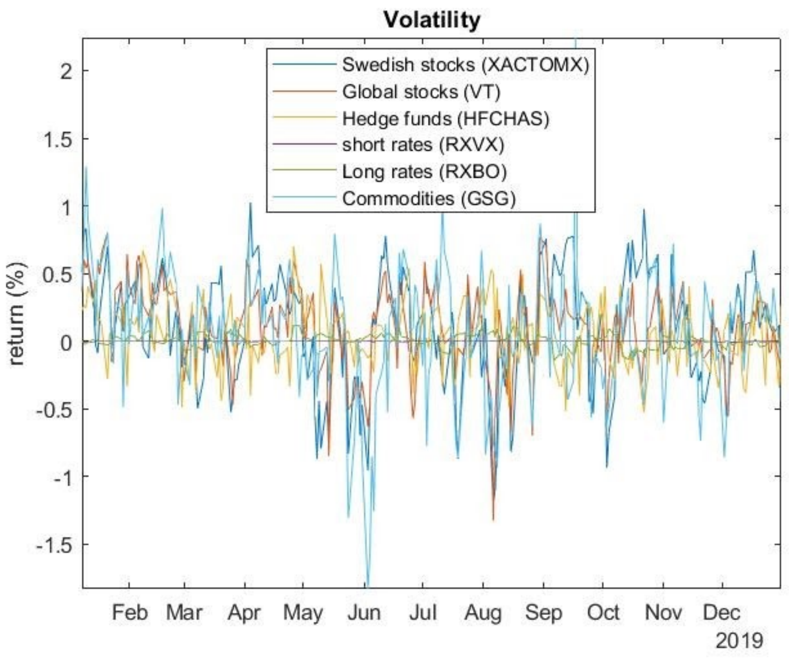


Figure 3.1.3: Daily volatility of each asset for all of 2019, plotted as linear graphs. It is clear that, relative to the other assets, the ETF with Swedish stocks and that with commodities were highly volatile throughout the whole of 2019, with commodities exhibiting a strong negative peak in June.

In Figure 3.1.4 we present the correlations between the assets. They are indicated by a heatmap from dark red to dark green, where red means a negative correlation coefficient and green a positive correlation coefficient. We have taken the whole period into account, i.e., from May 2011 to January 2020.

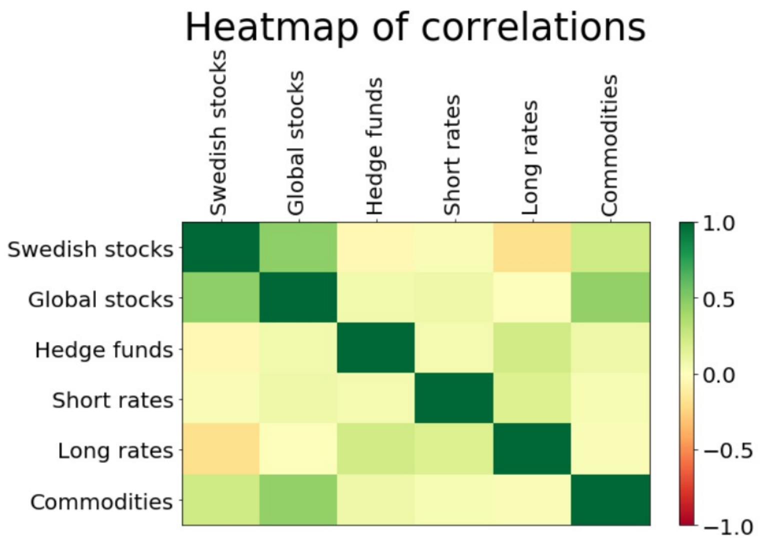


Figure 3.1.4: Correlations between assets presented as a heatmap.

As one might expect, there is a high positive correlation between Swedish stocks and global stocks. The rest of the assets are uncorrelated, which is positive for decreasing the risk in our final portfolio.

3.1.2 Pre-processing of data

The data were pre-analysed separately for the classic portfolio and the deep portfolio.

Pre-processing of data for the classic portfolio.

- **Time horizon**

  For the classic portfolio, we decided to reduce the data and only evaluate the most recent ones, i.e., data from 01/01/2019 to 31/12/2019. This is the period presented in figure 3.1.2. In the chosen time period, we applied an exponential moving average (EMA) with a window size of 5 to the price data of each asset. According to the following formula (3.1) [47], this will result in a weight of 33% for the most recent price in each window:

$$ExponentialPercentage = \frac{2}{WINDOWSIZE + 1} \tag{3.1}$$

  The choice of window size is taken from the paper *Forecasting with moving* *averages* by Robert Nau [18].

We then transformed the now exponentially weighted prices to returns, as required by the Markowitz model.

As our focus is on optimizing a portfolio in such a way that it yields the best return with the least short-term risk, we believe that reducing the data is acceptable. If we had been interested in creating a portfolio to be held for several years with no tampering, it would have made more sense to look even further back.

The data within this time frame had no missing values or NaNs. However, we noticed duplicates at the beginning of each year and decided to scan the data, which resulted in the identification of 3 duplicated rows. As one row was at the beginning and the two others at the end of the year, we decided to delete them to obtain a full sequence with no discontinuities.

- **Scaling**

  As mentioned in the previous chapter, the price data are in the local currency of the asset, which obliges an investor to consider currency appreciation and depreciation when trading on different markets. This is beyond the scope of the present thesis and we consider every asset as giving directly comparable returns.

Pre-processing of data for the Deep LSTM portfolio.

- **Time horizon**

  The data fed to our deep model stretched back to May 2011. Thus we fed our model with over eight years of data to enable it to make a proper prediction based on the test period at the end of 2019. It is reasonable to believe that the relationships between companies have changed within this time frame, but as our assets are in the form of ETFs and indices, we can assume that the correlations remain the same. Therefore, using the full time period is still relevant in that regard. Furthermore, we decided to exclude the beginning of the Corona pandemic from the test period because the model would clearly fail badly as it had no opportunity to train on something similar.

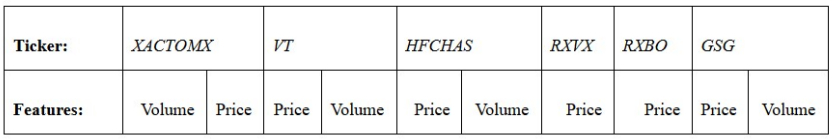
  Having said that, we had to ensure that the model was fed with a clean dataset. We still had a few duplicates, about 2-3 per year. However, we decided to keep them, as there were so few and deleting or keeping them would basically make no difference, because the model would discard them anyway in a “pattern recognition way”.

- **Scaling**

As mentioned in the literature, normalization of data is important for decreasing the training time and increasing the accuracy of the model. We applied two different z-score scalar objects, one for the features and the other for the targets of the training set and test set. In this way the model’s output values are not limited by the ranges of the input sets. We decided not to apply batch normalization between the layers, which will be discussed in greater detail in the next chapter where we outline the methodology.

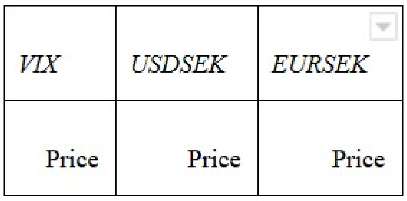
  - **Training, validation and test data.**

In the final step, we had to decide how much data should be allocated to the training, validation and test set. Our data comprised a total of 2,328 trading days and 13 categories of data, see below;



Ticker:	*XACTOMX*		*VT*		*HFCHAS*		*RXVX*	*RXBO*	*GSG*	
Features:	Volume	Price	Price	Volume	Price	Volume	Price	Price	Price	Volume

Table 3.1.4: Set 1



VIX	*USDSEK*	*EURSEK*
Price	Price	Price

Table 3.1.5: Set 2

Set 1 and set 2 are the data fed to the deep model to enable it to make price predictions on the assets in set 1. Values in set 2 are not actual prices but benchmark values. Because we are dealing with a fairly small data set, we decided to allocate 90% of the data to training and the remaining 10% to testing;

Training data	1992
Test data	226

Table 3.1.6: Training data and test data

During training we used the test data as the validation set to check our validation loss, as we stated section 2.9 that we wanted the loss to be as low as possible.

3.2 The methodology for and design of the classic portfolio

The final result of the Markowitz model is based on the mean of each asset. As stated in the previous chapter, we applied an EMA with a window size of 5 to the price data of each individual asset. The reason was that we wanted the final portfolio to have a high sensitivity to changes in the underlying, as we wanted to create a "portfolio for tomorrow". We were interested in capturing short term rather than long term trends. See the calculation of the EMA in the previous chapter.

In chapter 2, section 2.1.5 we introduced a "Trade-off" portfolio, i.e., a portfolio to maximize return and minimize risk. When we turned this into computations in Matlab, we used the max Sharpe ratio as the indicator of this type of portfolio, because this ratio is the point on the efficient frontier where we obtain the highest ratio in respect of return and risk. But before we could do these calculations, we had to choose which asset should be risk-free and which constraints our portfolio should have.

A portfolio can have so called risk-free assets. And we set the short rate, RXVX, as the risk-free asset, because this is the most stable asset, as can be seen by the volatility in Figure 3.1.3, where it is so small that it is barely noticeable. From Figure 3.1.1 it is clear that RXVX is, in fact, a money losing asset for the chosen period, which can be seen in greater detail in Figure 3.2.1 below;

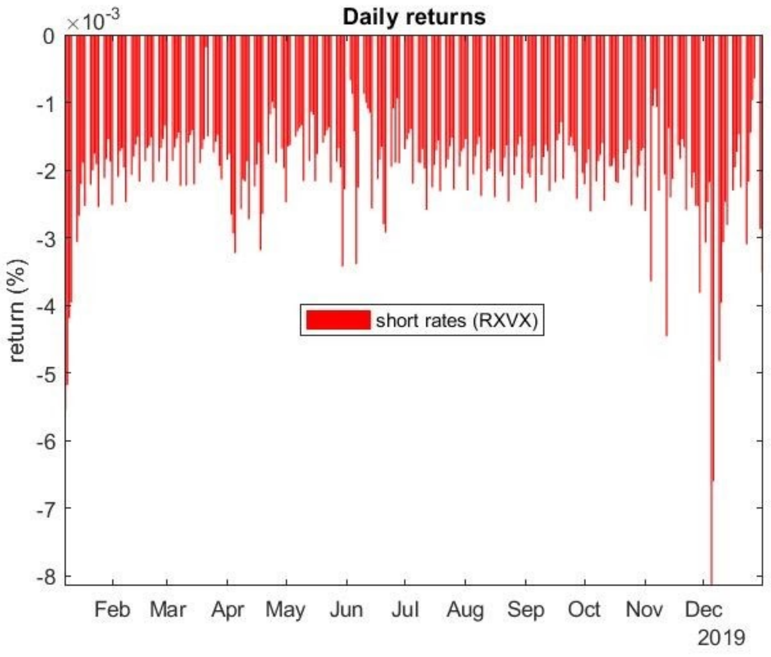


Figure 3.2.1: In reality, the theoretical risk-free asset has turned into a daily money-losing asset.

Our chosen risk-free asset will of course have a negative effect on our final portfolio, mostly in terms of the expected return.

Risk-free asset	RXVX
Class	Short rates

Table 3.2.1: The risk-free asset, chosen because it had the lowest historical volatility.

Our portfolio thereby consists of 5 risky assets and 1 risk-free asset. We set the following constraints on the assets;

- Shorting of assets is not allowed
- No more than 8% of the initial capital must be invested in the risk-free asset
- The risky assets are allocated in such a way that the max Sharpe ratio should be attained for the portfolio. This will result in our final classic portfolio
- No leverage is allowed.

A note on shorting of assets. If we plot two efficient frontiers, one when shorting is allowed and one when it is not, we obtain the following plot;

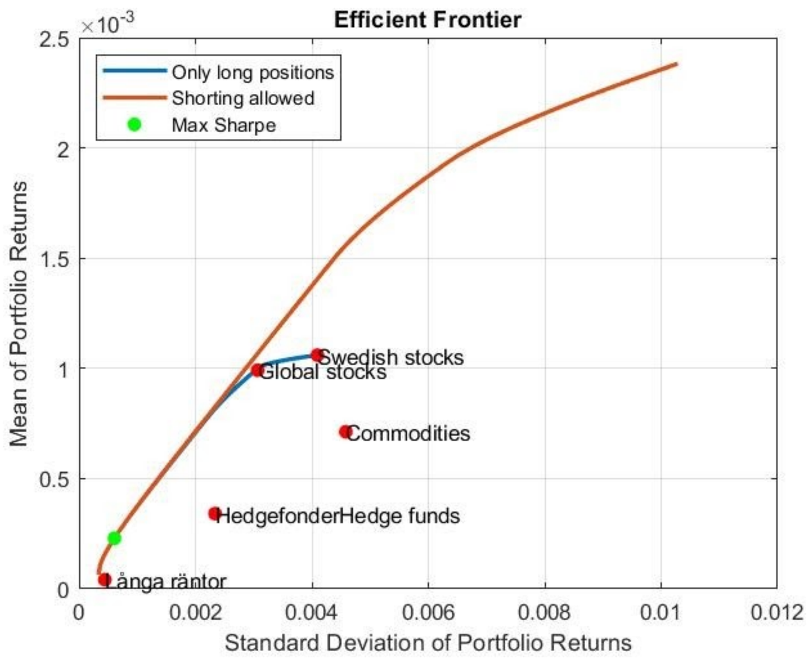


Figure 3.2.2: The red line is the efficient frontier when short selling is allowed in the portfolio. The blue line is when only long positions are allowed. The green dot is the max Sharpe portfolio, i.e., the portfolio that gives the most return in relation to risk.

When shorting is allowed it provides great leverage but also greater variance. The relationship is almost linear but then flattens out a little. The efficient frontier when shorting is allowed has no finite upper boundary. In theory, one can lose an infinite amount of money when short selling.

The rest of the procedure to create an optimal trade-off portfolio is coding portfolio objects, Sharpe ratios and efficient frontiers in Matlab. We refer to these codes for those who are interested in such details.

3.3 The methodology for and design of the deep model

To conclude this chapter, we will describe the RNN we built, how and why we built it as such.

3.3.1 Setting up the network

A neural network can be set up in several different ways and there is no definitive way of doing it because the data differ from case to case. Our approach was therefore to try different architectures and then retain the one that resulted in the lowest validation loss. We started off with a fairly complicated network architecture; three LSTM layers with 512, 256 and 128 neurons in each respective layer and between each such layer there was a dropout layer to deal with overfitting, ending with two dense layers, where a dense layer is a dot product between the input (to that layer), the weights and an added bias term. We simplified the architecture by decreasing the number of neurons in the LSTM layers and removing layers completely. By so doing and evaluating the performance of each network, we arrived at a much simpler network architecture;

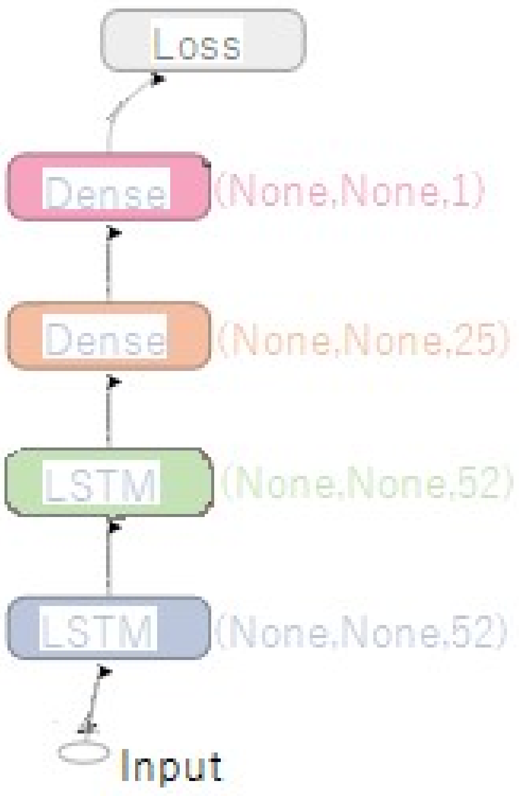


Figure 3.3.1: The architecture with the lowest validation loss. The values in parenthesis are the output shape where None denotes the arbitrary size.

Because of the rather small dataset, of which 90% was allocated for training, we decided to have just two sets, one for training and one for testing, where the test set was used for validation throughout the training process, as stated in the previous chapter. This meant that the network obtained as much training data as possible.

When calculating loss, the stepsize or learning rate is important, as it is that who steps us down the loss function. A learning rate that is too small may prevent the neural network from learning, while a stepsize rate that is too large might overshoot low loss areas. We addressed this problem by setting up a function that will reduce the learning rate when validation loss ceases to improve.

Reduce learning rate when it reaches a plateau	**Monitor**	**Factor**	**Minimum learning rate**	**Patience**
	Validation loss	0.61	1e-4	7

Table 3.3.1: The parameters of the function that will prevent an overshoot of the valleys of low loss.

Monitor is the parameter we want to observe, **factor** is how much we want to change the learning rate each time, **minimum learning rate** is the minimum boundary and **patience** is how many epochs we will wait before reducing the learning rate by the factor 0.61. The approach to selecting an optimal factor was the same as that employed above when choosing the optimal architecture. We tried factors as low as 0.1 and as high as 0.9., meeting in between at around 0.6 produced the lowest loss.

We ran the model for 50 epochs with a batch size of 5. As the intention is to enable our network to produce a prediction of tomorrow's price, the sequences in every batch had a length of 40 days, i.e. 2 months of previous trading days. This means that every batch had 40*5 days in a sequential row, i.e., no shuffling within the batch. However, we shuffled the batches for every epoch to increase the model's learning. As the best model with the lowest loss does not necessarily have to be for 50 epochs, we added a function to evaluate the model at each epoch and only retained the model with the lowest loss. To save time we also terminated the running of the program if the validation loss had not improved after 14 epochs. The reason for 14 epochs before termination was that we let the learning rate run for seven epochs and, if the validation loss had not decreased after the 7 runs, it meant that it had 7 more runs before the program was terminated.

Batch-size	*Sequence of days*	*Prediction*
5	40	one day forward

Table 3.3.2: Our RNN model was fed with batches of 5 sequences, each sequence being 40 days. This trains the model to be able to make a proper prediction of tomorrow's price.

3.3.2 Running the deep model

The data fed to our final deep model are presented in Chapter 3.1 and section 3.1.2. *Preprocessing of the Deep LSTM portfolio*. There are 13 columns of parameters that served as inputs. The model was trained with each of the risky assets, one at a time. When the model is run to produce price predictions on the XACTOMX asset comprising Swedish stocks, using the set up presented in the previous chapter, we obtain the following validation loss graph;

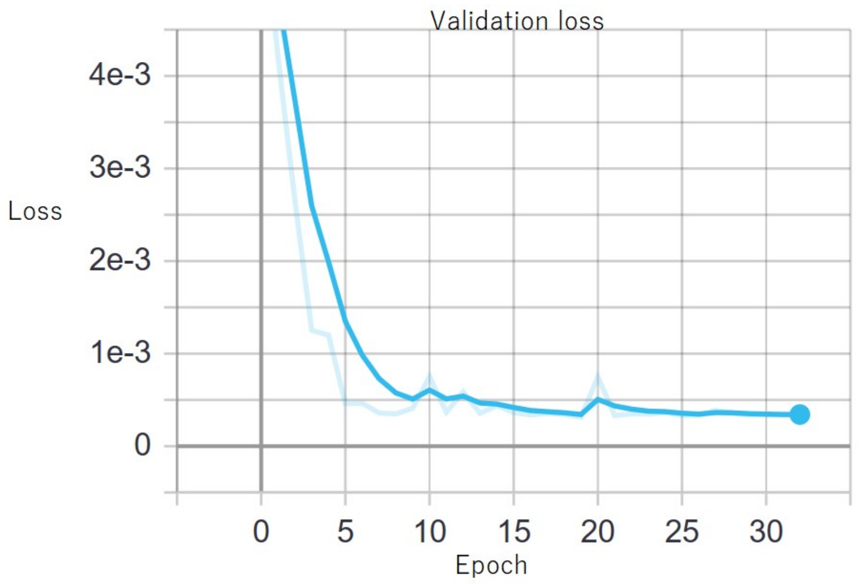


Figure 3.3.2: The development of validation loss of XACTOMX, when a decreasing learning rate is activated, as presented in section 3.3.1 Set up of the network. At around 33 epochs, the running of the model terminated because of the lack of updates on the validation loss parameter.

The pale blue line represents the actual values, while the dark blue line is a visual smoothing of these values. The lowest loss with a value of 0.00032 was achieved at the end of the 20th. epoch. From that point the model was unable to produce a lower validation loss.

The following graph presents the price predictions made by the deep model after being fed with the test data, which comprise 13 columns of the last 226 rows of the total data. It should be noted that the model is now trained but has not seen these data before, because when the test data were used for evaluating the validation loss during training, we did not let the model train on them. In order not to lose visual details of the predictions, only the first 60 values are presented.

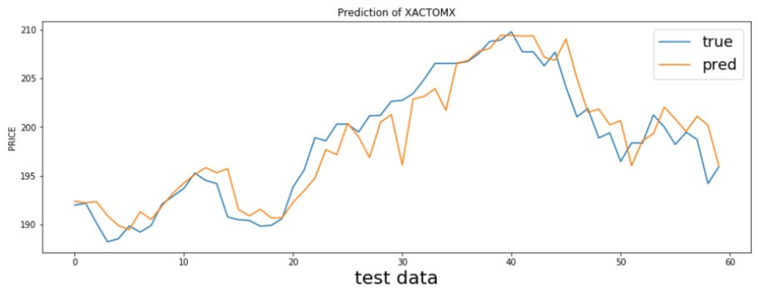


Figure 3.3.3: Our final deep model was trained to make price predictions one day ahead and produced the above predictions on data it had not been fed before, but of the same kind as it was trained on. The predictions were made on the XACTOMX asset, which consists of Swedish stocks.

We can see that the predictions have a slight delay relative to the true values. Furthermore, many of the peaks and valleys are exaggerated, over- or undershooting prices with a high frequency. But having said that, it nevertheless captures the overall pattern well when the prices go up or down but with a slight delay.

Let us investigate whether the predictions show any sign of heteroscedasticity by analysing the following figure;

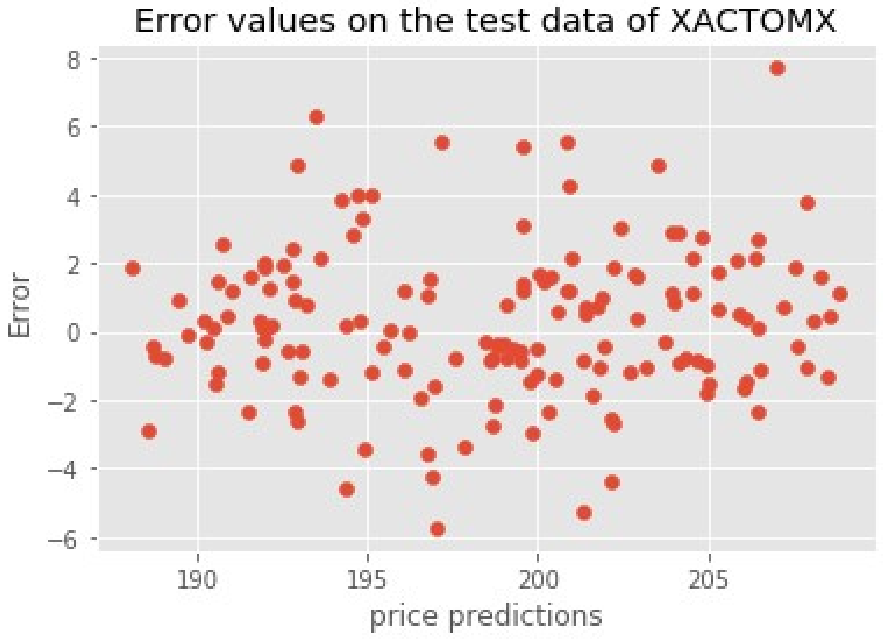


Figure 3.3.4: The y-axis is the error, i.e., the predicted value minus the true value for each price prediction. The plot shows homoscedasticity.

The error values have a randomness that definitely does not indicate heteroscedastic behaviour, which is good because it means that the errors do not underrepresent the true values. The error values are also plotted in a distribution graph, see Figure 3.3.5.

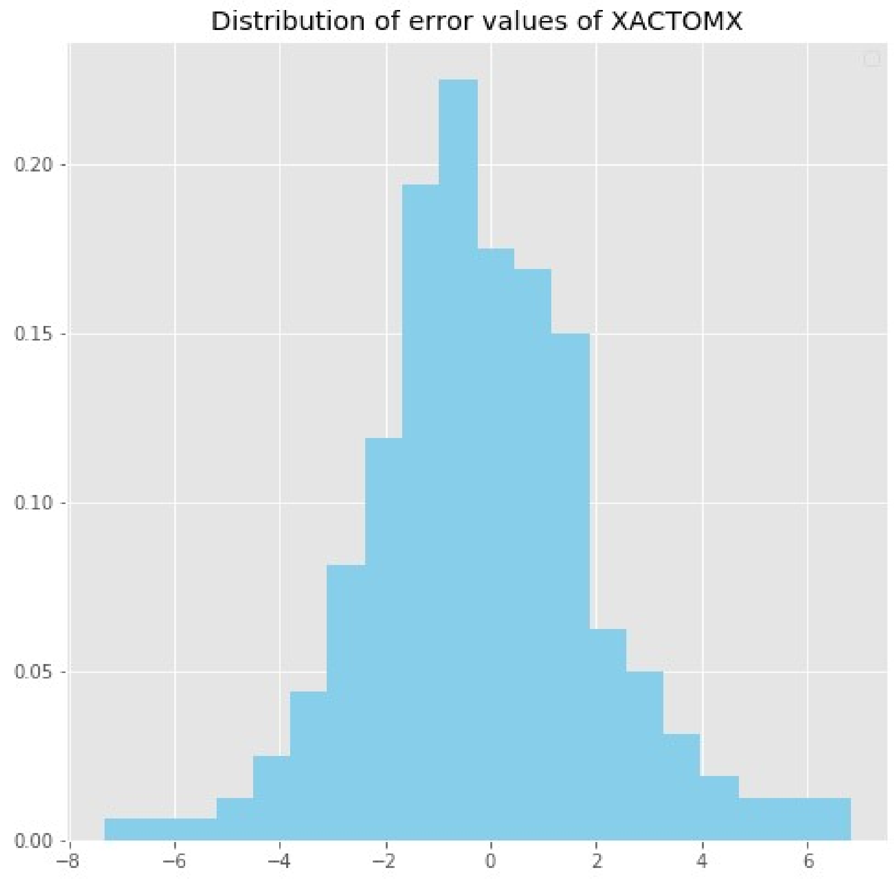


Figure 3.3.5: The prediction error range is [-8.7], but most of the errors centre around [-2.2].

The error values show a normal distribution that is slightly skewed to the left.

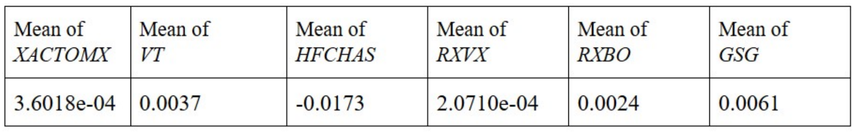
3.3.3 Mean and covariances of the deep model

In the previous chapter we created a model that gives tomorrow's price prediction for each asset presented in section 3.1 Data. Our aim is to design a portfolio based on these predictions. The classic portfolio, which is presented in its entirety in section 4.1 Results of the classic portfolio, is built on input data from 01/01/2019 to 31/12/2019. Therefore, the classic model creates an optimal portfolio that should be held on 1st January 2020. Thus, by taking the deep model's predicted price on 1st January, 2020, we can calculate tomorrow's return on an asset using the following formula;

$$Return^k_{i+1} = \frac{PP^k_{i+1} - P^k_i}{P^k_i} \tag{3.2}$$

where PP^k_{i+1} is tomorrow's predicted price of asset k, P^k_i is the previous day's true price value of asset k.

If we apply the above formula for each asset price prediction made with our deep model, we obtain the following list, where tomorrow's return is named "mean" as in the Markowitz framework. Please note that the predicted price is for 1st January, 2020.



Mean of *XACTOMX*	Mean of *VT*	Mean of *HFCHAS*	Mean of *RXVX*	Mean of *RXBO*	Mean of *GSG*
3.6018e-04	0.0037	-0.0173	2.0710e-04	0.0024	0.0061

Table 3.3.3: Mean of every asset calculated from predicted price values by our deep model.

The covariances are easily computed by the following formula;

$$cov(A,B) = \sum_{i=1}^{N} \frac{(r_i^A - r^{\overline{A}})(r_i^B - r^{\overline{B}})}{N-1} \tag{3.3}$$

where A and B are the assets, N is the length of test data up to 31/12/2019, r_i^A and r_i^B are the daily returns from this time period and $r^{\overline{A}}$ and $r^{\overline{B}}$ are the mean calculated by formula 3.2 and presented in table 3.3.3.

With the new values from our deep model we build our deep portfolio by applying the framework presented in section 3.2 *The methodology for and design of the classic portfolio*. The same conditions were set for the deep portfolio as for the classic portfolio and the results are presented in the following chapter.

Chapter 4

Results

We here present the final results of both the classic and the deep portfolio.

4.1 The results of the classic portfolio

After letting our coded program do its work, based on the frame set in 3.2 *The methodology for and design of the classic portfolio*, we arrived at a trade-off portfolio with the following allocation/weights of the assets;

Class	Swedish stocks	Global stocks	Hedge funds	Long rates	Commodities
Asset	XACTOMX	VT	HFCHAS	RXBO	GSG
Weight	0.088606	0.076355	0.054196	0.74272	0

Table 4.1.1: Allocation of risky assets

Class	Risk-free asset (short rates)
Asset	RXVX
Weight	0.0381

Table 4.1.2: Risk free asset

The risk-free asset has an allocation of 0.0381 because the total weight of all assets must equal to 1.

With these weights, we end up with a trade-off portfolio for tomorrow with the following mean and variance ;

Optimal portfolio	Mean (%)	Variance
Classic portfolio	0.021804	3.38e-05

Table 4.1.3: Optimal trade-off portfolio

where we calculated the variance by the square of the standard deviation. In Figure 4.1.1 We present our classic portfolio on the efficient frontier, i.e., the portfolio where the max Sharpe ratio is the highest, indicated by the green dot.

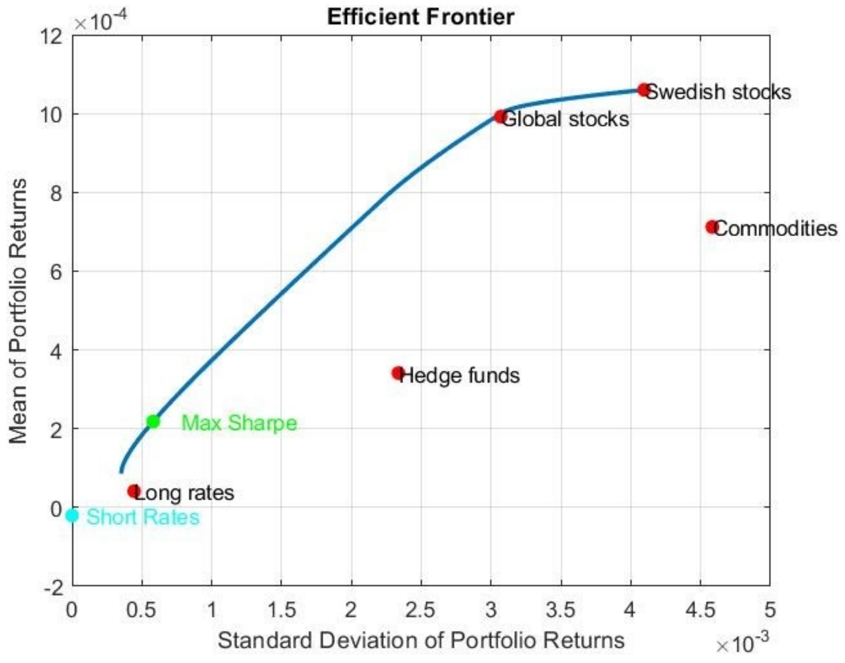


Figure 4.1.1: The risky assets are indicated by the red dots, the risk-free asset is represented by the short rates on the left and our final classic portfolio is where the Sharpe ratio is at its maximum, i.e., the green dot.

To ascertain that our final classic portfolio is really the Sharpe ratio portfolio, we can plot all the different Sharpe ratios and see where our portfolio will locate itself, see figure 4.1.2 below;

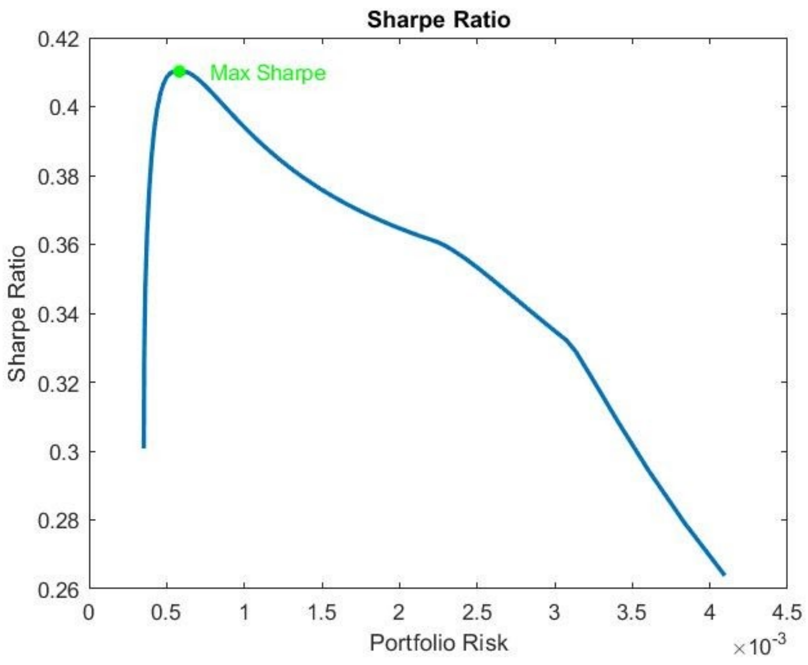


Figure 4.1.2: Our final portfolio has a ratio of return versus risk that peaks when the standard deviation is just above 0.0005.

Our final classic portfolio actually locates itself at the maximum Sharpe ratio value, which is 0.412.

4.2 The results of the deep portfolio

In section 3.3.3 *The mean and covariances of the deep model*, we showed the calculations of both the returns and the covariances of the assets based on the predictions of the deep model. The chapter ended with a reference to the procedure for creating a deep portfolio outlined in section 3.2, with the same conditions as for the classic portfolio. The weights/allocations of this deep portfolio are;

Class	Swedish stocks	Global stocks	Hedge funds	Long rates	Commodities
Asset	XACTOMX	VT	HFCHAS	RXBO	GSG
Weight	0.020056	0.033881	0	0.87719	0.014569

Table 4.2.1: Allocation of deep-portfolio

Class	Risk-free asset (short rates)
Asset	RXVX
Weight	0.0543

Table 4.2.2: weight of the riskfree asset

With these portfolio allocations, we end up with a portfolio for tomorrow with mean and variance as follows;

Optimal portfolio	Mean (%)	variance
Deep-portfolio	0.2338	5.8192e-06

Table 4.2.3: optimal deep-portfolio

where we calculated the variance by the square of the standard deviation. The following Figure 4.2.1 shows the location of the deep portfolio on the efficient frontier;

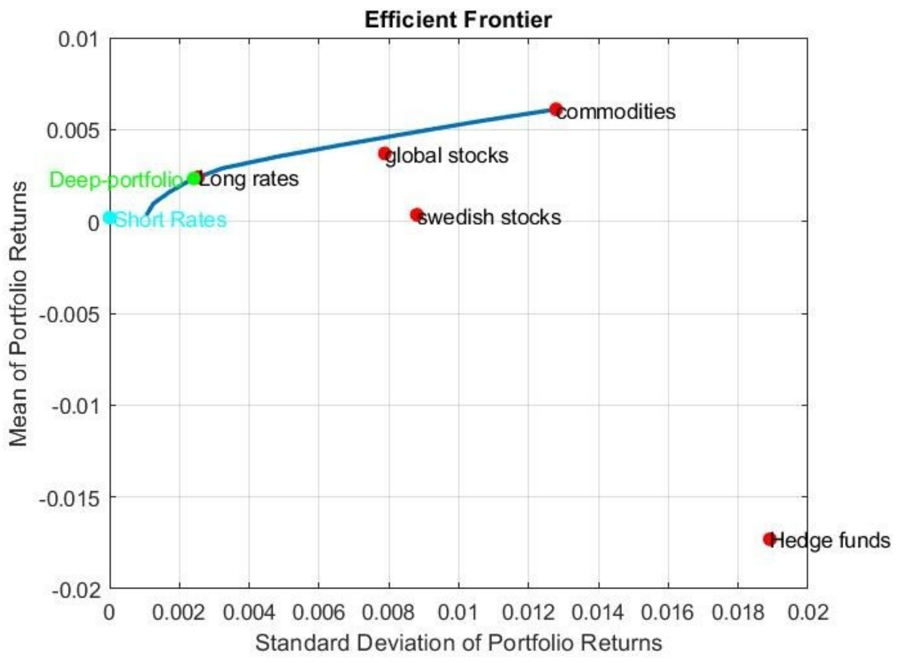


Figure 4.2.1: Deep portfolio allocation

where the locations of the assets are based on their means calculated from the prediction of their price on 1st January, 2020, which is of interest for identifying their optimal allocations in the portfolio if wishing to invest on 31st December, 2019. This is the same investment date as in the classic portfolio case.

To ensure that we are at the max Sharpe ratio, it is necessary to plot all possible Sharpe ratio values and observe where the deep portfolio locates itself;

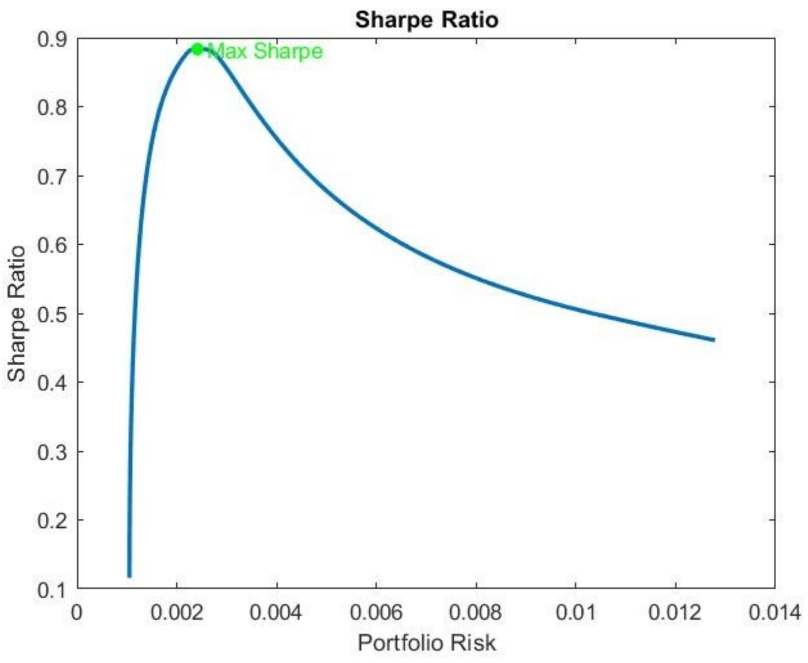


Figure 4.2.2: The deep portfolio, indicated by the green dot, has a max Sharpe ratio that is in fact a maximum value. The standard deviation is 0.0022.

The max Sharpe ratio value of the deep portfolio is 0.89.

Chapter 5

Analysis and discussion

5.1 Analysis

The result of the deep model is a portfolio that has a higher return and lower risk than that of the classic portfolio, with a Sharpe ratio that is almost two times greater. However, the allocations that we arrived at in the deep portfolio case can differ a great deal depending on the date chosen for the calculation of the optimal allocations. The deep model was better at showing tendency than predicting prices. Furthermore, if one chooses to look further ahead, for example a 10 day forward prediction, the errors increase significantly. One can argue that the stochastic behaviour of the market comes into play, or that we did not feed a network with a sufficient number of relevant input values. We tend to lean towards the latter.

In the Introduction we argued for the need to create a “portfolio for tomorrow” that can adapt to short-term trends, preferably with a “watchout” function that takes account of where we are heading in the long term. The classic model, i.e. the Markowitz model, is a stiff model in this respect, as it just divides historical returns by the number of days. We tweaked that a little in order to capture short-term trends and let the prices first go through an EMA. Despite this, the classic model lacks the ability to find underlying patterns in the same way as an LSTM has the potential to do. On the other hand, LSTM models need large amounts of carefully considered input data and computational power to achieve a reasonable standard.

5.2 Discussion, conclusion and future research

The conclusions of this work are that the length of the input data should be strongly reduced and expanded by data from the same time frame, for example one week, with a huge set of thoroughly considered indicators and minute data. However, this is only valid when considering allocations in short-term portfolios. But consider the following scenario: if the network is fed with many, many years of input data, there might be multiple regimes and the data will be subdivided over and over again. It would be similar to having a text generator that tries to emulate Shakespeare's style of writing but during training is also fed Fyodor Dostoevsky. One should instead consider having multiple networks, each specialized in a specific aspect. Additionally, it would be optimal to construct a dynamic network that tracks changes in the environment and adjusts its design, weights and hyperparameters accordingly. And finally, neural networks do not explain their decisions, they merely provide output values. Therefore, it would be interesting to develop a network that can explain the reason behind the output in a sentiment or a mathematical description, so that the investor can consider whether or not it is reasonable.

Bibliography

[1] Smartasset.com, Eric Reed, Jan 2020, ’A Guide to Portfolio Optimization Strategies’ [Online]. Available:
https://smartasset.com/investing/guide-portfolio-optimization-strategies

[2] Investopedia.com, James Chen, Feb 2020, ’What is a portfolio’ [Online]. Available:
https://www.investopedia.com/terms/p/portfolio.asp

[3] Myles E. Mangram, SMC University, Switzerland, 2013. A simplified perspective of the Markowitz portfolio theory

[4] Towards data science, Jahnavi Mahanta, July 2017, ’Introduction to Neural Networks, Advantages and Applications’ [Online]. Available:
https://towardsdatascience.com/introduction-to-neural-networks-advantages-and-applications-96851bd1a207

[5] The free dictionary.com, ’currency depreciation,’ [Online]. Available:
https://financial-dictionary.thefreedictionary.com/currency+depreciation

[6] Medium.com, G. Lemus, Jun 2018, ’Why Financial Time Series LSTM Prediction fails’ [online]. Available:
https://medium.com/datadriveninvestor/why-financial-time-series-lstm-prediction-fails-4d1486d336e0

[7] Itnext.io, Dmitry Rastorguev, Jan 2018, ’2017’s Deep Learning Papers on Investing’ [Online]. Available:
https://itnext.io/2017s-deep-learning-papers-on-investing-7489e8f59487

[8] The Journal of Finance, Harry Markowitz, March 1952, Portfolio Selection

[9] Investopedia.com, Sean Ross, 2019 ‘What are some common examples of unsystematic risk? [Online]. Available:

https://www.investopedia.com/ask/answers/040715/what-are-some-common-examples-unsystematic-risk.asp

[10] ResearchGate, Jorge Brusa, Rodrigo Hernandez, Pu Liu, Harold A. Dulan, Dec 2010, Systematic Risk, Unsystematic Risk and the Other January Effect

[11] Henrik Hult, Filip Lindskog, Ola Hammarlid, Carl Johan Rehn, 2012. Risk and portfolio analysis Principles and Methods.

[12] Assistant Professor of Psychology and Statistics University of Minnesota (Twin Cities), Nathaniel E. Helwig, Jan 2017, Data, Covariance, and Correlation Matrix

[13] The Correlation Coefficient: An Overview A. G. Asuero, A. Sayago, and A. G. Gonz´alez Department of Analytical Chemistry, Faculty of Pharmacy, The University of Seville, Seville, Spain

[14] Deborah J. Rumsey, 'How to Interpret a Correlation Coefficient r'. [Online]. Available:
https://www.dummies.com/education/math/statistics/how-to-interpret-a-correlation-coefficient-r/

[15] The balance, Melissa Phipps, July, 2019, 'Correlated and Non-Correlated Assets'[Online]. Available:
https://www.thebalance.com/what-is-asset-correlation-2894312

[16] Lazard Asset Management, Stephen Marra, 2015, Predicting volatility

[17] Macrooption.com, 2020, 'Historical Volatility Calculation'[Online]. Available:
https://www.macroption.com/historical-volatility-calculation/

[18] Forecasting with moving averages, Robert Nau, August 2014, Fuqua School of Business, Duke University.

[19] Inverstopia.com, Adam Hayes 2020, Exponential Moving Average (EMA) [Online]. Available:
https://www.investopedia.com/terms/e/ema.asp

[20] Researchgate.net, Seng Hansun, Nov 2013, A new approach of moving average method in time series analysis

[21] A new approach of moving average method in time series analysis, Jeffrey S. Simonoff, 2020, [Online]. Available:
http://people.stern.nyu.edu/jsimonof/classes/2301/pdf/regtime.pdf

[22] University of Notre Dame, Richard Williams, Jan, 2020, Heteroskedasticity, [Online]. Available:
https://www3.nd.edu/ rwilliam/stats2/l25.pdf

[23] [Online]. Available: https://i.stack.imgur.com/i3gnk.png

[24] Finance train, Constructing an Efficient Frontier [Online] Available:
https://financetrain.com/constructing-an-efficient-frontier/

[25] Investopedia.com, Akhilesh Ganti, 2020, Efficient Frontier [Online]. Available:
https://www.investopedia.com/terms/e/efficientfrontier.asp

[26] Sana Mushtaq, June 14, 2019, Data preprocessing in detail. [Online]. Available:
https://developer.ibm.com/technologies/data-science/articles/data-preprocessing-in-detail/

[27] The EconomicTimes.com, Definition of 'Sharpe Ratio', [Online]. Available:
https://economictimes.indiatimes.com/definition/Sharpe-Ratio

[28] Stanford University, William F. Sharpe, 1994 , The Sharpe Ratio, [Online]. Available:
https://web.stanford.edu/ wfsharpe/art/sr/sr.htm

[29] Sortino: A 'Sharper' Ratio, Thomas N. Rollinger, Scott T. Hoffman, [Online]. Available:
https://www.cmegroup.com/education/files/sortino-a-sharper-ratio.pdf

[30] Picture from 'if 2 random variables have exactly same mean and variance ' [Online]. Available:
https://stats.stackexchange.com/questions/314000/if-2-random-variables-have-exactly-same-mean-and-variance?fbclid=IwAR2eREB32Nbgp7N0dgxnxAFDEuK-hGQEFCveqXj9Z$_ryTHG3SQcCJ9rEqq8$

[31] Jonas DeMuro December 17, 2019, ’what is neural network’, [Online]. Available: https://www.techradar.com/news/what-is-a-neural-network

[32] Comparison of Supervised and Unsupervised Learning Algorithms for Pattern Classification, R. Sathya, Annamma Abraham, 2013

[33] Cornell university, César Laurent, Gabriel Pereyra, Philémon Brakel, Ying Zhang, Yoshua Bengio, ’Batch Normalized Recurrent Neural Networks’ [Online]. Available: https://arxiv.org/abs/1510.01378

[34] A Critical Review of Recurrent Neural Networks for Sequence Learning, Zachary C. Lipton, John Berkowitz, June 2015, [Online]. Available: https://arxiv.org/pdf/1506.00019.pdf

[35] Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) Network, Alex Sherstinsky, March 2020, [Online]. Available: https://arxiv.org/pdf/1808.03314.pdf

[36] Hojjat Salehinejad, Sharan Sankar, Joseph Barfett, Errol Colak, and Shahrokh Valaee, ’Recent Advances in Recurrent Neural Networks’. December 2017.

[37] Optimizers Explained - Adam, Momentum and Stochastic Gradient Descent, Casper Hansen, Oct 2019

[38] Machine learning Mastery,Jason Brownlee, Dec 2018 ’How to Avoid Overfitting in Deep Learning Neural Networks’ [Online]. Available: https://machinelearningmastery.com/introduction-to-regularization-to-reduce-overfitting-and-improve-generalization-error/

[39] Backpropagation Algorithm: An Artificial Neural Network Approach for Pattern Recognition, Dr. Rama Kishore, Taranjit Kaur, June 2012, [Online]. Available: https://www.ijser.org/researchpaper/Backpropagation-Algorithm-An-Artificial-Neural-Network-Approach-for-Pattern-Recognition.pdf

[40] Computer Science, University of Toronto, Roger Grosse, 2018, Exploding and Vanishing Gradients

[41] KTH, Deep Learning DD2424, 2018, Josephine Sullivan, Networks for Sequential RNNs and LSTMs

[42] Improving the way neural networks learn, Michael Nielsen, Dec 2019, [Online]. Available:
http://neuralnetworksanddeeplearning.com/chap3.html

[43] Statology.org,Feb 2020, [Online]. Available:
https://www.statology.org/root-mean-square-error-excel/

[44] Modeling Long- and Short-Term Temporal Patterns with Deep Neural Networks, Guokun Lai, Wei-Cheng Chang, [Online]. Available:
https://arxiv.org/pdf/1703.07015.pdf

[45] LSTM: A Search Space Odyssey, Klaus Greff, Rupesh Kumar Srivastava, Jan Koutn´ık, Bas R. Steunebrink, J¨urgen Schmidhuber, March 2015, [Online]. Available:
https://arxiv.org/pdf/1503.04069v1.pdf

[46] Computational Linguistics, Hyopil Shin (Dept. of Linguistics, Seoul National University), Understanding Stateful LSTM Recurrent Neural Network

[47] Mathworks.com, [Online]: Available:
https://se.mathworks.com/help/finance/tsmovavg.html

TRITA -SCI-GRU 2020:091