Charles University

Faculty of Social Sciences
Institute of Economic Studies

MASTER'S THESIS

Artificial Neural Networks in Option Pricing

Author: **Bc. et Bc. Dominik Vach**

Supervisor: **PhDr. Petr Gapko, Ph.D.**

Academic Year: **2018/2019**

Declaration of Authorship

The author hereby declares that he compiled this thesis independently, using only the listed resources and literature, and the thesis has not been used to obtain a different or the same degree.

The author grants to Charles University permission to reproduce and to distribute copies of this thesis document in whole or in part.

Prague, January 4, 2019

Signature

Acknowledgments

The author is grateful especially to his supervisor, PhDr. Petr Gapko Ph.D., for his valuable comments and good advice. Further, the author is grateful to doc. PhDr. Jozef Baruník Ph.D. who provided him an option dataset which was essential for the thesis completion. The author is also very thankful to his family and friends, who supported and motivated him throughout his university studies.

Abstract

This thesis examines the application of neural networks in the context of option pricing. Throughout the thesis, different architecture choices and prediction parameters are tested and compared in order to achieve better performance and higher accuracy in option valuation. Two different volatility forecast mechanisms are used to compare neural networks performance with Black Scholes parametric model. Moreover, the performance of a neural network is compared also to more advanced modular neural networks. A new technique of adding rational prediction assumptions to neural network prediction is tested and the thesis shows the importance of adding virtual options fulfilling these assumptions in order to achieve better training of the neural network. This method comes out to increase the prediction power of the network significantly. The thesis also shows the neural network prediction outperforms the traditional parametric methods. The size and number of hidden layers in a neural network is tested with an emphasis to provide a benchmark and a structured way how to choose neural network parameters for future applications in option pricing.

JEL Classification	C13, C14, G13
Keywords	Option pricing, Neural networks, Modular neural networks, S&P500 index options

Author's e-mail	vach.dominik@gmail.com
Supervisor's e-mail	petr.gapko@seznam.cz

Abstrakt

Tato diplomová práce zkoumá aplikaci neuronových sítí při oceňování opcí. V průběhu práce jsou vyzkoušeny a porovnány různé architektury a parametry neuronových sítí za účelem dosažení co nejpřesnější valuace opcí. Jsou porovnány dvě různé metody predikce volatility, které jsou vzápětí testovány při valuaci opcí neuronovými sítěmi oproti klasické metodě použití vzorce Blacka a Scholese. Navíc je přesnost neuronové sítě porovnávána s pokročilejší architekturou modulárních neuronových sítí. Nová technika zpřesnění predikce pomocí přidání tzv. racionálních predikčních předpokladů je použita a je ukázáno, že metoda přidávání virtuálních opcí znatelně zpřesňuje výkon neuronových sítí v opčním oceňování. Kromě toho práce ukazuje, že oceňování opcí neuronovými sítěmi je přesnější než výsledky získané parametrickými metodami. Na určení optimální velikosti a počtu vrstev neuronové sítě je v práci kladen důraz a je navrhnuta nová strukturovaná metoda, pomocí které lze určit velikost sítě pro budoucí aplikace v opčním oceňování.

Klasifikace JEL	C13, C14, G13
Klíčová slova	Oceňování opcí, Neuronové sítě, Modulární neuronové sítě, Opce S&P500 indexu

E-mail autora	`vach.dominik@gmail.com`
E-mail vedoucího práce	`petr.gapko@seznam.cz`

Contents

List of Tables

List of Figures

Acronyms

ANN Artificial Neural Network

ARCH Autoregressive Conditional Heteroskedasticity

ART Adaptive Resonant Theory

CBOE Chicago Board Options Exchange

ELU Exponential Linear Unit

GARCH Generalized Autoregressive Conditional Heteroskedasticity

MAE Mean Absolute Error

MAPE Mean Absolute Percentage Error

OLS Ordinary Least Squares

PDE Partial Differential Equations

ReLU Rectified Linear Unit

RESET Regression Equation Specification Error Test

RMSE Root Mean Square Error

Master's Thesis Proposal

Author	Bc. et Bc. Dominik Vach
Supervisor	PhDr. Petr Gapko, Ph.D.
Proposed topic	Artificial Neural Networks in Option Pricing

Motivation A great amount of literature has been written since Black and Scholes (1973) developed their option pricing formula by transforming the option pricing problem into the task of solving a parabolic partial differential equation with a final condition. This task can be done under several assumptions and the solution can be found in a closed form. However, some assumptions are in practice violated or are at least questionable. Therefore, a number of authors e.g. Scott (1987) tried to relax them which led to more accurate models. Despite some particular successes, unrealistic yet crucial assumptions such as log-normal distribution of prices and continuity of trading were still required.

Another approach was taken by the Hutchinson et al. (1994) who used an artificial neural network (ANN) model. This model is able to naturally explain the way input data affect output option price without any predetermined functional form of the relationship. It turned out that ANN models applied on American style options performed better than the commonly used Black-Scholes formula which served as a benchmark. This breakthrough paper was followed by e.g. Geigle and Aronson (1999), Ghaziri et al. (2000), Saito and Jun (2000), and many others who compared the performance of the ANN non-parametric approach to the Black-Scholes formula with varying results examined on both European and American style options. Nevertheless, the non-parametric approach is believed to have better out-of-sample prediction potential.

Even though many research papers have been written on this topic, not many of them analyse whether the two approaches could be combined or applied on non-standard derivates such as exotic options or barrier options. Furthermore, not much is known about appropriate sampling frequencies of data. Ultimately, there are various ways to estimate volatility parameter which is essential for the model performance. As stated by Hahn (2013), there is also no consensus about strategies used when deciding

how to select size, architecture, parameters, learning and evaluation functions for ANNs despite the extensive research in machine learning.

All the factors listed above contribute to the ambiguity of conclusions of the published papers and are not systematically treated. This thesis aims to find how some of these factors determine predictive power of the parametric models in comparison to the non-parametric ANN models.

Hypotheses

Hypothesis #1: Artificial neural network approach performs better in out-of-sample data prediction than conventional parametric models.

Hypothesis #2: Modular neural networks systematically outperform simpler neural networks in option pricing task.

Hypothesis #3: The prediction power of the neural networks in comparison to the parametric methods depends on the used volatility prediction mechanism and can yield contradictory results.

Methodology One of the proposed methods is to employ the modular neural network design approach. This approach takes advantage of decomposing the underlying complex problem into a number of subtasks and was found useful e.g. by Cofino et al. (2004) and many other authors. Modularity as a concept, also called divide and conquer method, can be very useful in approximative process in the case heterogeneous data are examined and helps to put a structure and patterns in otherwise unclear datasets. Modules used in this framework are represented by a single feedforward ANN model and their usefulness lies in the interaction with each other which can generalize complex market situations such are highly volatile periods with consequent relatively stable periods which simpler ANN models cannot effectively deal with. Moreover, selection of modules and data analysis can be based on more advanced techniques from the statistical learning theory such as fuzzy clustering (Jang et al. (1997)) or genetic algorithms (Cofino et al. (2004)) which is proposed by Gradojevic et al. (2007). The performance of particular models used in price prediction to test hypotheses is proposed to be measured by the total sum of squared differences of the predicted values from the actual data. The prediction power of modular ANNs will be compared to the basic backpropagation algorithm and also to the analytical formula for option pricing. The data used in this thesis to evaluate the hypotheses are expected to vary for each hypothesis in order to find whether the underlying asset, option style and possibly data frequency could change the inference.

Expected Contribution As stated above there are a lot of factors determining the performance of ANNs in comparison to the parametric models. This thesis aims to show how some of these factors could change the inference and results based on used data, neural network architecture, volatility prediction mechanism, sampling frequency or other important characteristics. On top of this, a new pattern in the process of choice of the particular factors shall be developed which is urgently yearned for by existing literature on this matter.

Outline

1. Introduction
2. Literature review
3. Theoretical part
3.1 Black-Scholes formula
3.2 Volatility prediction
3.3 Artificial neural networks
3.4 Modular neural networks
4. Empirical part
4.1 Description of data
4.2 Simulations
4.3 Results
4.4 Discussion
5. Summary and conclusion

Core bibliography

Black, F. and Scholes, M. (1973). The pricing of options and corporate liabilities. Journal of Political Economy, 81, 637–659.

Cofino, A. S., Gutierrez, J., and Ivanissevich, M. (2004). Evolving modular networks with genetic algorithms: Application to nonlinear time series. Expert Systems, 21(4), 208–216.

Geigle, D. S., Aronson, J. E. (1999). An artificial neural network approach to the valuation of options and forecasting of volatility. Journal of Computational Intelligence in finance, 7(6), 19–25.

Ghaziri. H., Elfakhani. S., Assi., J. (2000). Neural networks approach to pricing options. Neural Network World 10(1â€“2): 271–277.

Gradojevic, N.; Gencay, R.; and Kukolj, D. (2009). Option pricing with modular neural networks. IEEE Transactions on Neural Networks 20(4):626–637.

Hahn, T. (2013). Option Pricing Using Artificial Neural Networks : an Australian Perspective. Bond University.

Hutchinson, J. M., Lo, A. W., and Poggio, T. (1994). A nonparametric approach to pricing and hedging derivative securities via learning networks. Journal of Finance, 43(3), 851–889.

Jang, J. R., Sun, C., and Mizutani, E. (1997). Neuro-Fuzzy and Soft Computing. Prentice Hall, Inc.,

NJ. Saito. S, Jun. L. (2000). Neural network option pricing in connection with the Black and Scholes model. In Proceedings of the Fifth Conference of the Asian Pacific Operations Research Society, Singapore 2000.

Scott, L. (1987). Option Pricing when the Variance Changes Randomly: Theory, Estimation, and an Application, Journal of Financial and Quantitative Analysis, 22, 419–438.

Chapter 1

Introduction

Since the advent of computers, there have been numerous attempts to approach the complexity and precision of data processing the human brain has. The first attempt was done by McCulloch and Pitts already in 1943 who showed that simple neural networks could approximate any arithmetic or logical function. In the 1950s and 1960s, a first successful neuro-computer was built and scientists achieved many substantial successes in implementation and the development of neural network architectures.

Later in the 1970s the research in the area of neural networks was rather quiet as perceptrons showed to be unable to perform well enough. This thought was reinvestigated by a physicist of worldwide reputation John Hopfield in the 1980s who wrote two essential papers on neural networks that reached wide research audience all over the world. Since then many researchers significantly advanced in applying neural networks in various data processing fields such as image and object recognition, voice generation and recognition, physics, genetics, economics, finance, and many others. (Yadav *et al.* 2015)

In the meantime, stock markets significantly developed with many new asset pricing models for various financial instruments. In 1973, Black and Scholes published their famous formula for European-style option pricing which solves a parabolic partial differential equation with a final condition based on continuously revised delta hedging. Even though the model led to a massive boom in option trading and helped to legitimize activities of Chicago Board Options Exchange (CBOE), the model had several flaws as some assumptions were either questionable or explicitly violated in practice. (MacKenzie 2006) Hence, a number of authors e.g. Scott (1987) tried to relax the assumptions which led to more precise models. In spite of some partial successes, unrealistic

yet crucial assumptions such as log-normal distribution of prices and continuity of trading were still required.

A new approach was taken by Hutchinson *et al.* (1994) who used an artificial neural network model which was able to naturally explain the option price without assumptions made by parametric models (i.e. Black-Scholes formula). Moreover, this approach showed useful in pricing American-style option for which the pricing task is more complicated as it can be exercised anytime.

Since then, a great amount of literature comparing non-parametric (neural networks) and parametric (e.g. Black-Scholes formula) approaches have been written. Nevertheless, the literature is still ambiguous regarding the precision of volatility prediction mechanisms and other important characteristics of the model. Therefore, not much is known about appropriate sampling frequencies of data, there is also no consensus about strategies used when deciding how to select size, architecture, parameters, or learning and evaluation functions for neural networks.

As there is no closed form formula holding in general market conditions and neural networks are an alternative way to traditional models, the improvement of option pricing estimation using them is essential. This thesis aims to not only compare the performance of standard neural network with Black-Scholes formula, but also uses a concept of modular neural networks and compares it to the simple feedforward neural network prediction. In addition, the thesis aims to examine and propose a structured way how to choose size and architecture of the neural network specifically for option pricing. A new technique used by Yang et al. (2017) of adding rational prediction assumptions is also investigated and the goal of the thesis is also to show whether the technique of adding virtual options improves the prediction. Hence, the thesis aims to verify the following hypotheses.

Hypotheses

Hypothesis #1: Artificial neural network approach performs better in out-of-sample data prediction than conventional parametric Black-Scholes formula.

Hypothesis #2: Modular neural networks systematically outperform simpler feedforward neural networks in the option pricing task.

Hypothesis #3: Fulfilling of rational prediction assumptions including adding of virtual options outperforms simpler feedforward neural networks in the option pricing task.

Hypothesis #4: The out-of-sample prediction power of the neural networks depends on the used volatility prediction mechanism and can yield varying results.

On top of this, a structured pattern regarding choosing of the size and architecture is aimed to be developed reflecting the thesis results.

The thesis structure is divided as follows. The chapter Literature review discusses previous research done in the area of option pricing using neural networks. Third chapter is devoted to introduce the theoretical framework for both option pricing and neural networks. Fourth chapter presents the data analysis, numerical simulations and tests done to find the right choice of parameters and final results. The thesis concludes with a discussion of results and suggests possible improvements in the last chapter.

Chapter 2

Literature Review

In this chapter, the aim is to explain what the reasons were for investigating the topic chosen in this thesis and how it is related to the literature that has been written on this matter. This chapter also should help the reader to orientate in the field of option pricing and especially in the application of neural network approach. This literature review should help as a guide in case the reader would like to examine more about the areas related to the topic covered by this work. It is split to three sections according to the primary field of the research.

2.1 Option pricing literature

The option pricing models are very thoroughly investigated by the researchers from wide range of the fields. The popularity of this topic is given not only by the relation of this instrument to the stock markets, but also by the fact it occurs at the edge between the academic research and the practitioner use. For academics, it is a perfect tool that evaluates what exactly is happening at the financial markets whereas for practitioners this topic is considered as a profit making tool.

Probably the most influential paper written in this field is the paper by Black & Scholes (1973) who developed their famous formula which gives a theoretical estimate of European option price. Many further researchers tried to improve this model. Among them was e.g. Scott (1987) who tried to relax the key assumptions made by Black and Scholes. Despite several successes in the area of assumption relaxation, the models were still requiring too strict assumptions that made them not generally successful. A good review on the already existing parametric models was written e.g. by Bates (2003).

2.2 Econometric methods literature

As the pricing of any instrument in the financial markets is a quite active research area, the options were a natural choice for many researchers. The biggest problem that was observed soon after the Black-Scholes formula was developed is the non-constant volatility implied by the model. This is caused by the existence of the fat-tailed and skewed distributions. One of the proposed solution to this volatility problem was an approach done by Heston (1993) who considered random process in order to model the stochastic volatility. On the other hand, there is another approach to model the volatility by the inclusion of jump correcting for events that have low probability which might result in a good model including the volatility smile which was shown in Kou (2002) and Carr *et al.* (2002).

Further methods relied on the techniques of Fourier transform based option valuation (Carr & Madan 1999), finite difference option pricing method (Merton *et al.* 1977), or Monte Carlo option pricing method (Boyle 1977).

Regarding the volatility prediction mechanisms, the recent literature shows that implied volatility as a forecast method is biased and inefficient which was handled in Baruník & Hlínková (2016) by correcting the mechanism using wavelet regression. In the older literature, it was shown by Heston & Nandi (2000) that their GARCH(1,1) closed-form based option pricing model performed well.

GARCH(1,1) estimation procedure was employed also by Hahn (2013) who showed several different scenarios with varying volatility. He also mentions that there is no consensus about the strategies how to effectively use choose the volatility mechanisms and also size and architecture of the neural networks. Furthermore, in Figlewski (1997) it is stated that the prediction using historical volatility is not optimal as the true model has correlation in returns, time-varying volatility, and the returns are not normally distributed which led him to use the GARCH model, too.

The GARCH models were originally developed by Bollerslev (1986) who generalized the work regarding ARCH models by Engle (1982). However, as the financial markets do not have symmetric movements in prices, the EGARCH model was developed by Nelson (1991) and TARCH model by Zakoian (1994) where both models treat the positive and negative movements in a separate way. Further researchers such as Christoffersen & Jacobs (2004) or Berkowitz (2010) suggested that their option pricing model has good performance under

the condition of frequent parameters and volatility update.

2.3 Neural networks literature

Regarding the neural networks, there is a lot of research done on option pricing by computer scientists. The neural networks are not required to fulfill any economic assumptions or axioms as they yield results based on the universal approximation theorem which was shown in Hornik (1991). This was soon investigated in option pricing application by Hutchinson *et al.* (1994) who compared the neural network models with Black-Scholes formula in three distinct models on American option data and showed that the neural network predicted the price better. The application on both European and American styles option was comprehensively done by authors in the consequent years. The analysis has been also thoroughly done on the S&P500 index options data which is done also in this thesis and was done before e.g. by Bakshi *et al.* (1997); Garcia & Gençay (2000); Dugas *et al.* (2001); Baruníkova & Baruník (2011). It was shown in Geigle (1999) that in some cases the Black-Scholes formula may have better prediction power than the neural network. Malliaris & Salchenberger (1993) on the other hand showed, the Black-Scholes formula performs better in case of in-the-money options whereas the neural network approach is dominant in the out-of-sample prediction. Further evidence was made which favored the performance of neural networks e.g. by Ghaziri *et al.* (2000), or Saito & Jun (2000). Moreover, this evidence was found on various underlying assets e.g. in Lajbcygier *et al.* (1996) who compared the neural network with Black-Scholes for Australian Share Price Index futures.

The research was soon followed by the neural networks with specified requirements on the functional shape like in Garcia & Gençay (2000) or Dugas *et al.* (2001). These papers suggested that it is important to include the economic axioms so the models can better predict the price. The flaw of most of literature done on this topic was that the researchers tried to come up with one general model, that describes all market situations perfectly. This was often violated by deep in the money or deep out of the money options.

A completely new approach was made by Gradojevic *et al.* (2009) who solved the problem by creating the modular neural network with the modules based on the moneyness and time to maturity of the options. Similar idea was shown in Baruníkova & Baruník (2011), who compared the prediction power for different subsets of the data. This idea was developed in the recent paper

by Yang *et al.* (2017) who criticized the fixed heuristics that were used in Gradojevic *et al.* (2009), and thus they proposed a new method how to split the modules automatically based on the market conditions using the advantage of gating networks. This paper also investigated the importance of imposing assumptions on the data thorough the inductive bias made by structural change in data by option generation. The summary of more advanced modular neural network architectures was shown by Azam (2000) who points out the usage of adaptive mixture of local experts architecture by Jacobs *et al.* (1991). The mentioned option generation is investigated as a tool for verifying whether the size of the network is big enough also by another recent paper by Culkin & Das (2017) who generated data matching Black-Scholes formula in order to calibrate the parameters.

Moreover, it was mentioned by several authors that some procedures used in the other application of machine learning such as fuzzy clustering (Jang *et al.* 1997) or genetic algorithms (Cofiño *et al.* 2004) could be used in order to improve the prediction of the neural networks.

Chapter 3

Theoretical Part

3.1 Option pricing

3.1.1 Option Fundamentals

An option is a financial markets derivative which gives its holder a right but not an obligation to buy or sell (depending on the option type) a related underlying asset for a given fixed price which is set at the time of option contract purchase. As it is a benefit to own an option contract due to its possibility to buy (or sell) below (or above) the market price, the option contract costs a premium. There are two basic types of options, call option which gives the owner a right but not an obligation to buy an underlying stock for a specific price called exercise price (also strike price), and put option which gives the owner a right but not an obligation to sell an underlying asset at a given exercise price.

Options can be exercised at a certain date called maturity or expiration date or earlier depending on its features. The basic distinction is to European and American options where the European ones have a fixed maturity date whereas the American options can be exercised anytime prior to maturity which is usually supported also by a higher premium in case of American options. American options also have higher risks for the option sellers due to the early exercise which can be done by the option holder.

Based on the actual value of the underlying asset the option can be categorized as in-the-money, at-the-money, or out-of-the-money which depends on the relative price of the underlying asset to the exercise price. The option is said to be in-the-money (ITM) if the difference between the underlying asset price and exercise price is positive in case of call options and negative in case of put options, which implies the option has intrinsic value that would be gen-

erated as a profit if the option was immediately exercised. It is said to be at-the-money (ATM) if the underlying asset price exactly matches the strike price of the option and therefore it is exactly the break even point between no intrinsic value and positive intrinsic value of the option. The option is said to be out-of-the-money (OTM) if the strike price exceeds the underlying asset price in case of call options and not exceeds the underlying asset price in case of put options. Such an option does not bear any intrinsic value and as the option premium is not negative the profit would be negative if the option is exercised.

Practically, the option derivative trading is similar to futures trading in the sense that there is always one buyer (going long) and one seller of the contract (going short). The seller has an obligation to fulfill the right of the buyer whenever the buyer exercises the option, which is an advantage for an option holder in case of American options (most of equity options), who can decide the optimal time for the option exercise.

3.1.2 Black-Scholes formula

As the options are widely used and allow for more specific trades than futures contracts, the urge for a valuation technique was sought by many researchers and practitioners. The option pricing task is frequently performed by the application of the Black-Scholes formula which is the result of the following Black-Scholes equation (parabolic partial differential equation)

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0 \tag{3.1}$$

where V is the price of a derivative, S is the price of the stock, σ is the standard deviation of the stock returns, r is the annual risk-free interest rate (continuously compounded) and t is the time. The motivation for the solution of this equation is in the minimization of risk on the option markets by buying or selling the underlying asset in exactly the number that is needed to eliminate the risk which is called continuously revised delta hedging. As the parabolic equation can be solved by the theory of Partial Differential Equations (PDE) given the specific initial (or final) and boundary conditions (see e.g. Evans (2010)), the following Black-Scholes formulas are obtained for the pricing of European call and put option

$$C(S,t) = N(d_1)S - N(d_2)Xe^{-rt} \tag{3.2}$$

$$P(S,t) = N(-d_2)Xe^{-rt} - N(-d_1)S \tag{3.3}$$

where C is the call option price, P is the put option price, $N(\cdot)$ is the cumulative distribution function of the standard normal distribution, t is the time to maturity, X is the strike price, S is the spot price of the underlying asset and

$$d_1 = \frac{1}{\sigma\sqrt{t}}(\ln(S/X) + (r + \frac{\sigma^2}{2}t)) \tag{3.4}$$

$$d_2 = d_1 - \sigma\sqrt{t}. \tag{3.5}$$

This model is widely used due to its simplicity, however in practice there are some assumptions to which the model is limited. First, the model underestimates possibility of extreme market movements and therefore neglects the fat tails. Second, it supposes the instantaneous and free execution of trading which is in practice also violated and yields a liquidity risk. Third, the model assumes also that the process is stationary which yields also a volatility risk. Last but not least, there is an assumption of continuous time and continuous trading which is generally not satisfied.

3.1.3 Option Greeks and delta hedging

The option pricing task can be analyzed also by the analyzing of the so called Greeks. In mathematical finance it is often described as a sensitivity of the price of options under change of some parameter the price depends on. These sensitivities are used in particular in risk management. These sensitivities are both in Black-Scholes model and in the models described by the artificial neural networks easily obtained by direct computation or by taking numerical derivatives (Haug 2007). The most important Greek is Delta defined as $\Delta = \frac{\partial V}{\partial S}$ where the variables are the same as in the Black-Scholes formula section (V is the value of the option instrument and S is the value of the underlying asset). Delta is measuring the rate of change of the option price based on its underlying stock. The Δ_c coefficient for call options lies in the interval $[0, 1]$ whereas Δ_p coefficient for put options lies in the interval $[-1, 0]$. These two values are mutually interconnected and for the same underlying asset with the

same exercise price and expiry it holds that

$$\Delta_c - \Delta_p = 1, \tag{3.6}$$

which is called put-call parity. The put-call parity requires fewer assumptions than Black-Scholes model as the most important is to have an existence of a forward contract. It is a useful tool when calculating how to hedge portfolio in order to have a delta-neutrality. Delta neutral portfolio is such a portfolio where the portfolio value remains unchanged in case of a small change of the underlying stock price. The delta neutral portfolio is obtained if the positive and negative delta components offset and "neutralize" each other (Haug 2007).

Another important Greek is Vega defined as $\nu = \frac{\partial V}{\partial \sigma}$ and it measures the amount of money gain (loss) in case of an increase (decrease) in the volatility. Further first order Greek is Theta, which is defined as $\Theta = -\frac{\partial V}{\partial \tau}$ and is important as it is measuring the time sensitivity of the option value. Theta is typically negative for call options and positive for put options, which means the option holder gains or losses a time value of the option as it is held. The last important first order Greek is Rho, which is defined as $\rho = \frac{\partial V}{\partial r}$ and measures the sensitivity to changes in risk-free rates and is the least used first order Greek.

Among the Greeks of higher order, the most important is Gamma defined as $\Gamma = \frac{\partial^2 V}{\partial S^2} = \frac{\partial \Delta}{\partial S}$, which corrects for the convexity issues as it is the second order term of the option value Taylor expansion. The Γ is the same for both call options and put options with the otherwise same parameters (Haug 2007).

3.2 Volatility prediction mechanisms

One of the most important parameters in the option pricing task is the volatility of the underlying asset due to its non-observability in the price data whereas other parameters such as stock price, strike price, time to maturity and the interest rate are often directly observable. As it is a measure of how the price of an asset fluctuates around its mean, it is important to use the correct value in the price prediction. It significantly influences the time value of the option as the expectations of high volatility in the future usually inclines to increase the option prices (Van Horne, 1969). There are several ways how to predict volatility. One of the aims in this thesis will be to find out the differences and

compare their performance by focusing on the implementation of more types of volatility estimation techniques.

3.2.1 Historical volatility

This method how to estimate volatility relies on the usage of historical stock price data at a given time frequency. It is obtained as the standard deviation of the underlying stock's return over some period. The return of the stocks is often calculated in the logarithm scale from the closing prices (Enke and Dagli, 2007)

$$r_t = \ln\left(\frac{S_t}{S_{t-1}}\right), \tag{3.7}$$

where S_t is the underlying stock price in the period t and r_t is the return between the periods $t-1$ and t. Historical volatility is then obtained by the standard statistical formula for the second statistical moment, the variance, and taking its square root

$$\sigma_{HV} = \sqrt{\frac{1}{T-1}\sum_{t=1}^{T} r_t - \bar{r}}, \tag{3.8}$$

where σ_{HV} denotes the historical volatility estimator, T is the number of periods over which the volatility is measured and $\bar{r}$ is the mean value of the stock returns r_t (Enke and Dagli, 2007). The choice of the time span of the interval over which the volatility is measured is questionable and there is a rule of thumb to use T as a number of days which the volatility estimation will be applied to. It is also possible to calculate the historical volatility from the standard deviation from the moving average of returns, which better handles the altering value of the mean returns value. (Hull, 2003) In cases where σ_{HV} historical volatility is higher, it follows from the previous unstable period with a greater amplitude in the price returns movements. On the other hand, the calm periods of stable trading or low activity in the market might be characterized by the low value of the historical volatility estimator σ_{HV}. The historical volatility estimation is often used for the cross-sectional comparison between various stocks or one specific stock to the whole general market (or possibly some part of it). It is also possible to compare two historical volatilities over different time periods which yields the information whether the volatility recently changed positively or negatively.

3.2.2 Implied volatility

In the option pricing, the commonly used volatility prediction mechanism is the implied volatility measure. Its application arises from the Black-Scholes model as it uses a lot of market information including the predicted option price and tries to explain the market expectations including the future volatility. The implied volatility measure then calculates the last remaining unknown variable which is the volatility of the option. Similarly to the historical volatility, in order to find the relative size of the volatility, the measure should be applied for various periods in the past in order to evaluate the market expectations.

The option contracts vary according to the implied volatility and it is not constant in time. This is in direct contradiction with one of the assumption of the Black-Scholes model which assumes the same volatility during the time for one particular underlying asset, however the options deep in-the-money or deep out-of-the-money usually do not respond to volatility as near at-the-money options (Enke and Dagli, 2007).

Volatility smile

In most cases, the implied volatility computed from the option prices in the market constitutes a persistent pattern which varies according to the remaining time to maturity and its exercise price, while the Black-Scholes model assumes no pattern and constant volatility instead. This pattern is also known as volatility smile, where the name stems from the shape of the implied volatility function given the exercise prices as it mostly has convex shape with global minimum near at-the-money situation. Volatility smile is an empirical phenomenon which rises for example in the currency option markets and implies that deep out-of-the-money and deep in-the-money options exhibit higher prices than the theoretical values given by the Black-Scholes model. In stock and stock indices markets, the implied volatility generally tends to have a negative relationship to the strike price meaning that with increased strike price, the implied volatility decreases - this phenomenon is called the volatility skew and it follows from the usage of the leverage. During the decline of the price of the underlying asset, the leverage increases and in turn the volatility also increases which is why the probability that the price decreases is higher. This holds similarly also vice versa (Enke and Dagli, 2007).

3.2.3 Volatility forecast with the ARCH family models

As stated in Figlewski (1997), there are many reasons why classical prediction from the historical volatility is not correct. The problems are stemming from the serial correlation in returns, time varying volatility, noisy estimate of the mean, and also the fact that returns are not normally distributed. The time varying volatility can be handled by the models which involve the volatility prediction over time.

The basic model which assumes time-invariant volatility could be specified as follows

$$r_t = E(r_t) + \epsilon_t \tag{3.9}$$

s.t.

$$E(\epsilon) = 0 \tag{3.10}$$

$$Var(\epsilon) = \sigma_t^2 \tag{3.11}$$

where the simplest model would be assuming constant variance

$$\sigma_t^2 = C. \tag{3.12}$$

Instead of this simple model, the volatility could be rather computed by the Autoregressive Conditional Heteroskedasticity (ARCH) model first used by Engle (1982) which handles the time-varying volatility quite effectively. The variance in the period t is given by

$$\sigma_t^2 = \beta_0 + \beta_1 \epsilon_{t-1}^2 \tag{3.13}$$

for the most simple ARCH(1) model which is reflecting one lag of the errors in estimation of the variance. If we added more lags we would obtain ARCH(q) model with $q \in \mathbb{N}$ lagged residual terms. The model is given by

$$\sigma_t^2 = \beta_0 + \beta_1 \epsilon_{t-1}^2 + \cdots + \beta_q \epsilon_{t-q}^2. \tag{3.14}$$

In order to achieve numerical stability of the stochastic process, it should not be unit root and therefore the sum of coefficients should be less than one

$$\sum_{i=1}^{q} \beta_i \leqslant 1 \tag{3.15}$$

In practical applications, q parameter is usually quite low as higher number of parameters decrease the precision of the out-of-sample prediction. Longer du-

ration effects are also better described by GARCH models (Generalized ARCH) by Bollerslev (1986) which reflects in the dependent variables also the variance from the previous periods. The simplest and widely used GARCH model is GARCH(1,1) and is given by the following specification

$$\sigma_t^2 = \omega + \alpha_1 \sigma_{t-1}^2 + \beta_1 \epsilon_{t-1}^2 \tag{3.16}$$

where ω, α_1, and β_1 are the estimation parameters and ϵ_{t-1} is the residual term from the (3.35) regression. This model has again possible generalization to GARCH(p,q) given by

$$\sigma_t^2 = \omega + \alpha_1 \sigma_{t-1}^2 + \cdots + \alpha_p \sigma_{t-p}^2 + \beta_1 \epsilon_{t-1}^2 + \cdots + \beta_q \epsilon_{t-q}^2, \tag{3.17}$$

where all α_i and β_j for all i and j are also regression coefficients. The long run steady state in case of GARCH(1,1) is given by

$$\sigma_{LR}^2 = \frac{\omega}{1 - \alpha_1 - \beta_1} \tag{3.18}$$

since the expected value of ϵ^2 is σ^2 and therefore the model generally converges to the steady state value (if the coefficients sum up to less than 1). The GARCH models, however, have a strong shortcomings given by the fact that they are hard to fit and they also restrict the impact of the shock to be independent of its sign (Figlewski, 1997). It is well known that in some markets the shocks are not symmetric as market traders usually react differently in bullish and bearish markets (e.g. Black (1976)). This problem was solved by Nelson (1991) who modeled a logarithm of preceding variances in so called EGARCH (Exponential GARCH). Even though in this thesis this model is not used, it is worth mentioning it extends the application of GARCH models as it allows for asymmetric treatment for positive and negative movements as well as it can handle negativity of the dependent variable which could be an improvement of the approach used in this thesis. The model specification is given by

$$\log \sigma_t^2 = \omega + \alpha_1 \log \sigma_{t-1}^2 + \cdots + \alpha_p \log \sigma_{t-p}^2 + \beta_1 g(\epsilon_{t-1}) + \cdots + \beta_q g(\epsilon_{t-q}) \tag{3.19}$$

where $g(\epsilon_t) = \theta \epsilon_t + \lambda(|\epsilon_t| - E(|\epsilon_t|))$ and θ and λ are the estimation coefficients. The functional form of the $g(\epsilon_t)$ function enables the magnitude and sign to have different effects on the volatility which is useful in the price estimation (St. Pierre and Eilleen, 1998). Another possibility is to use TARCH (Threshold

ARCH) model developed by Zakoian (1994) which uses different coefficients for different signs of price movements with the model specified by

$$\sigma_t^2 = \omega + \alpha\sigma_{t-1}^2 + \beta_1\epsilon_{t-1}^2 + \gamma_1\epsilon_{t-1}^2 I_{t-1} \tag{3.20}$$

where $I_{t-1} = 0$ if $\epsilon_{t-1} \geqslant 0$ and $I_{t-1} = 1$ if $\epsilon_{t-1} < 0$.

The ARCH family models are widely used in economics and finance and the main focus is put on in-sample variance explanation rather than out-of-sample forecast (Figlewski, 1997). The model is fitted by performing maximum likelihood estimation for ϵ_t terms with frequent use of the normal distribution for the log prices but sometimes the t-distribution is used in order to allow for the fat tails effect. However, the log-normality assumption of the model does not fit the real situation enough because of the too frequent large price changes which is often explained by the fat tails in returns distribution (Figlewski, 1997).

The problem with ARCH models is often doing the estimation as the models require large data sets to train on, otherwise they do not behave well enough. Another problem with the ARCH models may be there might be a long term shock, which cannot be involved in the model as it would diminish the explanation power of the model. On top of that, the econometrically estimated coefficients might become negative which is why the model simplicity is in this case favored. Last but not least, the models are designed to predict the variance for one or a few steps ahead instead of a long term forecast which makes it unsuitable for long term predictions (Figlewski, 1997).

3.3 Artificial neural networks

3.3.1 Motivation and biological neuron

Artificial neural networks are data processing entities which have similar information processing properties and functions as the biological neural networks i.e. human or animal brains. The usage of brain-like structure in the calculation of various problems was always sought due to its adaptability and problem solving attributes. Biological neural systems are naturally evolved tangles of neurons, brain cells, that are able to transfer the impulse through the network of mutually interconnected neurons. A biological neuron consists of a cell with many small parts called dendrites which are receiving the signals from the other

neurons. The collected signal goes through an axon which is another part of the neuron body and at the end of the axon there are connections called synapses which connect the axons to dendrites of other neurons (Azam, 2000).

The brain is generally considered a black box with many functionalities not thoroughly understood. It obtains input signals and returns output response which as a byproduct strengthens the links between particular neurons that were frequently used. The brain usually decides to fulfill particular goals by the most likely way how to do the given task which wrongly gives an intuition that the brain works as a whole (Azam, 2000). There is an evidence from neuropsychology and neurobiology that some parts of the brain are independent on another and some specific brain functions can be even done without being connected to other parts. The neurobiologists have long thought that the particular parts of the brain are distinguished in distinct modules mutually cooperating with each other and the advances in neurobiology confirm these thoughts and point on the fact that the human and animal brains are consisting of different modules with different physiological composition which are responsible for various tasks (e.g. Huble, 1988). This makes a valid argument why it makes sense to apply artificial neural networks to the problems with blurry forms of results without any particular easily seen relationship as it is exactly the natural way how the brain processes the information.

3.3.2 Neural networks as a non-linear approximation tool

In order to predict the option prices Hutchinson et al. (1994) used as the first group an Artificial Neural Network (ANN) approach. ANN models are naturally capable of identifying the empirical relationship between input and output data without prior knowledge about the closed form functional relationship which makes it a useful tool for the option pricing task.

The usage of neural network models is also called a non-parametric approach and has several advantages in comparison to the parametric models (Black-Scholes etc.). In non-parametric approach there are not assumptions such as lognormality of returns or time continuity of trading. Moreover the models are adaptive and can immediately react to unexpected shocks by changing the functional form.

There is also a problem with this approach which is its uneffectiveness in cases when there is not enough data. The ANN has to be trained on a very large datasets and thus requires a large database of historical prices. The approach

is therefore not very good in case of thinly traded options or other derivatives (Can and Fadda, 2014).

ANNs are parallel distributed statistical models which process information in the input data set and generalize the relationships between the data in order to forecast the new situations. The usage of ANNs for the time series data is not binding as it can be used also to cross-sectional applications (Ng and Lippman 1991). The usage of ANNs as a non-parametric method is a quite recent way how to predict unknown nonlinear functional relationship. Hornik et al. (1991) showed that the feedforward networks can approximate any arbitrary function with an arbitrary accuracy which opened door to all diverse applications for nonlinear modelling.

3.3.3 Perceptron and feedforward neural networks

Perceptron is the most basic form of neural network. It consists of only one element which classifies whether the input belongs to the target class or not. It uses a function

$$\phi(x) = \begin{cases} 1 & \text{if } w \cdot x + b > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{3.21}$$

where w is the vector of weights, x are the input values and b is the bias which shifts the decision boundary which enables the perceptron to form various outputs, the $w \cdot x$ stands for the dot product of two vectors. As the function goes up in the values 0 or 1, the perceptron is an artificial neuron which uses Heaviside function as its activation function.

A feedforward neural network is a neural network, where the connections between the elements of the network are arranged into separate layers and each layer of neurons can influence only its output in the next layer whereas the layer from which it draws input data stays fixed. This procedure guarantees the one-way transfer of data which is the reason why the neural network is called feedforward. The layers between the input and output layers are called hidden layers and each hidden layer can have any number of neurons (nodes). The number of hidden layers depends on the nature of the approximation problem and there are various empirical techniques how to choose it. The feedforward network is characteristic by nonexistence of any closed cycles or loops in the network. In comparison to the perceptron, the feedforward network can have a continous output where the mathematical function processing the inputs is called an activation function and is commonly taken as the logistic (or equally

sigmoid) function because of the existence of its derivative in the closed form

$$f(x) = \frac{1}{1 + e^{-x}}. \tag{3.22}$$

As one of the goals of machine learning is to find the best possible functional form of inputs providing predictions for other situations, the process of learning during the data evaluation is important. In multi-layered neural networks, the commonly used algorithm is backpropagation, which calculates the error contribution after the data are processed. The backpropagation algorithm then calculates the differences between the training pattern target and the actual value. Based on the differences, the weights are updated to find the gradient of the weight which is consequently split and its fraction is subtracted. This optimization problem results after several steps in a steady state where the weights at particular layers are fixed. The most common approach to optimize these weights is to use a gradient descent optimization scheme.

3.3.4 Activation functions

When constructing a feedforward neural network, it is often questionable not only how to choose the size and architecture of the network, but also what the activation functions between the individual layers should be. The assumption which has to be done in the choice of the activation function is the continuity of the activation function and the existence of derivatives at all points. Even though the process done by biological neuron is very close to the mathematical definition of sigmoid activation function, there are also other possibilities how to choose the activation function in case of more hidden layers which may in some application perform better. The following functions represent all activation functions used and compared in this thesis and will be denoted by ϕ_i where i is a coefficient determining the functional form (Glorot *et al.* 2011)

$$\phi_1(x) = \frac{\exp(x)}{1 + \exp(x)} = \frac{1}{1 + e^{-x}}, \tag{3.23}$$

$$\phi_2(x) = \ln(1 + \exp(x)), \tag{3.24}$$

$$\phi_3(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{otherwise,} \end{cases} \tag{3.25}$$

$$\phi_4(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{otherwise,} \end{cases} \tag{3.26}$$

$$\phi_5(x) = \begin{cases} x & \text{if } x > 0 \\ \exp(x) - 1 & \text{otherwise.} \end{cases} \tag{3.27}$$

The first enumerated function is the sigmoidal (or logistic) function $\phi_1(x)$ and it converges to 1 (full signal transfer) as it approches $+\infty$ and 0 (no signal transfer) as it approches $-\infty$. Due to these convergence properties, its simple derivative, and its simplicity it is very frequently used by many researchers. Second function $\phi_2(x)$ is the softplus function which offers an interesting property for the positive part of the input, as it grows asymptotically linearly as it approches $+\infty$ and behaves as an exponential function for the negative input. It is also a smooth approximation of the next function called Rectified Linear Unit (ReLU). This function is denoted $\phi_3(x)$ and completely ignores the input with negative sign whereas the positive input is treated linearly. The advantage of ReLU activation function is that it quickly separates the input and works well as a filter. The next activation function $\phi_4(x)$ is called Leaky ReLU activation function, which is almost identical to the previous ReLU function but also a small fraction of negative input is transmissible through the neuron. The last activation function used in this thesis $\phi_5(x)$ is the Exponential Linear Unit (ELU) activation function which makes the mean activations closer to zero which speeds up the learning. (Glorot et al., 2011)

3.4 Modular neural networks

In the thesis, the modular neural network framework is of an utmost importance. It decomposes complex problem to smaller modules which interact with each other. In order to bring it more to the option pricing context, we follow (Jang et al. 1997) and we specify the activation function for the modular neural network architecture as a weighted average of outputs of individual modules

$$\varphi(x_1, x_2) = \omega_1 c_{1t}(x_1, x_2) + \cdots + \omega_M c_M(x_1, x_2) = \sum_{k=1}^{M} \omega_k c_k(x_1, x_2). \tag{3.28}$$

where M denotes the number of modules which are approximated by functions $c_1, \ldots, c_M$ and their weights $\omega_1, \ldots, \omega_M$. The weights ω_k can be estimated by several methods. McCullagh & Nelder (1989) and Bridle (1990) proposed a

softmax activation function which they defined with weights given as

$$\omega_k = \frac{\exp(c_k)}{\sum_{j=1}^{M} \exp(c_j)}, \tag{3.29}$$

s.t.

$$\sum_{k=1}^{M} \omega_k = 1. \tag{3.30}$$

Another possibility is to utilize Gaussian mixture models which is a way how to characterize each module by parametric Gaussian distribution. The model is then considered as a mixture of these distributions. The Gaussian mixture models can be also considered as gated neural networks which pass their output to multiplication gates, where all used modules are weighted.

3.4.1 Modular neural network modules

This thesis, however, follows a slightly different approach which is shown as first by Gradojevic et al. (2009) that it makes sense to choose the particular modules by using option parameters such as moneyness (S/X) and time to maturity t. In such a setup it is also shown to be a good approach to split the options in different groups according to the named parameters and switch the neural network modules according to the subset the input data comes from. This technique guarantees a better treatment of the input data than standard weighting of particular modules as shown in equation (3.28). The particular modules are considered as feedforward neural networks and the overall output of the modular network is estimated by the following equation in case of one hidden layer

$$c_k = \phi_k \left(\beta_{k0} + \sum_{j=1}^{q_k} \beta_{kj} \psi_k \left(\alpha_{kj0} + \sum_{i=1}^{s} \alpha_{kij} x_i \right) \right), k = 1, \ldots, M \tag{3.31}$$

where q_k denotes number of hidden nodes in k^{th} module, α_{kij} and β_{kj} denote the connection weights between the adjacent layers for the modules and subscripts equal to zero denote the neural network biases. ϕ_k and ψ_k are activation functions which gives the model flexibility as the types of this function are considered sigmoid (logistic) functions for this modular architecture. This choice of activation function can handle any input functions as stated in the simple feedforward setting.

The option pricing formula is considered in the same form as in Gradojevic et al. (2009)

$$C = \phi(S, X, \tau) \tag{3.32}$$

where C is the call option price, S is the price of the underlying asset, X is the strike price and τ is the time to maturity. Similarly as in Gradojevic et al. (2009), one can assume homogeneity of degree one in terms of pricing which means we can rearrange the function by normalizing the equation by X

$$\frac{C}{X} = \phi(\frac{S}{X}, 1, \tau) \implies c = \phi(s, \tau), \tag{3.33}$$

where $s := \frac{S}{X}$ and $c := \frac{C}{X}$. This is particularly useful in this thesis, as this normalization is a very useful in terms of increasing of the dataset and thus simplifying of the inference. Of course, it is questionable, whether this assumption is completely valid, as there might be psychological reasons in the market that make two situations with the same normalized variables completely different (e.g. exceeding $S = 1000$ with $X = 1000$ is much more likely to have strong resistance or support than exceeding $S = 1175$ with $X = 1175$ even though their normalized moneyness is the same $m = 1$). This is usually not treated in literature and most of authors treat either moneyness as we do, or inverse moneyness m with the assumption of homogeneity of degree one in terms of S. However, not assuming this assumption while doing option pricing with neural networks would make the inference much harder to do reliably as each strike price would need to have a separate model.

The modular neural network architecture we use in the thesis divides the options according to moneyness (3 groups) and time to maturity (3 groups) which constitutes in total 9 categories, therefore the number of trained modules is exactly the number of these categories. These modules are then filled with the moneyness $m = S/X$, time to maturity τ, and volatility σ and based on the first two variables the optimal module corresponding to the given moneyness and time to maturity is chosen. The schematic view of this process is depicted in the figure 3.1 where it is shown how input data switch the module and one of the 9 neural networks computes the output given the inputs.

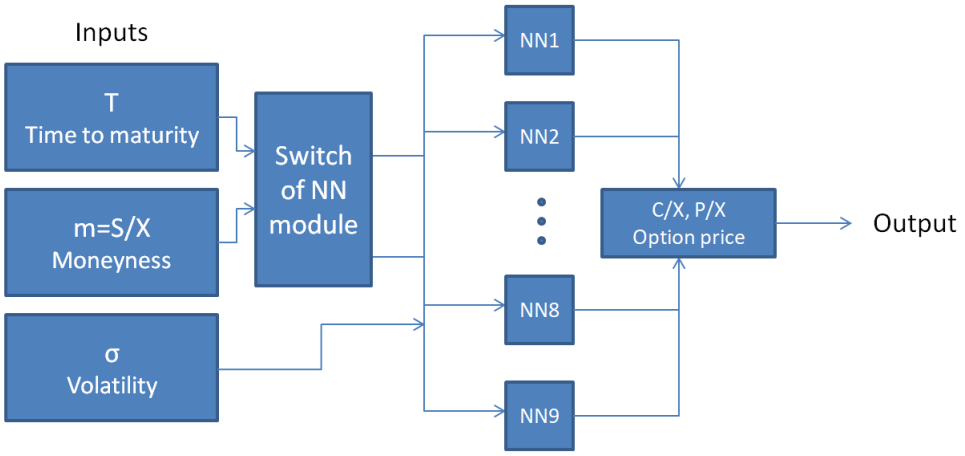


Figure 3.1: Modular network choosing modules according to moneyness m and time to maturity T

3.4.2 Other modular neural network architectures

The modular neural networks architectures were quite thoroughly investigated by many researchers in the computer science field, however, in the option pricing field there was not thoroughly explained which architecture to choose and why. In order to fulfill the requirement to have distinct modules that operate for the whole computation distinctly only few architectures match the required modularity. (Azam, 2000)

One of the first suggested architectures called decoupled modules (De Bollivier et al., 1991) uses unsupervised and supervised learning in stages proceeded sequentially. The first stage comprises of an Adaptive Resonant Theory (ART) network categorizing the input data into various clusters without supervision. In the second part the models are trained separately with supervisor according to the clusters the first stage created. The final output is then given by maximal activation function output of the separate modules.

Another option is to use so called other-output model introduced by Tsai et al. (1994). It differs from the previous architecture only by the fact the output is only binary which controls for whether the input is classified in the given region of the input space.

Next modular neural network architecture is called the hierarchical network which is quite similar to the previous models as it has two level hierarchy both with supervisor which makes clusters in the first level of hierarchy, and

the second hierarchical level makes learning and classification processes by the specialist modules.

Among several other extensions or alternatives to the mentioned models, the most widely accepted modular network architecture is called adaptive mixture of local experts (Jacobs et al., 1991). This architecture is quite simple and has an intuitive structure as it is close to the biological analogy. It draws from the Gaussian mixtures theory and local learning algorithms that try to adjust locally the capacity to the properties of the training system. This method is also used by Yang et al. (2017) which is in a slightly different way followed in this thesis as we do not use the modular network with the same architecture.

The last presented modular neural network architecture is called hierarchical mixture of experts introduced by Jordan & Jacobs (1994). It has a potential to be widely used for the classification and regression purposes such as in cases of the feedforward multi-layered artificial neural networks. The architecture is functionally derived from the divide and conquer models and it is able to perform tasks such as classification and regression trees together with multivariate adaptive regression splines. These fitting algorithms divide the input space in such a nested sequences of regions which later helps to classify and regress the data (Azam 2000)

The hierarchical mixture of experts is comprised of two kinds of modules, specialist (expert) neural networks and gating networks where the experts networks are at the bottom level of the hierarchy and divide the problem in the smaller parts for which solutions can be found. The particular sub-solutions are then joined together by so called gating networks and form the final solution. The difference between the gating and expert artificial neural network is that the gating networks transform the input values to the confidence levels for the outputs while the specialist networks use the data to find the estimate of the output value. (Azam 2000).

As shown above, there is a wide range of different modular neural networks that can be applied in various application including option pricing, and as such this thesis is able to cover only the simple modular neural network as done in Gradojevic et al. (2009). It is worth mentioning the other architectures as a possible improvement for future works.

3.5 Rational Prediction Assumptions

3.5.1 Motivation

In order to get as good neural network model as possible, it raises naturally a question whether there can be some extra information added which would improve the performance of the model. A completely general neural network model regardless of its architecture relies purely on the input data that are provided, and thus it employs the advantage of a universal approximation theorem. This theorem says that neural network with one hidden layer can approximate an arbitrary continuous function on compact subsets of real numbers given that the activation function of the neural network is continuous and differentiable. (Balász et al., 2001)

Even though there is a strong support for using simple neural networks due to this theorem, there is a need for a really large training dataset in order to ensure that the training procedure converges quickly and the function can be taught. Moreover, in option pricing, there is no guarantee that the function, that would generalize to arbitrary market conditions, even exists, as the option price relies on behavior of buyers and sellers in the markets which is limited by many factors such as trading fees, non-continuity of trading, limited supply of traders' funds, psychological phenomena such as changing behavior of traders, etc. Therefore, this thesis follows and investigates the method used by Yang et al. (2017) who applied six conditions that any meaningful option pricing model should satisfy according to Fölmer and Schmied (2004). Nevertheless, this method is used for a new choice of architecture and therefore the verification of rationality had to be derived in a completely new original way in the subsection 3.5.3.

3.5.2 Requirements for Rational Predictions

The conditions are set to meet basic assumptions about the behavior of traders on option markets. If these conditions were not met and the market instrument had enough trading volume, the no-arbitrage rule would immediately lead to a market correction that would meet the assumptions. These conditions are specified for call options as put options can be transformed to call options via put-call parity and similar inference can be done. The call option price is precisely mathematically determined for any market situation following the definition from Yang et al. (2017) as

$$\tilde{C}(S, X, \tau) = e^{-r\tau} \int_0^\infty \max(0, S_T - X) f(S_T|S_t, \tau) dS_T. \tag{3.34}$$

where r is the risk-free rate, the option price at time of expiration T is estimated at time t with the conditional risk neutral probability density function $f(x|S_t, \tau)$ and $\tau = T - t$ is the time difference. This can be understood very easily as a discounted sum of all possible profit streams of an option weighted by their probability. The integral integrates over the underlying stock price at the time of expiration which can be arbitrarily large, however, with negligible probability for unrealistic stock prices. Notice, that we denote the option price $\tilde{C}$ but the risk-free rate and discount term is obtained independently, therefore we estimate by neural network only the following price C without the discount factor

$$C(S, X, \tau) = \int_0^\infty \max(0, S_T - X) f(S_T|S_t, \tau) dS_T. \tag{3.35}$$

Hence, for call options the following six conditions from Yang et al. (2017) have to hold in order to have rational prediction assumptions

$$\frac{\partial C}{\partial X} \leqslant 0. \tag{3.36}$$

The first condition ensures a growing price with respect to the strike price given all other parameters are fixed. This seems reasonable as with increase of the strike price X, the difference from underlying asset price S can only diminish in case of in the money call option where the option price main driver is the difference $S-X$. Mathematically rigorously, if we take a derivative of (3.35), we obtain $\frac{\partial C}{\partial X} = \int_0^X f(S_T|S_t, \tau) dS_T - 1$ which actually is a cumulative distribution function $P(S_T \leqslant X)$ which cannot exceed 1 due to its statistical properties.

The second condition ensures convexity of option price with respect to strike price

$$\frac{\partial^2 C}{\partial X^2} \geqslant 0, \tag{3.37}$$

which follows directly from the derivative of the first derivative $\frac{\partial^2 C}{\partial X^2} = f(S_T|S_t, \tau) \geqslant 0$.

The third condition guarantees the longer is the time to maturity, the bigger is the probability of exceeding the strike price and therefore the option price should not decrease

$$\frac{\partial C}{\partial \tau} \geqslant 0. \tag{3.38}$$

The fourth condition says, when the exercise price is high enough, there is a zero probability for obtaining any profit from the option and thus the option price should be zero. Particularly

$$\lim_{X \to +\infty} C(S_t, X, \tau) = C(S_t, +\infty, \tau) = 0. \tag{3.39}$$

The fifth condition handles the option price at the time of exercise, at this point the price should exactly correspond to its theoretical value, which is for options typical ramp function

$$C(S, X, \tau = 0) = \max(0, S_t - X). \tag{3.40}$$

Last but not least, the sixth condition provides a lower and upper bound for the option price as it cannot exceed the underlying stock price without the violation of no-arbitrage rule, otherwise an investor would do an arbitrage by buying the underlying asset and selling the call option simultaneously and closing both of them at the time of option maturity

$$\max(0, S_t - X) \leqslant C(S_t, X, \tau) \leqslant S_t. \tag{3.41}$$

In order to have these conditions valid, Yang et al. (2017) also state assumptions that have been made. It is necessary to assume the option price function C twice differentiable with respect to X and once differentiable with respect to τ.

3.5.3 Verifying rationality

Opposing to Yang et al. (2017) this thesis uses different model for the neural network, as they use a complicated gated neural network which treats inputs individually with different activation functions and then combine the outputs by multiplication gate. Their choice which is compatible with the rationality assumptions is based on using sigmoid function $\phi_1(x) = \frac{e^x}{e^x+1}$ for the inverse moneyness $1/m$ and softplus function $\phi_2(x) = \log(1 + e^x)$ for time to maturity τ input. Their model combines both activation functions at every step and as this thesis uses completely different architectures, their verification of the

rationality conditions cannot be used and must be derived in order to match the models used in this thesis.

In option pricing literature, the sigmoid function is used very frequently for the neural network action function as it is probably the most predictable choice in machine learning. However, not many research articles regarding option pricing use the softplus activation function which has one clear advantage, its values have to be positive. This is very desirable property for the output layer of the neural network as the option price has to have a positive value naturally. Therefore, a new architecture used in this thesis is proposed to have arbitrary function in hidden layers (we choose sigmoid) and softplus function as an output.

One can argue, this is the same choice as Yang et al. (2017) made, however, there is a difference as we use sigmoid function in hidden layers and softplus function in output layer and they used in their single layer gated network both activations combined together. This provides us with nice properties of the activation functions and therefore it is not hard to verify the rationality. The neural network with one hidden layer respecting our choice of activation functions would price the call option (normalised for the strike price) as following

$$c = C/X = \phi_2(a + \sum_j w_j \phi_1(b_j + \sum_i \tilde{w}_{ij} x_i)) \tag{3.42}$$

where x_i stand for the input variables such as moneyness m, time to maturity τ, and volatility σ, a and b_j are the standard neural network bias terms, w_j and $\tilde{w}_{ij}$ are the weights for each neuron i and j. In case of more than one hidden layer, the functions would be nested which will not change the inference as all assumptions will hold trivially using chain rule for taking derivatives and because of a good choice of activation function, the derivative of an arbitrary number of nested functions does not change a sign of the result.

To verify the first assumption it is enough to consider

$$\frac{\partial C}{\partial X} = \underbrace{\frac{\partial C}{\partial c}}_{=X} \frac{\partial c}{\partial X} = X \frac{\partial c}{\partial m} \frac{\partial m}{\partial X} = \underbrace{X}_{\geqslant 0} \underbrace{\sum_j w_j \phi_1'(\cdot) \tilde{w}_{ij}}_{\geqslant 0} \underbrace{\phi_2'(\cdot)}_{\geqslant 0} \underbrace{(-\frac{S}{X^2})}_{\leqslant 0} \leqslant 0 \tag{3.43}$$

where we used chain rule and $(\cdot)$ stands for the inner function in order to make the equation simpler to read, and i stands for the index of neuron used for the moneyness input. The derivatives of activation functions are always positive

as $\phi_1'(x) = \phi_1(x)(1-\phi_1(x)) \geqslant 0$ and $\phi_2'(x) = \phi_1(x) \geqslant 0$ as $x \to \infty$.

The second assumption is a bit more tricky as it requires to calculate a second derivative of the equation 3.42. In sake of simplicity, we try to compute the result using the knowledge of the first derivative.

$$\begin{aligned}\frac{\partial^2 C}{\partial X^2} = \frac{\partial}{\partial X}(\underbrace{\frac{\partial C}{\partial c}}_{=X}\frac{\partial c}{\partial m}\underbrace{\frac{\partial m}{\partial X}}_{-\frac{S}{X^2}}) = \frac{\partial}{\partial X}(-\frac{\partial c}{\partial m}\frac{S}{X}) = \frac{\partial}{\partial m}(-\frac{\partial c}{\partial m}m)\frac{\partial m}{\partial X} \\ = (\frac{\partial c}{\partial m} + \frac{\partial^2 c}{\partial m^2}m)\frac{m^2}{S}.\end{aligned} \tag{3.44}$$

As the last step in the derivation compares a value known from the first derivative for which it was shown that is positive, it is enough to implicitly assume the biases b_j in the functional form of c such that the second derivative $\frac{\partial^2 c}{\partial m^2}$ is also positive, which can be always done by assuming the biases high enough. Then the whole term will be positive, and thus $\frac{\partial^2 C}{\partial X^2} \geqslant 0$. Due to this, this assumption imposes a condition on the structure of the bias terms that in practical computations do not have to be necessary valid. This can be mitigated by choosing more complicated architecture, however, out of scope of this thesis which would guarantee the second derivative to be always positive.

The remaining assumptions are, however, not that complicated to prove. Similarly as the verification of the first assumption, we get the validity of the third assumption,

$$\frac{\partial C}{\partial \tau} = \underbrace{\frac{\partial C}{\partial c}}_{=X}\frac{\partial c}{\partial \tau} = X\frac{\partial c}{\partial \tau} = \underbrace{X}_{\geqslant 0}\underbrace{\sum_j w_j \phi_1'(\cdot)\tilde{w}_{ij}}_{\geqslant 0}\underbrace{\phi_2'(\cdot)}_{\geqslant 0} \geqslant 0 \tag{3.45}$$

The only exception is that i stands for the time to maturity input.

The fourth assumption can be derived using the following idea. If $X \to +\infty$ then in a scenario where $C > 0$ it would automatically lead to $c = C/X \to 0$ as C is bounded by S and as c is definitely non-zero for various time to maturity τ, therefore it must hold that $C = 0$. The fifth and sixth assumptions are not easily achieved by choice of network architecture. Therefore, we generate virtual options in order to train the neural network the last two assumptions from the data. These generated options do not exist in the real data, however, their presence in the training set can learn the neural network better the upper and lower bound for the option price and if they existed in the real data they would coincide by the price due to the no-arbitrage rule. The fifth condition

is guaranteed by synthesising for all S_t from our dataset many possible strike prices X at the the moment when time to maturity is zero $\tau = 0$. The last condition is also learned from the data by generating virtual options with $X \to 0$ for many different τ, this is numerically problematic as if $X = 0$ is chosen, the moneyness m is diverging, which can solve by approximating this condition with very small nonzero X.

Chapter 4

Empirical Part

4.1 Description of data

The option dataset used in this thesis consists of both call and put European style options on cash settled SPX index (representing S&P500 index). The reason why this option index was chosen is that it belongs to the most liquid option assets in the U.S. market and is a suitable choice for the evaluation of the option price prediction. The main advantage is that this index is traded as a European option which is easier to do the valuation task on as it cannot be exercised earlier than at the time of maturity. Moreover, the index reflects the situation of the top 500 U.S. listed stocks which is broad enough to capture, at least partially, the situation of the whole worldwide market situation. This choice of index is also better than equity options traded on big companies as it is not so susceptible to the negative shocks happening in various industries and rather covers the overall market situation.

The data consist of daily data in the period from June 2004 to June 2007. The dataset originally had 491819 unique option prices comprised of both call and put options and cover 761 trading days. The dataset used in this thesis is corresponding to the data of option quotes at CBOE which are common data choice in the option pricing literature. The original dataset is drawn from the dataset used in Baruníková and Baruník (2011), who investigated the performance of the basic feedforward neural network. This gives this thesis a possibility to relate to their work, even though the measures are slightly different and on different subsets of the data and therefore are not directly comparable.

The possibility to expand the dataset by newer data was not chosen due

to the unavailability of this data in the same scale at the original source and its relatively high cost (usually between hundreds and thousands of USD) at different option data sources. Furthermore, if more data sources were combined, the data reliability might not be guaranteed to be consistent enough throughout the dataset. Moreover, if newer data were examined, there would be a possible risk of existent biases in the pricing as the 2008 global financial crisis took place in the consequent period.

The dataset consists of various data regarding daily option prices such as the daily best bid and ask price for each exercise price listed at the given date. The dataset provides for every option also the time to maturity which spans between 1 day and 3 years.

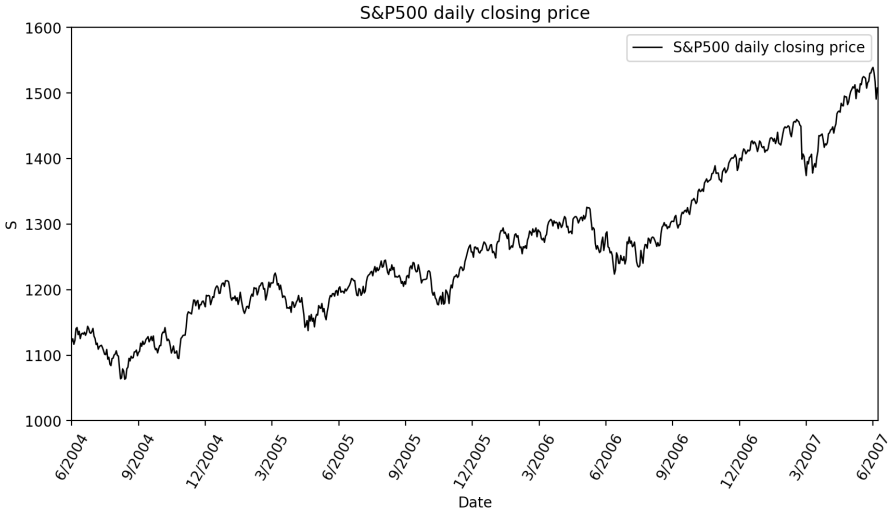
In order to get the risk-free rate, which is a necessary input to the Black-Scholes pricing model, the data from the official Federal Reserve website are gathered on the monthly yield rate of the zero-coupon Treasury bills which serve as a proxy of the risk-free rate.

As it is necessary to use only one input price in Black-Scholes formula, the price of the option is evaluated as the average value of the bid and ask price provided in the dataset.

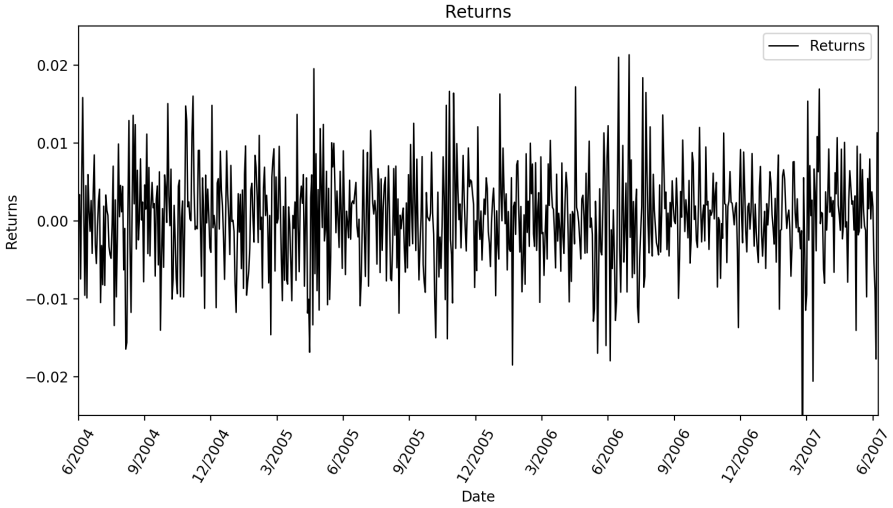
Similarly as Baruníková and Baruník (2011), we applied the exclusion filters on the options with less than six days to expiration as there might occur the liquidity related bias as stated by Bakshi et al. (1997). Moreover, the option price quotes lower than \$0.4 were not used as the effects of the discrete price could affect the overall inference. Last but not least, the assumptions from the section 3.5 were verified for each option and as there was a possibility in the market to do the arbitrage, these options were cleaned from the dataset to not bias the inference.

The resulting dataset consists of 203671 call options and 150184 put options which is a reasonably large dataset to train the various scenarios and parameters of the neural network examined in this thesis.

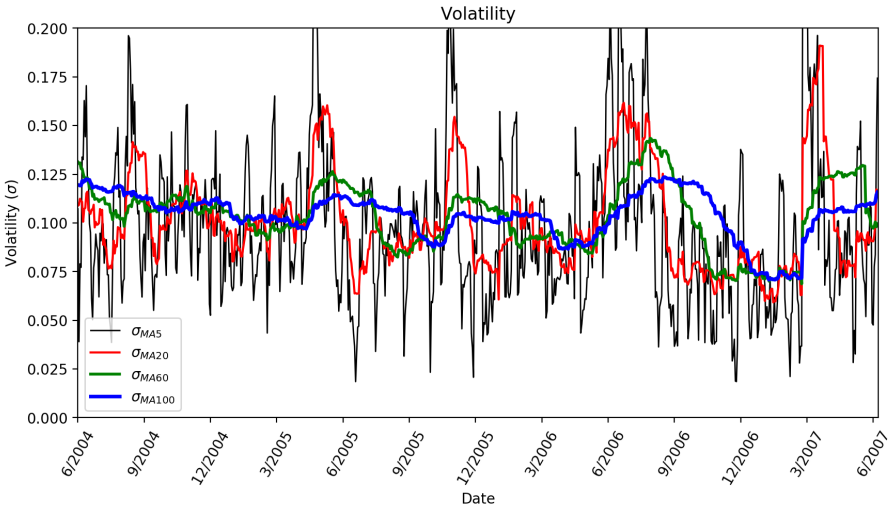
In order to enable the reader of this thesis to grasp better the characteristics of the dataset, the following figures summarizing some key dataset properties were created. The first following figure 4.1 shows three different characteristics of the used data.



(a) S&P500 daily closing prices between 2004-2007



(b) Returns on S&P500 index between 2004-2007



(c) Volatility of S&P500 index between 2004-2007

Figure 4.1: Underlying index summary statistics

The first subfigure (a) shows the closing prices of the S&P500 index in the period from June 2004 to June 2007 which is exactly matching the 761 trading days which correspond to the option dataset. The range of the underlying price starts from \$1063 per one S&P500 index share and ends at \$1539. The closing price is generally growing throughout the examined period, however, as the price is not treated as an absolute value directly and all option prices are normalized, this trend does not yield an effect that would significantly affect the inference for our further estimation as the options are compared with each other based on their other characteristics such as moneyness m and time to maturity τ which were checked to be distributed similarly across the whole dataset as there exist options with differing time to maturity and variations in moneyness on each trading day.

Based on these daily close prices, the returns were calculated according to the formula (3.7). From the subfigure (b) the reader can see the returns are generally oscillating around the mean close to 0, this mean is later revisited as the thesis discusses the GARCH(1,1) estimation method. As a measure of volatility the estimators of the historical volatility were chosen as the variance computed from the values of moving averages over four distinct periods.

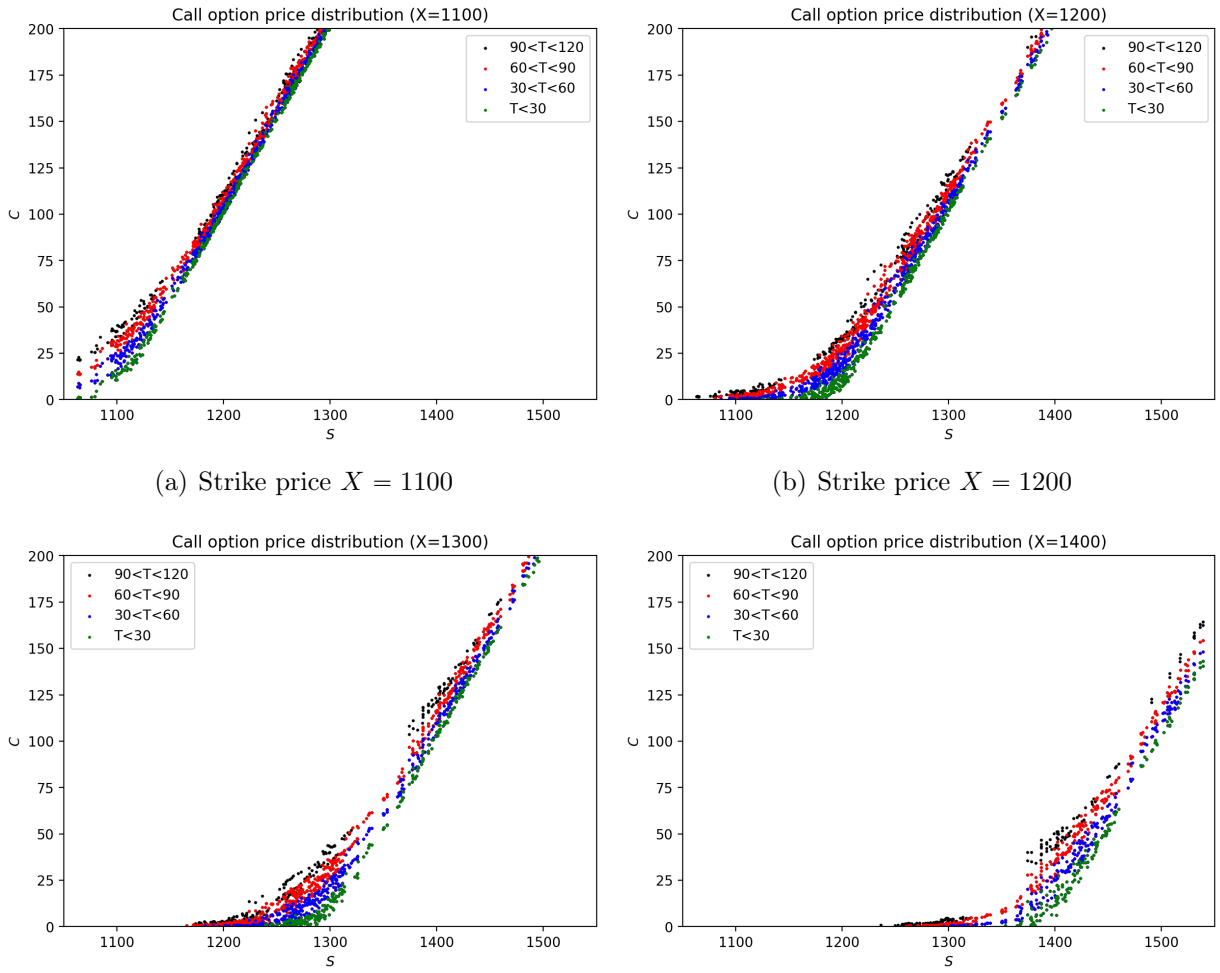
The short-run volatility is estimated according to the formula (3.8) by variance from the 5 and 20 days window moving averages over the returns and are denoted by σ_{MA5} and σ_{MA20}.

On the other hand, the long-run volatility is estimated by longer time spans of the moving window. The 60 days and 100 days are chosen as it was suggested by Hahn (2013), and thus the long-run historical volatilities are denoted as σ_{MA60} and σ_{MA100}. All of these four measures for historical volatilities are depicted in the subfigure (c) where the long-run measures are naturally lagged following the periods of higher short-run volatility.

In the following figures 4.2 and 4.3, the ramp functional form which is typical for call options and the put option equivalent is shown in the data structure. Moreover, the subfigures also depict the time value of the options as each individual colour distinguish the different time to maturity (in the following figures denoted as T). The closer the option is to expiration, the more similar it is to the theoretical shape.

In the figure 4.2, the subfigures (a),(b),(c),(d) show the data for the call option prices which have strike prices \$1100,\$1200,\$1300,\$1400, respectively. The call options in the dataset have also different strike prices with the smallest difference between two strike prices 5\$. The most observations are, however,

the options that correspond for the 25$ difference in the strike prices (i.e. 1000,1025,1050,etc.). The smallest and highest strike price in the dataset are $400 and $2000 for the call options.



(a) Strike price $X = 1100$

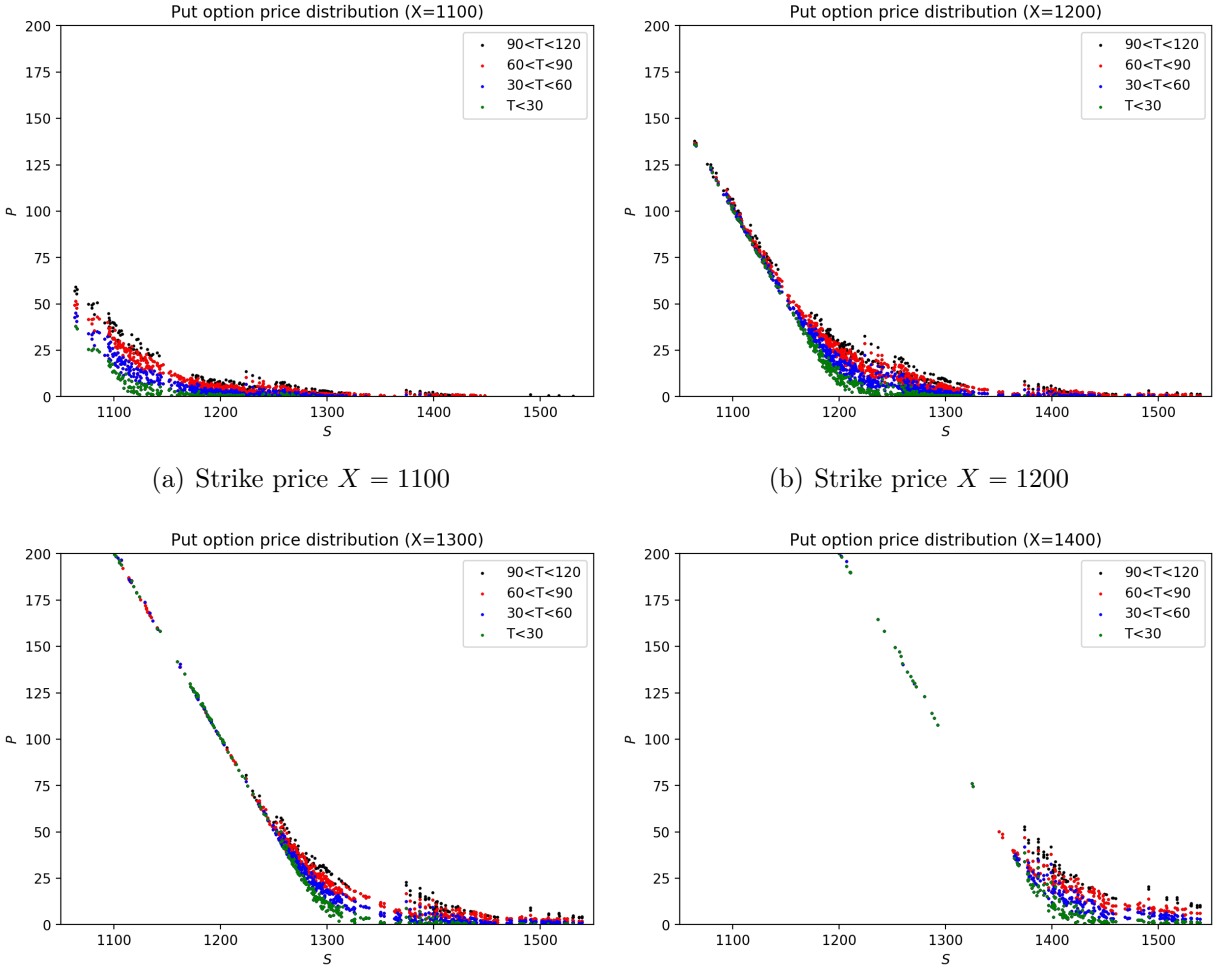
(b) Strike price $X = 1200$

(c) Strike price $X = 1300$

(d) Strike price $X = 1400$

Figure 4.2: Call options price distribution

Similarly, the put options are less traded in this period and the data are available for the trades that offer strike prices between $500 and $1700. For the most traded exercise prices the put options were depicted in the figure 4.3, where the subfigures (a),(b),(c),(d) show the data for the put option prices again with strike prices $1100, $1200, $1300, $1400, respectively. The reason we depict both, the call options and the put options for these strike prices, is to show the call and put options that were at the money (ATM) at some point during the period we study.



(a) Strike price $X = 1100$

(b) Strike price $X = 1200$

(c) Strike price $X = 1300$

(d) Strike price $X = 1400$

Figure 4.3: Put options price distribution

In both of the previous figures there were noticeable subsets of data which were richer in terms of data density and prices, for which there was smaller number of observations. This is the case partially because of the cleaning of the data which withdrew some data and thus there is a need to somehow normalize the remaining options.

As was already mentioned above, the underlying asset price is divided by the exercise price in order to get the moneyness $m = S/X$ of the option, we split the options in three subsets according to moneyness and in three subsets according to the time to maturity τ (in figures denoted T). All combinations provide us with 9 subsets for call options and 9 subsets for put options. In this thesis these 18 subsets are of high importance as they are considered as separate for the modular neural network architecture which is solved. Each of

these subsets is estimated by a different neural network module and therefore it is necessary to know the characteristics of these subsets.

Most of the options with a relatively high volume have their moneyness m in the range of 0.8-1.3 which includes over 84% of all options in our dataset. The subsets boundaries are chosen in order to have each of the modules big enough and so that the training set of the network is not too small for each distinct module. The values $m = 0.97$ and $m = 1.05$ were chosen as boundary values for the subset division, as this is done in the same way by the original paper examining modular neural networks by Gradojevic et al. (2009)

Similarly, this was done also with the division of the options according to the time to maturity τ. The division boundaries were set up at 60 days and 180 days to maturity which separates the short-term and long-term options by a reasonably wide subset of options in the middle.

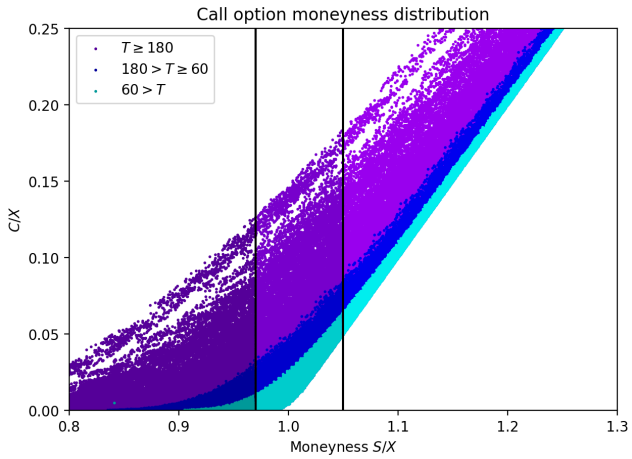
In order to provide a better view of how the options are distributed in the modules, the figure 4.4 shows both, the call option and put option distribution. The figure (a) shows a call option division into separate subsets by a vertical line (division by moneyness m) and by time to maturity τ (in figure denoted T) depicted in the figure by different colors. The figure (b) shows exactly the same division, except for that the options in the money lie at the left side of the graph from the definition of put options.

In order to show a detailed summary statistics of the individual subsets that are used in a modular neural network scenario, we use the table 4.1, which follows after the figures. It describes the size of the individual modules in terms of option data, and for each module it provides an average option premium that the options hold.

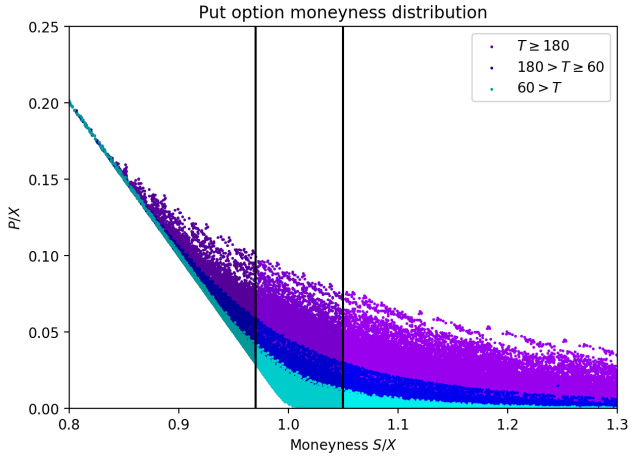
Two theoretical phenomena can be seen in the table summarizing data regarding the average option price premium. The first one shows that the time value of the option grows with the time to expiration both for calls and for puts and thus the premium grows. The second observable phenomenon is the growing price of the option as its moneyness increases. This is intuitive as the higher moneyness means the option has a higher probability of expiring in the money and therefore the price is higher.

The smallest of the subsets are the subsets with 4230 and 4120 put option observations, which are the options with short time to maturity and are in the money (ITM). The subset with the shortest time to maturity is the smallest also for call option with 6769 observations, however, these options are out of the money (OTM).

On the other hand, the biggest subset is in both call and put cases the subset with options which have moneyness $m < 0.97$ and time to maturity $\tau \geqslant 180$. This is the case because of the possibility to trade a lot of option quotes with future expiration dates greater than 180 days.



(a) Call options



(b) Put options

Figure 4.4: Options moneyness distribution

Notes: The figures show distribution of call and put options in the dataset, respectively. Two vertical lines are splitting the dataset according to moneyness (ITM < 0.97, ATM 0.97-1.05, OTM $\geqslant$1.05) and time to maturity (in figure denoted T) to 9 disjoint sets (modules) for which we perform separate inference in the thesis.

Table 4.1: S&P500 Dataset Summary

	Calls			
Days to expiration	< 60	60-180	$\geqslant 180$	All options
Moneyness (S/X)		Average Option Premium		
OTM (< 0.97)	\$2.39	\$7.52	\$30.67	\$21.05
ATM ($0.97 - 1.05$)	\$28.10	\$44.79	\$105.91	\$53.58
ITM ($\geqslant 1.05$)	\$211.08	\$264.71	\$326.38	\$275.03
Moneyness (S/X)		Observations		
OTM (< 0.97)	6,769	9,378	26,311	42,458
ATM ($0.97 - 1.05$)	25,053	12,810	14,347	52,210
ITM ($\geqslant 1.05$)	34,627	26,028	48,348	109,003
	Puts			
Days to expiration	< 60	60-180	$\geqslant 180$	All options
Moneyness (S/X)		Average Option Premium		
ITM (< 0.97)	\$84.43	\$81.66	\$101.50	\$92.42
ATM ($0.97 - 1.05$)	\$14.70	\$27.49	\$56.81	\$29.69
OTM ($\geqslant 1.05$)	\$2.12	\$5.45	\$15.96	\$10.77
Moneyness (S/X)		Observations		
ITM (< 0.97)	4,230	4,120	8,600	16,950
ATM ($0.97 - 1.05$)	24,080	12,787	14,336	51,203
OTM ($\geqslant 1.05$)	16,377	18,932	46,722	82,031

In addition, there is also a table in the appendix A.1 sumarizing the volatility prediction done for the particular subsets. The table shows for all modules the average volatility in range between 9%-15% for the historical volatility measures and slightly higher for the implied volatility measure ranging almost for all modules between 13%-21%. Implied volatility is computed using the Black-Scholes formula inversely to get the volatility from the price. This table also shows the empirical phenomenon of volatility smile described in the section 3.2.2 which is noticeable mainly in the case of put options as the options at the money (ATM) have a much lower implied volatility than the options that are OTM and ITM. In the case of call options this is not observable in the table due to the boundary choice of the modules, but the options that are deep in the money or deep out of the money show the same empirical phenomenon in the data. This only enhances the urge to value these options with a different method than by the Black-Scholes formula. In the mentioned table there are also the estimated values for the GARCH(1,1) process, which is discussed later in the results chapter.

4.2 Simulations

Even though there are various rules of thumb how to choose the parameters of neural networks in machine learning literature, the literature regarding the neural network model selection particularly in option pricing is rather rare. There is a range of complicated algorithms for automatic neural network generation, e.g. using evolutionary neural networks which takes advantage of genetic algorithms (Stathakis, 2008). However, in this thesis, we would like to adjust the parameters explicitly for the option pricing problem and propose an easily reproducible standardized approach.

Based on this, we run a series of simulations with many different sizes of network in order to find out the optimal parameters of the network. The predictive power of the neural network is tested during the process by measuring Root Mean Square Error (RMSE) and Mean Absolute Error (MAE) specifically for the option pricing problem. After the errors are calculated, the optimal parameters are chosen to minimize the errors. In order to find out what size of network is large enough to not underfit the neural network and not overfit it at the same time independently on our option dataset, we follow a way similar to the one presented in Culkin and Das (2017) which is explained in the next subsection.

4.2.1 Resolution tests

In this subsection, the main objective is to find out the ideal number of nodes and number of hidden layers in the neural network. Even though it can be proved one hidden layer network can approximate any arbitrary continuous function by the universal approximation theorem, its usually empirically preferred to use deeper neural networks in cases, where the learning task is performed for difficult object e.g. handwritten characters or face recognition, which helps the network to learn a difficult pattern more quickly (Balázs et al., 2001).

In the following tests we will try to find out what is the ideal number of hidden layers and the ideal number of nodes for the task of option pricing. Similarly to Culkin and Das (2017), we generate new data with parameters closely matching our true dataset with the exception that their price precisely matches the original Black-Scholes formula. The points are generated uniformly

randomly in range of parameters of the original option dataset. The parameters are chosen according to the table 4.2.

Parameter	Range
Stock price (S)	\$1100-\$1500
Strike price (X)	\$1000-\$1600 in \$25 steps
Maturity (T)	1 day - 1000 days
Underlying asset date	1/6/2004 - 31/5/2007 (Only trading days)
Risk free rate (r)	Based on underlying asset
Volatility (σ)	Based on underlying asset

Table 4.2: The range of parameters used to simulate call option prices

The prices of the newly generated options are consequently calculated according to the Black-Scholes formula (3.2). The generated dataset does not have to match precisely the real situation as it is used only to independently only to test how large our option dataset needs to be to predict reliably the true relationship between the option price and given parameters that is not known in the closed form. Culkin and Das (2017) use this approach only to find the optimal size of architecture that is able to learn Black-Scholes formula as the relationship for the real dataset needs to learn an unknown relationship which is, however, well enough approximated by Black-Scholes. Therefore it is assumed, the network size that is able to learn Black-Scholes formula will be able also to learn similar, yet more precise relationship based on the real option data.

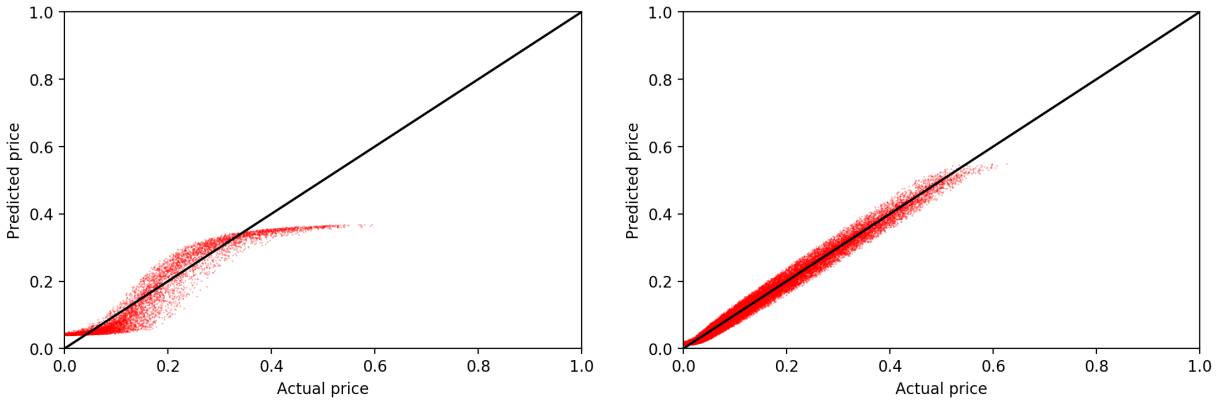
Nevertheless, in Culkin and Das (2017), the authors used this generation only to confirm, that their one particularly chosen architecture worked well, which does not contribute in any way to how to choose the ideal choice of parameters. This thesis, on the other hand, compares 360 models of different sizes of feedforward neural networks with parameters range chosen based on frequently used choices in option pricing articles with similarly sized dataset. The choice of the parameters is available in the table 4.3.

Parameter	Range
Hidden layers	1-4
Nodes in each layer	10-100
Dropout	0-0.2
Size of the training sample	10000-40000 (60% of the data)
Size of the validation sample	20% of the data
Size of the test sample	20% of the data
Number of models	360

Table 4.3: The choices of neural network size

In order to test for the effect of the parameters, we had to fix the activation function as sigmoid activation function and we fixed the softplus function at the output layer as it is proposed by this thesis as a new well performing architecture. The reason for fixing it is that even comparing 360 neural network models is computationally difficult and in practice it is not possible to test all other possible scenarios of parameters choices.

Based on these parameters we estimated option prices and compared in-sample and out-of-sample prediction power. Throughout the whole thesis, the training set, validation set and test set is split in ratio 60:20:20. The in-sample prediction is naturally very good for neural network predictions, therefore, the thesis concentrates more on the ability to generalize the relationship and out-of-sample performance. The dropout variable stands for number of nodes that are dropped from the neural network architecture when the backpropagation step is done, which is used in machine learning literature to increase the accuracy. For some choices of parameters the prediction power in out-of-sample prediction was, however, very weak whereas there were also very well matching predictions which we demonstrate in the following figure 4.5.



(a) Bad out-of-sample prediction

(b) Good out-of-sample prediction

Notes: The prediction in the left graph is generated using following parameters: 4 hidden layers, 40 nodes, 10% dropout, size of the training sample: 10000. The prediction in the right graph is generated using following parameters: 4 hidden layers, 80 nodes, 0% dropout, size of the training sample: 40000

Figure 4.5: Dependence of out-of-sample price prediction on neural network parameters

In order to evaluate how good is the out-of-sample prediction of neural network, we measure MAE and RMSE for each model and then compare the models with each other. It is worth mentioning also to measure Mean Absolute Percentage Error (MAPE) which we do not measure for these simulations as non-precise predictions for small prices makes this measure biased a lot. In order to find the effect of the model parameters an Ordinary Least Squares (OLS) regression was estimated based on the following formulas (4.1) and (4.2).

$$MAE_i = \alpha + \beta_1 hidden_i + \beta_2 nodes_i + \beta_3 size_i + \beta_4 dropout_i + \varepsilon_i \tag{4.1}$$

$$RMSE_i = \alpha + \beta_1 hidden_i + \beta_2 nodes_i + \beta_3 size_i + \beta_4 dropout_i + \varepsilon_i \tag{4.2}$$

Using Ramsey Regression Equation Specification Error Test (RESET) (Ramsey, 1969), it was found out that these models were misspecified which is also quite intuitive as it seems believable that the effect of number of hidden layers is related to number of nodes and also to the size of the training sample. Therefore we improved the model by adding some of the squared variables and mixed terms in order to make the models specified correctly.

$$\begin{aligned} MAE_i = \alpha &+ \beta_1 hidden_i + \beta_2 hidden_i^2 + \beta_3 nodes_i \\ &+ \beta_4 nodes_i^2 + \beta_5 size_i + \beta_6 size_i^2 + \beta_7 dropout_i \\ &+ \beta_8 hidden_i nodes_i + \beta_9 hidden_i size_i + \beta_{10} nodes_i size_i + \varepsilon_i. \end{aligned} \tag{4.3}$$

$$\begin{aligned} RMSE_i = \alpha &+ \beta_1 hidden_i + \beta_2 hidden_i^2 + \beta_3 nodes_i \\ &+ \beta_4 nodes_i^2 + \beta_5 size_i + \beta_6 size_i^2 + \beta_7 dropout_i \\ &+ \beta_8 hidden_i nodes_i + \beta_9 hidden_i size_i + \beta_{10} nodes_i size_i + \varepsilon_i. \end{aligned} \tag{4.4}$$

These models turned out to be not misspecified with the choice of keeping the dropout variable not squared and not in mixed terms with other variables. As the next step, we ran Breusch-Pagan test (Breusch and Pagan, 1979) which is used to test for heteroskedasticity in OLS models. The homoskedasticity was rejected in both models, we therefore estimated both regression models with robust White standard errors. The results of the models are in the tables 4.4 and 4.5.

Number of observations		R^2	
360		0.5808	
MAE	β	Robust Std. Err.	t
hidden	$-1.89 \cdot 10^{-2}$	$6.51 \cdot 10^{-3}$	-2.90**
$hidden^2$	$3.37 \cdot 10^{-3}$	$1.19 \cdot 10^{-3}$	2.74**
nodes	$-2.15 \cdot 10^{-3}$	$2.01 \cdot 10^{-4}$	-10.71***
$nodes^2$	$1.26 \cdot 10^{-5}$	$1.61 \cdot 10^{-6}$	7.83***
size	$-5.23 \cdot 10^{-6}$	$7.41 \cdot 10^{-7}$	-7.05***
$size^2$	$5.61 \cdot 10^{-11}$	$1.32 \cdot 10^{-11}$	4.25***
dropout	$1.57 \cdot 10^{-2}$	$1.51 \cdot 10^{-2}$	1.04
$hidden \cdot nodes$	$-7.63 \cdot 10^{-5}$	$3.75 \cdot 10^{-5}$	-2.04*
$hidden \cdot size$	$2.23 \cdot 10^{-7}$	$8.62 \cdot 10^{-8}$	2.59**
$nodes \cdot size$	$1.30 \cdot 10^{-8}$	$3.40 \cdot 10^{-9}$	3.82***
intercept	0.190	0.012	16.42***

** 0.05 significance level, ** 0.01 significance level, *** 0.001 significance level*

Table 4.4: OLS estimate of MAE depending on neural network parameters

Number of observations		R^2	
360		0.5851	
RMSE	β	Robust Std. Err.	t
hidden	$-2.19 \cdot 10^{-2}$	$8.17 \cdot 10^{-3}$	-2.68**
$hidden^2$	$3.91 \cdot 10^{-3}$	$1.50 \cdot 10^{-3}$	2.62**
nodes	$-2.79 \cdot 10^{-3}$	$2.53 \cdot 10^{-4}$	-11.04***
$nodes^2$	$1.64 \cdot 10^{-5}$	$2.03 \cdot 10^{-6}$	8.10***
size	$-6.55 \cdot 10^{-6}$	$9.29 \cdot 10^{-7}$	-7.05***
$size^2$	$7.10 \cdot 10^{-11}$	$1.66 \cdot 10^{-11}$	4.27***
dropout	$3.31 \cdot 10^{-2}$	$1.89 \cdot 10^{-2}$	1.75
$hidden \cdot nodes$	$-9.38 \cdot 10^{-5}$	$4.73 \cdot 10^{-5}$	-1.98*
$hidden \cdot size$	$2.59 \cdot 10^{-7}$	$1.08 \cdot 10^{-7}$	2.39*
$nodes \cdot size$	$1.70 \cdot 10^{-8}$	$4.26 \cdot 10^{-9}$	3.99***
intercept	0.237	0.015	16.24***

** 0.05 significance level, ** 0.01 significance level, *** 0.001 significance level*

Table 4.5: OLS estimate of RMSE depending on neural network parameters

Both regressions estimate very similarly defined dependent variables $RMSE$ and MAE, therefore it is not surprising the results are very similar both in terms of coefficients and also in terms of standard errors and significance. Therefore, we will analyze only table 4.4 estimating MAE. From the estimated coefficients we see all variables except *dropout* are significantly different from zero at least at 0.05 significance level and some of them are significant even at 0.001 significance level (i.e. variables *nodes*, $nodes^2$, *size*, $size^2$ and $nodes \cdot size$). The R^2 coefficient is 0.5808 which suggests the model explains the variance from the independent variables reasonably well.

The sign at quadratic terms shows convexity in variables *hidden*, *nodes*, and *size*. The optimal choice of parameters *hidden* and *nodes* can be found by finding a number of nodes and hidden layers corresponding to the minimal MAE. The results of estimation suggest by finding the minimal effect of the variable *hidden* that the number of hidden layers minimizing the error is $hidden \doteq 2.8$. Similarly, by finding the minimal effect of variable *nodes*, the optimal number of nodes is $nodes \doteq 85$. These optimal parameters are slightly influenced by the size of the training set which increases the error which is given by the multiplication terms $hidden \cdot size$ and $nodes \cdot size$.

Therefore, based on the results of this estimation we set the number of hidden layers in the future models as 2-3 and number of nodes in each layer to around 50-80 depending on the size of the training network in the network

modules used. In the choice of the size of network it is also good to think about the generally recognized empirical rules in machine learning literature which rather penalize higher number of nodes and higher number of layers. Nevertheless, the choice to fix the parameters has to be made in order to make the comparisons between different models numerically feasible as there is too many free parameters that could potentially influence the estimation.

Based on this estimation we also suggest to not use the dropout technique (which was used e.g. in Culkin and Das (2017) for the option pricing application) in the future models as it did not have a significant effect on the out-of-sample option price prediction. Moreover, the sign of the coefficient was positive which suggested the higher the dropout ratio, the higher the out-of-sample prediction error. Therefore, this technique will not be used in the following models. For the particular choice of call options the results of applying this strategy on the real dataset for different number of nodes and hidden layers are summarized in table A.2 which shows the errors are increasing with too high number of nodes as well as with not enough of them. This is shown on the real dataset which partially legitimize the simulation method we use in this chapter in order to estimate the size of the neural network.

These simulations are part of the newly developed pattern which tries to answer the question how to choose the neural networks in an optimal way so it performs well in the option pricing task. This pattern is in its complete form discussed in the results section.

4.2.2 Virtual options

According to the paper written by Yang et al. (2017), it is useful to constrain the neural networks in the option pricing task by adding several assumptions on the option price function. The mentioned authors suggested six improvements by employing assumptions from Föllmer and Schied (2004) which are required to make rational predictions in order to achieve better accuracy in option pricing using neural networks. In this thesis, a rigorous discussion regarding these assumption has been made in theoretical part (see section 3.5). Fifth and sixth assumption can be however hardly fulfilled in neural network only by choice of architecture or activation functions. The practical usage of these two assumptions is firstly done by Yang et al. (2017). In this thesis it is however identified, that the crucial assumptions of adding so called virtual options can in some cases dramatically improve the performace of the network whereas the

other suggested constraints are less important as many other authors have not violated the first three or four assumptions in their articles with the standard choice of the feedforward networks. A lot of done research fulfilled first three or four assumptions implicitly as many neural network architectures do not violate them.

Practical realization of fulfilling the fifth and sixth assumptions is done by generating new originally non-existing (virtual) options. In case of the fifth assumption these options fulfill the no-arbitrage condition when their time to maturity is zero. That means, that the shape of option prices of the generated options match typical for options ramp function (i.e. in case of calls $max(0, S - X)$). This ensures that the neural network learns where the price bottom boundary is and does not over or underestimate options with small time to maturity.

On the other hand, fulfilling of the sixth assumption is more tricky. It leads to a generation of virtual options with zero strike price, where both lower bound and upper bound for option price coincide. This means such option should cost exactly the same as the underlying instrument as there is no risk of falling below the price of the underlying asset. In other words, these virtual options are guaranteed to be the most expensive ones. Many such options can be generated as it can be generated for any arbitrary time to maturity τ and corresponding to any price of the underlying asset (which we draw from the historical range). There is a practical problem with such options, which is that options with zero strike price X have moneyness going to infinity $lim_{X->0}m = +\infty$. This problem can be, however, mitigated by supposing strike price very low but positive (in our case \$1). This is an approximation but as all other options in our dataset have an underlying asset price in range between \$1000-\$1600, this instantly generates option with very high moneyness, but that is exactly what the neural network needs to learn. This problem can be also mitigated by using inverse moneyness $1/m = X/S$ as an input in the neural networks, which was done by Yang et. al (2017). However, in this thesis the standard moneyness m is used in the way it was described before in the theoretical part as most of other research articles use it, too.

Hence, virtual options are extending the given dataset by adding new options respecting the theoretical no-arbitrage rule at the time of option expiration or simulating the most expensive option possible. Notice the difference in comparison to the previous simulations, where we generated completely new options (we did not call these virtual) which were all satisfying exactly the for-

mula of Black and Scholes. The generated options in the previous subsection cannot be added to the real dataset as the exact Black-Scholes formula does not hold precisely for the real option data. The dataset was generated only to find out what is approximately the right size of the network to learn the relationship between the input and the output which is very similar (but not identical) to Black-Scholes formula.

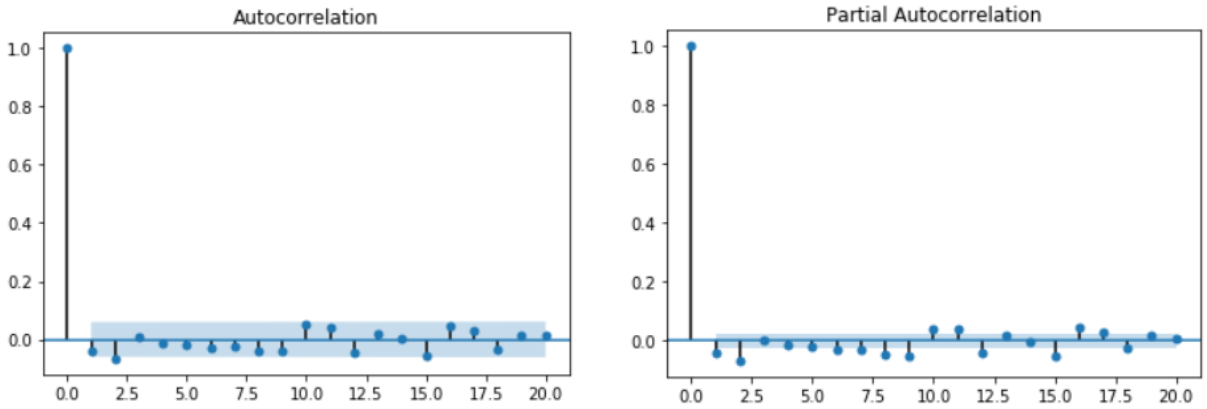
On the other hand, virtual options generated according to the mentioned assumptions, are fully compatible with the real data and can be added to the real data in the training set for the neural network in order to improve the learning process of the neural network. The results of this improved training are investigated in the next section.

4.3 Results

4.3.1 Volatility prediction

In order to estimate the volatility we use two methods in this thesis. First the historical volatility is estimated by measuring of standard deviation from the moving average over a historical window which is described both in the theoretical part and the data description part. The length of these rolling windows was chosen from short term 5 trading days, over 20 and 60 to long term 100 trading days. The neural network that learns this volatility parameter uses all four volatility measures.

On the other hand, in case of different volatility prediction mechanism using GARCH(1,1), the neural network learns only this measure which, however, proved to be for some models a better input than the historical volatility. In the following figure 4.6, there is an autocorrelation test for the log returns input which confirms that GARCH(1,1) is correctly parametrized with one lag in both parameters.



(a) Autocorrelation function (ACF) (b) Partial Autocorrelation function (PACF)

Figure 4.6: Autocorrelation function (ACF) and Partial Autocorrelation function (PACF)

The resulting parameters of the GARCH(1,1) model were estimated on the sample of 761 trading days with the following results in the table 4.6 where the parameters are denoted in the same way as in 3.16 in theoretical part.

Number of observations		761	
$\sigma_{GARCH(1,1)}$	β	Robust Std. Err.	t
$\text{mean}_{returns}$	$4.60{\cdot}10^{-4}$	$4.64{\cdot}10^{-5}$	9.93***
ω	$4.25{\cdot}10^{-6}$	$1.78{\cdot}10^{-11}$	$2.38{\cdot}10^{5}$ ***
α_1	$5.00{\cdot}10^{-2}$	$2.02{\cdot}10^{-2}$	2.48*
β_1	0.85	$2.94{\cdot}10^{-2}$	28.90***

** 0.05 significance level, ** 0.01 significance level, *** 0.001 significance level*

Table 4.6: Estimate of the GARCH(1,1) model based on the underlying asset closing price

The table shows the mean returns around which the volatility is measured are almost negligible at $4.60 \cdot 10^{-4}$. The important results from the model are its α_1 and β_1 magnitudes which show that the model is rather determined by the residual of the mean equation than by the past volatility as β_1 is 17 times higher than α_1.

The prediction of volatility by GARCH(1,1) is depicted in the figure 4.7, where it can be seen that it matches relatively well the short-term historical volatility of five day moving average σ_{MA5}.

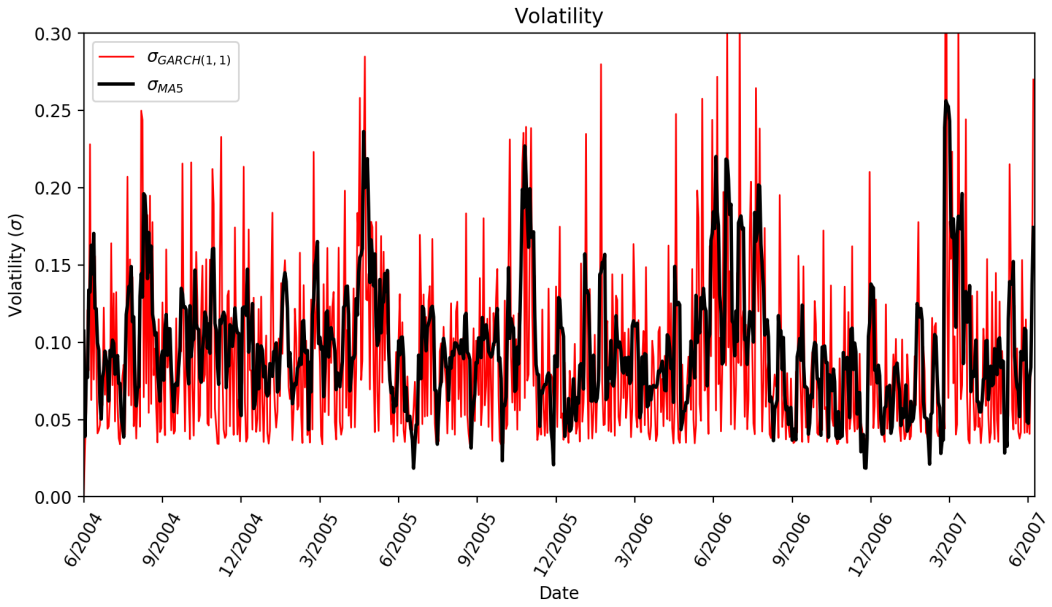


Figure 4.7: Comparison of the GARCH(1,1) volatility model with the short-run historical volatility

4.3.2 Neural networks prediction

We have compared the prediction power of various neural networks architecture both with and without using virtual options for training of the dataset. The size of training, validation and testing dataset was everywhere throughout this thesis chosen in ratio 60:20:20. We estimated three different architectures with fixed 2 hidden layers and 50 nodes in each of them. This choice of size was done after the results were obtained from the simulations section.

As activation functions were chosen either sigmoid function or special function which is inspired by Culkin and Das (2017), who used a sequence of Leaky ReLU, ELU, ReLU and ELU hidden layers (see section 3.3.4 for detailed explanation), which we test in comparison with other choices of neural network. In the following models, the usage of virtual options is also compared, which are sampled both according to the fifth and sixth assumption as described in 4.2.2.

In the table 4.7 showing the prediction on call options, the effect of improved training procedure can be seen at all three measures MAE, RMSE, and MAPE. The prediction power is evaluated by Diebold Mariano test, which is used for out-of-sample comparison of two models (in our case neural network prediction and Black-Scholes formula). The Diebold-Mariano (DM) statistics tests whether the two compared samples have equal predictive accuracy or not. The

null hypothesis of this test is that both samples have the same predictive power and therefore, if the hypothesis can be rejected, the first model outperforms the second model. For further details on this model see Diebold & Mariano (1995). In all studied cases the neural networks were much better at price prediction than Black-Scholes formula which is shown by the fact the Diebold-Mariano statistics is large and positive, which means it has significantly better out-of-sample prediction performance than in case of classical Black-Scholes formula. The significance level at which the neural network predictions are better than Black-Scholes is in all cases 0.001.

This means all of the used models are in the out-of-sample prediction outperforming Black-Scholes model and thus confirming our hypothesis #1 of better out-of-sample prediction of feedforward networks than parametric models. In case of GARCH(1,1) volatility prediction, the Diebold Mariano test has smaller but still significant value as this volatility is used also for the Black-Scholes prediction whereas in case of historical volatility the long-term σ_{MA100} was chosen for the evaluation of the Black-Scholes price.

In the following tables 4.7 for call options and 4.8 for put options, several effects can be observed. These two tables are bearing the most important results of this thesis, as they confirm two of the stated hypotheses. First, all three errors MAE, RMSE, and MAPE dramatically diminish as we increase virtual options for scenarios with softplus output function and special activation function. This confirms the hypothesis #3 as the errors are much lower using virtual options than if we do not use them.

Generally, both of these scenarios had better out-of-sample predictions for historical volatility if there were no virtual options, however, as the number of virtual options increased, the GARCH(1,1) volatility predictor started to predict the option price better than the historical volatility. This shows the hypothesis #4 is also valid that the prediction performance differs as the volatility mechanism changes. In case of put options, the historical volatility systematically outperformed GARCH(1,1) in almost all cases except for the scenario with linear output, which however proved to be the worst of the three examined models with error between 2 and 4 multiples of the other two architectures in almost all setups.

Nevertheless, for the particular choice of linear output function and sigmoid activation function in the hidden layers, the effect of adding virtual options was the highest even though it was not generally performing well. The best choice overall was a scenario which was originally suggested by this thesis which

uses softplus output function with sigmoid activation function which had the smallest error of all call options when 76100 virtual options were used to train the dataset.

activation	output	volatility	virtual	MAE	RMSE	MAPE	DM-test
sigmoid	linear	σ_{HV}	0	0.00798	0.00902	0.693	23.77***
sigmoid	softplus	σ_{HV}	0	0.00331	0.00461	0.129	39.51***
special	exp	σ_{HV}	0	0.00407	0.00583	0.119	37.25***
sigmoid	linear	σ_{HV}	7610	0.00359	0.00466	0.446	38.96***
sigmoid	softplus	σ_{HV}	7610	0.00315	0.00453	0.109	39.71***
special	exp	σ_{HV}	7610	0.00448	0.00615	0.117	34.95***
sigmoid	linear	σ_{HV}	30440	0.00513	0.00639	0.675	34.83***
sigmoid	softplus	σ_{HV}	30440	0.00307	0.00429	0.106	39.94***
special	exp	σ_{HV}	30440	0.00416	0.00639	0.171	35.82***
sigmoid	linear	σ_{HV}	76100	0.00614	0.00762	0.514	31.09***
sigmoid	softplus	σ_{HV}	76100	0.003	0.00431	0.103	40.14***
special	exp	σ_{HV}	76100	0.00322	0.00486	0.109	38.84***
sigmoid	linear	$\sigma_{GARCH(1,1)}$	0	0.00399	0.00524	0.48	8.48***
sigmoid	softplus	$\sigma_{GARCH(1,1)}$	0	0.00522	0.00734	0.147	8.15***
special	exp	$\sigma_{GARCH(1,1)}$	0	0.00429	0.00622	0.144	8.48***
sigmoid	linear	$\sigma_{GARCH(1,1)}$	7610	0.00456	0.0058	0.525	8.41***
sigmoid	softplus	$\sigma_{GARCH(1,1)}$	7610	0.00325	0.00459	0.129	8.56***
special	exp	$\sigma_{GARCH(1,1)}$	7610	0.00306	0.00464	0.128	8.7***
sigmoid	linear	$\sigma_{GARCH(1,1)}$	30440	0.00578	0.00833	0.366	7.95***
sigmoid	softplus	$\sigma_{GARCH(1,1)}$	30440	0.00342	0.00454	0.123	8.57***
special	exp	$\sigma_{GARCH(1,1)}$	30440	0.00376	0.0051	0.14	8.63***
sigmoid	linear	$\sigma_{GARCH(1,1)}$	76100	0.00302	0.00428	0.198	8.6***
sigmoid	softplus	$\sigma_{GARCH(1,1)}$	76100	0.00287	0.00432	0.089	8.74***
special	exp	$\sigma_{GARCH(1,1)}$	76100	0.00293	0.00436	0.098	8.59***

Table 4.7: Prediction accuracy comparison for different models with various number of virtual options for call options

** 0.05 significance level, ** 0.01 significance level, *** 0.001 significance level*

activation	output	volatility	virtual	MAE	RMSE	MAPE	DM-test
sigmoid	linear	σ_{HV}	0	0.0038	0.00523	0.592	78.09***
sigmoid	softplus	σ_{HV}	0	0.00262	0.00426	0.25	83.77***
special	exp	σ_{HV}	0	0.00281	0.00426	0.302	83.63***
sigmoid	linear	σ_{HV}	7610	0.00402	0.00557	0.944	75.76***
sigmoid	softplus	σ_{HV}	7610	0.00266	0.00413	0.257	83.81***
special	exp	σ_{HV}	7610	0.00277	0.00421	0.277	82.54***
sigmoid	linear	σ_{HV}	30440	0.00368	0.00519	0.792	79.76***
sigmoid	softplus	σ_{HV}	30440	0.0027	0.0041	0.286	83.72***
special	exp	σ_{HV}	30440	0.00287	0.00423	0.284	82.37***
sigmoid	linear	σ_{HV}	76100	0.00323	0.00455	0.731	82.47***
sigmoid	softplus	σ_{HV}	76100	0.00223	0.0035	0.217	84.76***
special	exp	σ_{HV}	76100	0.00273	0.00411	0.275	83.73***
sigmoid	linear	$\sigma_{GARCH(1,1)}$	0	0.00602	0.00748	1.2	8.12***
sigmoid	softplus	$\sigma_{GARCH(1,1)}$	0	0.00297	0.00501	0.296	8.41***
special	exp	$\sigma_{GARCH(1,1)}$	0	0.00346	0.00503	0.345	8.15***
sigmoid	linear	$\sigma_{GARCH(1,1)}$	7610	0.00556	0.0077	1.251	8.09***
sigmoid	softplus	$\sigma_{GARCH(1,1)}$	7610	0.00304	0.00466	0.323	8.45***
special	exp	$\sigma_{GARCH(1,1)}$	7610	0.003	0.00458	0.345	8.18***
sigmoid	linear	$\sigma_{GARCH(1,1)}$	30440	0.00399	0.00545	0.73	8.37***
sigmoid	softplus	$\sigma_{GARCH(1,1)}$	30440	0.00282	0.00449	0.275	8.43***
special	exp	$\sigma_{GARCH(1,1)}$	30440	0.00298	0.00479	0.309	8.19***
sigmoid	linear	$\sigma_{GARCH(1,1)}$	76100	0.00346	0.00519	0.566	8.4***
sigmoid	softplus	$\sigma_{GARCH(1,1)}$	76100	0.00273	0.00437	0.251	8.46***
special	exp	$\sigma_{GARCH(1,1)}$	76100	0.00281	0.00449	0.257	8.2***

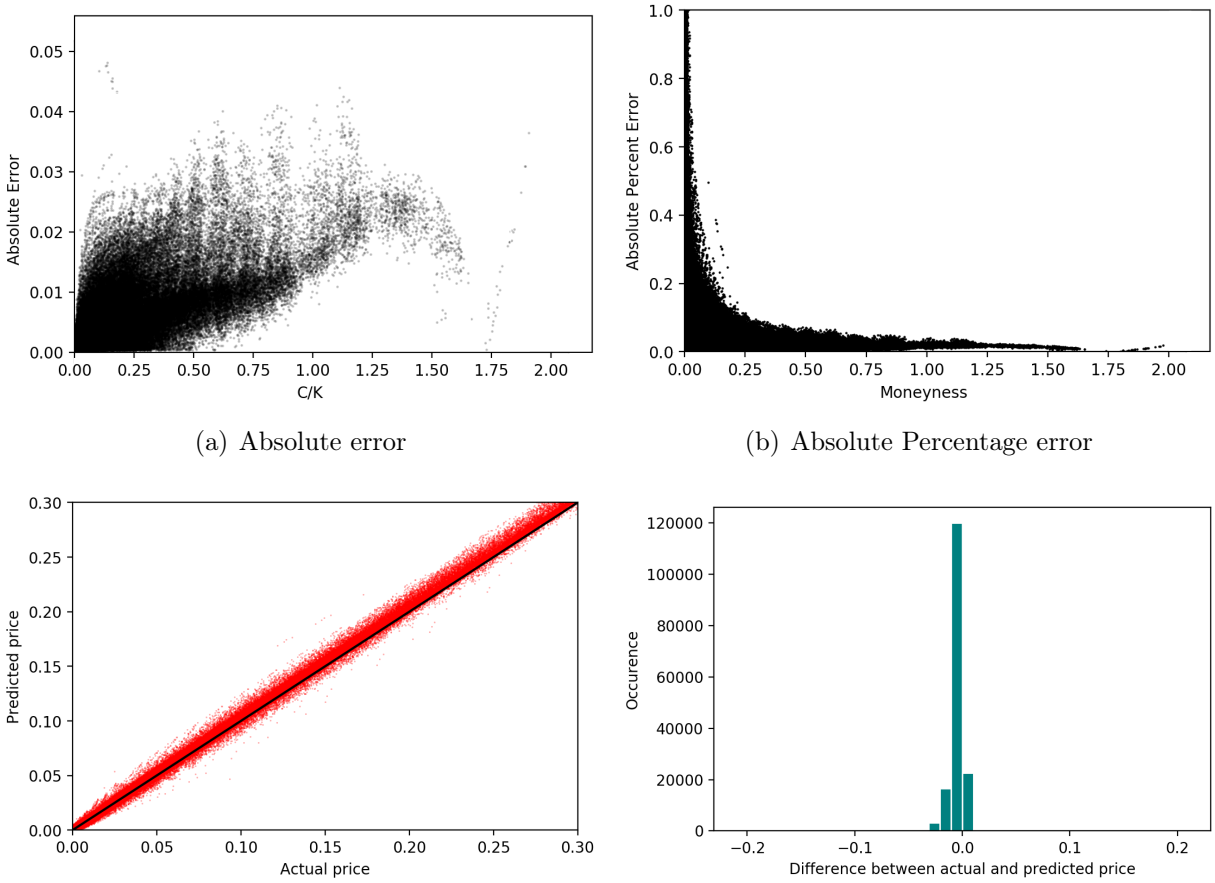
Table 4.8: Prediction accuracy comparison for different models with various number of virtual options for put options

** 0.05 significance level, ** 0.01 significance level, *** 0.001 significance level*

In order to demonstrate the precision of the prediction by our choices of architecture, we depict the measures of prediction in the following figure 4.8 for a softplus output and sigmoid activation function in the hidden layers. We depict the same measures also for the special choice of activation functions from the paper by Culkin and Das (2017), which were in the previous tables denoted as "special", in figure 4.9. For both of the figures the historical volatility was used. The graphical form of the figures is shown in the similar way as in the mentioned paper.

Both of the figures demonstrate that the neural network is very successful in the learning of the correct functional form and that its predicted price nearly matches the true price observed in the market data. Both of the models in figures are shown for call option without the usage of virtual options (which even improve the performance). Four subfigures are depicted in both of these

figures which show the absolute error and absolute percentage error used for the MAE and MAPE estimation. The options have generally high absolute percentage error for small moneyness, as the small differences from the correct pricing result in high percentage error. Last subfigure in both figures shows the histogram of the difference between prediction and real price.



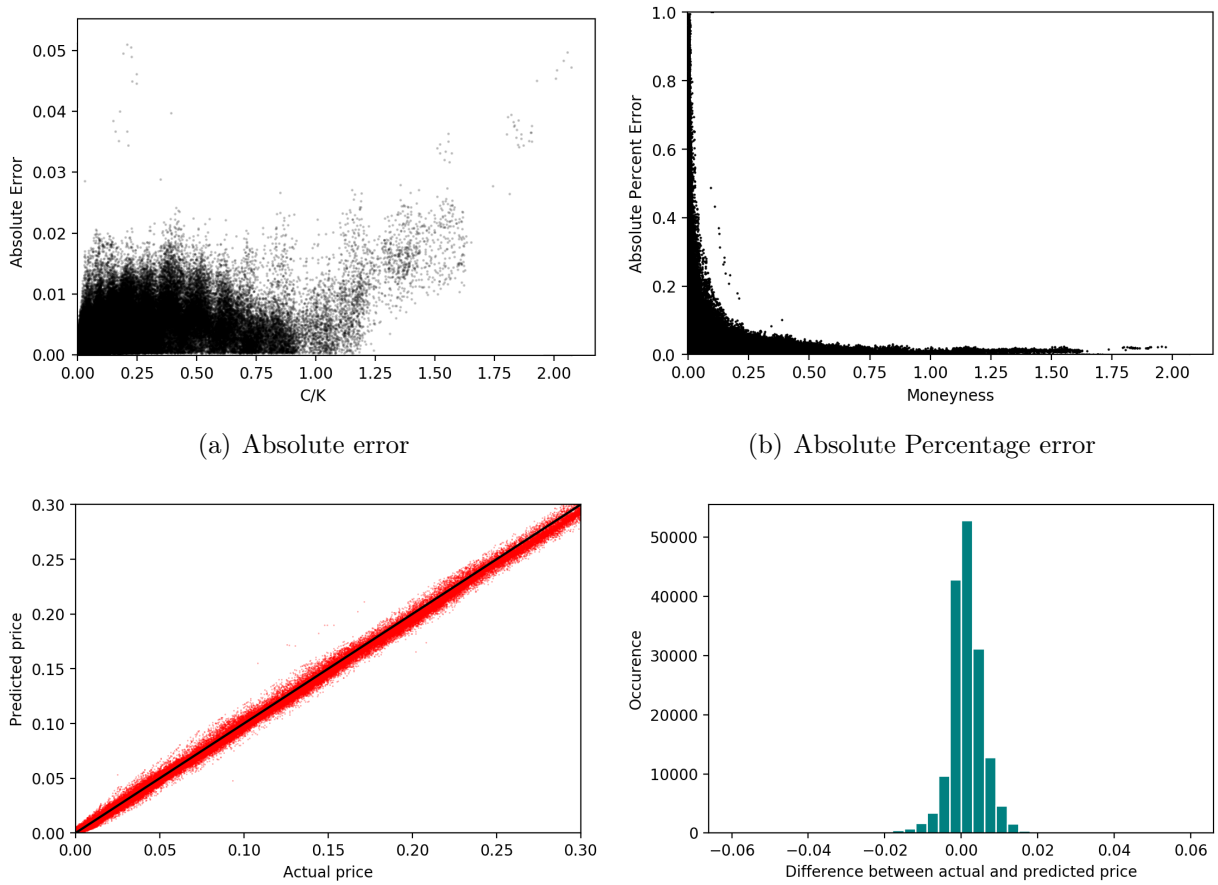
(a) Absolute error

(b) Absolute Percentage error

(c) Prediction precision

(d) Histogram of difference

Figure 4.8: Call options price prediction for network with softplus output function without the usage of virtual options



(a) Absolute error

(b) Absolute Percentage error

(c) Prediction precision

(d) Histogram of difference

Figure 4.9: Call options price prediction for network with special activation functions without the usage of virtual options

4.3.3 Modular neural networks prediction

In the previous subsection the thesis compared three different choices of feedforward neural network with the usage of two volatility measures. In this subsection, the thesis compares the feedforward architecture with the modular neural network architecture as was described in section 3.4. The modular neural network split the input data to separate subsets according to moneyness m and time to maturity τ. This separation is done to separate the input data to 9 subsets defined by the three choices of moneyness $m < 0.97$,$0.97 \leqslant m < 1.05$, $1.05 \leqslant m$, and by three choices of time to maturity $\tau < 60$, $60 \leqslant \tau < 180$, $180 \leqslant \tau$.

The neural network assessed the pattern in the data for each subset separately and learned for all 9 subsets different prediction mechanism. Each of these subsets was processed by a different module, which is basically a standalone neural network with the choice of 30 nodes in two hidden layers (slightly lower to not overfit the neural network as the subsets have smaller size). After the all modules of the network are trained, the test data comes to the whole network which switches the module so the right evaluation module is chosen which outputs the option price. The whole architecture of the modular neural network is depicted in the theoretical part in figure 3.1.

The results are compared according to the same measures as in the previous sections, i.e. MAE, RMSE, MAPE, and prediction power in comparison to the classical Black-Scholes formula by Diebold-Mariano statistics. In the table 4.9 there is an evaluation of the modular neural network architecture in comparison to the previously favored simple feedforward architecture with sigmoid activation function in the hidden layers and softplus function at the output. It can be seen, the out-of-sample prediction in comparison to the Black-Scholes formula is slightly better as the Diebold-Mariano test shows higher value. Moreover it also outperforms the mentioned feedforward neural network in all three error measures. This confirms the hypothesis #2 which said that the performance of the modular neural network will be better than the performance of the simple feedforward network.

Nevertheless, it is not better than the feedforward neural network in terms of MAE and RMSE if we compare the modular neural network to the case where 76100 virtual options were added. The mentioned table leads to the conclusion, that if we add small amount of virtual options, the modular neural network still outperforms the feedforward neural network, however, if the amount of

virtual options added is high enough as 76100 (which stands for over 46% of the original call option dataset, and over 63% of the put option dataset), the prediction has lower error measures for the simpler architecture with added virtual options rather than modular neural network.

Nonetheless, it is worth mentioning both models have very good prediction power and there is also a possibility to merge these two approaches into the one which would generate virtual options in some particular models and is suggested as a future extension of the topic.

option	architecture	MAE	RMSE	MAPE	DM-test	virtual	size
Call	modular	0.00308	0.00446	0.085	41.03***	0	162939
Call	softplus	0.00331	0.00461	0.129	39.51***	0	162939
Call	softplus	0.00315	0.00453	0.109	39.71***	7610	170549
Call	softplus	0.00307	0.00429	0.106	39.94***	30440	193379
Call	softplus	0.00300	0.00431	0.103	40.14***	76100	239039
Put	modular	0.00249	0.00400	0.229	84.50***	0	120149
Put	softplus	0.00262	0.00426	0.25	83.77***	0	120149
Put	softplus	0.00266	0.00413	0.257	83.81***	7610	127759
Put	softplus	0.00270	0.00410	0.286	83.72***	30440	150589
Put	softplus	0.00223	0.00350	0.217	84.76***	76100	196249

** 0.05 significance level, ** 0.01 significance level, *** 0.001 significance level*

Table 4.9: Prediction accuracy comparison of modular neural network to the simple feedforward network fulfilling rational prediction assumptions

In the following table 4.10, there is a performance of the individual modules in the modular neural network estimation. The modules are sorted according to moneyness (first three modules $m < 0.97$, the middle three modules $0.97 \leqslant m < 1.05$,last three modules $1.05 \leqslant m$) and also in the each triplet of modules, they are sorted by time to maturity in ascending order (i.e. module 1 has $\tau < 60$, module 2 has $60 \leqslant \tau < 180$, etc.).

As it can be seen in the table, modules that are trained over small sized samples are relatively poorly performing in comparison to those with bigger sample. These are mostly first three modules 1-3 for both put and call option cases. The higher the moneyness is however, the bigger is also the training sample in the module and in the end results in far better out-of-sample prediction for modules 7-9 which outperform even the best scenarios of the simple feedforward network.

As the sample size for the poorly predicting modules is really small, the

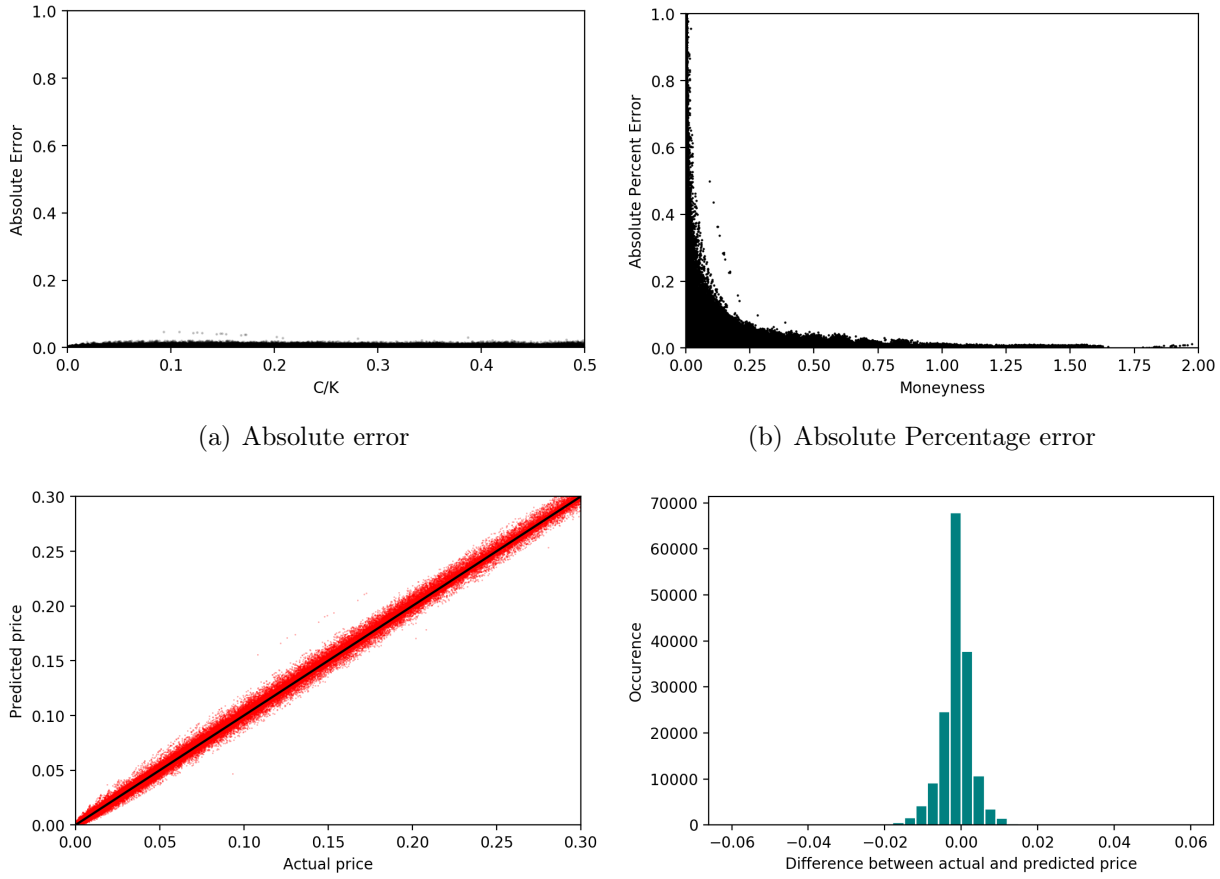
overall result in the prediction power is weighted more by very precisely predicted modules 7-9. Hence, as a whole modular neural network this architecture performs better than the non-modular equivalent which was compared and discussed in the table 4.9. The prediction power of the whole modular neural network is also depicted in the figure 4.10. In the figure it is shown, it matches the predicted price very accurately as was discussed based on the results from the table.

option style	module	MAE	RMSE	MAPE	DM-test	training size
Call	module 1	0.009	0.00915	0.523	7.48***	5415
Call	module 2	0.00438	0.00536	0.182	9.04***	7502
Call	module 3	0.0026	0.00347	0.175	11.03***	21048
Call	module 4	0.01377	0.01609	0.235	8.02***	20042
Call	module 5	0.00142	0.00168	0.009	42.8***	10248
Call	module 6	0.006	0.00645	0.204	32.8***	11477
Call	module 7	0.00183	0.00242	0.013	49.4***	27701
Call	module 8	0.00254	0.00313	0.013	46.59***	20822
Call	module 9	0.00312	0.00337	0.013	43.75***	38678
Put	module 1	0.02775	0.03723	0.441	4.64***	3384
Put	module 2	0.02186	0.02706	0.374	5.24***	3296
Put	module 3	0.01663	0.01989	0.213	6.96***	6880
Put	module 4	0.00602	0.00646	0.193	33.25***	19264
Put	module 5	0.00625	0.00685	0.185	34.41***	10229
Put	module 6	0.00617	0.00705	0.172	46.28***	11468
Put	module 7	0.00133	0.0016	0.126	89.82***	13101
Put	module 8	0.00352	0.00447	0.152	78.9***	15145
Put	module 9	0.00281	0.00411	0.185	85.68***	37377

** 0.05 significance level, ** 0.01 significance level, *** 0.001 significance level*

Table 4.10: Prediction accuracy for particular modules in the modular neural network

Notes: For each module a sigmoid activation in 2 hidden layers was chosen, and softplus function at the output of modules, considering 30 nodes in each layer to not overfit the modules)



(a) Absolute error

(b) Absolute Percentage error

(c) Prediction precision

(d) Histogram of difference

Figure 4.10: Prediction power estimation for modular neural network with two hidden layers with 30 nodes in each module

4.3.4 The proposed pattern for the choice of architecture

One of the goals of this thesis is to find out how to estimate the right size and the correct functional form of the neural network. This is urgently yearned for by the existing literature as the majority of research articles regarding the option pricing focused more on the comparison of the parametric and non-parametric methods than on the choice of the neural networks parameters. This thesis follows the methods done in several recent papers on the matter, and proposes to set up a pattern which is based on the findings stemming from this thesis.

As the number of parameters that can be adjusted in case of neural network prediction is relatively large, the proposed method in the choice of the optimal parameters is to split the architecture choice to two major steps and then do several adjustments.

The first big step in this pattern is to evaluate the right size of the network,

which can be done either by using some of the empirical rules of thumbs used in machine learning or a new tailor-made empirical rule can be estimated by the method we used in the section 4.2.1. This method simulates completely new options that corresponded to the real dataset only by the choice of the underlying asset parameters, but nothing from the real option dataset regarding the price of the option was used. The price of the generated options is calculated by the Black-Scholes formula.

The next step is to estimate a number of neural network models with an architecture of the reader's choice but varying number of hidden layers and nodes in each layer. We evaluated also the dropout technique which drops a part of the nodes that have little or no weights after the backpropagation. The following step is to run an OLS regression of the out-of-sample prediction error on the neural network parameters which evaluates the effect of adding hidden layers or number of neurons in each layer. It is useful to run several models with different sizes of training samples as with increasing training sample the error tends to diminish.

In our case, the OLS estimation led us to the omitting of the dropout method as its effect was not significant. Consequently, the error-minimizing number of nodes and hidden layers was estimated as the OLS regression had specified also quadratic terms which enables to find the parameter value with the lowest effect on the error.

The second big step in this pattern is to choose the right activation function and the output function. This partially depends on the answer the reader obtains from the first step, and offers a large amount of possibilities how to choose the optimal activation function. The results of this thesis analyzed three different choices and it showed, that it is much better to use softplus function as the output function instead of the linear output. The performance of the softplus function is shown to be the best in this thesis, however, only slightly better and overall similarly successful to the architecture chosen by Culkin and Das (2017), who used the exponential output function and a special choice of activation functions for each layer (see tables 4.7 and 4.8, where we denote this architecture "special"),

The important step in choosing of the activation and output function is to find the functional form of activation function which respects the rational prediction assumptions mentioned by Yang et al. (2017). These assumptions were individually verified in section 3.5 of this thesis for our architecture choice. This procedure includes the generation of virtual options in order to fulfill two

of these assumptions. Such a technique proved to be very effective and overall improved the performance of all models with different activation functions and outputs. Another possibility is to impose a modular structure, which also improved the performance of the model we used comparably with the adding of the virtual options.

One of the possible improvements to this pattern is to find a way to combine the empirical rule with the rules standardly used in machine learning. Another might be to combine the usage of modular neural networks with the fulfillment of rational assumptions which would result in improved performance in some modules by the presence of virtual options in the data subsets. This has not been examined by this thesis but surely would yield interesting results.

4.3.5 Discussion of the results

The results were mostly discussed individually throughout the whole empirical chapter, however, this subsection aims to summarize and simplify the comparisons done in this thesis. This thesis had several goals which were described by the hypotheses which were stated in the introduction chapter. Moreover, the thesis proposed a new way, how to choice the size and the architecture of the network which was discussed in the previous subsection 4.3.4.

The first hypothesis was shown true in all tests that have been done with any architecture of the neural network that was tested in the thesis. The out-of-sample prediction power was measured by the Diebold-Mariano test and this measure proved the neural network price prediction to be significantly better than Black-Scholes formula.

The second hypothesis was confirmed in the subsection 4.3.3 where it was shown, the modular neural network with the same parameters as the feed-forward neural network outperformed the latter in case there were no virtual options added. The addition of virtual options improved slightly the performance of the feedforward neural network and then it became more accurate even than the modular neural network.

The third hypothesis examined whether the fulfilling of the rational prediction assumptions improve the network. It was shown in the theoretical part in section 3.5, that the choice of architecture with the softplus function at the output and sigmoid activation function in the hidden layers satisfies at least 5 out of 6 rational prediction assumptions given that the virtual options of both types are generated.

This is compared in the results section 4.3.2 where the virtual options are added and two other setups, which were not guaranteed to match the assumptions, were compared to the mentioned architecture with the softplus function at the output. The third hypothesis was shown correct as the virtual options were improving the performance by at least 10% in terms of MAE and RMSE. Moreover, the softplus function was better than the other choices for which the assumptions were not verified and were highly likely to not satisfy them (e.g. the "special" choice of activation functions) Due to the results that the addition of virtual options, and having a modular neural network structure have improving effect on the option price prediction, it is suggested to combine these two approaches in the future research.

The last fourth hypothesis is also shown to be valid by comparing two volatility forecast methods. The historical volatility which was based on the moving window of historical prices of 5,20,60, or 100 trading days was shown it can be in several cases outperformed by the GARCH(1,1) method. However, none of the volatility forecast method showed to be dominantly better for all types of options. In case of put options the price prediction using historical volatility had overall lower error in terms of all three measures MAE, RMSE, and MAPE.

In the end, the thesis formulated the pattern in the previous subsection, which shows it is reasonable to run several testing simulations with generated data matching Black-Scholes formula, that evaluates the ideal size of the neural network architecture in option pricing. This resolution test was not done in case of modular neural network, which might be also considered as an expansion of this work to make the pattern more reliable. After the estimation of the optimal size, the several activation function and output function choices were compared on the real data with fixed number of hidden layers and nodes. This approach made the whole neural network estimation significantly easier in terms of computational power as there is significantly smaller number of possible choices. Even though this approach simplified the choice of the architecture significantly, there is still a lot of possible choices as only negligible small subset of all architecture choices was investigated in this thesis.

Chapter 5

Conclusion

It has been more than 40 years since the famous option pricing formula was developed by Black and Scholes. Many improvements to their formula have been done by both the academic researchers and the practitioners since then. One of the critical point for the option pricing theory came with the release of the article with a new approach taken by Hutchinson et al. (1994) who used a neural network method to predict the option price.

This method was successfully followed by many other researchers as the neural networks are able to very accurately reconstruct any arbitrary continuous function only from the structure of the data they are given. A lot of researchers interested in option pricing compared the results of the classical Black-Scholes formula and the new non-parametric approach using neural networks for various types of options and different datasets.

However, the current literature on this topic still does not have clear answers on the questions related to the machine learning problems such as the optimal choice of architecture and size of the neural network. There are also various types of volatility which can be used as a prediction mechanism that plays a key role in the option pricing task.

As it was shown by a comprehensive amount of literature that in most cases the neural networks are better prediction mechanism than the closed-form Black-Scholes formula, it is very important to have this approach as precise as possible in order to find out the true price of options.

This leads this thesis to examine methods that are tested with a specific goal to increase the prediction power of the neural networks. This is done by investigation of several improvements to the simplest feedforward network setup.

The contribution of this thesis includes the following results. First, a new pattern in the choice of the neural network size and architecture was proposed based on the results of this thesis. This pattern enables the future researchers to evaluate how big the architecture should be in order to minimize the error of the price prediction.

Second, it was confirmed that all choices of the neural network architecture done in this thesis led to increased prediction power and thus better price prediction. Moreover, the specific contribution is in the evaluation of the effect of adding rational prediction assumption to the model which enforces the neural network to learn some key theoretical concepts from option pricing theory, and thus has better performance in the pricing task. There is an added value in verification of the assumptions for the newly chosen architecture choice assuming the softplus output function in the last layer of the neural network. To comply with the assumptions, it is necessary to add virtual options that are matching some of these assumptions and their presence in the model increased the performance for various architectures significantly.

Third, a concept of modular neural networks was revisited and shown that this approach leads to better out-of-sample prediction than the simple feedforward networks. This is shown to be comparably good with the addition of virtual options.

Last but not least, it is shown that the performance of the neural network can vary with respect to the used volatility prediction mechanism. In this thesis, the historical volatility and GARCH estimation method were used and it was shown that even though both mechanism yield generally better results for feedforward networks than for Black-Scholes formula, none of the used volatilities is significantly dominant over the another, and thus in some of the investigated models it is better to use historical volatility whereas in other models use GARCH volatility estimation.

Even though many possible improvements to the simple neural networks were examined by this thesis, there is still a great amount of work ahead in the option pricing in terms of using more advanced gated neural networks and advanced modular neural network architectures which were only mentioned by the thesis. One of the directly suggested improvements corresponding to the results of this thesis is to combine the approach of generating of virtual options with the modularity, which would imply some of the modules in the modular neural network would have much better prediction power.

Moreover, as every module in the modular neural network estimated its

own prediction mechanism, each two neighbouring modules should satisfy that the prediction at their boundaries is in alignment with the prediction of the opposite module. This seems as a natural condition that should be matched in case of modular neural networks and could find an answer in the theory of finite element method widely used in the partial differential equations modelling. In addition, some authors suggest to use a different class of genetic algorithms to perform the option pricing instead of the choices we made in this thesis.

There also exists a lot of various volatility prediction mechanisms some of which were summarized in the theoretical part of this thesis, however, this thesis investigated this topic only slightly and therefore the reader is encouraged to use more sophisticated models in combination with the other concepts used in this thesis in the future works.

To sum up, several methods were thoroughly examined by this thesis, however, the option pricing field can be still investigated in much deeper way with the use of more advanced neural networks and different time frames of the data.

Bibliography

Azam, F. (2000): *Biologically inspired modular neural networks.* Ph.D. thesis, Virginia Tech.

Bakshi, G., C. Cao, & Z. Chen (1997): "Empirical performance of alternative option pricing models." *The Journal of finance* **52(5)**: pp. 2003–2049.

Baruník, J. & M. Hlínková (2016): "Revisiting the long memory dynamics of the implied–realized volatility relationship: New evidence from the wavelet regression." *Economic Modelling* **54**: pp. 503–514.

Baruníkova, M. & J. Baruník (2011): "Neural networks as a semiparametric option pricing tool." .

Bates, D. S. (2003): "Empirical option pricing: A retrospection." *Journal of Econometrics* **116(1-2)**: pp. 387–404.

Berkowitz, J. (2010): "On justifications for the ad hoc black-scholes method of option pricing." *Studies in Nonlinear Dynamics & Econometrics* **14(1)**.

Black, F. & M. Scholes (1973): "The pricing of options and corporate liabilities." *Journal of political economy* **81(3)**: pp. 637–654.

Bollerslev, T. (1986): "Generalized autoregressive conditional heteroskedasticity." *Journal of econometrics* **31(3)**: pp. 307–327.

Boyle, P. P. (1977): "Options: A monte carlo approach." *Journal of financial economics* **4(3)**: pp. 323–338.

Bridle, J. S. (1990): "Probabilistic interpretation of feedforward classification network outputs, with relationships to statistical pattern recognition." In "Neurocomputing," pp. 227–236. Springer.

Carr, P., H. Geman, D. B. Madan, & M. Yor (2002): "The fine structure of asset returns: An empirical investigation." *The Journal of Business* **75(2)**: pp. 305–332.

Carr, P. & D. Madan (1999): "Option valuation using the fast fourier transform." *Journal of computational finance* **2(4)**: pp. 61–73.

Christoffersen, P. & K. Jacobs (2004): "Which garch model for option valuation?" *Management science* **50(9)**: pp. 1204–1221.

Cofiño, A. S., J. M. Gutiérrez, & M. L. Ivanissevich (2004): "Evolving modular networks with genetic algorithms: application to nonlinear time series." *Expert Systems* **21(4)**: pp. 208–216.

Culkin, R. & S. R. Das (2017): "Machine learning in finance: The case of deep learning for option pricing." *Journal of Investment Management* **15(4)**: pp. 92–100.

Diebold, F. X. & R. S. Mariano (1995): "Comparing predictive accuracy." *Journal of Business & Economic Statistics* pp. 253–263.

Dugas, C., Y. Bengio, F. Bélisle, C. Nadeau, & R. Garcia (2001): "Incorporating second-order functional knowledge for better option pricing." In "Advances in neural information processing systems," pp. 472–478.

Engle, R. F. (1982): "Autoregressive conditional heteroscedasticity with estimates of the variance of united kingdom inflation." *Econometrica: Journal of the Econometric Society* pp. 987–1007.

Evans, L. C. (2010): "Partial differential equations." .

Figlewski, S. (1997): "Forecasting volatility." *Financial markets, institutions & instruments* **6(1)**: pp. 1–88.

Garcia, R. & R. Gençay (2000): "Pricing and hedging derivative securities with neural networks and a homogeneity hint." *Journal of Econometrics* **94(1-2)**: pp. 93–115.

Geigle, D. S. (1999): *An artificial neural network approach to the valuation of options and forecasting of volatility.* Nova Southeastern University.

Ghaziri, H., S. Elfakhani, & J. Assi (2000): "Neural, netvvorks approach to pricinc, options." *Neural Network World* **1(2/00)**: pp. 271–277.

Glorot, X., A. Bordes, & Y. Bengio (2011): "Deep sparse rectifier neural networks." In "Proceedings of the fourteenth international conference on artificial intelligence and statistics," pp. 315–323.

Gradojevic, N., R. Gençay, & D. Kukolj (2009): "Option pricing with modular neural networks." *IEEE transactions on neural networks* **20(4)**: pp. 626–637.

Hahn, T. (2013): "Option pricing using artificial neural networks: an australian perspective." .

Haug, E. G. (2007): *The complete guide to option pricing formulas*, volume 2. McGraw-Hill New York.

Heston, S. L. (1993): "A closed-form solution for options with stochastic volatility with applications to bond and currency options." *The review of financial studies* **6(2)**: pp. 327–343.

Heston, S. L. & S. Nandi (2000): "A closed-form garch option valuation model." *The review of financial studies* **13(3)**: pp. 585–625.

Hornik, K. (1991): "Approximation capabilities of multilayer feedforward networks." *Neural networks* **4(2)**: pp. 251–257.

Hutchinson, J. M., A. W. Lo, & T. Poggio (1994): "A nonparametric approach to pricing and hedging derivative securities via learning networks." *The Journal of Finance* **49(3)**: pp. 851–889.

Jacobs, R. A., M. I. Jordan, S. J. Nowlan, & G. E. Hinton (1991): "Adaptive mixtures of local experts." *Neural computation* **3(1)**: pp. 79–87.

Jang, J.-S. R., C.-T. Sun, & E. Mizutani (1997): "Neuro-fuzzy and soft computing; a computational approach to learning and machine intelligence." .

Jordan, M. I. & R. A. Jacobs (1994): "Hierarchical mixtures of experts and the em algorithm." *Neural computation* **6(2)**: pp. 181–214.

Kou, S. G. (2002): "A jump-diffusion model for option pricing." *Management science* **48(8)**: pp. 1086–1101.

Lajbcygier, P., C. Boek, A. Flitman, & M. Palaniswami (1996): "Comparing conventional and artificial neural network models for the pricing of options on futures." *Neurovest Journali* **4(5)**: pp. 16–24.

MacKenzie, D. (2006): "Is economics performative? option theory and the construction of derivatives markets." *Journal of the history of economic thought* **28(1)**: pp. 29–55.

Malliaris, M. & L. Salchenberger (1993): "A neural network model for estimating option prices." *Applied Intelligence* **3(3)**: pp. 193–206.

McCullagh, P. & J. A. Nelder (1989): *Generalized linear models*, volume 37. CRC press.

Merton, R. C., M. J. Brennan, & E. S. Schwartz (1977): "The valuation of american put options." *The Journal of Finance* **32(2)**: pp. 449–462.

Nelson, D. B. (1991): "Conditional heteroskedasticity in asset returns: A new approach." *Econometrica: Journal of the Econometric Society* pp. 347–370.

Saito, S. & L. Jun (2000): "Neural network option pricing in connection with the black and scholes model." In "Proceedings of the Fifth Conference of the Asian Pacific Operations Research Society, Singapore 2000," .

Scott, L. O. (1987): "Option pricing when the variance changes randomly: Theory, estimation, and an application." *Journal of Financial and Quantitative analysis* **22(4)**: pp. 419–438.

Yadav, N., A. Yadav, & M. Kumar (2015): *An introduction to neural network methods for differential equations*. Springer.

Yang, Y., Y. Zheng, T. M. Hospedales *et al.* (2017): "Gated neural networks for option pricing: Rationality by design." In "AAAI," pp. 52–58.

Zakoian, J.-M. (1994): "Threshold heteroskedastic models." *Journal of Economic Dynamics and control* **18(5)**: pp. 931–955.

Appendix A

Appendix

A.1 Source code

The Python source code originally developed by the author used for the programming of the neural networks in this thesis can be obtained either in the attachment of this thesis or by contacting of the author on his email `vach.dominik@gmail.com`. The Google's Tensorflow package and Keras were used in order to develop the architectures of neural networks.

A.2 Additional tables

The following table A.1 regards the chapter 4.1 which is summarizing the volatility of the dataset used in this thesis for all subsets used in the modular neural network. The table A.2 verifies on the real data for call options the usage of simulations from the chapter 4.2.1. Both of these tables were mentioned and discussed in the corresponding sections.

	Calls			
Days to expiration	< 60	60-180	$\geqslant 180$	All options
ITM ($\geqslant 1.05$)		Volatility (σ)		
σ_{IV}	10.36%	11.04%	12.31%	11.72%
σ_{MA5}	10.37%	10.02%	10.04%	10.09%
σ_{MA20}	10.47%	10.34%	10.36%	10.37%
σ_{MA60}	10.48%	10.43%	10.27%	10.44%
σ_{MA100}	10.41%	10.44%	10.41%	10.42%
$\sigma_{GARCH(1,1)}$ 9.67%	9.35%	9.5%	9.49%	
ATM ($0.97 - 1.05$)		Volatility (σ)		
σ_{IV}	12.63%	13.15%	14.84%	13.37%
σ_{MA5}	9.66%	9.39%	9.75%	9.62%
σ_{MA20}	10.02%	9.97%	10.14%	10.04%
σ_{MA60}	10.27%	10.30%	10.33%	10.29%
σ_{MA100}	12.63%	13.15%	14.84%	10.30%
$\sigma_{GARCH(1,1)}$	9.06%	8.76%	9.24%	9.04%
OTM (< 0.97)		Volatility (σ)		
σ_{IV}	28.09%	19.55%	17.41%	21.31%
σ_{MA5}	9.32%	9.38%	9.52%	9.43%
σ_{MA20}	9.77%	9.87%	9.94%	9.87%
σ_{MA60}	10.13%	10.21%	10.23%	10.20%
σ_{MA100}	10.19%	10.29%	10.27%	10.25%
$\sigma_{GARCH(1,1)}$	8.68%	8.7%	8.97%	8.81%
	Puts			
Days to expiration	< 60	60-180	$\geqslant 180$	All options
OTM ($\geqslant 1.05$)		Volatility (σ)		
σ_{IV}	19.33%	13.28%	14.09%	15.20%
σ_{MA5}	11.35%	10.35%	10.24%	10.55%
σ_{MA20}	11.03%	10.54%	10.5%	10.64%
σ_{MA60}	10.48%	10.43%	10.27%	10.66%
σ_{MA100}	10.73%	10.85%	10.60%	10.69%
$\sigma_{GARCH(1,1)}$	11.75%	10.36%	10.3%	10.68%
ATM ($0.97 - 1.05$)		Volatility (σ)		
σ_{IV}	12.81%	13.22%	15.13%	13.56%
σ_{MA5}	9.73%	9.39%	9.75%	9.65%
σ_{MA20}	10.08%	9.98%	10.14%	10.07%
σ_{MA60}	10.29%	10.31%	10.33%	10.31%
σ_{MA100}	10.32%	10.29%	10.30%	10.31%
$\sigma_{GARCH(1,1)}$	9.13%	8.76%	9.24%	9.07%
ITM (< 0.97)		Volatility (σ)		
σ_{IV}	19.60%	20.02%	20.38%	20.14%
σ_{MA5}	9.96%	9.55%	9.60%	9.66%
σ_{MA20}	10.30%	10.01%	10.01%	10.07%
σ_{MA60}	10.41%	10.32%	10.29%	10.32%
σ_{MA100}	10.32%	10.38%	10.33%	10.34%
$\sigma_{GARCH(1,1)}$	9.32%	8.98%	9.11%	9.12%

** 0.05 significance level, ** 0.01 significance level, *** 0.001 significance level*

Table A.1: Volatility prediction

hidden	nodes	MAE	RMSE	MAPE	DM-test	output
1	10	0.00384	0.00527	0.261	41.36***	softplus
1	20	0.00422	0.00571	0.174	40.39***	softplus
1	30	0.0034	0.00463	0.161	42.25***	softplus
1	40	0.00534	0.00741	0.203	37.41***	softplus
1	50	0.00357	0.00489	0.202	42.16***	softplus
1	60	0.00347	0.00475	0.173	42.04***	softplus
1	70	0.0036	0.00484	0.21	41.96***	softplus
1	80	0.00563	0.00793	0.141	34.9***	softplus
1	90	0.00448	0.00611	0.137	39.49***	softplus
1	100	0.00385	0.00542	0.173	41.4***	softplus
2	10	0.00344	0.00467	0.308	42.41***	softplus
2	20	0.00334	0.00458	0.19	42.72***	softplus
2	30	0.00666	0.00872	0.167	32.98***	softplus
2	40	0.00341	0.00471	0.163	42.03***	softplus
2	50	0.00313	0.00444	0.173	43.2***	softplus
2	60	0.00323	0.00472	0.183	42.84***	softplus
2	70	0.00387	0.00523	0.15	41.02***	softplus
2	80	0.00521	0.00755	0.152	36.6***	softplus
2	90	0.00372	0.00553	0.172	41.65***	softplus
2	100	0.00432	0.00614	0.131	39.96***	softplus
3	10	0.00371	0.00572	0.396	40.36***	softplus
3	20	0.00341	0.0048	0.259	42.02***	softplus
3	30	0.00337	0.00455	0.244	42.41***	softplus
3	40	0.00314	0.00432	0.186	42.67***	softplus
3	50	0.00365	0.00496	0.178	41.78***	softplus
3	60	0.00369	0.00501	0.167	41.5***	softplus
3	70	0.00541	0.00702	0.201	37.5***	softplus
3	80	0.00454	0.00635	0.187	38.98***	softplus
3	90	0.00696	0.0101	0.168	29.91***	softplus
3	100	0.00715	0.00922	0.164	31.81***	softplus

** 0.05 significance level, ** 0.01 significance level, *** 0.001 significance level*

Table A.2: Prediction accuracy for call options using sigmoid activation in hidden layers, softplus at the output