Tampere University

Van Huynh

NEURAL NETWORK APPROACH TO FORECASTING OPTION PRICE

Faculty of Engineering and Natural Sciences
Bachelor of Science thesis
April 2022

ABSTRACT

Tampere University
Bachelor of Science and Engineering
Author: Van Huynh
Title of the thesis: Neural network approach to forecasting option price
Date: April 2022

This thesis evaluates whether a Long Short-Term Memory (LSTM) neural network perform better than the Black-Scholes model in pricing European-type options. The S&P 500 index daily calls and puts option data from January 2018 to October 2018 is used in this thesis to train, validate and test the model. The same test data is also used on the Black-Scholes model to form a comparison.
.
Since early literature on option pricing using neural networks has already tested extensively the capabilities of MLP-type neural networks, the study opts to apply an LSTM network architecture instead. LTSM is a recurrent neural network (RNN) structure suited to learn from sequential data while not suffering from the vanishing gradient problem faced by other RNN structures.

The thesis will first introduce basic concepts of neural networks to the readers, as well as explaining the advantage of the neural network structure used in the study. Chapter 3 of the thesis provides a brief survey of related literature on the topic of option pricing using neural network. The final chapters of the thesis will explain the methodology behind the study and discuss the empirical results.

The study behind the thesis tests the performance of an option pricing model learned by Long Short-Term Memory architecture against the Black-Scholes pricing model. Three different performance metrics are used to test the performance of both models using a set of test data. The performance metrics are calculated by putting the price predictions by the neural network and the Black-Scholes model against the actual prices of the option.

The results are consistent with previous literature in terms of performance in pricing options and show that an LSTM-type neural network is superior in all performance metrics to the Black-Scholes model.

Keywords: Neural Networks, Black-Scholes, Option pricing, Long Short-Term Memory.

PREFACE

This thesis and study was conducted at Tampere University for my Bachelor's Programme in Science and Engineering. Above all, I am thankful to my parents and sister who gave and are always giving their all to support my journey.

I would also like to express my deepest and sincere gratitude toward Prof. Juho Kanniainen, who provided me with the topic and invaluable input necessary for my work.

Lastly, I would like to show appreciation to my friends who provided me with the strength to push through my struggles and who shared joy and discussion daily.

Tampere, April 2022

Huynh Khuong Van

CONTENTS

LIST OF SYMBOLS AND ABBREVIATIONS

ANN	Artificial neural network
FNN	Feedforward neural network
RNN	Recurrent neural network
LSTM	Long short-term memory
MLP	Multilayer perceptron
BS	Black-Scholes
NTM	Near-the-money
ITM	In-the-money
OTM	Out-of-the-money
GARCH	Generalized Auto Regressive Conditional Heteroskedasticity

1 INTRODUCTION

The first supervised neural networks date back to the early 19th century in the form of linear regression variants (Gauss, 1809, 1821). Nearing the latter half of the 20th century ideas of unsupervised learning were published (Hebb, 1949). The following decades brought forth frequent and drastic improvements to NN architectures: the invention of the perceptron (Rosenblatt, 1958), convolutional NNs and subsampling (Fukushima, 1980), popularisation of backpropagation (Rumelhart et al., 1986), ... The applications of neural networks during the late 20th century saw limited success due to hardware limitations resulting in inadequately trained neural networks. Furthermore, many modern deep learning advancements are still being researched and developed during this period, explaining the poor optimization of early neural networks.

In the modern world, the average person possesses exponentially greater computing power and data than any 20th century researchers did while having unlimited access to relevant academic discussions and materials. Tools such as TensorFlow, Keras and Torch supported by programming languages like Python and R allow any person, regardless of academic background, to learn and implement deep learning solutions to any problems definable in deep learning terms.

Option pricing is among the most studied topics in finance and a primary target for deep learning applications in the field. Options are contracts that represent the right to buy or sell a security on (European type) or before (American type) the expiration date. Arguably the most famous option pricing formula is the Black-Scholes model (Black & Scholes, 1973)

$$c = SN(d_1) - Xe^{-rT}N(d_2) \tag{1}$$

$$p = Xe^{-rT}N(d_1) - SN(d_2) \tag{2}$$

where $d_1 = \frac{\ln\left(\frac{S}{X}\right)+(r+\frac{\sigma^2}{2})T}{\sigma\sqrt{T}}$; $d_2 = d_1 - \sigma\sqrt{T}$; $N()$ is the cumulative density function of the standard normal distribution; c is the call price; p the put price; X the exercise price; S the underlying price; T the annualized fraction of time until maturity; r the risk-free rate; and σ the volatility of the underlying asset, which cannot be observed directly. The model is based on the assumptions that stock price movements are continuous and follow geometric Brownian motion, r and σ are constant and the market is frictionless, i.e., there are no transaction costs. However, many of these assumptions are often violated in practice. For example, the model's prices may show a volatility smile, implying the model's prices are lower for deep out-of-the-money and deep in-the-money options, a result of the moneyness bias. The model also suffers from other biases such as the term-structure bias and the put-skew bias.

Many modifications and approaches were made with the limitations of the Black-Scholes model in mind such as the Heston model (Heston, 1993) and jump-diffusion processes (Merton, 1976). An alternative solution is to consider an option's price a function of X, T, S, r and σ. This serves as a foundation upon which a computational model can be built such that there's no reliance on assumptions concerning financial mechanics but instead on dynamics behind historical data. Inspired by the success of deep learning in fields such as medical, automation and visual recognition, the thesis aims to build a neural network model and test its performance against the Black-Scholes model. The inputs of the model are the terms of the option and the price of the underlying at transaction time and the output would be the price of the option. The thesis explores a neural network structure that take into consideration previous state information known as Long Short-Term Memory.

2 ARTIFICIAL NEURAL NETWORKS

This chapter provide an overview of the different types of neural network, their basic architecture, their shortcomings, and the solutions to these shortcomings. Section 2.3 will briefly discuss the main type of neural network used in the thesis, the Long Short-Term Memory architecture.

2.1 Artificial Neural Networks

The first artificial neural networks (ANNs) were modelled after the neural network of a biological brain. A neural network is a collection of simple processing blocks, called nodes, connected by weighted connections. Biologically, the nodes of a neural network represent neurons, while the weights of the connections are the strength of the synapses connecting the neurons. The activation of a neural network, when an input is provided to some or all nodes, emulates the electrical activity of a neuron.

The typical form of a neural network consists of layers of neurons, connections, a propagation function, and a learning rule. Input data is first fed to neurons in the input layer. The size, i.e., the number of neurons in the input layer, is dependent on the dimension of the learning problem's input. The input layer will then transfer data to hidden layers of the neural network. In each hidden layer the input data will be computed by the propagation function and output to the next layer. Connections, including weights and biases dictate the rules by which outputs of a layer is transferred to the next. The learning rule modifies the weights and thresholds of the variables in the network. The process is repeated for each hidden layer until the output reach the output layer.

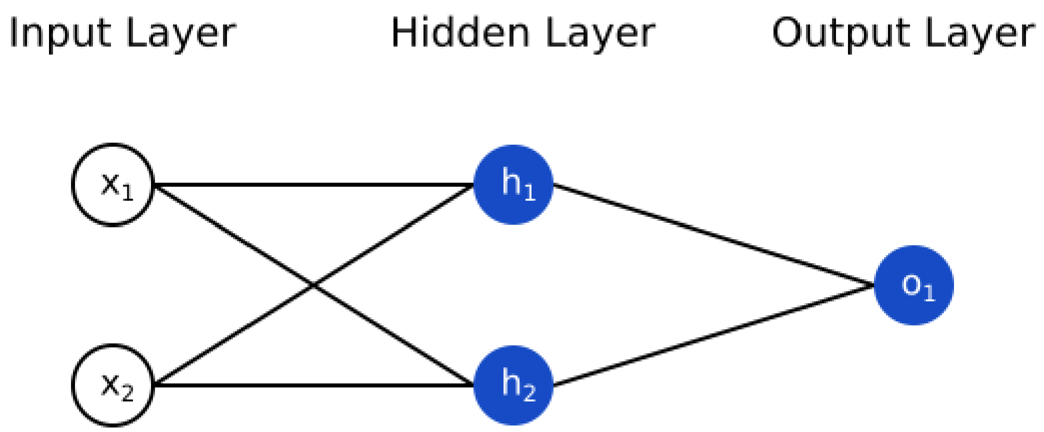

Figure 1. Example of a one-hidden-layer neural network.

We can partition neural networks into two types based on the connections between nodes, neural networks which with acyclic connections, called feedforward neural networks (FNNs), and neural networks with cyclical connections, called recurrent neural networks (RNNs). Famous examples of FNNs include perceptron (Rosenblatt, 1958), Kohonen maps (Kohonen, 1989) and the multilayer perceptron (MLP; Rumelhart et al., 1986) ... Varieties of RNNs include Elman networks (Elman, 1990), Jordan networks (Jordan, 1990) and Long Short-Term Memory (LTSM; Hochreiter and Schmidhuber, 1997), which will be the primary type of neural network used in the thesis.

2.2 Recurrent Neural Networks

Recurrent neural networks (RNNs) are obtained when the connections between nodes are extended to allow cycles. Despite the seemingly trivial extension from MLPs to RNNs, there are significant implications for the sequence learning ability of neural networks. FNNs such as the multilayer perceptron map only from input vectors to output vectors, while a RNN may theoretically map every past state of earlier inputs to every output. In other words, recurrent connections allow the history of earlier inputs to persist through the network's hidden layers, which in turn influence the entire network output.

Similar to FNNs, the basic structure of a RNN consists of three types of layers: input layer, recurrent hidden layer and output layer. Like FNNs, the most used activation functions of RNNs are sigmoid, tanh, ReLU and leaky ReLU. Figure 2 below shows the basic structure of an RNN.

Due to the sequential nature of the learning problem, RNNs hold great advantages over other forms of neural networks such as MLPs and convolutional neural networks (CNNs) in terms of predicting option prices while CNNs in particular are better suited for feature detection.

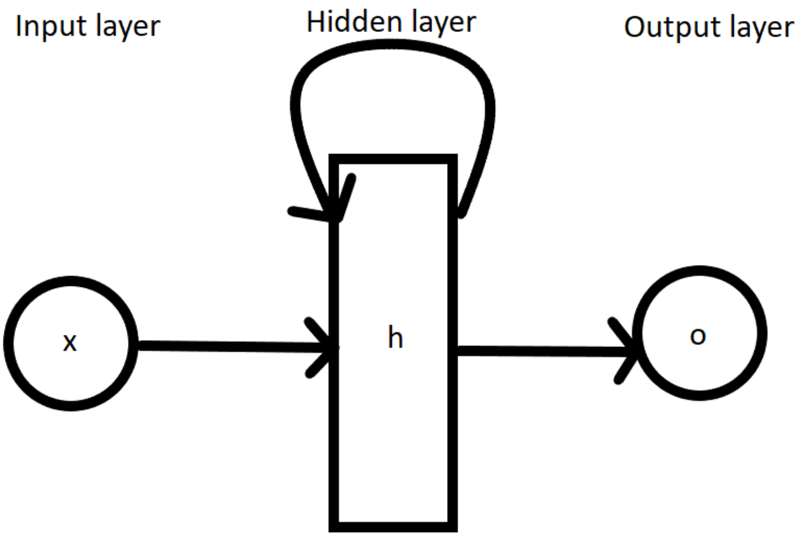

Figure 2. Structure of a simple recurrent neural network.

2.3 Long Short-Term Memory

2.3.1 The vanishing gradient problem.

While recurrent neural networks has an advantage by using contextual information as additional input in their hidden layers, simple RNN architectures suffer from a limited range on this context. The main drawback is that the effect of an input on the output hidden layers, and by extension on the the entire network' output, either increases or decreases exponentially as it goes through the network's cyclical connections. This shortcoming, termed the vanishing gradient problem (Hochreiter, 1991) poses a difficulty for an RNN to learn from problems with more than 10 timesteps delay between input and output (Hochreiter et al., 2001). Figure 3 shown below illustrates schematically the vanishing gradient problem. Various solutions were proposed in the 1990s as an attempt to address the vanishing gradient problem. Among these are non-gradient based training algorithms like simulated annealing and discrete error propagation (Bengio et al., 1994), explicitly introduced time delays (Lang et al., 1990; Lin et al., 1996; Plate, 1993), and hierarchical sequence compression (Schmidhuber, 1992). One of the most effective solution, however, is the Long Short-Term Memory (LSTM) architecture (Hochreiter and Schmidhuber, 1997).

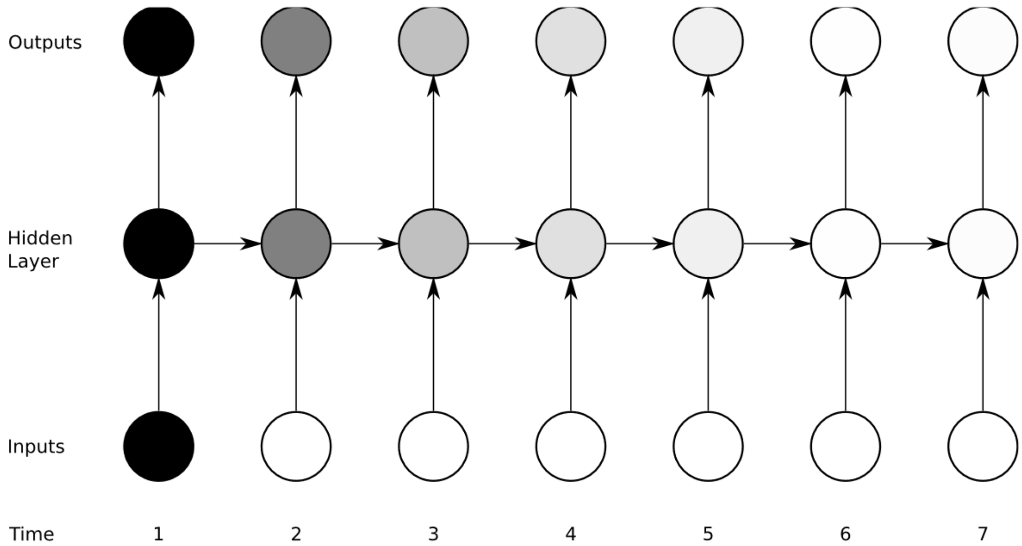

Figure 3. Vanishing gradient problem for RNNs. The shading denotes the sensitivity of the nodes to an input over a period (darker shade means greater sensitivity).

2.3.2 LSTM Architecture

The LSTM architecture is built upon a collection of recurrently connected subnets called memory blocks. The basic structure of an LSTM network is nearly identical to a simple RNN, except that the nonlinear units in the hidden layer are replaced by memory blocks. Each block contains one or more self-connecting memory cells and three multiplicative units: the input gate, the output gate and the forget gate. Each gates provide continuous analogues of write, read, and reset operations for the cells.

The LSTM architecture avoid the vanishing gradient problem by using multiplicative gates of its memory cells for storage and use information for an extended period. For instance, when the input gate stays closed, the cell activation will not be overwritten by the arriving inputs, thereby allowing the information to be available to the network at a later time in the sequence, whenever the output gate is opened.

Figure 4 illustrates the structure of an LSTM memory block with one cell. Activations from inside and outside the block are collected by the three gates and the cell is controlled by multiplicative units (small circles in the figure). The input and output of the cell is scaled by the input and output gate, respectively, while the internal state is scaled by the forget gate. The input activation function *g()* and output activation function *h()* of the cell are applied at the designated places.

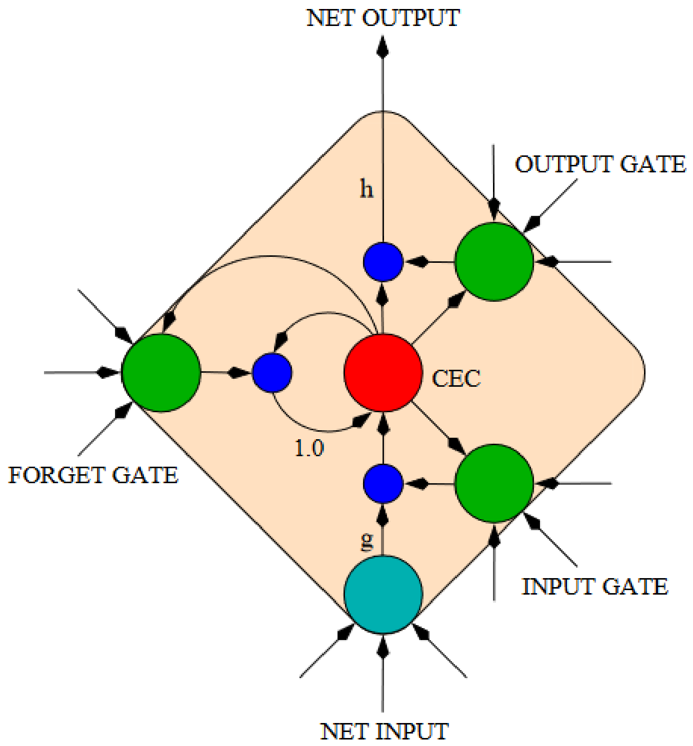

Figure 4: A memory block of LSTM containing a single cell. Recurrent connection maintains the cell' state.

3 BRIEF OVERVIEW OF DEEP LEARNING AS AN OPTION FORECASTING METHOD

Among the first to examine the capabilities of neural networks to price option and perhaps one of the most cited was the paper by Hutchinson et al. (1994). The paper examined the results of three non-parametric approaches to price call options and compared them to the results of the Black-Scholes model. Hutchinson et al. (1994) studied and compared the performance between a multilayer perceptron, a radial basis function (Powell, 1987) and a projection pursuit approach (Huber, 1985) to determine whether the models were able to learn the dynamics between the underlying and the price of an option. Hutchinson et al. (1994) initiated several papers with similar topics in the following years and until now.

While numerous papers on the capabilities of neural networks in pricing options have been published since the paper by Hutchinson et al. (1994), the majority of these studies only provided additional insights and improved upon the approach that was already taken in the original paper. These papers, due to a combination of hardware limitation and relative infancy of deep learning techniques, rely on simple feed-forward networks that have already been used and studied extensively. Among the insights provided and the improvements made were the introduction of statistical inference in choosing networks (Anders et al. 1996), using statistical inference techniques in the results of the models (Garcia & Gencay 2000), ability to estimate underlying volatility of neural networks (Yao et al. 2000) and different regularization techniques of neural networks (Gencay & Salih 2003).

Nonetheless, these papers inspired future academia to examine the performance of more modern inventions of neural works such as recurrent neural networks in option pricing and to compare the performance between themselves and with other known models.

4 DATASET AND METHODOLOGY

4.1 Dataset

The data used in this thesis are the transaction data of S&P 500 stock index options. Over 600,000 The data is supplemented with treasury yields obtained from the US Treasury Resource Center, which informs the risk-free rate r for the model.

The dataset provides information on the contract terms X and T, as well as the underlying price S. The yield on the US Treasury instrument having maturity closest is matched to the time until expiration of each option to find the risk-free rate, a widely accepted options trading practice.

To find the volatility σ for the Black-Scholes model, historical volatility from the previous 20 trading days (approximately one trading month) is assumed to represent of the volatility over the life of the option. We can then feed this new feature into the Black-Scholes model. The thesis uses equilibrium price, which is the average of bid and ask prices.

Anders et al. (1996) suggested using exclusion criteria to remove non-representative data that represent illiquid or extraordinary options occurrences. The criteria exclude options that are too deep in-the-money or out-of-the-money, options with over 2 years to expiration, or options that are traded at such low prices that the discrete nature of security prices becomes a consideration. While this suggestion has been widely adopted among previous literature, this thesis will not follow such practice in the hope that a neural network model is able to learn using these rare circumstances to better predict the behaviour of options.

The research studied a total of 685,196 data points of roughly half calls and half puts. 98% of the data is used as a training set while the last 2% of the data are split between validation set and test set.

4.2 Neural network architecture

LSTM is used in this paper to capture state information in the hope that the architecture can learn the implied volatility from historic observational data for an improved pricing performance. An LSTM consisting of eight units are inputted with the daily closing price at every timestep over 20 timesteps. The number of timesteps were chosen since we assumed the historical volatility to be from the previous 20 days. The output sequence is fed forward for three layers of 8-unit LSTMs. The final timestep's prediction is then concatenated with the S, X, T, and r, and fed through a fully connected network (FCN) to output the equilibrium price. The architecture is shown in Figure 5.

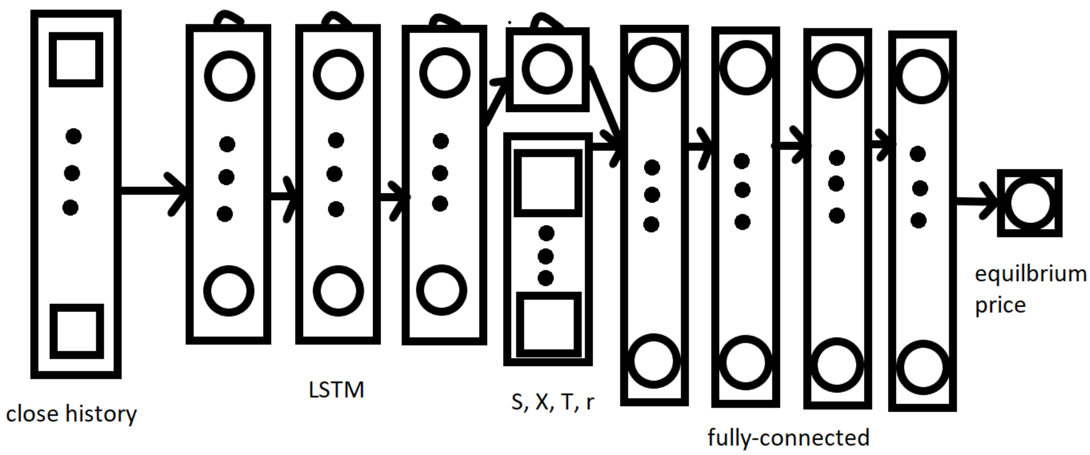

Figure 5. Architecture of the neural network model used in the empirical study.

The network is trained with 100 epochs and on a batch size of 64. Since batch normalization is used, Adam optimizer (Kingma & Lei Ba 2014) is chosen as the gradient descent algorithm, with a learning rate of δ = 0.0001.

4.3 Performance metrics

The performance of the LSTM architecture is tested against the original Black-Scholes model, without Merton's (1973) adjustment for dividends. The Black-Scholes model is defined by equation (1) and (2):

$$c = SN(d_1) - Xe^{-rT}N(d_2) \tag{1}$$

$$p = Xe^{-rT}N(d_1) - SN(d_2) \quad (2)$$

where $d_1 = \frac{\ln\left(\frac{S}{X}\right)+(r+\frac{\sigma^2}{2})T}{\sigma\sqrt{T}}$; $d_2 = d_1 - \sigma\sqrt{T}$; $N()$ is the cumulative density function of the standard normal distribution; c is the call price; p the put price; X the exercise price; S the underlying price; T the annualized fraction of time until maturity; r the risk-free rate; and σ the volatility of the underlying asset.

The performance metrics used in the thesis are MSE, MAPE and PE, where train-MSE shows the mean squared error on the training set, and each of these metrics are calculated over the test set for both models. MAPE is the mean absolute percent error, and PEX% is the percentage of observations within ±X% of the actual price. These error estimates are defined in the equations below

$$MSE = \frac{1}{N}\sum_{i}^{N}(A_i - F_i)^2 \quad (3)$$

$$MAPE = \frac{1}{N}\sum_{i}^{N}\left|\frac{A_i - F_i}{A_i}\right| \quad (4)$$

$$PE = \left|\frac{A_i - F_i}{A_i}\right| * 100 \quad (5)$$

where N is the size of the test data, A_i is the i_{th} actual price of the option and F_i is the i_{th} forecasted value.

5 RESULTS

Since train-MSEs for the LSTM structure are approximately equal to the training MSEs, there is no evidence of overfitting. The error metrics for call options is shown in table 1, while the error metrics for put options is displayed in table 2.

As shown in the tables below, the LSTM architecture outperforms the Black-Scholes in every performance metrics, for both type of options. Additionally, the PEX% metrics reveal that LSTM prices illiquid options with much greater accuracy than Black-Scholes, which struggled as shown by the PE.

The Black-Scholes model particularly struggled in pricing put options. While the LSTM provides less accuracy in pricing put options, it stills improve over the Black-Scholes significantly.

Model	train-MSE	MSE	MAPE(%)	PE5(%)	PE10(%)	PE20(%)
BS	0.000223	0.000272	76.12	50.53	57.19	68.81
LSTM	0.000035	0.000066	21.58	60.49	64.32	75.21

Table 1. Error metrics comparing LSTM prices with Black-Scholes prices for call options.

Model	train-MSE	MSE	MAPE(%)	PE5(%)	PE10(%)	PE20(%)
BS	0.000554	0.000513	66.56	11.52	19.33	22.59
LSTM	0.000093	0.000095	41.35	27.45	34.25	43.86

Table 2. Error metrics comparing LSTM prices with Black-Scholes prices for put options.

6 CONCLUSION

This thesis examines deep learning for option pricing, in particular using the LSTM architecture to learn from historic data and estimate the volatility of the options for better forecasting performance. The thesis used performance metrics to determine the relative performance of the LSTM to the Black-Scholes model.

Despite a naive assumption of volatility, the LSTM architecture nonetheless vastly outperformed the Black-Scholes model on every performance metrics. By learning from historical options data, the model was able to evade financial assumptions and view option's price as a function that can be approximated by a neural network. Since the study only used equilibrium price, the possibility of using bid and ask prices individually, or even closing prices, leave much to be explored about the topic.

For future research, optimal and/or unusual hyperparameters may still to be found for the LSTM approach. It's also possible and perhaps academically significant to implement deep learning for the reverse problem, i.e., to find the implied volatility using historical option price. This will allow comparison between the plotted volatility surface and volatility surfaces from the Heston or GARCH models. Furthermore, deeper error analysis can be done to see if neural networks perform better or worse provided certain constraints (NTM, ITM or OTM options, different expiration dates, etc.).

REFERENCES

Gauss, C. F. (1809). Theoria motus corporum coelestium in sectionibus conicis solem ambientium.

Gauss, C. F. (1821). Theoria combinationis observationum erroribus minimis obnoxiae

Hebb, D. O. (1949). The Organization of Behavior. Wiley, New York.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. Psychological review, 65(6):386.

Fukushima, K. (1980). Neocognitron: A self-organizing neural network for a mechanism of pattern recognition unaffected by shift in position. Biological Cybernetics, 36(4):193–202.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning internal representations by error propagation. In Rumelhart, D. E. and McClelland, J. L., editors, Parallel Distributed Processing,volume 1, pages 318–362. MIT Press.

Black, F. & Scholes, M. (1973). The pricing of options and corporate liabilities. Journal of Political Economy 81(3), 637-654.

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), 327-343.

Merton, R. C. (1976). Option pricing when underlying stock returns are discontinuous.

T. Kohonen. Self-organization and associative memory: 3rd edition. Springer-Verlag New York, Inc., New York, NY, USA, 1989. ISBN 0-387-51387-6.

J. L. Elman. Finding structure in time. Cognitive Science, 14:179–211, 1990.

M. I. Jordan. Attractor dynamics and parallelism in a connectionist sequential machine, pages 112–127. IEEE Press, Piscataway, NJ, USA, 1990. ISBN 0-8186-2015-3.

S. Hochreiter and J. Schmidhuber. Long Short-Term Memory. Neural Computation, 9(8):1735–1780, 1997.

S. Hochreiter. Untersuchungen zu dynamischen neuronalen Netzen. PhD thesis, Institut für Informatik, Technische Universität München, 1991. See-http://ni.cs.tu-berlin.de/~hochreit/papers/hochreiter.dipl.ps.gz.

Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. IEEE Transactions on Neural Networks, 5(2):157–166, March 1994. URL citeseer.ist.psu.edu/bengio94learning.html

T. Lin, B. G. Horne, P. Tĩno, and C. L. Giles. Learning long-term dependencies in NARX recurrent neural networks. IEEE Transactions on Neural Networks, 7(6):1329–1338, November 1996. URL citeseer.ist.psu.edu/lin96learning.html.

K. J. Lang, A. H. Waibel, and G. E. Hinton. A time-delay neural network architecture for isolated word recognition. Neural Netw., 3(1):23–43, 1990. ISSN 0893-6080. doi: http://dx.doi.org/10.1016/0893-6080(90)90044-L.

J. Schmidhuber. Learning complex extended sequences using the principle of history compression. Neural Computing, 4(2):234–242, 1992. URL citeseer.ist.psu.edu/article/schmidhuber92learning.html.

S. Liu, C. Oosterlee, and S. Bohte. Pricing options and computing implied volatilities using neural networks. Risks, 7:16, 02 2019.

Anders, U., Korn, O. & Schmitt, C. (1996). Improving the pricing of options: A neural network approach. Journal of Forecasting 17(5), 369-388.

Yao, J., Li, Y. & Tan, C. L. (2000). Option price forecasting using neural networks.

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

Amilon, H. (2003). A neural network versus Black–Scholes: A comparison of pricing and hedging performances. Journal of Forecasting 22(4), 317-335.

D.P., K. & J., B. (2014). Adam: A method for stochastic optimization. ArXiv e-prints.

Bollerslev, Tim (1986). "Generalized Autoregressive Conditional Heteroskedasticity". Journal of Econometrics. 31 (3): 307–327. CiteSeerX 10.1.1.468.2892. doi:10.1016/0304-4076(86)90063-1.