Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Economics and Management
Dept. of Industrial Economics and Technology Management

Erlend Stegavik Rygg
Hjalmar Jacob Vinje
Cassandra Wu

Enhanced Option Pricing Using Deep Learning

A Time-Series Approach with a Combined LSTM-MLP Model

Master's thesis in Industrial Economics and Technology Management
Supervisor: Sjur Westgaard
Co-supervisor: Morten Risstad
June 2023

NTNU
Norwegian University of Science and Technology

Erlend Stegavik Rygg
Hjalmar Jacob Vinje
Cassandra Wu

Enhanced Option Pricing Using Deep Learning

A Time-Series Approach with a Combined LSTM-MLP Model

Master's thesis in Industrial Economics and Technology Management
Supervisor: Sjur Westgaard
Co-supervisor: Morten Risstad
June 2023

Norwegian University of Science and Technology
Faculty of Economics and Management
Dept. of Industrial Economics and Technology Management

NTNU
Norwegian University of
Science and Technology

Preface

This research represents the conclusion of our master's degree at the Norwegian University of Science and Technology (NTNU). It reflects the knowledge we've gathered and the skills we've honed throughout our studies. We've had a range of experiences at NTNU that have shaped our understanding and approach to learning and research. It's been a meaningful journey of growth within this academic environment.

We are grateful for the support and guidance from our supervisors, Professor Sjur Westgaard, and the Head of FX and interest rate derivatives at SpareBank 1 Markets, Ph.D. Morten Risstad. Their expertise and advice have been instrumental in accomplishing this research. Furthermore, we would like to thank Professor Christian Oliver Ewald for inspiring discussions and valuable input on our research topic. We also want to thank our friends and peers for their continuous support and encouragement.

With the support from the people mentioned above, we are grateful for the experience of researching the complex yet fascinating problem of option pricing. We hope our research not only contributes to the current body of knowledge, but also opens new doors for the exploration of the exciting intersection of deep learning with finance.

Erlend Stegavik Rygg, Hjalmar Jacob Vinje and Cassandra Wu

Trondheim, Norway, June 11, 2023

Abstract

This paper presents a deep learning approach to option pricing that integrates a long short-term memory (LSTM) network with a multi-layer perceptron (MLP) to form a combined LSTM-MLP model. The proposed model uses its LSTM component to extract time series information from the historical returns of the underlying asset. This information is then used by the MLP component, along with the option characteristics for a given contract, to determine the option price. By training the pricing model on historical returns, we enable it to extract time series information that encapsulates market dynamics including volatility. This can replace the need for an explicit volatility measure, which is required by established option pricing methods.

We empirically test the proposed LSTM-MLP pricing model by applying it to European call options on the S&P 500 index from 2015 to 2022. Using sliding windows, we simulate real-life deployment with monthly retraining. Our results show that the LSTM-MLP model outperforms benchmark models in pricing accuracy, predictive performance, and risk-adjusted trading returns. This demonstrates its potential usefulness as a valuation benchmark for options market makers or as trading signals for options investors.

Sammendrag

Denne artikkelen presenterer en dyplæringstilnærming til opsjonsprising som integrerer et long short-term memory (LSTM) nettverk med et multi-layer perceptron (MLP) netverk for å danne en kombinert LSTM-MLP modell. Den foreslåtte modellen bruker sin LSTM-komponent til å trekke ut tidsserieinformasjon fra de historiske avkastningene til den underliggende eiendelen. Denne informasjonen brukes deretter av MLP-komponenten, sammen med opsjonskarakteristikkene for en gitt kontrakt, for å bestemme opsjonsprisen. Ved å trene prisingsmodellen på historiske avkastninger, gjør vi det mulig for den å trekke ut tidsserieinformasjon, inkluderer volatilitet. Dette kan erstatte behovet for et eksplisitt volatilitetsmål som kreves av etablerte metoder for opsjonsprising.

Vi tester empirisk den foreslåtte LSTM-MLP prismodellen ved å bruke den på europeiske kjøpsopsjoner på S&P 500-indeksen fra 2015 til 2022. Ved å bruke rullerende vindu, simulerer vi virkelighetsnær implementering med månedlig trening. Våre resultater viser at LSTM-MLP modellen overgår benchmark-modeller i prisingsnøyaktighet, prediktiv ytelse og risikojustert avkastning. Dette demonstrerer modellens potensielle praktiske nytte som en prisingsreferanse for markedsdeltakere eller som handelssignaler for opsjonsinvestorer.

Table of Contents

List of Figures vi

List of Tables vii

List of Abbreviations viii

1 Introduction 1

2 Literature review 3

2.1 Option pricing models . . . 3

2.2 Volatility measures for neural network-based option pricing models . . . 5

2.3 Performance measures for neural network-based pricing models . . . 6

2.4 Interpretable machine learning . . . 7

3 Data 8

3.1 Data set . . . 8

3.2 Data preprocessing . . . 8

3.3 Descriptive statistics . . . 9

4 Methods 11

4.1 Combined LSTM-MLP option pricing model . . . 11

4.2 Backtesting trading strategy based on price differences . . . 19

4.3 Benchmark models . . . 20

4.4 Performance metrics . . . 22

4.5 Interpretability analysis . . . 23

5 Results and discussion 26

5.1 Pricing performance . . . 26

5.2 Predictive ability of directional price movements . . . 39

5.3 Trading performance . . . 40

6 Conclusion 45

Bibliography 47

Appendix 51

A Nelson–Siegel–Svensson for generating rate curves 51

B MLP model hyperparameter search **52**

C Interpretability analysis **53**

D Market conditions **54**

E Trading performance by year **54**

List of Figures

3.1 Number of options quotes at different levels of maturity and moneyness 10
3.2 Average daily trading volume per contract for different maturity and moneyness . 10
4.3 Architecture of the combined LSTM-MLP option pricing model 11
4.4 Structure of an LSTM memory cell at a specific time step t. 13
4.5 Data structure for the LSTM-MLP 15
4.6 Sliding windows setup for model training and testing 16
4.7 Hyperparameter search for the LSTM-MLP 17
5.8 Boxplot of pricing difference for all models 27
5.9 Boxplot of pricing differences for all models across all years 30
5.10 Pricing performance of the LSTM-MLP with MACD overlay 32
5.11 Pricing performance of the LSTM-MLP with VIX overlay 33
5.12 RMSE at different maturities 34
5.13 RMSE at different moneyness 35
5.14 SHAP summary plot of the option characteristics for the LSTM-MLP 37
5.15 SHAP summary plot of the 10 most recent returns for the LSTM-MLP 37
5.16 ALE feature importance for the LSTM-MLP 38
5.17 ALE feature importance for the MLP benchmark 39
5.18 Annual cash, net option portfolio and total portfolio for the LSTM-MLP 43
5.19 Distributions of options bought and sold at the time of the transaction 44
A.1 18 Random interest rate curves generated using the NSS method 51
B.1 Hyperparameter search for the MLP 52
C.1 SHAP summary plot for all returns for the LSTM-MLP 53
C.1 MACD during the testing period 54
C.2 VIX during the testing period 54

List of Tables

3.1 Average trading volume per option contract segmented by filtering criteria 8
3.2 Descriptive statistics for all options data used 9
4.3 LSTM-MLP configuration grouped by model component 18
5.4 RMSE and MAE values for the pricing performance by all models 26
5.5 Diebold-Mariano test statistics for each pair of models 27
5.6 RMSE for each model for each year 28
5.7 MAE for each model for each year 28
5.8 Pricing performance during different S&P 500 regimes 31
5.9 Pricing performance during different VIX regimes 33
5.10 RMSE grouped by moneyness and maturity 36
5.11 Correct directional prediction 40
5.12 Average yearly trading performance 41
5.13 Alpha and Beta values based on the CAPM 41
B.1 MLP model configuration 52
E.1 BS 30-day trading performance by year 55
E.2 BS GARCH trading performance by year 55
E.3 BS IV trading performance by year 55
E.4 Heston trading performance by year 55
E.5 MLP trading performance by year 56
E.6 LSTM-MLP trading performance by year 56

List of Abbreviations

AIC Akaike Information Criterion

ALE Accumulated Local Effects

BIC Bayesian Information Criterion

BS Black-Scholes

CAPM Capital Asset Pricing Model

CNN Convolutional Neural Networks

DM Diebold-Mariano

IQR Interquartile range

IV Implied Volatility

LSTM Long Short-Term Memory

MACD Moving Average Convergence/Divergence

MAE Mean Absolute Error

MDD Maximum Drawdown

MLP Multi-Layer Perceptron

MSE Mean Squared Error

NSS Nelson-Siegel-Svensson

PDP Partial Dependence Plots

RMSE Root Mean Squared Error

SABR Stochastic alpha, beta, rho

SHAP SHapley Additive exPlanantions

TTM Time-to-maturity

XAI Explainable Artificial Intelligence

1 Introduction

This paper studies the application of deep learning to option pricing and proposes a pricing model that combines a long short-term memory (LSTM) network with a multi-layer perceptron (MLP). The proposed architecture enables the model to use both time series and options data in its pricing process. First, time series data in the form of historical S&P 500 returns is provided as input to the LSTM network for extraction of the embedded information. Subsequently, the extracted information is used by the MLP, along with the option characteristic for a given contract, to determine the option price. The combined LSTM-MLP architecture is motivated by our hypothesis that the time series information extracted by the LSTM is more informative for option pricing than the explicit volatility input commonly used by option pricing models. Therefore, we replace the explicit volatility input to the MLP with extracted time series information conveyed via the LSTM outputs. Our research objective extends beyond achieving a high pricing accuracy to also include generating positive risk-adjusted trading returns. This aims to demonstrate the practical value of the proposed pricing model for different market participants, such as a valuation benchmark for options market makers or trading signals for investors.

In the context of the literature on neural network-based option pricing models, our research contributes in the following ways:

1. We demonstrate that the proposed LSTM-MLP achieves better pricing and trading performance than the benchmark models. This indicates that the time series information extracted by an LSTM network is more informative for option pricing than an explicit volatility input. To the best of our knowledge, combining an LSTM network with an MLP for option pricing has only been explored by Ke and Yang (2019). However, their model does not perform better than an MLP with an explicit volatility input. This makes our research to be the first LSTM-MLP implementation to do so. Our implementation is also novel in that the LSTM uses returns rather than prices for the underlying asset as input, and we use a significantly longer period of historical data[1].

2. We evaluate the practical relevance of the proposed pricing model with a trading implementation. This goes beyond common practice for evaluating model performance in the literature, which is often limited to pricing accuracy measured as the difference between the model price and the market price. We also provide a more comprehensive analysis of trading performance compared to Andreou et al. (2008), which is the only previous study identified that uses trading to evaluate a neural network-based option pricing model.

3. We simulate how a market participant could deploy and retrain the model in real life by using sliding windows to train and test our model, following Andreou et al. (2008)[2] and Cao et al. (2021)[3]. This differs from most previous studies that split the entire data set chronologically for training, validation, and testing. Our setup is novel in that we implement more sliding windows over a longer time period, and we use a different combination of the length of the training, validation, and test set.

4. We apply our option pricing model to some of the most recent data for options on the S&P 500 index. Our data set ends in December 2022 and thereby includes the market turbulences related to Covid-19. Besides Gradojevic and Kukolj (2022), our research is one of few studies to date that analyzes the impact of Covid-19 on neural network-based option pricing models.

[1] Ke and Yang (2019) use 20-days lags, while our model implementation uses 140-days lags.

[2] Andreou et al. (2008) divide their data set into ten different overlapping training and validation sets, each followed by a non-overlapping test set. For visualization of the rolling training-validation-testing procedure, see Andreou et al. (2008) Figure 2.

[3] Cao et al. (2021) uses 300 days for testing and validation, and the following 7 or 30 days for testing. Then the training and validation window rolls forward 7 or 30 days for the second testing. This goes on until the end of the sample period.

Option pricing is complex and challenging due to the uncertainty of outcomes at the future time when the option expires. Numerous methods have been proposed for the option pricing task, of which the most well-known one is presented by Black and Scholes (1973). Although it has been empirically proven that the Black-Scholes model does not fit real financial markets, it has still gained widespread popularity due to its relatively simple arithmetic and the limited number of inputs, most of which are easily observable (Natenberg, 2015). Since its publication, the Black-Scholes model has significantly influenced how traders price and hedge derivatives, and to date plays a key role in how option portfolios are managed (Hull, 2023).

A challenge central to option pricing is about how to most suitably incorporate information about volatility. This is fundamental to option pricing as an option holder is sensitive to both the speed and the direction of the market for the underlying asset. If this market does not move with sufficient speed, options on the asset will be priced lower to reflect the reduced likelihood of the asset price going above the strike (Natenberg, 2015). Several volatility measures have been explored in the literature, including historical and implied volatility, as well as volatility forecasts and volatility indices. Part of the difficulty with choosing the most appropriate measure is that each requires its own set of decisions, such as the time horizon for calculating historical volatility or the choice of forecasting methods. Moreover, option pricing should also account for varying volatility across different levels of moneyness and maturity.

In contrast to traditional option pricing models, modern-day research on option pricing introduces more data-driven approaches by applying machine learning methods, such as neural networks. As these methods can approximate any continuous function without restrictive assumptions (Hornik et al., 1989), they are well-suited for modeling the complex and non-linear relationship between option characteristics and option prices. Multiple works in the literature demonstrate that machine learning-based option pricing models can achieve better pricing accuracy than established pricing methods. Despite these findings, the adoption of machine learning-based pricing models in real financial markets has been slow, with one of the key reasons being their lack of interpretability. An options market maker or investor may be reluctant to act on model outputs when the factors determining these outputs are unknown. To address the issue of interpretability, explainable artificial intelligence (XAI) has gained recognition over recent years. However, applications to machine learning-based option pricing models remain rather limited. We intend to promote the adoption of XAI practices by applying these methods to our proposed pricing model. This not only helps us provide economic explanations of our model behavior, but also provides guidance for further model development.

The rest of this paper is structured as follows. Section 2 presents previous studies that lay the foundation for our work. Section 3 describes the data set used and how it has been processed. Section 4 details the implementation of the proposed LSTM-MLP option pricing model, and its training process. This section also outlines the methods for evaluating model performance in terms of pricing accuracy and risk-adjusted trading returns, as well as the XAI frameworks used for interpretability analysis. Section 5 presents the results for pricing, prediction and trading performance, and discusses these in comparison to benchmark models. Finally, Section 6 summarizes our findings and provides recommendations for future research.

2 Literature review

This section presents previous studies that lay the foundation for our research in the following order: Section 2.1 presents different option pricing models ranging from traditional analytical and numerical approaches to state-of-the-art machine learning methods. For the latter, we primarily focus on neural networks to stay consistent with the focus of our research. Section 2.2 discusses ways to incorporate volatility information in previously proposed neural network-based option pricing models. The volatility input is of particular interest as it cannot be directly observed in the market, and therefore often subject to debate about the most appropriate way to quantify it. Subsequently, Section 2.3 provides an overview of how previous studies evaluate the performance of neural network-based option pricing models. Finally, Section 2.4 covers XAI through various methods for interpretability analysis and provides examples of its applications.

2.1 Option pricing models

The most well-known option pricing model is widely regarded as the work by Black and Scholes (1973), and its extended version by Merton (1973). Contrary to the commonly held belief that Black and Scholes (1973) and Merton (1973) came up with a *new* model, their model is actually a special case of preceding work such as Sprenkle (1961), Boness (1964) and Samuelson and Merton (1969), among others (Bloch, 2023). More specifically, it is a theoretical economic argument based on a new way of deriving an already-existing formula (Haug and Taleb, 2011). In practice, the pricing model by Black and Scholes (1973) and Merton (1973) plays a key role for traders to manage option portfolios (Hull, 2023). Meanwhile, in academia, several shortcomings of the model have been discovered when comparing it to empirical evidence. This has driven the development of alternative methods that aim to better capture the dynamics in the options market.

Of the alternative methods that have been proposed, the most interesting to our research is the set of methods that address the assumption of constant volatility over the life of the option. This includes alternative option pricing methods that account for stochastic volatility, and thereby provide a more realistic representation of real market dynamics. Advances in this area include the work of Hull and White (1987), Heston (1993), and Hagan et al. (2002). Early works by Hull and White (1987) propose an option pricing model for when volatility is stochastic and uncorrelated to the asset. They later modify this to allow for volatility that is correlated to the asset in Hull and White (1988). Along similar lines, Heston (1993) develop a closed-form solution for European call options when correlations exist between the underlying asset and volatility. Hagan et al. (2002) introduce the SABR (stochastic alpha, beta, rho) model and claim that it captures the dynamics of the volatility smile. However, SABR implied volatility surfaces are found to not always align with market data (Fukasawa and Gatheral, 2021).

With regard to the state-of-the-art literature on option pricing, the application of machine learning methods has attracted the attention of many researchers. The emergence of this field of research is motivated by the ability of machine learning algorithms to learn non-linear relationships between input and output variables (Hornik et al., 1989), without necessarily being limited by economic and statistical assumptions as the traditional models (Ivaşcu, 2021). Most previous research on machine learning-based option pricing models focuses on neural networks (Ludkovski, 2023), of which multiple studies show that neural network-based models perform better than established benchmark models in terms of estimating the observed market price (Ruf and Wang, 2020). Among the neural network-based option pricing models, MLPs seem to be the most commonly implemented architecture. The work of Hutchinson et al. (1994) is one of the first studies to use neural networks for option pricing. They train an MLP with one hidden layer with four hidden units on simulated data and demonstrate that it is capable of learning the Black-Scholes formula with a high degree of accuracy. The practical relevance of the proposed model is tested by using it to price options on S&P 500 futures, for which it outperforms Black-Scholes. Following Hutchinson et al. (1994), several later works provide supportive evidence of better performance by MLPs compared to traditional option pricing methods (Amilon, 2003; Gençay and Gibson, 2007; Andreou et al., 2008; Wang et al., 2012; Culkin and Das, 2017; Fadda, 2020; Iltüzer, 2022). These studies propose a variety of MLP architectures using different combinations of input variables, different

methods to quantify volatility, or deeper neural networks compared to the earliest studies. Bennell and Sutcliffe (2004) implement a number of MLP pricing models to test different combinations of the standard Black-Scholes input variables. They find that the MLP generally performs better than Black-Scholes, especially for out-of-the-money options. Further analysis shows that the MLP overprices options that are deep in-the-money and underprices options with a long maturity. By excluding these options from the sample space, the MLP pricing performance becomes comparable to Black-Scholes.

Different volatility measures for neural network-based option pricing models are explored in Amilon (2003), Gençay and Gibson (2007), Andreou et al. (2008), Wang et al. (2012), Hsu et al. (2018), Fadda (2020) and Iltüzer (2022). This body of literature will be discussed separately in Section 2.2 given its relevance to our research. Furthermore, advancement in machine learning methods and computing has enabled the implementation of deep neural networks. Culkin and Das (2017) follow the approach in Hutchinson et al. (1994) by training and testing their model on simulated call option prices. In contrast to Hutchinson et al. (1994), they use a deep MLP consisting of four hidden layers with 100 units each and incorporate more input variables. Their results show that the proposed model learns the Black-Scholes option pricing model based on simulated data with high accuracy.

Besides MLPs, improved pricing performance is also demonstrated by other neural network architectures. Ruf and Wang (2020) notes that the overall trend in recent research is more complex architectures. Modular neural networks are proposed by Gradojevic et al. (2009) and Gradojevic (2016) in which each module is represented by a single MLP. Empirical results show that modularity improves the pricing performance compared to a standard MLP. The advantage of this architecture becomes clear when considering a data set that contains a highly volatile period followed by a relatively stable period. While it would be challenging for a single neural network to learn to generalize based on such different data sequences, a modular structure can overcome this by generalizing through interaction across modules. Gated neural networks are presented in Yang et al. (2017) and Cao et al. (2021). Both studies demonstrate that this particular architecture can be used to introduce economic intuition into the pricing models, for instance by enforcing the no-arbitrage principle. Yang et al. (2017) demonstrate high pricing performance with a multi-gated pricing model that jointly trains multiple individual gated pricing models. Improved pricing performance is also shown in Cao et al. (2021), although with a gated pricing model that uses a different set of input variables. Wei et al. (2021) propose both a stand-alone convolutional neural networks (CNN) pricing model, and an ensemble model that combines the CNN with three non-machine learning option pricing models, namely a stochastic volatility model, a jump diffusion model, and ad hoc Black-Scholes[4]. The stand-alone CNN outperforms each of the non-machine learning models, and high pricing accuracy is also reported for the ensemble model. Both the CNN and the LSTM proposed by Liang and Cai (2022) achieve better pricing performance than the benchmark models Black-Scholes, Merton, and Heston. Liang and Cai (2022) attribute the enhanced performance of the neural network-based pricing models to their ability to absorb time-series information embedded in the time-sequenced data.

Recently, options pricing has also been studied using machine learning methods other than neural networks (Ivaşcu, 2021; Sood et al., 2022; Gradojevic and Kukolj, 2022) In addition to neural networks, Ivaşcu (2021) also study the performance of support vector regression, genetic algorithms, decision trees, random forests, XGBoost and LightGMB. All machine learning algorithms outperform the benchmark models Black-Scholes and Corrado-Su across maturity and moneyness. Sood et al. (2022) implement pricing models based on MLPs, LSTM, support vector machines and XGBoost. They find that both the MLP and the LSTM outperform Black-Scholes, while the SVM and XGBoost fail to do so. Gradojevic and Kukolj (2022) study the pricing performance of various machine learning algorithms during the significant market changes, or 'regime switch', caused by Covid-19. They implement MLPs, support vector machines, random forests, and XGBoost. All machine learning models outperform Black-Scholes in more stable conditions pre-Covid-19, while the pricing performance by Black-Scholes becomes comparable to the machine learning models during Covid-19. This indicates that the regime switches to a certain extent limited the ability of

[4]The ad hoc Black Scholes pricing model estimates implied volatilities using the determined volatility function, which is a popular method for estimating implied volatility based on strike and maturity. For further details, see Wei et al. (2021)

the machine learning pricing models to learn and generalize.

2.2 Volatility measures for neural network-based option pricing models

As mentioned previously, appropriately quantifying volatility remains a challenge for both traditional and neural network-based option pricing methods. In the existing literature on neural network-based approaches, the most common volatility measures seem to be historical volatility (Hutchinson et al., 1994; Amilon, 2003; Hsu et al., 2018; Ivaşcu, 2021), implied volatility (Gradojevic and Kukolj, 2022), as well as GARCH and its extended versions. A possible drawback with these approaches is that the volatility is often not adapted to the moneyness and the maturity of the option being priced. This could therefore lead to inaccurate option prices that do not suitably reflect the range of possible outcomes at option expiry.

Historical volatility measures used in the existing literature cover a wide range of time periods. The lowest range identified is seen in Hsu et al. (2018) which proposes a deep neural network with historical 1-minute and 5-minute volatility to price monthly and quarterly options. However, no significant improvement in pricing accuracy is achieved by including these very short-term volatilities. Longer historical periods of 10 to 30 days are seen in Amilon (2003), and 60 days in Hutchinson et al. (1994) and Ivaşcu (2021). Iltüzer (2022) tests historical volatility over five different time horizons, including 360 days, 30 days and 10 days for calendar days and 21 and 252 days for trading days. Liang and Cai (2022) give their proposed LSTM model 5 lags of every input used, treating each input, including the stationary inputs strike and the linearly decreasing input of maturity, as time-series data. They argue that this allows the model to learn valuable time-series information from the past five trading days. In addition, the network gets the 90-day historical volatility on the past five trading days as inputs. Amilon (2003) also provide lagged values of the underlying asset for the last four days as inputs to their MLP pricing model. This is motivated by a similar hypothesis as our research about the potential of the pricing model to learn the volatility structure, or any useful distribution structure, from the historical data.

Among the studies that use implied volatility, different ways have been proposed for how to incorporate it into a neural network-based pricing model. Fundamentally, implied volatility is the value of the volatility variable in Black-Scholes in order to get an option price that matches the market price. In contrast to historical volatility which is backward-looking, implied volatility is considered to be forward-looking and therefore used to monitor market views about the volatility of a particular asset (Hull, 2022). Andreou et al. (2008) calculate the implied volatility for a specific option on day $t-1$ and use it to price the same option on day t. The MLP pricing model that uses this implied volatility appears to be the overall best-performing model. The superiority of pricing models using implied volatility is also demonstrated in Wang et al. (2012). However, it should be noted that they calculate implied volatility in a different way by using the equal-weighted average of implied volatilities for all traded options on day $t-1$ to price on day t. Along similar lines, Iltüzer (2022) achieves the best pricing performance with an MLP with implied volatility represented by the volatility of Black-Scholes that perfectly fits the closing price of the at-the-money call on day t. Fadda (2020) demonstrate that the MLP using implied volatility significantly outperforms the MLP using the GJR-GARCH model. However, a dual-volatility pricing model that takes both volatility measures as input outperforms both of the single-volatility pricing models.

Other neural network-based pricing models use volatility calculated using GARCH. Gençay and Gibson (2007) use GARCH(1, 1) as the volatility input to the proposed MLP pricing model that otherwise uses the same input variables as Black-Scholes. The proposed model performs better across all maturities and moneyness compared to benchmark models including Black-Scholes with historical volatility, Black-Scholes with GARCH volatility, stochastic volatility model, and stochastic volatility random jump model. Lin and Yeh (2009) show that a pricing model that uses GARCH outperforms pricing models that use historical, implied[5], or the Grey prediction approach. Various modifications of GARCH have also been able to achieve superior pricing performance, such as GM(1, 1)-GARCH (Wang, 2009b), Grey-GJR-GARCH (Wang, 2009a), and Grey-EGARCH

[5]Lin and Yeh (2009) estimates the implied volatility for day t as the average implied volatilities calculated from option prices on that day.

(Tseng et al., 2008).

In contrast to the aforementioned literature which all present option pricing models that use an explicit volatility input, some studies omit volatility and instead argue that the neural network is able to extract volatility information from alternative historical data during training. Yao et al. (2000) suggest that volatility is not required as input to their MLP pricing model when daily prices for the underlying asset are fed to the network. They propose that this should enable the daily returns, being the fundamental factor of volatility, to be captured by the network. A similar line of reasoning is expressed in Gradojevic et al. (2009), in which they suggest that the modular neural network pricing model proposed extracts volatility information during training using data for the first two quarters of each year. Ke and Yang (2019) replace the volatility input with outputs from an LSTM network. They motivate their proposed model architecture with the ability of recurrent neural networks, like the LSTM, to capture state information that can be more useful to option pricing than historical volatility. The MLP using outputs from the LSTM network achieves better pricing accuracy than Black-Scholes, but fails to outperform the MLP pricing model with historical volatility as an input. Motivated by their novel approach to incorporating volatility information into option pricing in this way, we show that a pricing model that combines an LSTM and an MLP component can indeed deliver superior pricing performance.

2.3 Performance measures for neural network-based pricing models

In their literature review on neural networks for option pricing, Ruf and Wang (2020) found that the Black-Scholes formula with historical volatility is the most common benchmark for pricing accuracy. This includes studies by Garcia and Gencay (2000), Amilon (2003), and Ivaşcu (2021). Benchmarking against Black-Scholes with implied volatility is seen in the works of Andreou et al. (2008), Iltüzer (2022), and Gradojevic and Kukolj (2022). Besides Black-Scholes, other alternative option pricing models have also been used for benchmarking, such as Heston (Jerbi and Chaabene, 2020; Liang and Cai, 2022; Wang et al., 2022) and Corrado-Su (Andreou et al., 2008; Ivaşcu, 2021). These methods arguably provide a fairer comparison to machine learning-based option pricing models as they provide more degrees of freedom compared to Black-Scholes. More recently, neural network-based pricing models have also been compared to other machine learning methods (Ivaşcu, 2021; Wang et al., 2022; Gradojevic and Kukolj, 2022).

To measure neural network-based option pricing models by their practical relevance, the literature proposes hedging (Hutchinson et al., 1994; Andreou et al., 2008; Cao et al., 2021) and trading implementations (Amilon, 2003; Andreou et al., 2008). Our research is centered on trading and hence this will be the focus of the following review. Amilon (2003) suggests, but does not implement, a trading strategy that buys (sells) options that are underpriced (overpriced) by the market when compared to model prices. He suggests that different pricing models can be ranked by comparing the sum of terminal profits or the standard deviation of the profits. A model that can successfully identify mispriced options would yield a positive value for the terminal position, and the model that yields the highest terminal value would be the preferred one. To the best of our knowledge, Andreou et al. (2008) is the only study on neural network-based option pricing models that implement trading strategies to evaluate model performance. They find that trading strategies based on both neural network-based and traditional pricing models are profitable for certain combinations of transaction cost and mispricing margin. The best models yield profits in 77-82% of the transactions made. Overall trading performance, measured by absolute profits, is similar across neural network-based and traditional models. The neural network-based models provide the most improvement compared to the traditional methods that use less sophisticated volatility measures, such as 60-day historical volatility. The trading strategy implemented by Andreou et al. (2008) is based on the single-instrument hedging approach presented in Bakshi et al. (1997), meaning only the underlying asset is used as the hedging instrument. Portfolios are created by buying (selling) undervalued (overvalued) options by comparing the option price predicted by the model to the market price. Additionally, they take a delta hedging position in the underlying asset by which a short (long) position in a call option is hedged using a long (short) position in the underlying asset. A position is held as long as the option is undervalued or overvalued, after which the position is liquidated, the profit or loss is computed, and a new position is entered based on current market

conditions. A limitation in the trading approach by Andreou et al. (2008) is the lack of other performance measures than absolute profits as they do not implement an actual trading portfolio.

2.4 Interpretable machine learning

Despite the promising results achieved by machine learning models exemplified by the literature presented above, a major drawback is that the insights learned by these models are hidden in their architecture, and hence they are often referred to as black box models. This lack of interpretability poses a challenge for real-life model deployment, and the importance of addressing the black box challenge associated with machine learning methods is highlighted by the emergence of XAI. At a basic level, interpretability in machine learning can be achieved by using inherently interpretable model architectures, such as regression models or decision trees. Another approach is to apply model-agnostic interpretability methods. In the following parts, we will focus on the latter approach as it is most suitable for the proposed LSTM-MLP model.

Model-agnostic interpretability methods work by changing the inputs to a machine learning model and measuring how it changes the output value (Molnar, 2022). The literature on this can be broadly categorized into global and local methods. Global methods explain the average model behavior using methods such as Partial Dependence Plot (Friedman, 2001) and Accumulated Local Effects (ALE) (Apley and Zhu, 2016). On the other hand, local methods explain single instances of model output, as achieved by Local Interpretable Model-Agnostic Explanations (Ribeiro et al., 2016) and SHapley Additive exPlanantions (SHAP) (Lundberg and Lee, 2017).

In the context of option pricing, the literature that applies XAI remains rather limited. Liang and Cai (2022) use ALE to interpret their proposed option pricing models. For the MLP pricing model used to price call options on the S&P 500 index, they find that the strike is the most important input, followed by the underlying asset and the time to maturity has relatively less influence. In comparison, the risk-free rate and volatility appear to have almost no influence. They also conduct the interpretability analysis on option pricing models using convolutional neural networks and LSTM and find that feature importance differs across models. Gradojevic and Kukolj (2022) apply SHAP in their study to the proposed option pricing models and investigate how feature importance changes across market regimes caused by Covid-19. They find that moneyness is the most important feature across regimes, and more precisely that option prices increase as moneyness increases. Less contribution is shown for all other input features including time to maturity, implied volatility, open interest, and volume.

3 Data

3.1 Data set

The primary data set for our research consists of daily closing prices and relevant attributes of SPX European call options on the S&P500 index, provided by optionsDX[6]. The data set starts on November 1st, 2011 and on December 31st, 2022. The data before January 1st, 2015 is used exclusively for training and validation for hyperparameter tuning. To create the initial sequence of 140 lagged S&P500 returns required by the model, the S&P500 data series begins on April 14th, 2011. The complete data set consists of 13,108,756 observations, each representing a specific option contract on a particular date.

For the risk-free rate, we use daily Treasury yields from the U.S. Department of the Treasury[7].

3.2 Data preprocessing

The data preprocessing stage removes approximately 13.58% of the initial dataset, leading to a final total of 11,328,707 observations. The initial screening process eliminates 3,582 data points due to invalid or missing values in the options data. Furthermore, we remove 249,566 data points representing options with zero time to maturity, as their pricing, equating to their intrinsic value[8], is trivial and does not contribute to our research objectives. An additional 285,293 data points, representing options with more than two years of time-to-maturity (TTM), are also removed. To counteract the influence of extreme moneyness values, we exclude options that fall approximately within the top and bottom 5% of moneyness, resulting in the removal of another 1,241,608 data points with moneyness below 0.8 or above 2.0.

The primary rationale behind this filtering strategy is to mitigate the impact of outliers that could potentially disrupt model learning. The options removed from the dataset, often less traded and potentially less efficiently priced, fall outside the typical moneyness and maturity ranges. Consequently, these options restrict the efficacy of data-driven methods in learning their pricing dynamics and may also be of less interest to market participants, given their lower trading volumes and quantities. Table 3.1 provides more detailed insights into the trading volumes associated with these filtering criteria.

Table 3.1
Average daily trading volume per contract, where each contract represents 100 options, segmented by the filtering criteria

	Criterion	Average Volume
TTM	> 2	13.216
	< 2	24.144
Moneyness	> 2.0	7.045
	< 0.8	20.922
	$0.8 - 2.0$	25.245

The interest rate data from the U.S. Department of the Treasury specifies rates only for distinct maturities: 1, 3, 6, 12, and 24 months. To derive the risk-free rate applicable to the time to maturity of each option, we apply the Nelson-Siegel-Svensson (NSS) method, allowing us to construct continuous risk-free rate curves. A more in-depth explanation of the NSS method applied can be found in Appendix A.

[6] https://www.optionsdx.com
[7] https://home.treasury.gov
[8] Intrinsic value = Underlying - Strike

3.3 Descriptive statistics

Table 3.2 presents descriptive statistics by year. There is a general upward trend in the mean option price, S&P 500 index, and strike price, with an observable increase in their variability, which signifies heightened market volatility in later years. Time to maturity and moneyness exhibit relatively similar behavior throughout the period. The risk-free rate is generally increasing in the latter years, except for a downturn in 2020 and 2021. The total number of available options increases through the period except from 2021 to 2022.

Table 3.2
Descriptive statistics of all options data used for hyperparameter search, training, validation, and testing

Year	Measure	Option price	S&P 500	Strike	TTM (Yrs)	Rate (%)	Moneyness
2011	count	33,080	33,080	33,080	33,080	33,080	33,080
	mean	192.32	1236.11	1098.89	0.32	0.03	1.18
	std	173.97	27.61	228.79	0.39	0.05	0.28
2012	count	259,343	259,343	259,343	259,343	259,343	259,343
	mean	191.6	1,383.49	1,239.55	0.34	0.10	1.17
	std	187.5	45.79	245.32	0.42	0.05	0.26
2013	count	345,503	345,503	345,503	345,503	345,503	345,503
	mean	227.38	1,653.16	1,462.91	0.32	0.07	1.17
	std	218.48	98.33	276.85	0.4	0.06	0.24
2014	count	554,101	554,101	554,101	554,101	554,101	554,101
	mean	277.43	1,942.64	1,700.84	0.28	0.05	1.18
	std	250.94	80.19	302.33	0.38	0.07	0.24
2015	count	869,602	869,602	869,602	869,602	869,602	869,602
	mean	294.05	2,059.37	1,812.5	0.25	0.08	1.18
	std	268.59	56.83	326.75	0.32	0.13	0.24
2016	count	811,805	811,805	811,805	811,805	811,805	811,805
	mean	275.46	2,097.31	1,870.22	0.24	0.33	1.16
	std	272.01	102.25	338.59	0.34	0.14	0.24
2017	count	919,659	919,659	919,659	919,659	919,659	919,659
	mean	280.88	2,451.22	2,211.72	0.21	0.92	1.14
	std	299.24	111.24	358.84	0.33	0.26	0.22
2018	count	1,249,550	1,249,550	1,249,550	1,249,550	1,249,550	1,249,550
	mean	299.29	2,745.12	2,525.13	0.23	1.97	1.12
	std	330.92	101.72	404.58	0.32	0.34	0.22
2019	count	1,317,850	1,317,850	1,317,850	1,317,850	1,317,850	1,317,850
	mean	343.64	2,917.99	2,644.67	0.23	2.09	1.14
	std	358.96	155.2	445.94	0.32	0.34	0.22
2020	count	1,454,860	1,454,860	1,454,860	1,454,860	1,454,860	1,454,860
	mean	408.38	3,219.94	2,928.58	0.24	0.38	1.14
	std	404.78	314.85	553.86	0.32	0.55	0.23
2021	count	1,762,540	1,762,540	1,762,540	1,762,540	1,762,540	1,762,540
	mean	526.87	4,295.38	3,887.66	0.28	0.05	1.14
	std	525.04	286.16	695.49	0.34	0.05	0.22
2022	count	1,750,820	1,750,820	1,750,820	1,750,820	1,750,820	1,750,820
	mean	397.96	4,108.23	3,916.73	0.27	2.00	1.08
	std	456.77	295.95	639.11	0.34	1.52	0.20

Figure 3.1 and Figure 3.2 illustrate the dataset characteristics across the dimensions of time to maturity and moneyness to provide context for the discussions in later sections. Notably, our dataset contains a higher density of shorter-maturity options. This observation may not only be attributable to the fact that all options transit through these shorter maturities during their lifecycle, but also to the increased writing of options with shorter maturities. When inspecting moneyness, we observe a concentration of data points around at-the-money options. As for trading volumes, the data indicates higher volumes for the shortest-maturity and at-the-money options, with a substantial decrease in volume for options more deeply in-the-money.

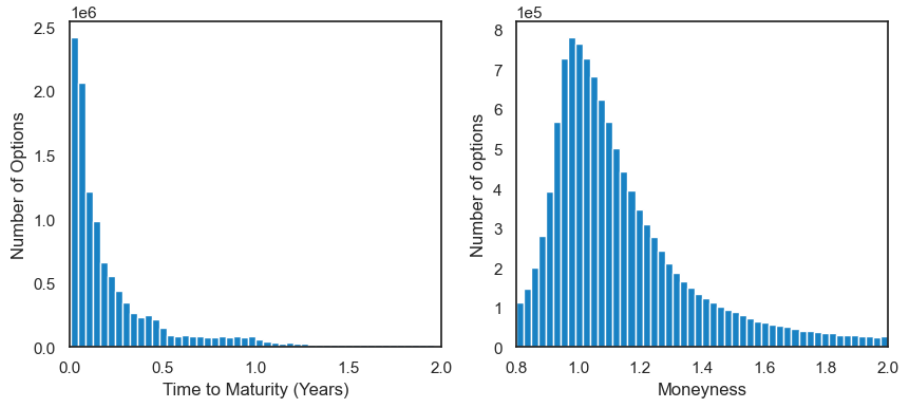


Figure 3.1 Histogram for the number of options quotes at different levels of maturity and moneyness in the processed data. The left figure shows the maturity distribution and the right figure shows the moneyness distribution.

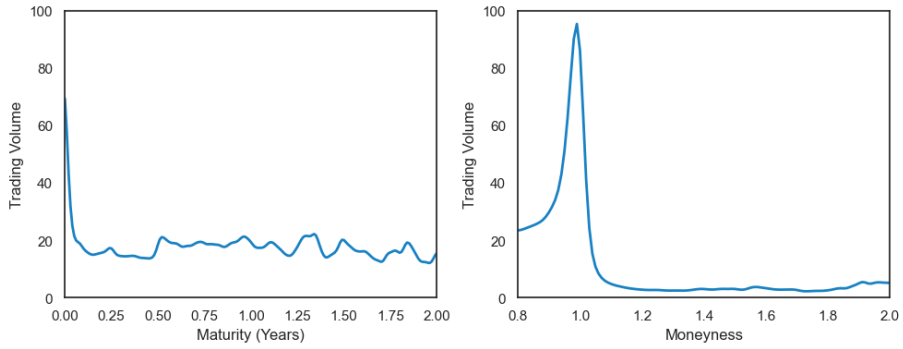


Figure 3.2 Average daily trading volume per contract, where each contract represents 100 options, for each level of maturity and moneyness. The left figure shows the volume based on maturity and the right figure shows the volume based on moneyness.

4 Methods

4.1 Combined LSTM-MLP option pricing model

4.1.1 Model implementation

The proposed option pricing model combines an LSTM network with an MLP, as illustrated in Figure 4.3. The LSTM network takes as input historical S&P 500 returns over the past 140 days and outputs information extracted from the time series data through eight output nodes. The MLP takes as input the LSTM outputs and the option characteristics including the price of the underlying asset, the strike, the risk-free rate, and the time to maturity. The final output of the combined LSTM-MLP is the option price[9]. The model hyperparameters are decided based on results from the hyperparameter search presented in Section 4.1.3.

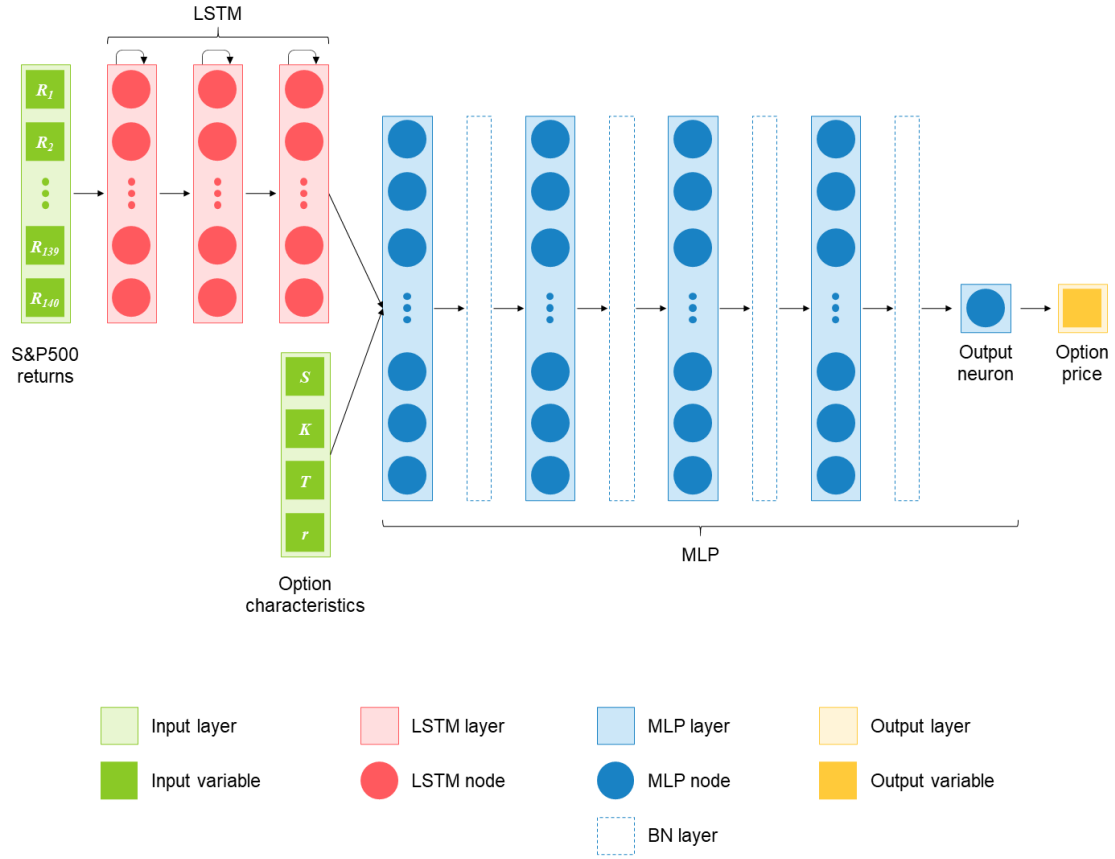


Figure 4.3 Architecture of the combined LSTM-MLP option pricing model. R_i is the return on the S&P 500 index i days ago. S is the price of the underlying asset. K is the strike. T is the time to maturity. r is the risk-free rate. The illustration is inspired by Ke and Yang (2019).

[9]Following Yang et al. (2017), Liang and Cai (2022) and others, the bid-ask midpoint is used as a proxy for the market price. This enables comparison with the benchmark models as these do not output bid and ask prices separately.

The LSTM network is made up of three layers, each consisting of eight nodes each. Each node is composed of memory cells with an internal structure as shown in Figure 4.4.

Central to the LSTM network (Hochreiter and Schmidhuber, 1997) is the cell state, as illustrated by the horizontal line at the top of the memory cell. This can be thought of as the long-term memory component. Updates to the cell state are regulated by three types of gates: an input gate i_t, a forget gate f_t and an output gate o_t. The subscript t represents a single time step. For short-term memory, information from the previous calculation step is stored in the hidden state, as illustrated by the horizontal line at the bottom of the memory cell.

The computation for updating the cell state at a specific time step is based on the previous cell state C_{t-1}, the previous hidden state h_{t-1}, and the current input x_t. The first step of the computation is to decide what information to throw away by using the forget gate as given by Equation (4.1). The previous hidden state h_{t-1} and the current input x_t are passed through the sigmoid function, which outputs values between 0 and 1 for each number in the previous cell state C_{t-1}. An output of 0 means that the information should be thrown away, and an output of 1 means that the information should be kept. The results from the forget gate are then multiplied by the cell state, as given by the first term of the addition in Equation (4.4).

$$f_t = \sigma(W_{f,x}x_t + W_{f,h}h_{t-1} + b_f) \tag{4.1}$$

where σ denotes the sigmoid function, $W_{f,x}$ and $W_{f,h}$ denote weight matrices, and b_f denotes the bias term.

In the next step, the input gate is used to decide what information to store. This is done through a two-step process. First, the input gate decides which values to update, as given by Equation (4.2). Then, a vector of new candidate values $\tilde{C}_t$ is generated using tanh, as given by Equation (4.3). To create an update to the state, we combine the outputs from these two steps, as given by the second term of the addition in Equation (4.4).

$$i_t = \sigma(W_{i,x}x_t + W_{i,h}h_{t-1} + b_i) \tag{4.2}$$

$$\tilde{C}_t = tanh(W_{\tilde{C},x}x_t + W_{\tilde{C},h}h_{t-1} + b_{\tilde{C}}) \tag{4.3}$$

where $W_{i,x}$, $W_{i,h}$, $W_{\tilde{C},x}$ and $W_{\tilde{C},h}$ denote weight matrices, and b_i and $b_{\tilde{C}}$ denote bias terms.

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{4.4}$$

where $\odot$ represents element-wise multiplication.

Finally, the new hidden state h_t can be seen as a filtered version of the cell state C_t. First, a sigmoid layer decides which parts of the cell state to output, as given by Equation (4.5). Then, tanh is applied to the cell state C_t to transform its values between -1 and 1. These two outputs are multiplied so that only the desired parts are given as output, as given by Equation (4.6).

$$o_t = \sigma(W_{o,x}x_t + W_{o,h}h_{t-1} + b_o) \tag{4.5}$$

$$h_t = o_t \odot tanh(C_t) \tag{4.6}$$

where $W_{o,x}$ and $W_{o,h}$ denote weight matrices, and b_o denotes the bias term.

The exact number of interface units, equivalent to nodes in the final LSTM layer, is specifically investigated in the hyperparameter search to determine the optimal number of units to encode the information passed by the LSTM to the MLP component.

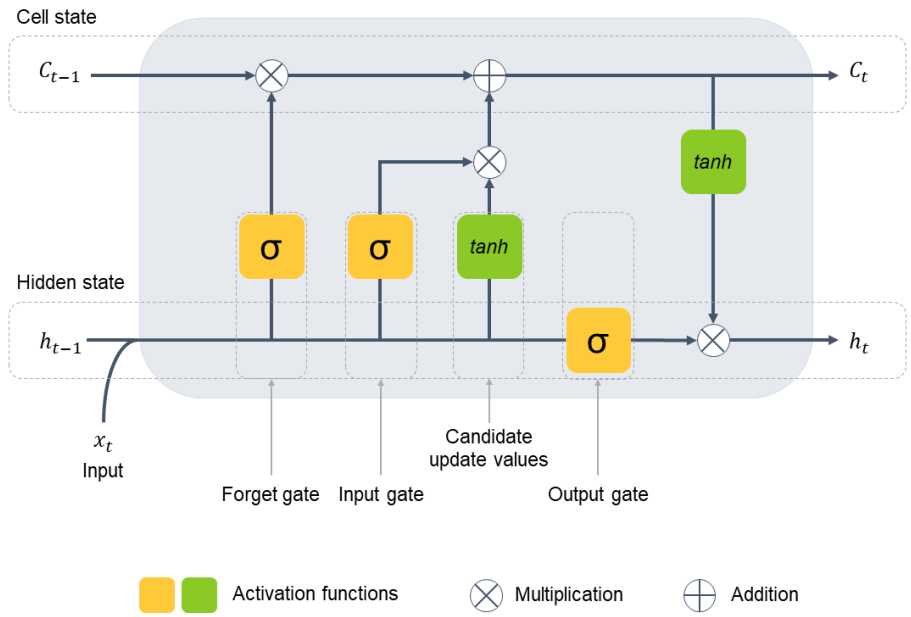


Figure 4.4 Structure of an LSTM memory cell at a specific time step t. C_t is the cell state. h_t is the hidden state. x_t is the input. σ is the sigmoid function. The illustration is inspired by Olah (2015).

The MLP component of the LSTM-MLP model consists of four stacking structures, each consisting of a dense layer followed by a batch normalization layer. The dense layer forms a fully-connected network with the nodes of the preceding layer. Batch normalization layers are implemented between each dense layer to normalize the activations of the preceding layer in each batch. This strategy addresses common challenges in deep neural networks, such as vanishing or exploding gradients, curbs overfitting, and accelerates the training process by reducing the number of iterations required for convergence. The BN normalization algorithm is presented below. For further details, see Ioffe and Szegedy (2015).

Input: Values of x over a mini-batch $B = \{x_1, ..., x_m\}$

Parameters to be learned: γ and β

Output: $y_i = BN_{\gamma,\beta}(x_i)$

$$\mu_\beta \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{Mini-batch mean}$$

$$\sigma_\beta^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_\beta)^2 \qquad \text{Mini-batch variance}$$

$$\hat{x}_i \leftarrow \frac{x_i - \mu_\beta}{\sqrt{\sigma_\beta^2 + \epsilon}} \qquad \text{Normalize}$$

$$y_i \leftarrow \gamma\hat{x}_i + \beta \equiv BN_{\gamma,\beta}(x_i) \qquad \text{Scale and shift}$$

where ϵ is a small constant for numerical stability.

The MLP uses leaky ReLu as its activation function, as given by Equation (4.8). Unlike the conventional ReLu function given in Equation (4.7), leaky ReLu replaces zero values with small negative values. This design decision mitigates the "dead neuron" problem that can arise with regular ReLu. The "dead neuron" problem occurs when a neuron gets stuck during the learning process and only outputs zero, effectively rendering it inactive in any network-wide computation or learning. By allowing for small negative values instead of zero, the leaky ReLu function ensures that these neurons remain active and continue learning, which can lead to a more robust and effective network Maas et al. (2013). For the final dense layer, which consists of a single output node, the

conventional ReLu activation function is used to ensure that the final output is a non-negative option price.

$$\text{ReLu (x)} = \begin{cases} x & \text{if } x \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{4.7}$$

$$\text{Leaky ReLu (x)} = \begin{cases} x & \text{if } x \geq 0 \\ \alpha x & \text{otherwise} \end{cases} \tag{4.8}$$

where α is a small positive constant found during the hyperparameter search.

A key strength of the LSTM-MLP architecture is the direct connection between the LSTM network and the MLP. This collaborative design enables training and backpropagation throughout the entire network as a unified entity. In other words, extracting time series information is not treated as a separate process from the pricing of the options, but rather as one integrative task to be solved by the LSTM-MLP. The intention of this particular setup is to align the training of the LSTM network and the MLP in terms of maximizing pricing accuracy. Rather than training the LSTM network to extract pre-specified measures from the time series data it receives as input, it should instead learn to output the information that is optimally useful for the MLP to price options.

Another strength of the model architecture is that it eliminates the need for an explicit volatility input to an option pricing model. Instead, the LSTM-MLP is trained to extract information directly from the historical return series for the underlying S&P 500 index. While the precise nature of the information extracted by the LSTM remains uncertain, it seems reasonable to hypothesize that volatility information is embedded within the time series information extracted. The historical return series could also contain other market information, such as directional expectations or other factors influencing option pricing behavior by market participants. Nonetheless, we hypothesize that the LSTM-MLP can learn the appropriate volatility measures used by market participants during its training period, as well as tailor the volatility measure for specific option characteristics as there are multiple connections between the LSTM and MLP components. This integrative model structure thereby demonstrates a potential approach for addressing the inherent complexities associated with volatility representation in option pricing, while also allowing for additional information extraction from the time series of the asset returns.

While standalone LSTM models, as examined in Liang and Cai (2022) and Liu and Wei (2022), are indeed capable of capturing time series information, they require all input variables to be lagged. This requirement introduces complexity and computational cost. Computational expenses may outweigh the benefits in predictive accuracy when lagging constants such as the strike price, linearly decreasing variables like maturity, or variables with infrequent changes like the risk-free rate. This could limit the feasible size of the network. Despite not explicitly citing computational constraints, Liang and Cai (2022) only consider five lags in their LSTM implementation, supplementing it with a historical volatility input. This implies that handling an array of lagged variables over extensive time series in large datasets can become computationally burdensome, and potentially unfeasible. A remedy to this problem lies in the integration of LSTM with MLP in our proposed model. The LSTM segment captures temporal dependencies within the time series data, while the MLP component efficiently processes option data at a given time, capturing the complex nonlinear relationships inherent in option characteristics. This combined architecture thus streamlines computational resources and enhances model performance.

In order to feed the input data to this model architecture, it is structured in two parts. For the LSTM model to utilize time series information, it needs data in a time-sequenced format. This results in a 3D format of its input consisting of the number of options, number of time steps and number of features, where the latter is only the returns in our case. The MLP component requires a 2D input with the option characteristics for each option. These two components together constitute our model input. The result of this data processing, along with the target value of the call option price, is illustrated in Figure 4.5

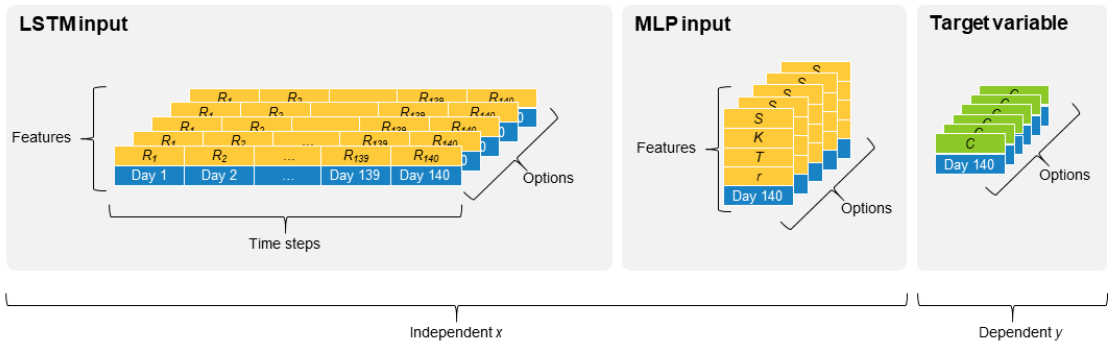


Figure 4.5 Data structure for the LSTM-MLP model. The MLP component of the model also receives the output from the LSTM component directly in addition to the external variables of the option characteristics.

4.1.2 Model training

The proposed LSTM-MLP model is trained and tested using sliding windows, following the approach in Andreou et al. (2008) and Cao et al. (2021). Compared to their implementation, our setup is different in that we use a greater number of sliding windows and cover a longer time period overall. Additionally, we propose a novel combination of the length of the training, validation and test set. This is illustrated in Figure 4.6, in which each model instance corresponds to a single sliding window. Each model instance uses a 3-year training set, a 1-month validation set, and a 1-month test set. As an example, the model that prices options in January 2015 is trained on data from December 2011 to November 2014, and validated on data for December 2014. Then, to price options in February 2015, the training and validation window slides forward by one month. This sliding procedure continues until the end of the sample period in December 2022 and thereby creates 96 sliding windows in total. In a real-life setting, a market participant could retrain the model at a higher frequency as needed, such as on a daily or weekly basis. We have chosen to retrain every month in our research to limit the computing time.

Our implementation of sliding windows for model training and testing provides several advantages. First, it keeps the training data up-to-date, such that it is relevant for and representative of the out-of-sample options to be priced. At the same time, we also provide sufficient historical training data for the model to learn the past pricing behavior of the market. The model should therefore be able to adapt to changing market conditions. This is key given the time-inhomogeneity inherent in financial data (Ruf and Wang, 2020), where the statistical properties and underlying dynamics of the data can change over time. Second, it reinforces the robustness of our proposed model architecture by testing across multiple time periods. With this setup, we thereby intend to improve upon most previous studies which only assess model performance based on one or a few out-of-sample test periods. Thus, our out-of-sample results obtained through this training and testing procedure provide a stronger argument for the generalization capability of our model.

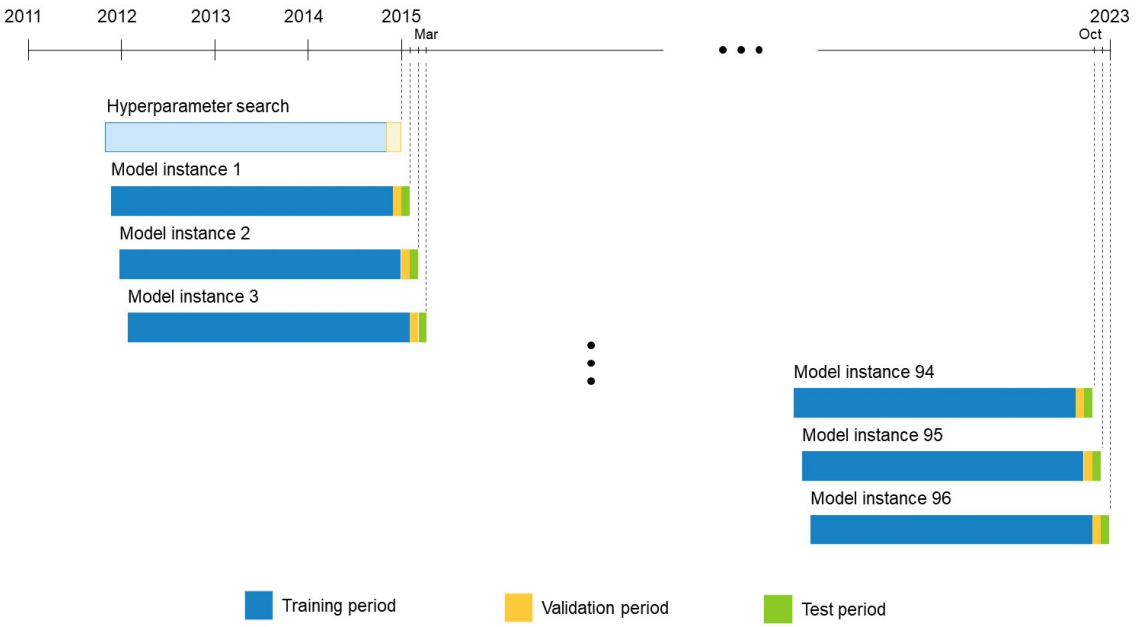


Figure 4.6 Sliding windows setup for model training and testing. Each of the 96 separate model instances uses a 3-year training set, a 1-month validation set, and a 1-month test set. The full out-of-sample test period runs from 2015 through 2022.

In order to ensure consistency across training periods and expedite the training process, we fix the number of training samples at 1,000,000. This utilizes most of the available options in the first sliding window shown in Figure 4.6. As the number of options in the market grows in subsequent periods, we randomly sample 1,000,000 options based on a uniform distribution from each training set. This approach provides a balanced and representative sample for training and facilitates computational efficiency. For validation and testing, we utilize all options in the data set for the one-month periods.

Furthermore, to prevent any potential biases in the training process, data points corresponding to specific option contracts in the training set are randomly shuffled. This ensures that the model performance is not impacted by the order in which the data points are presented during training. To avoid any confusion, it should be noted that the lagged data points provided as input to the LSTM network are not shuffled. These are always kept in chronological order for the past 140 days from the pricing date of the given option, which is necessary for the LSTM to accurately capture temporal dependencies.

The learning process incorporates an exponentially decaying learning rate to optimize parameter tuning, as expressed in Equation (4.9). This approach promotes faster learning in the early stages, while minimizing the risk of bypassing the global minimum during gradient descent. Furthermore, it facilitates refined parameter adjustments as training evolves. This decay mechanism encourages substantial initial changes to weights and biases when the model is far from the optimal solution. As the model approaches the optimal solution, adjustments become smaller and more cautious. This strategy strikes a balance between rapid convergence and the precision of the model's performance.

$$Lr_{epoch} = Lr_{initial} \times \text{decay rate}^{epoch} \tag{4.9}$$

To ensure that the features are on a similar scale and to avoid any potential issues with the training process, the features are scaled. The scaler is fitted and used on the training set and then applied to the validation and test sets, ensuring that the scaling is consistent across all data sets and that there is no data leakage. Min-max scaling is used to scale each input value between 0 and 1, as given by Equation (4.10).

$$X_{scaled} = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{4.10}$$

Model training begins with randomly generated initial weights for the LSTM-MLP model. Through an iterative process, these weights are continuously calibrated using the Adam optimizer, a variant of stochastic gradient descent, where mean squared error (MSE) serves as the objective function. For an in-depth exploration of this technique, see Kingma and Ba (2014). Early stopping is implemented and stops the model training if there is no improvement in validation loss for 20 consecutive epochs. To ensure that the best model configuration is retained, we utilize checkpointing during the final model training, which saves the model configuration each time a reduction in validation loss is observed. This process continues iteratively, gradually fine-tuning the model performance.

4.1.3 Hyperparameter configuration

In developing machine learning models, the choice of hyperparameters is important because it can have a meaningful impact on model performance. Therefore, to ensure a systematic and efficient approach, we conduct our hyperparameter search using Wandb[10], a machine learning development tool for experiment tracking. Wandb is useful in that it keeps a detailed record of model runs and facilitates parallel exploration of multiple hyperparameter configurations. This is highly beneficial for the proposed LSTM-MLP architecture due to the significant number of hyperparameters that need to be specified.

The hyperparameter search for the proposed LSTM-MLP is conducted using a window ending in December 2014, as illustrated in Figure 4.6. The validation period is purposefully set before the first out-of-sample test period in January 2015 to avoid data leakage between validation and out-of-sample testing. To match the total duration of the one-month validation and one-month test period in the 96 sliding windows used for out-of-sample testing, a two-month validation period is used for the hyperparameter search. With this setup, the hyperparameter search uses training data for the three years from November 2011 to October 2014, and validation data for the two months from November to December 2014.

The ranges of hyperparameter values to be searched over are determined based on values that have been used in previous studies, including Ke and Yang (2019), Liu et al. (2019), Liang and Cai (2022). Further adjustments to these ranges are made based on observed results. Due to the extensive number of hyperparameters, it is not computationally feasible to conduct a grid search testing all possible combinations. Exploration of the hyperparameter space is therefore carried out using a random search algorithm where 150 unique combinations of hyperparameters are generated and tested. Figure 4.7 shows the model run for each unique combination of hyperparameters and the resulting validation loss.

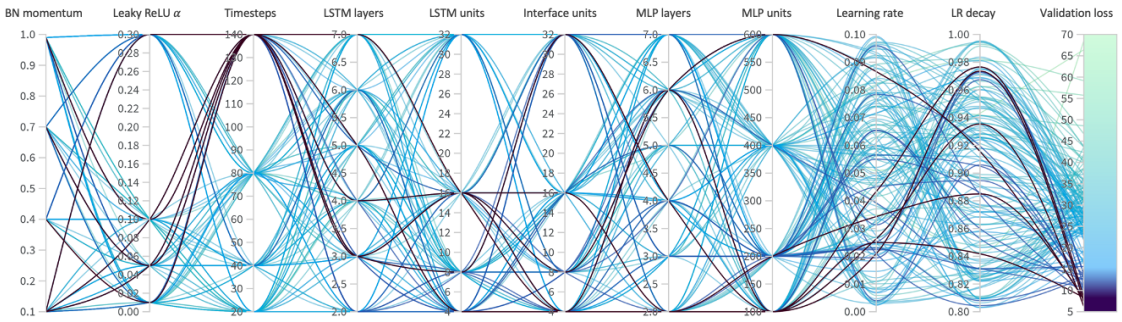


Figure 4.7 Hyperparameter search for the LSTM-MLP model in terms of hyperparameter combinations, hyperparameter value ranges and MSE validation loss. Results are shown for model runs testing 150 distinct combinations of hyperparameter values. The data set used consists of training data for the three years from November 2011 to October 2014, and validation data for the two months from November to December 2014.

Interestingly, the hyperparameter search shows a clear preference for a higher number of LSTM timesteps, meaning more lagged values of S&P returns. It seems intuitive that providing more

[10]https://wandb.ai/

historical data can improve model performance, assuming that the additional information does not introduce too much unnecessary noise or lead to excessive overfitting. This is where the strength of the LSTM network comes in, with its ability to selectively forget irrelevant historical data while retaining valuable information. The clear preference for longer periods of historical data provides support for our hypothesis that the LSTM is capable of extracting useful information from long-term patterns in historical data.

A limitation of our current approach to setting hyperparameters is that the same values are applied to all 96 model instances across the entire test period. This could potentially lead to suboptimal results, as different market conditions might require different hyperparameters for optimal performance. For example, the same hyperparameters might not have been suitable before and after the onset of the Covid-19 pandemic. Despite this, we have opted for to use the same hyperparameters for all model instances due to computation limitations. To improve our process, we could consider conducting separate hyperparameter searches for each of the 96 model instances or performing a search at fixed intervals, like every six months. Alternatively, we could keep the hyperparameters constant across model instances until a significant change in market conditions necessitates a new hyperparameter search.

The final hyperparameters are chosen based on inspection of Figure 4.7 and analysis of the highest performing runs during the hyperparameter search. The parameters chosen and configuration choices made based on the hyperparameter search and discussion in Section 4.1 are summarized in Table 4.3.

Table 4.3
LSTM-MLP model configuration grouped by model component. The parameters layers, units, interface units, timesteps, BN momentum, alpha constant, learning rate and learning rate decay were found during the hyperparameter search using Wandb while the others were determined manually beforehand to reduce the hyperparameter space.

	Parameter	Value
LSTM	Layers	3
	Units per layer	8
	Units in the last layer (Interface units)	8
	Lagged returns (Timesteps)	140
	Activation function	Tanh
	Recurrent activation function	Sigmoid
MLP	Layers	4
	Units per layer	200
	Batch normalization momentum	0.10
	Activation function	Leaky ReLu
	Alpha constant for Leaky ReLu	0.05
Training	Training set samples	1,000,000
	Learning rate	5.7×10^{-2}
	Learning rate decay	0.91
	Minibatch size	4 096
	Epochs	Early stopping
	Optimizer	Adam
Sliding windows setup	Training length	3 years
	Validation length	1 month
	Test length	1 month
	Number of windows	96
Computations	Hardware	Nvidia Tesla T4 GPU
	Hyperparameter search run time	26 hours
	Final model average run time per window	13 min
	Total run time all windows	21 hours

4.2 Backtesting trading strategy based on price differences

Evaluating an option pricing model based on its ability to accurately estimate the observed market price implies that the observed market price is the most accurate option price. Fundamentally, an accurate option price can be defined as the discounted risk-neutral[11] expectation of its payout at maturity (Blyth, 2014). As an academic exercise to investigate whether our proposed model can price options more accurately than the market per this definition, we implementing trading strategies based on the difference between the model price and the market price. Following the trading implementation in Andreou et al. (2008), we trade by buying (selling) options that are undervalued (overvalued) by the market when compared to the model price. This approach also enables us to demonstrate the ability of the proposed model to achieve positive risk-adjusted returns, thereby strengthening our argument about its practical significance.

The technical specifications of the trading implementation include the following:

1. The trading portfolio is initialized with a starting capital of $1,000,000
2. Trading signals are generated when the difference between the model price and the observed market price is greater than 15%[12]
3. Options are bought at the ask price and sold at the bid price with a mispricing margin of 15%. A buy signal is generated if the model price is 15% above the ask price, and a sell signal is generated if the model price is 15% below the bid price
4. Each buy or sell position the trading strategy takes is for an amount equivalent to 0.005% of the available cash in the portfolio at the time of the trading signal[13]
5. If an option is bought (sold) on day t, it cannot be bought (sold) again on subsequent days
6. If an option has been bought (sold) on day t, it can be sold (bought) again on subsequent days if the price passes sell (buy) threshold, i.e. it becomes mispriced in the opposite direction
7. A stop-loss threshold is set at 500%. This implies that when going short, if the bid price is 500% higher than the price originally sold for, the algorithm will buy back the option for the current ask. This will not be triggered when going long as it is only possible to lose 100% of the invested amount in that case
8. The difference in the number of long and short positions cannot exceed 10% of the total number of positions. If the difference exceeds 10%, only new positions that lower this percentage are permitted. This rule is implemented to limit the directional exposure in the portfolio, ensuring that the model primarily generates profits from market mispricings rather than directional bets
9. A transaction cost of 0.5% is imposed[14]. This is meant to represent the exchange fees, clearing fees, regulatory fees, and technology and infrastructure a market participant would have to pay when deploying a trading strategy like this
10. Options with a mid-price under 50 cents are disregarded to avoid extreme percentage fluctuations

To bridge the trading implementation from an academic exercise to real-life deployment, we identify three main simplifications made that could prevent a market participant from achieving the same trading results showcased in our research. First, with regard to slippage, our implementation assumes infinite liquidity at every bid and ask price. In reality, the liquidity at a specific price

[11] The concept of risk-neutral probability means that these probabilities depend on the price the underlying asset can take in different states of the world, but not on the actual probability of these states occurring (Blyth, 2014)

[12] Andreou et al. (2008) show that a mispricing margin of 15% yields the highest absolute returns for their trading implementation. They also observe that the P&L increases in a diminishing fashion for higher mispricing thresholds, indicating that there is an optional threshold for maximizing trading profits.

[13] In practice, this might not be entirely realistic as it requires the options trader to buy a discrete number of options to exactly fit the specified percentage position.

[14] Andreou et al. (2008) find that the best strategies retained profitability up to transaction costs of 0.5%.

may be limited such that the rest of the position is executed at a worse price. Secondly, we do not incorporate the market impact of trading decisions in the sense that trades on one day do not affect the price on the next day. It is worth highlighting that the effect of both of these limitations depends on the transaction size. Third, we do not impose margin requirements. The only limitation of writing options we currently have, is that there needs to be a balance of going long and short options as described above. At the same time, the portfolio has significant cash at all times, which could act as collateral. However, in real life, more formal collateral and margin requirements to write options will be required than what we have implemented.

4.3 Benchmark models

4.3.1 Black-Scholes

First introduced by Black and Scholes (1973), the Black-Scholes model (BS) defines the price for a European call option as:

$$C = S\,N(d_1) - Ke^{-rt}N(d_2) \tag{4.11}$$

$$d_1 = \frac{log(S/K) + (r + \sigma^2/2)/\tau}{\sigma\sqrt{\tau}} \tag{4.12}$$

$$d_2 = \frac{log(S/K) + (r - \sigma^2/2)/\tau}{\sigma\sqrt{\tau}} = d_1 - \sigma\sqrt{\tau} \tag{4.13}$$

where S is the current price of the underlying asset, K is the strike, τ is the time to maturity, σ is the volatility of the underlying asset, r is the risk-free interest rate and $N(\cdot)$ is the cumulative normal distribution.

Despite its widespread use in practice, a straightforward implementation of the BS model can underperform due to its reliance on potentially biased information. In contrast, our LSTM-MLP model is trained with actual options prices, which allows it to potentially capture higher-order moments. The BS model, notably, is sensitive to the choice of volatility measure, which is the only input variable that is unobservable. This issue is compounded by the inability of the model to adjust its functional form, unlike neural networks (Anders et al., 1998). Consequently, the same volatility measure must be applied to all options, leading to inherent inflexibility. Despite these limitations, the BS model is widely employed as a benchmark due to its widespread acceptance and understanding within the financial community (Hutchinson et al., 1994; Ruf and Wang, 2020).

To provide a fairer comparison, we use three different versions of the BS model as our benchmark models, each incorporating a different volatility measure: historical, implied, or volatility predicted using GARCH(1,1).

Historical volatility

Historical volatility is computed from the past 30 trading days using the following formula:

$$\sigma_{30} = \sqrt{252}\sqrt{\frac{\sum_{i=1}^{30}(R_i - \overline{R})^2}{29}} \tag{4.14}$$

$$R_i = \ln(\frac{S_i}{S_{i-1}}) \tag{4.15}$$

where σ_{30} is the 30-day historical volatility at time t, R_i is the log return of the underlying asset price at time i, $\overline{R}$ is the average log return, and S_i is the closing price of the underlying asset on day i. We annualize the volatility by multiplying by $\sqrt{252}$, assuming 252 trading days per year.

GARCH(1, 1) volatility

The GARCH model introduced by Bollerslev (1986) provides a more sophisticated measure for capturing the inherent time-varying volatility in financial markets. This could yield a more robust and dynamic measure of the risk associated with the option. We employ the Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC) to determine the optimal order of p and q for the GARCH model, which penalizes model complexity while promoting goodness-of-fit.

The GARCH(1, 1) model was selected as it returned the lowest AIC and BIC values among different combinations of p and q. The model is specified as follows:

$$\sigma_t^2 = \omega + \alpha R_{t-1}^2 + \beta \sigma_{t-1}^2 \tag{4.16}$$

where σ_t^2 is the conditional variance at time t, ω is the constant term, α and β are the GARCH parameters, and R_{t-1} is the log return at time $t-1$. In this model, we assume a normal distribution for the error terms. After obtaining the conditional volatilities, we annualize them by multiplying by $\sqrt{252}$.

Implied Volatility

Following Wang et al. (2012) we calculate an implied volatility (IV) measure for day $t-1$, which is then employed to price all options on day t. Since implied volatility is often considered a superior predictor of future volatility than historical volatility (Jorion, 1995; Egelkraut and Garcia, 2006), it might enhance the pricing accuracy of BS model. Given the volatility smile, the deep in- and out-of-the-money options have a disproportional effect on the calculated implied volatility. Using the median as opposed to the average as seen in Wang et al. (2012) yielded better results and BS with median implied volatility is therefore reported. The Newton-Raphson numerical method is used to calculate the implied volatilities at $t-1$ from the option prices and the BS formula given in Equation (4.11).

4.3.2 Heston

Heston is one of the most popular models for option pricing (Larikka and Kanniainen, 2012; Jerbi and Chaabene, 2020; Liang and Cai, 2022). It is a stochastic volatility model where the asset price and its volatility are described as two stochastic processes. The model is described by the following system of stochastic differential equations:

$$dS_t = \mu S_t dt + \sqrt{\nu_t} S_t dz_{1t} \tag{4.17}$$

$$d\nu_t = \kappa(\theta - \nu_t)dt + \sigma\sqrt{\nu_t} dz_{2t} \tag{4.18}$$

$$dz_{1t} dz_{2t} = \rho dt \tag{4.19}$$

where S_t is the spot asset price, ν_t is the mean-reverting stochastic volatility, σ is the volatility of the variance, μ is the rate of return of the underlying asset, θ is the long-term variance, κ is the mean reversion rate and ρ is the correlation between the two Brownian motions z_{1t} and z_{2t}.

The formula for pricing a European call option under the Heston model, as described by Heston (1993), involves solving the complex integral below:

$$C_0 = \frac{1}{2}(S_0 - Ke^{-r\tau}) + \frac{1}{\pi}\,\mathrm{Re}\int_0^\infty \left(e^{r\tau}\frac{\phi(u-i)}{iuK^{iu}} - \frac{\phi(u)}{iuK^{iu}} \right) du \tag{4.20}$$

$$\phi(u) = e^{u\tau} S_0^{iu} \left(\frac{1-ge^{-d\tau}}{1-g} \right)^{-2\frac{\theta_\kappa}{\sigma^2}} exp\left(\frac{\theta_\kappa \tau}{\sigma^2}(\kappa - \rho\sigma iu - d) + \frac{V_0}{\sigma^2}(\kappa - \rho\sigma iu + d)\frac{1-e^{d\tau}}{1-ge^{d\tau}} \right) \tag{4.21}$$

$$d = \sqrt{(\rho\sigma ui - \kappa)^2 + \sigma^2(iu + u^2)} \tag{4.22}$$

$$g = \frac{\kappa - \rho\sigma iu - d}{\kappa - \rho\sigma iu + d} \tag{4.23}$$

where i is the complex number, V_0 is the square of the initial volatility, r is the risk free rate, K is the exercise price of the option and τ is the time to maturity and.

Solving this complex integral requires a discretization strategy, as demonstrated in (Mikhailov and Nögel, 2004). Five parameters have to be calibrated: $\kappa, \rho, \theta, \sigma$ and V_0. We calibrate the Heston

model on the last trading day prior to each test set using the Levenberg-Marquardt algorithm (Levenberg, 1944; Marquardt, 1963). The MSE loss function was used for the Levenberg-Marquardt algorithm to maintain consistency with the loss function used during LSTM-MLP training and the primary performance metric.

4.3.3 MLP

A standard MLP model is implemented as a benchmark model to understand the impact of substituting the explicit volatility input with the LSTM component of the LSTM-MLP model. The MLP model uses historical volatility as volatility input, as also implemented in Iltüzer (2022). The 30-day historical volatility calculated in Section 4.3.1 is used. The MLP model is developed and trained following a similar methodological approach to that used for the LSTM-MLP model, as detailed in Section 4.1. The MLP model undergoes its independent hyperparameter search, conducted following the same process as that outlined for the LSTM-MLP model in Section 4.1.3. Details of the hyperparameter search for the MLP model, including the final optimized hyperparameters, can be found in Appendix B.

4.4 Performance metrics

4.4.1 Pricing performance

To evaluate the pricing performance of the LSTM-MLP model and the benchmark models, we use root mean squared error (RMSE) and mean absolute error (MAE). These metrics measure the degree of pricing error, with lower values indicating better accuracy. RMSE is a quadratic scoring rule which weighs large errors more heavily, thus punishing large outliers to a greater extent. MAE is a linear scoring rule and therefore treats all individual differences equally in the average, resulting in less sensitivity to outliers. By employing both metrics, we gain a nuanced perspective on the model's performance, considering both its overall predictive accuracy and its robustness to large errors. RMSE and MAE are defined as:

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2} \tag{4.24}$$

$$MAE = \frac{1}{m}\sum_{i=1}^{m}|y_i - \hat{y}_i| \tag{4.25}$$

Where m is the total number of options, y_i is the observed price, $\hat{y}_i$ is the model price estimates.

To perform pairwise comparisons of the models, the Diebold-Mariano (DM) test is employed using the MSE loss function to asses the equality of model predictions. For a given loss function $l(\cdot)$, the difference series, d, between the two prediction series is defined as:

$$d_i = l(\hat{y}_{i,1} - y_{t,1}) - l(\hat{y}_{i,2} - y_{i,2}) \tag{4.26}$$

Given the null hypothesis of:

$$H_0 : E[d_i] = 0 \tag{4.27}$$

The DM test statistic is:

$$DM = \frac{\frac{1}{m}\sum_{i=1}^{m} d_i}{\sqrt{2\pi \hat{f}_d(0)/m}} \tag{4.28}$$

Where $\hat{f}_d(\cdot)$ is the spectral density of $\{d_i\}$.

4.4.2 Trading performance

We assess the risk-adjusted performance of our options portfolio using several metrics: the Sharpe ratio, the Sortino ratio, the Capital Asset Pricing Model (CAPM) α, returns and maximum drawdown (MDD). These metrics are intended to assess both the trading returns and risks.

The Sharpe ratio, introduced by Sharpe (1966), is a measure of risk-adjusted return, taking into account the total volatility of returns. The Sortino ratio, a variant of the Sharpe ratio introduced by Sortino (1994), distinguishes harmful volatility from total volatility by considering only the downside deviation. We compute the Sharpe and Sortino ratios using the daily returns and annualize the measures. The formulas are given in:

$$\text{Sharpe Ratio} = \frac{R_p - r_f}{\sigma_p} \tag{4.29}$$

$$\text{Sortino Ratio} = \frac{R_p - r_f}{\sigma_d} \tag{4.30}$$

where R_p is the return of the portfolio, r_f is the risk-free rate, σ_p is the standard deviation of the portfolio returns, σ_d is the standard deviation of negative portfolio returns.

The CAPM α provides a measure of the value a portfolio manager adds over a benchmark, with a positive α signifying market outperformance after adjusting for market risk. In the CAPM model, α is the intercept in the linear OLS regression model that relates the excess return of a security or portfolio (over the risk-free rate) to the excess return of the market (over the risk-free rate). The α and β can be extracted from the following formula:

$$R_p - r_f = \alpha + \beta \cdot (R_m - R_f) + \epsilon \tag{4.31}$$

where R_p is the return of the portfolio, r_f is the one-month risk-free rate, R_m is the return of the market, β represents the part of the excess return attributed to taking market risk and ϵ is an error term.

The annual portfolio returns are computed by determining the ratio of the ending balance to the starting balance, subtracting 1 from the result. This provides a measure of the overall growth or contraction of the portfolio over a year. Maximum drawdown, on the other hand, serves as an assessment of risk. It quantifies the largest relative decline from a peak to a trough in the portfolio value over the specified period. As such, it reflects the most severe potential loss that could have been incurred, underlining the exposure of the portfolio to downside risk.

4.5 Interpretability analysis

Interpretability analysis using both SHAP and ALE is conducted to gain a better understanding of the pricing behavior demonstrated by the LSTM-MLP model and to ensure a sanity check of its behavior. As outlined in Section 2.4, SHAP and ALE distinguish themselves based on the local and global scope of their interpretability respectively. Thus, the implementation of both these methods makes our analysis more comprehensive: SHAP contributes insights at a granular level, revealing individual instance-based characteristics, while ALE provides a broader perspective, capturing average effects to illuminate overarching model behaviors. While there may be areas of overlap in their analyses, the contrasting techniques of SHAP and ALE bring about a more comprehensive and robust understanding of the LSTM-MLP model. This combination, in turn, bolsters the reliability of our analysis and enriches the interpretability of our model in the complex domain of option pricing.

For both the ALE and the SHAP method, we choose to use the training set of the model for the interpretability analysis due to their larger size and a wider variety of feature values compared to the test sets, which span only one month each. Our rationale is that a broader dataset would better show the learned behavior by the model, which is the primary goal of the interpretability analysis. This choice aligns with our focus on understanding the overall model behavior rather than the specific pricing patterns in the shorter test periods. However, if the goal was to explain specific pricing estimates on the test set, the test set could be employed.

4.5.1 SHapley Additive exPlanations (SHAP)

SHAP (Lundberg and Lee, 2017) serves as a unified measure for interpreting model predictions. It was developed to consolidate the expanding number of interpretability methods, due to the lack of understanding about their relationships and the criteria for choosing one over another. Rather than depending on a single method, SHAP integrates six methods to offer a unified measure of feature importance, which can be assigned individually to each feature.

In introducing these methods, SHAP treats every explanation of a model's prediction as an explanation model in its own right. All six methods that comprise SHAP utilize the same explanation model. Mathematically, if f is the original prediction model to be explained and g is the explanation model, the explanation models generally transform the original inputs x through a mapping function $x = h_x(x')$ to acquire simplified inputs x'. The explanation model g can then be expressed as:

$$g(z') = \phi_0 + \sum_{i=1}^{M} \phi_i z_i' \tag{4.32}$$

where $z' \in \{0,1\}^M$, M is the number of simplified input features, and $\phi_i \in R$ is the effect attributed to each feature by methods with explanation models matching Equation (4.32). For more detailed explanations, see Lundberg and Lee (2017).

Given that SHAP is a local interpretability method it calculates the impact for each data point individually, thus making it computationally intensive. In response to this, we apply SHAP using K-means with $k = 75$ to identify representative option quotes from the training set for use by the explainer. We then sample 10,000 options to compute SHAP values. The analysis is performed on the final model trained on data from November 2019 - October 2022.

4.5.2 Accumulated Local Effects (ALE)

ALE (Apley and Zhu, 2016) provides an average measure of a feature's influence on the predictions of a machine learning model. ALE is a faster, and unbiased extension of Partial Dependence Plots (PDP), which notably overcomes the inability of PDPs to account for correlations between input features.

The ALE methodology can be broken down into several steps. Firstly, the values of the target feature are divided into intervals. For the data within each interval, the difference in predictions is calculated when the target feature value is replaced from the upper limit to the lower limit of the interval. These differences are then accumulated to yield the uncentered ALE values. Mathematically, this estimation process for the uncentered effect $g_{j,\mathrm{ALE}}(\cdot)$ can be expressed as:

$$\hat{g}_{j,\mathrm{ALE}}(x) = \sum_{k=1}^{k_j(x)} \frac{1}{n_j(k)} \sum_{\{i:x_{i,j} \in N_j(k)\}} \left[f\left(z_{k,j}, x_{i,\setminus j}\right) - f\left(z_{k-1,j}, x_{i,\setminus j}\right) \right] \tag{4.33}$$

where $f(\cdot)$ is the function of the machine learning model, x_j is the target feature, $N_j(k)$ is the k^{th} interval, $z_{k,j}$ is the limit value of feature j in the k^{th} interval, and $x_{i,\setminus j}$ is the i^{th} sample excluding feature j.

To ensure that the mean effect is zero, the computed ALE values are centered to form an ALE curve. The centering process to estimate the main effect function $f_{j,ALE}(\cdot)$ can be expressed as:

$$\hat{f}_{j,ALE}(x) = \hat{g}_{j,ALE}(x) - \frac{1}{n} \sum_{i=1}^{n} \hat{g}_{j,ALE}(x_{i,j}) \tag{4.34}$$

where n is the number of samples. For a more comprehensive explanation, see Apley and Zhu (2016).

We conduct the ALE method on the final model trained each year from 2015 to 2022. The feature importance measures for each year are then averaged over the eight-year period. Even though ALE

is a global interpretability method and is therefore significantly quicker than local interpretability methods such as SHAP, it can demand substantial GPU memory when processing large amounts of data simultaneously. To tackle this issue, we opt to select a sample of 500,000 data points for the ALE estimation of each model.

5 Results and discussion

This section measures the performance of the proposed LSTM-MLP model from three perspectives. Section 5.1 analyzes pricing performance in terms of accurately estimating the market price. Section 5.2 studies the predictive ability of option price movements, testing whether the model is able to correctly identify overpriced or underpriced options. This is applied in practice in Section 5.3 with trading strategies based on the mispricings identified.

Comparisons are made between the proposed LSTM-MLP model and the benchmark models, as well as other models in the literature. For comparisons with the literature, it should be noted that fair comparison is limited due to the novelty of our approach. However, we believe it is still relevant to compare our results to the literature, albeit on a more general basis due to the unavailability of truly comparable model architectures, data sets, and time periods studied.

5.1 Pricing performance

Pricing results for the proposed LSTM-MLP model are generated by using it to price European call options from 2015 to 2022, during which the model is evaluated using 96 sliding windows with a one-month out-of-sample test period each. In total, 10,136,680 options are priced out-of-sample by the LSTM-MLP model and each of the benchmark models.

In the following analysis, Section 5.1.1 evaluates the overall pricing performance across the full test period from 2015 to 2022. Section 5.1.2 tests the robustness of the model in different market conditions, including Covid-19 induced regime shifts, changes in underlying returns, and volatility variations measured by the VIX index. Section 5.1.3 examines the pricing performance for different levels of maturity and moneyness. Finally, Section 5.1.4 applies the XAI frameworks SHAP and ALE to interpret the influence of input features on model output.

5.1.1 Overall pricing performance

Table 5.4 presents the overall pricing performance by each model from 2015 to 2022. The proposed LSTM-MLP pricing model outperforms all benchmark models in terms of pricing accuracy measured by RMSE and MAE. The Diebold-Mariano test confirms that these pricing differences are statistically significant, as reported in Table 5.5.

Our finding that both neural network-based models perform better than BS is supported by multiple previous works in the literature (Hutchinson et al., 1994; Amilon, 2003; Ke and Yang, 2019; Wang et al., 2022; Iltüzer, 2022). The observation that the LSTM-MLP, as a neural network-based model, performs better than Heston is consistent with the general conclusion in Jerbi and Chaabene (2020), which states that neural network-based approaches have higher pricing accuracy than Heston. However, we also note that this is to some extent contradicted by the fact that our MLP is beaten by the Heston implementation. Nonetheless, the positive results provide supportive evidence of the LSTM-MLP being able to capture useful relationships inherent in the options market.

Table 5.4
RMSE and MAE values for the pricing performance by each pricing model across the period from 2015 to 2022. The lowest (best) value for each metric is highlighted in bold.

Metric	BS 30-day	BS GARCH	BS IV	Heston	MLP	LSTM-MLP
RMSE	35.92	36.26	20.42	13.15	17.10	**11.84**
MAE	15.48	15.52	10.78	7.47	8.86	**6.61**

Table 5.5
Diebold-Mariano test statistics for each pair of models. All values are significant at the $p < 0.01$ level.

	BS 30-day	BS GARCH	BS IV	Heston	MLP	LSTM-MLP
BS 30-day		6.27	-255.91	-302.42	-271.42	-311.73
BS GARCH	-6.27		-250.20	-281.67	-254.95	-288.92
BS IV	255.91	250.20		-325.54	-143.26	-368.01
Heston	302.42	281.67	325.54		266.82	-180.88
MLP	271.42	254.95	143.26	-266.82		-344.71
LSTM-MLP	311.73	288.92	368.01	180.88	344.71	

The improved performance of the LSTM-MLP model compared to the BS models and the Heston model is likely due to its ability to capture the complex, non-linear relationship between the input variables and option prices. In comparison to the baseline MLP, the increased performance of the LSTM-MLP suggests that the combination of two neural network architectures does enhance pricing accuracy. More specifically, it demonstrates that an MLP with inputs from an LSTM outperforms an MLP with a single 30-day historical volatility input. These findings contradict Ke and Yang (2019) which found that a combined LSTM-MLP model could outperform the BS model, but not the standalone MLP model. Our implementation of the LSTM-MLP is able to do this, which hints at the proficiency of our model implementation described in Section 4.1, and suggests their usage of only 20 timesteps was insufficient to extract time series information more valuable than historical volatility. In contrast, we use 140 timesteps.

Figure 5.8 complements these results by illustrating the range of pricing differences defined by the model price minus the market price. Most models exhibit a median pricing difference near zero, with BS IV and Heston models slightly overpricing the median. Distinctively, the proposed LSTM-MLP model displays a narrower range of pricing differences, signifying a more consistent pricing behavior. These observations contribute to validating our hypothesis that the LSTM-MLP model is able to adapt its pricing behavior to varying option characteristics and market conditions, thereby minimizing the variability in pricing differences.

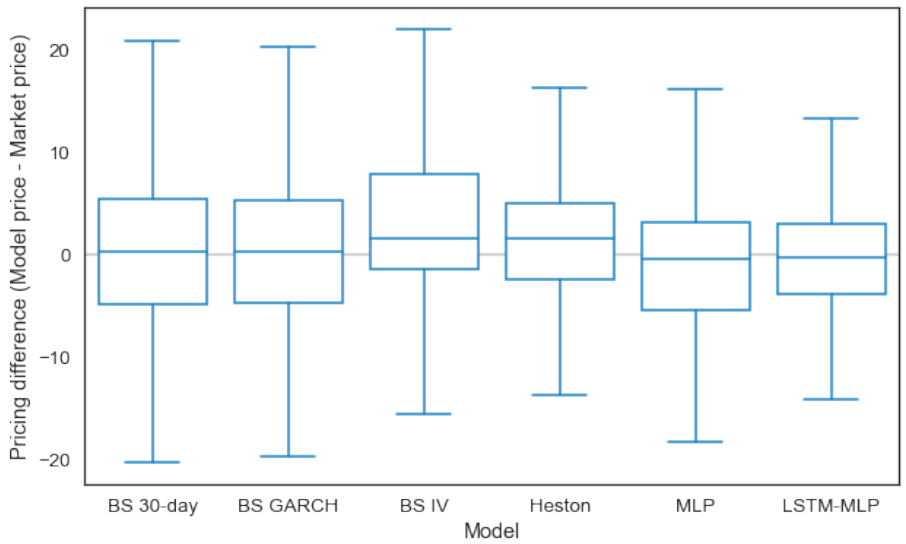


Figure 5.8 Boxplot of pricing differences, defined as the model price minus the observed market price, in terms of USD for all pricing models studied. The rectangular body represents the first and third quartiles of the error distribution (interquartile range, or IQR), with a line indicating the median error. The whiskers represent the minimum and maximum error within a conventional range, defined as 1.5 times the IQR below the first quartile or above the third quartile.

5.1.2 Impact of market conditions over time on pricing performance

Covid-19 pandemic

Table 5.6 and Table 5.7 display RMSE and MAE pricing errors for all models per year. In every year except 2022, our LSTM-MLP model is the top performer, achieving the lowest RMSE and MAE in comparison to the benchmark models. In 2022, however, the Heston model marginally surpasses the LSTM-MLP model. These findings align with the results documented in Gradojevic and Kukolj (2022), where a neural network-based model more distinctly outperformed the BS model before the Covid-19 pandemic. However, it's worth noting that their findings are based on a dataset limited to the first half of 2020.

Table 5.6
Pricing performance measured in RMSE for each pricing model. Results are reported for each year between 2015 and 2022. For each year, the lowest value is printed in bold to highlight the best-performing pricing model for the given year.

Year	BS 30-day	BS GARCH	BS IV	Heston	MLP	LSTM-MLP
2015	11.99	13.99	8.70	6.55	7.10	**3.12**
2016	11.23	12.37	8.72	4.85	5.30	**3.33**
2017	11.79	9.81	9.03	3.81	3.41	**3.20**
2018	18.25	22.75	13.07	7.76	11.46	**5.24**
2019	19.82	18.50	12.51	6.52	7.32	**6.12**
2020	72.90	70.83	31.24	22.79	34.88	**21.41**
2021	36.57	34.66	25.98	16.15	14.39	**12.54**
2022	31.20	37.28	23.84	**13.84**	17.21	14.45

Table 5.7
Pricing performance measured in MAE for each pricing model. Results are reported for each year between 2015 and 2022. For each year, the lowest value is printed in bold to highlight the best-performing pricing model for the given year.

Year	BS 30-day	BS GARCH	BS IV	Heston	MLP	LSTM-MLP
2015	6.91	7.18	5.51	4.47	4.11	**2.19**
2016	6.47	6.68	5.18	3.46	3.26	**2.30**
2017	5.83	5.04	4.98	2.92	2.39	**2.28**
2018	9.83	10.54	7.08	4.98	7.37	**3.57**
2019	10.58	9.72	7.28	4.68	5.02	**4.42**
2020	30.73	30.19	16.13	12.97	19.28	**12.88**
2021	20.87	19.9	16.41	10.53	9.71	**8.38**
2022	18.59	20.58	14.20	**9.43**	11.66	9.93

Up until 2022, the LSTM-MLP model consistently outperforms all benchmark models. This indicates its effective learning of the relationship between market variables and option prices. It exhibits flexibility in adapting to changing market conditions over time, demonstrated by its better performance across these years. In contrast, the BS models and the Heston model cannot match this adaptability. Their limitations potentially arise from the stringent statistical and economic assumptions inherent in their designs.

Interestingly, despite its flexibility, the MLP model still falls short of consistently outperforming the Heston model. One possible reason could be the reliance of the MLP on the 30-day historical volatility, which might not capture rapid market changes effectively. It is conceivable that the MLP could be improved with alternative volatility measures, but this raises a crucial point of our research: the assumption-laden process of choosing specific volatility measures. Given this, it might be more effective to enable the model to learn its own volatility representation, freeing it from predefined assumptions.

The year 2020, characterized by unprecedented market volatility due to the Covid-19 pandemic,

posed a significant challenge for all pricing models. Although the Heston model was competitive, it failed to match the LSTM-MLP's robust performance. The LSTM-MLP model also demonstrated better adaptability by displaying the smallest increase in RMSE from 2019 to 2020 compared to other models, but a slightly higher increase in MAE than the Heston model.

Post-2020, we observe broader ranges of pricing differences for all models as illustrated in Figure 5.9, signaling a shift in their performance. The difference in RMSE and MAE performance between models also varies more during and post-Covid-19 than prior, implying varying consistency in model performance. Heightened volatility during this period could contribute to these challenging pricing conditions for all models and the subsequent fluctuation in relative model performance. This change in trend might be due to the significant market transformations brought about by the Covid-19 market crash and recovery period. Particularly for the deep learning models MLP and LSTM-MLP, these changes might have hampered their ability to effectively apply pre-pandemic historical data (Gradojevic and Kukolj, 2022).

One potential solution to enhance the model performance post-Covid-19 could involve adjusting the length of the training window. A shorter window might allow the model to learn from and adapt more rapidly to recent market trends. However, this approach comes with trade-offs. However, a shorter training window might limit the diversity of training data, potentially negatively affecting the overall model performance. Therefore, the selection of the training window length presents a delicate balance: a longer window might enhance the model's ability to generalize through exposure to diverse market conditions, while a shorter window could facilitate quicker adaptation to recent market trends.

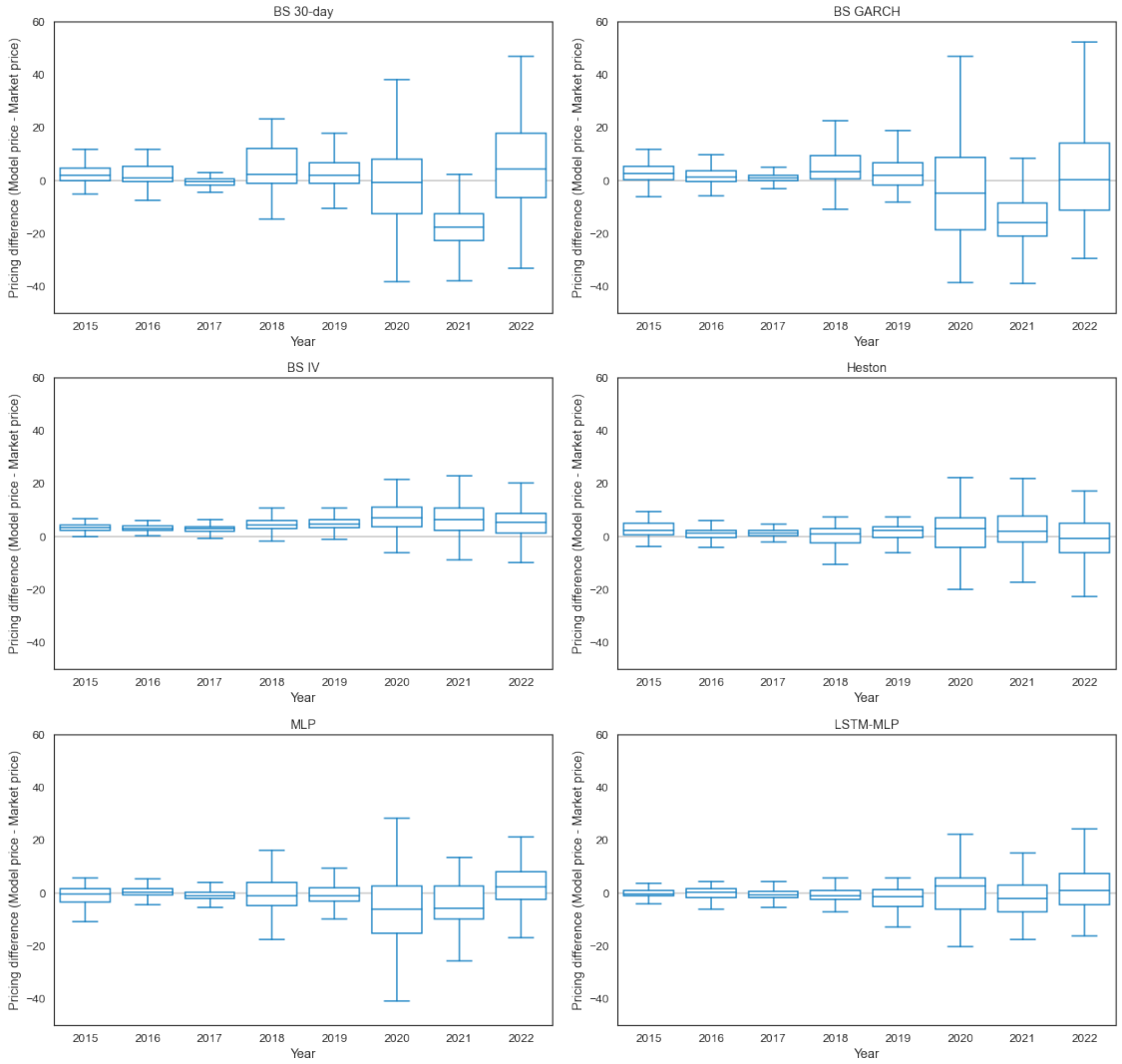


Figure 5.9 Boxplot of pricing differences, defined as the model price minus the observed market price, in terms of USD for all pricing models. Errors are reported for each year between 2015 and 2022. The rectangular body represents the first and third quartiles of the error distribution (interquartile range, or IQR), with a line indicating the median error. The whiskers represent the minimum and maximum error within a conventional range, defined as 1.5 times the IQR below the first quartile or above the third quartile.

S&P 500 index returns

Our analysis of the SP500 market conditions involves identifying three distinct regimes: stable, upwards-trending, and downwards-trending. These regimes are established based on the Moving Average Convergence/Divergence (MACD) technical indicator. In essence, we subtract the 26-day exponential moving average of the SP500 index from its 12-day exponential moving average to get the MACD. Then, we split the MACD range of all the test period days into three equal parts, creating threshold values of 0.93 and 21.12. The MACD plot with these threshold values during the testing period can be found in Appendix D.

The model performance under the three S&P500 return market regimes are presented in Table 5.8. In all three market conditions, the LSTM-MLP model consistently surpasses all benchmark models. Interestingly, underperformance by Heston is slightly lower in stable markets when evaluated on RMSE and standard deviation of absolute errors. This suggests that the complexity of the LSTM-MLP model may be less necessary in stable markets where the dynamics are simpler. Alternatively, it could suggest that the assumptions for Heston hold more accurately in stable conditions, improving the parameter estimation process and leading to better pricing performance.

In contrast, the MLP model underperforms consistently compared to both Heston and LSTM-MLP models across all market conditions. This performance gap is notably higher in both upward and downward trending markets, where the pricing error for the MLP escalates to levels similar to the BS IV model in downward trends. The MLP model's underperformance could be attributed to its dependence on the 30-day historical volatility measure, which does not account for market movement direction. Consequently, it struggles to adjust promptly and accurately to the different market directions. On the other hand, the LSTM-MLP model can leverage the capacity of the LSTM component to memorize and exploit data temporal dependencies, potentially capturing and reacting to underlying market trends, including direction, more accurately.

Hence, integrating an LSTM component into an MLP model can significantly improve its performance across various market trends. Most notably, the superior adaptability of the LSTM-MLP to market directionality provides it with a distinct advantage over the MLP model when markets are in motion, especially during downward trends, making it a potentially more reliable tool in these dynamic scenarios.

Table 5.8
Pricing performance during different S&P 500 regimes, measured in RMSE, MAE, and the standard deviation of the absolute value of pricing differences defined as model price minus the observed market price. Upwards-trending is defined as MACD over 21.12. Stable conditions is defined as MACD between 0.93 and 21.12. Downwards-trending is defined as MACD under 0.93. Each regime consists of 1/3 of the total days in the period.

	BS 30-day	BS GARCH	BS IV	Heston	MLP	LSTM-MLP
Downwards-trending						
RMSE	44.54	51.86	24.13	16.00	23.16	**14.91**
MAE	18.38	21.36	12.06	8.58	11.29	**7.72**
St.dev.	40.57	47.25	20.90	13.50	20.22	**12.76**
Stable conditions						
RMSE	25.44	16.53	15.63	7.88	9.39	**7.09**
MAE	10.51	8.48	8.05	4.90	5.24	**3.90**
St.dev.	23.17	14.19	13.40	6.17	7.79	**5.93**
Upwards-trending						
RMSE	34.07	29.73	20.01	13.49	15.18	**11.61**
MAE	16.63	15.62	11.69	8.40	9.41	**7.66**
St.dev.	29.74	25.30	16.24	10.56	11.91	**8.72**

The pricing behavior of the LSTM-MLP across different S&P 500 regimes over time is illustrated in Figure 5.10. The largest absolute mean pricing differences are observed in downward-trending regimes, which might be expected due to the asymmetric effect of negative shocks on volatility, making downward-trending markets typically more volatile (Black, 1976). As increased volatility boosts option prices, higher pricing difference values might be expected in absolute terms. Interestingly, the widest ranges, including 90% of the pricing differences, and the most drastic changes in mean pricing difference are also noticed for upward- and downward-trending markets, most notably in downward-trending markets.

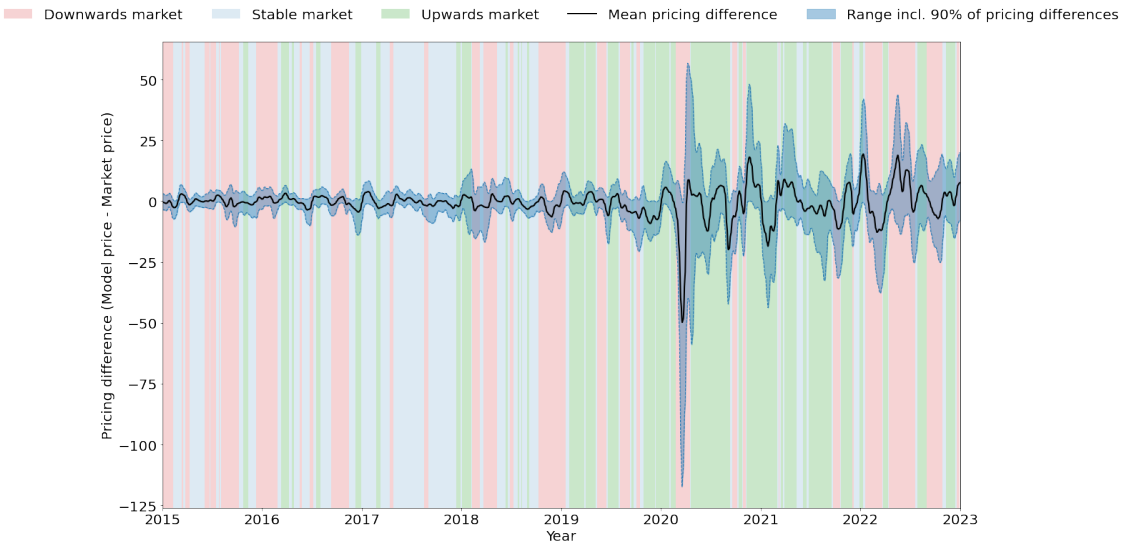


Figure 5.10 Pricing performance of the LSTM-MLP model, measured in mean pricing difference and the broadness of the range including 90% of pricing differences. The colored overlay represents different market regimes defined by S&P 500 returns. Upwards-trending is defined as MACD over 21.12. Stable conditions are defined as MACD between 0.93 and 21.12. Downwards-trending is defined as MACD under 0.93. Each regime consists of 1/3 of the total days in the period. The mean error and error interval is slightly smoothed for readability.

VIX

The VIX index, often referred to as the "fear gauge", measures the expectation by the market of future volatility. This makes it a key factor in option pricing, as higher volatility generally leads to higher option prices due to the increased risk. We explore this aspect further by examining the pricing performance of our models across three distinct VIX[15] regimes. Table 5.9 presents the pricing performance of each model for three distinct market regimes defined by the VIX. We divide the VIX range of all days in the test period into three equal parts, resulting in threshold values of 14.22 and 20.48. A plot of the VIX in the testing period with threshold values is included in Appendix D.

The LSTM-MLP model stands out in all scenarios with the lowest RMSE, MAE, and standard deviation. This suggests that the LSTM-MLP model can effectively adjust its pricing behavior to prevailing market volatility conditions due to its LSTM component, which captures the temporal dependencies in the sequential return data for the SP 500 index. In contrast, the MLP model only surpasses all other benchmark models in the medium VIX regime in terms of RMSE and standard deviation and in the low VIX regime in terms of MAE. Compared to the closest competitors, MLP exhibits a worsening relative performance in the high VIX regime. This implies that the MLP model may not be as robust when it comes to adjusting to different market volatility conditions.

[15]Daily VIX data from Cboe used: https://www.cboe.com/

Table 5.9
Pricing performance during different VIX regimes, measured in RMSE, MAE, and standard deviation of the absolute value of the pricing difference defined as model price minus the observed market price. Low VIX regime is defined as VIX values below 14.22. Medium VIX regime is defined as VIX values between 14.22 and 20.48. High VIX regime is defined as VIX values above 20.48. Each regime consists of 1/3 of the total days in the period.

	BS 30-day	BS GARCH	BS IV	Heston	MLP	LSTM-MLP
Low VIX						
RMSE	12.64	11.21	9.84	4.74	4.91	**4.42**
MAE	6.67	6.01	5.61	3.54	3.23	**3.02**
St.dev.	10.74	9.46	8.08	3.16	3.70	**3.22**
Medium VIX						
RMSE	25.51	25.07	17.33	10.78	10.29	**8.69**
MAE	13.45	12.92	10.07	6.62	6.69	**5.27**
St.dev.	21.67	21.48	14.11	8.51	7.82	**6.91**
High VIX						
RMSE	51.23	52.25	27.18	18.04	25.28	**16.71**
MAE	23.32	24.33	14.96	10.91	14.62	**10.26**
St.dev.	45.62	46.24	22.69	14.36	20.63	**13.19**

Figure 5.11 further illustrates the pricing behavior of the LSTM-MLP model across different VIX regimes over time. The high VIX regimes visually exhibit more fluctuations in mean pricing difference and larger 90% ranges, indicating the more difficult pricing conditions during these periods.

As previously discussed, absolute pricing errors could be expected to increase with higher VIX as option prices are higher when volatility is higher. This is especially true since we use the VIX index to determine different volatility periods. As the VIX index is directly tied to the implied volatility for options, it essentially approximates the pricing level of options.

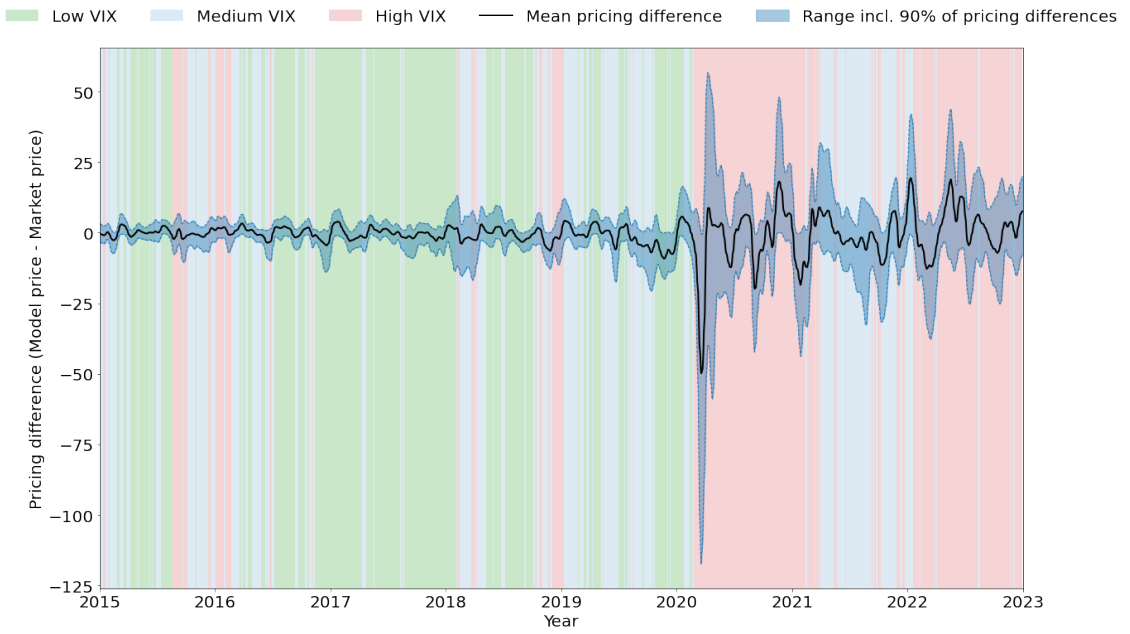


Figure 5.11 Pricing performance of the LSTM-MLP model, measured in mean pricing difference and the range including 90% of pricing differences. The colored overlay represents different market regimes defined by VIX values. Low VIX regime is defined as VIX values below 14.22. Medium VIX regime is defined as VIX values between 14.22 and 20.48. High VIX regime is defined as VIX values above 20.48. Each regime consists of 1/3 of the total days in the period. The mean error and error interval is slightly smoothed for readability.

5.1.3 Impact of moneyness and maturity on pricing performance

Figure 5.12 shows the pricing performance of each model for different maturities. For all models, the RMSE generally increases for longer maturities. However, some increase is expected given the increased option values for longer maturities.

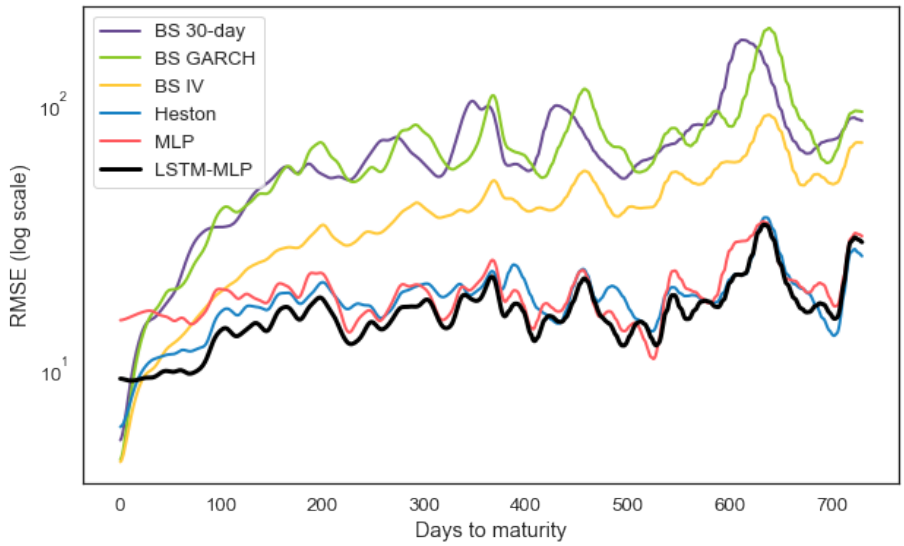


Figure 5.12 RMSE at different maturities. The graph is slightly smoothed for readability.

Both the LSTM-MLP and MLP show worse performance than the benchmark models for the shortest maturities. However, the RMSE increases more across maturities for the other benchmark models, and especially the BS models. This is also reported by Liang and Cai (2022), who noted comparable performance by conventional option pricing models like BS and machine learning methods like MLP for the shortest maturity options and then a strengthening of the relative performance by the machine learning methods for longer maturities.

This trend could be linked to the limitations of the BS assumptions, such as constant volatility and log-normally distributed asset prices, which may not accurately reflect the market dynamics over a long period. As a result, the market may price options for longer maturities differently than the BS model would predict, leading to greater pricing errors. In contrast, the LSTM-MLP model, being data-driven, is not constrained by such assumptions and can adjust its predictions based on the data it is trained on. This could explain its relatively stable performance across maturities compared to the BS models. The same reasoning can be applied to the Heston model, which accounts for stochastic volatility and does not require a lognormal distribution of the underlying asset prices.

When comparing the LSTM-MLP model with the MLP model, it is noticeable that the LSTM-MLP outperforms the MLP for short-maturity options but shows similar performance for long-maturity options. This suggests that the flexibility of the MLP may not be sufficient to accurately adapt to lower-maturity options using the 30-day volatility input. It, therefore, seems that the LSTM input is especially more suitable to extract the relevant volatility for the short and medium maturities relative to the explicit volatility input given to the MLP model.

Figure 5.13 illustrates the pricing performance of each model at varying levels of moneyness. The LSTM-MLP outperforms all the benchmark models for out-of-the-money and at-the-money options. For deeper in-the-money options, the underperformance by the MLP lessens and the Heston model slightly beats the LSTM-MLP. This is in line with the findings of Amilon (2003) and Iltüzer (2022), which highlighted the improved performance of neural network-based models over the BS model with historical volatility. Further, Iltüzer (2022) also documented an edge by neural networks over BS with GARCH volatility across all levels of moneyness. Comparing neural network-based models to BS with implied volatility, the most challenging point for outperformance

arises for at-the-money options, a finding also echoed in Iltüzer (2022).

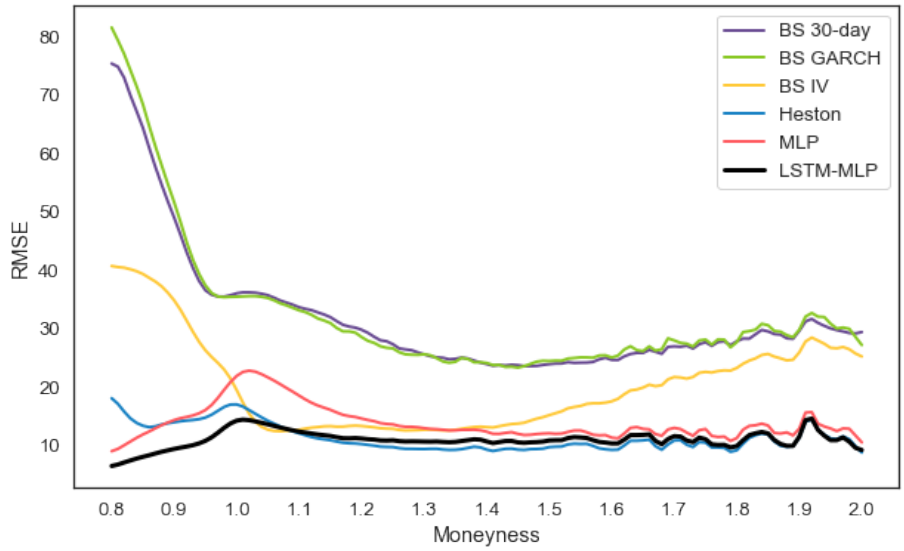


Figure 5.13 RMSE at different moneyness. The graph is slightly smoothed for readability.

The performance of the LSTM-MLP indicates that it is more suitable for capturing the pricing dynamics for out-of-the-money and at-the-money options, which can be more complex than in-the-money options as there is more uncertainty about whether they will become in-the-money at expiry. For out-of-the-money options, the lower RMSE for the neural network-based pricing model, and the higher RMSE for BS are also observed in Amilon (2003). Moreover, the observation that the neural network-based model has difficulty pricing options deep in the money, whereas BS is more capable of pricing in-the-money than out-of-the-money options is consistent with Bennell and Sutcliffe (2004).

For further in-the-money options, the Heston model becomes a stronger competitor for the LSTM-MLP model. The relative underperformance could be attributed to the generally lower liquidity of these options, as shown in Figure 3.2 and cited by Fan et al. (2017). Reduced liquidity could imply less efficient market prices and, consequently, a greater challenge in discerning a robust pricing relationship. As depicted in Figure 3.1, the number of options in the dataset drops considerably at higher moneyness levels, implying that the reported RMSE for these levels is based on a smaller sample size. Furthermore, the reduced sample could potentially impede the LSTM-MLP model's ability to learn a representative mapping from option characteristics to option price, due to insufficient data and variation. This is supported by Yang et al. (2017), who excluded in-the-money options from their study on gated neural networks for option pricing on the S&P500 index, citing the low trading activity and unreliability of prices for these options.

Table 5.10 shows the RMSE for each pricing model at different maturities within the specified groups of moneyness. For out-of-the-money and at-the-money options, the LSTM-MLP outperforms benchmark models across maturities. This is also in line with the one-dimensional analysis of pricing performance by the moneyness dimension shown in Figure 5.13. The outperformance across models is more mixed for in-the-money options. The LSTM-MLP struggles for shorter and medium-term options, for which BS with implied volatility performs best. However, the LSTM-MLP performs best again for the long and very long maturities of in-the-money options. Our observations for the LSTM-MLP are similar to the results for the MLP in Liang and Cai (2022), although our MLP implementation does not demonstrate equally strong performance.

Comparatively, our results diverge somewhat from those presented in Gradojevic and Kukolj (2022), who found that pricing options with a maturity greater than 180 days and not near-the-money is generally more challenging. Contrary to their findings, our LSTM-MLP model outperforms all benchmark models for options with long and very long maturities. This discrepancy may highlight the strengths of the LSTM-MLP model in handling these more challenging options.

Wang et al. (2022) demonstrate that a deep neural network-based pricing model outperforms both the BS and Heston models across all maturities and moneyness levels. Our results mostly align with these findings, although our LSTM-MLP model falls short in pricing short-maturity, in-the-money options.

Table 5.10
RMSE grouped by moneyness and maturity. For moneyness, out-of-the-money is defined as less than 0.97, at-the-money is defined as between 0.97 and 1.03, and in-the-money is defined as greater than 1.03. For maturity, shorter maturity is defined as 0-30 days, medium maturity is defined as 31-90 days, longer maturity is defined as 91-300 days, and very long maturity is defined as 301-730 days. For each combination of moneyness and maturity, the lowest value is printed in bold to highlight the best-performing pricing model for the given combination.

Moneyness	Maturity	BS 30-day	BS GARCH	BS IV	Heston	MLP	LSTM-MLP
<0.97	0-30	12.15	13.65	6.85	7.22	7.33	**4.98**
	31-90	31.20	31.99	18.66	13.04	13.79	**8.52**
	91-300	65.39	68.98	43.57	20.20	20.26	**13.41**
	301-730	104.34	107.84	73.23	23.27	23.61	**17.22**
0.97-1.03	0-30	15.68	13.43	12.35	12.94	19.26	**11.71**
	31-90	26.74	25.30	17.19	16.99	22.24	**12.83**
	91-300	56.32	56.94	27.24	22.69	25.38	**18.67**
	301-730	100.49	101.99	48.95	23.97	25.08	**21.76**
>1.03	0-30	8.04	6.61	**4.71**	6.23	16.60	9.11
	31-90	16.62	15.13	**7.74**	9.19	14.57	9.42
	91-300	41.12	40.75	17.17	14.77	16.78	**14.29**
	301-730	72.45	72.59	36.43	21.41	20.51	**20.31**

5.1.4 Interpretability analysis of pricing behavior

Interpretability analysis using SHAP and ALE is implemented to better understand the pricing behavior of the LSTM-MLP model.

In the SHAP summary plots, the x-axis represents the SHAP values. The SHAP values estimate the impact of each feature value in terms of explaining deviations in output values from a baseline of expected option value if no feature values were given. The interpretations of feature effects are thus relative to the other options in the data set. A positive SHAP value indicates an increased option price from the baseline price and a negative value represents a reduced option price. A red color indicates a high feature value, and a blue color indicates a low feature value.

Figure 5.14 presents the SHAP summary plot ordered by feature importance for model outputs for the option characteristics used as input in the MLP component of the LSTM-MLP model. The strike price is found to be the most important feature, followed by the price of the underlying asset, time to maturity, and the risk-free rate. This ordering is consistent with the interpretability analysis by Liang and Cai (2022) using ALE for LSTM and MLP option pricing models. Higher values for the S&P 500 index, time to maturity and risk-free rate contribute to increasing the estimated option price, while higher values for the strike contribute to reducing the estimated option price. This makes intuitive sense to what we would expect for call options and thus serve as a sanity check of the model.

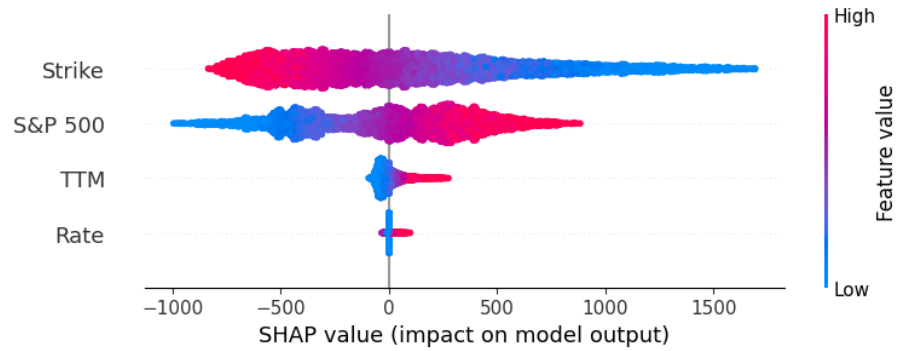


Figure 5.14 SHAP summary plot of feature importance and positive or negative impact on model output. The SHAP values are calculated for 10,000 options sampled from the latest LSTM-MLP option pricing model trained on options data from November 2019 - October 2022.

Figure 5.15 presents the SHAP values for the 10 most recent returns of the return series given to the LSTM component of the model. The full return series is included in Appendix C. The most recent returns seem the most important, with a declining impact of older returns. Negative returns tend to have a positive impact on option price for the most recent returns, while positive returns tend to have a negative impact. Large negative shocks tend to have a higher absolute SHAP value than positive shocks. This indicates that negative shocks have a greater impact on the option price, which could be partly explained by the fact that negative shocks are typically larger in magnitude as well. Many of the return data points are clustered around lower absolute SHAP values, while the endpoints of high and low returns show higher absolute SHAP values. This indicates that the most important information from the time series comes from the large movements and shocks.

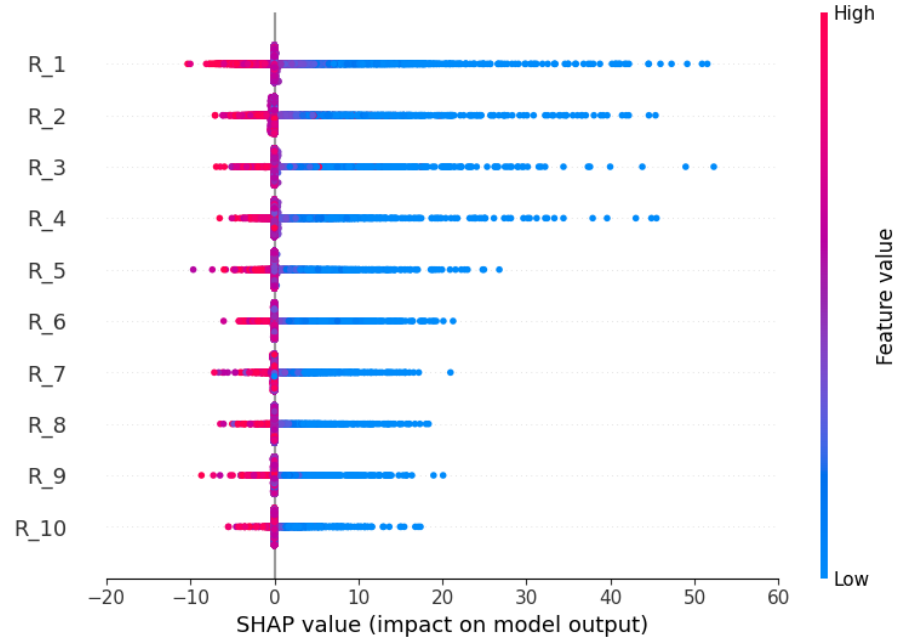


Figure 5.15 SHAP summary plot for the 10 most returns in the return series of the underlying asset given to the LSTM component. The full summary plot for the entire return series is given in Appendix C. The SHAP values are calculated for 10,000 options sampled from the latest LSTM-MLP option pricing model trained on options data from November 2019 - October 2022.

The pronounced and positive impact of recent negative returns aligns with the well-documented leverage effect. Initially identified by Black (1976) and empirically confirmed in numerous studies such as Christie (1982), the leverage effect suggests that negative returns and shocks tend to have a more pronounced impact on future volatility than positive shocks. Furthermore, asset prices and

volatility are typically negatively correlated (Bouchaud et al., 2001). Increased volatility typically leads to higher option prices due to the asymmetrical payoff profile of options. Potential losses are capped at the cost of the option premium, while potential gains can be theoretically unlimited for call options. Hence, increased volatility, indicative of greater price uncertainty, generally leads to higher option prices. Therefore, it is reasonable that negative shocks, which are typically associated with greater volatility, exert a greater positive influence on option prices. This dynamic underscores the benefit of incorporating return series into the option pricing model, allowing the model to capture this complex relationship more effectively than if it were to rely solely on explicit volatility measures like historical volatility.

Another plausible reason for the apparent higher option prices following negative returns might be investors' risk perception and behavior. Investors could perceive negative returns as a sign of increased risk, prompting them to buy more options to limit downside risk, which in turn drives up the option prices. On the other hand, during periods of positive returns, the perceived risk may decrease, and investors might not feel the need for such risk management, leading to a decrease in demand for options and consequently lower prices.

The relationship between the impact of positive and negative returns on option prices changes for older lags and becomes more mixed for the oldest. For some of the middle returns, the relationship appears the opposite. This might indicate some form of reversal effects. The SHAP method, while powerful, can struggle with the interpretation complexities in the presence of high-dimensional data and does not consider the time ordering of input features, which could typically have an effect on time series data. Therefore, while the SHAP analysis provides valuable insights into the behavior of our LSTM-MLP model, it may not fully reveal all patterns captured by the model due to these interpretation challenges.

Using the global explainability method ALE, Figure 5.16 illustrates the average feature importance of each feature for the last models trained each year from 2015 to 2022. The ordering of the option characteristics is the same as shown with SHAP. When aggregating the impact of each lag in the return series, it achieves an importance of 7%. This makes the return series less important than the strike and S&P500 price, but more important than the time to maturity and risk-free rate. Breaking down the impact of the return series, the most recent lag accounts for about 6 % of the total impact of the return series. After this, there is a rapid decline until around lag 15. For later lags there is a more stable and less rapid decline in importance. This indicates that the model considers the entire return series it is given, but with higher attention given to the most recent lags.

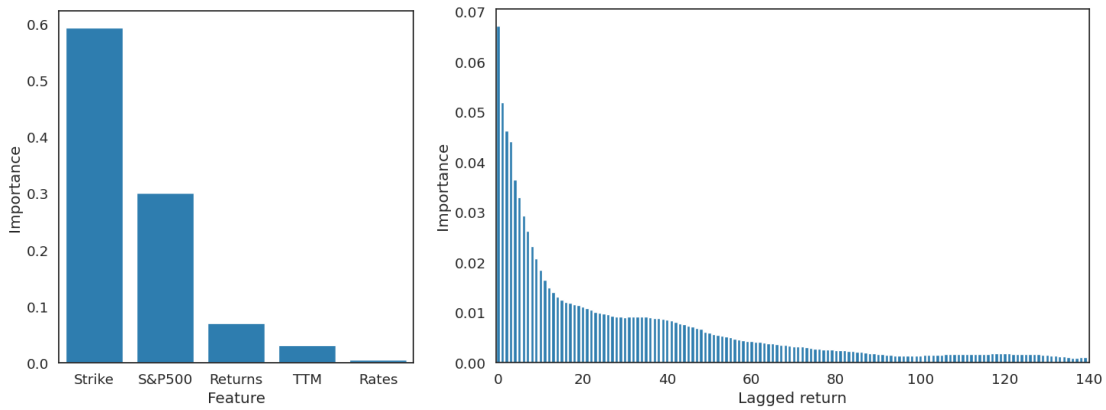


Figure 5.16 ALE feature importance for the LSTM-MLP model averaged across the last model trained each year during 2015-2022. The plot to the left shows the feature importance across the option characteristics with aggregated returns impact. The plot to the right show the feature importance of each individual lagged return towards the overall impact of the return series.

Figure 5.17 illustrates the feature importance for the MLP benchmark model when an explicit volatility input is employed. The features strike, S&P 500 value, time-to-maturity and risk-free rate maintain the same order and similar importance as observed in the LSTM-MLP model. On the

contrary, the MLP model assigns low importance to the volatility input at 0.2%. This observation aligns with Liang and Cai (2022) who also identified historical volatility as the least important feature. They reported a comparable importance percentage for the volatility input for both their MLP and LSTM models in pricing S&P 500 options.

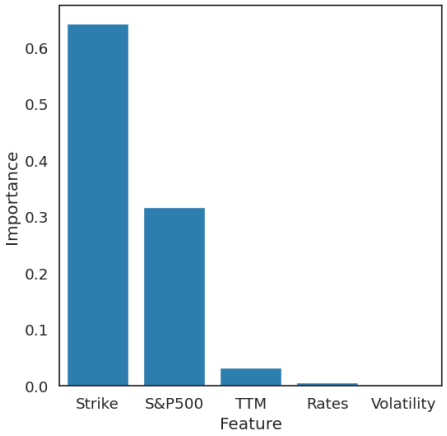


Figure 5.17 ALE feature importance for the MLP benchmark model averaged across the last model trained each year during 2015-2022.

The low attention given to the volatility input by the MLP strengthens the argument that providing explicit volatility input to these deep learning models yields less benefit to the models. One possible explanation could be that the MLP model extracts the average volatility from the training set and makes minor adjustments based on the historical volatility input. The implementation of sliding windows could further enable the MLP model to adjust learned volatility levels. If this hypothesis holds, it could explain the MLPs poor performance during periods with fluctuating volatility and returns as seen in the analysis in section 5.1.2. It could also hint at why we are seeing higher variability observed in the boxplot for the MLP as shown previously in Figure 5.9.

Nonetheless, the stark contrast between the importance assigned to the return series in the LSTM-MLP model and the volatility input in the MLP model provides additional evidence supporting the efficacy of time series information extraction using an LSTM component in the combined LSTM-MLP model. While strike and S&P500 price naturally prove the most important in determining the option price, we argue that the 7% feature importance attributed to the return series appears to be the differentiating factor causing the LSTM-MLP model to outperform the MLP.

5.2 Predictive ability of directional price movements

Table 5.11 shows the percentage of correct directional predictions of the market price by each pricing model. The model predicts an upward movement when it deems the option to be undervalued by the market, and a decrease when it sees the option as overvalued. The analysis considers different time horizons in terms of days after the pricing date, as well as different thresholds for classifying an option as underpriced or overpriced by the market[16]. It should be noted that this is not directly what the models are trained for, and ignores effects like changing market conditions or theta-decay which should reduce the value of the options as the maturity decreases all else being equal. Nonetheless, it still provides some insights into the ability of the model to identify options that are mispriced by the market.

For threshold 0%, Heston achieves the highest percentage of correct directions. However, most models achieve close to 50% prediction for the 0% threshold. For the other thresholds considered, the LSTM-MLP achieved the highest predictive performance. Interestingly, the LSTM-MLP has better predictive performance for higher thresholds. This indicates that when it is more certain

[16]A threshold of 5% means that an option is underpriced (overpriced) by the market if the model price is higher (lower) than the market price by 5% or more.

about potential mispricings by the market, it also tends to be more correct about the directional movement of the mispriced options for the days ahead.

The MLP has the second greatest predictive power considering thresholds larger than 0%, followed by Heston. The BS models appear more random in their mispricing direction. The LSTM-MLP and MLP model are trained on how the market has priced options in the last years. Misspricings by these models could therefore be interpreted as the market deviating from how it has previously priced options. We, therefore, argue that there could be more information in the mispricings by the LSTM-MLP model than in the benchmarking models. This gives the model value beyond simply trying to hit the market price. The information in the mispricings could thus be used by market participants as trading signals or other indicators of market conditions.

Table 5.11
Correct directional predictions for each model for different days ahead. Only options still alive after the given number of days are included in the calculation. Model overpricing compared to market price gives a downward prediction and underprediction gives an upward prediction. Different thresholds t for relative pricing difference compared to the market in order to generate a directional signal.

	BS 30-day	BS GARCH	BS IV	Heston	MLP	LSTM-MLP
t = 0%						
1-day (%)	49.08	49.29	48.76	**50.33**	49.46	50.06
5-day (%)	49.83	50.03	49.09	**52.78**	51.06	51.31
10-day (%)	49.95	49.06	48.66	**54.16**	52.28	50.38
t = 5%						
1-day (%)	46.23	46.56	44.63	48.00	48.88	**50.14**
5-day (%)	46.30	46.63	44.05	50.98	53.32	**54.61**
10-day (%)	46.28	44.74	43.14	52.60	56.01	**56.10**
t = 10%						
1-day (%)	45.52	46.00	43.66	47.62	48.71	**50.69**
5-day (%)	45.89	46.48	43.12	51.35	54.12	**56.64**
10-day (%)	45.57	44.53	41.69	53.37	57.48	**58.92**
t = 15%						
1-day (%)	45.24	45.66	43.17	47.38	48.61	**51.06**
5-day (%)	45.81	46.34	42.39	51.68	54.76	**58.29**
10-day (%)	45.34	44.31	40.68	53.80	58.89	**61.04**
t = 30%						
1-day (%)	44.76	44.74	42.22	46.82	48.28	**51.58**
5-day (%)	45.82	45.53	40.78	52.26	56.42	**61.80**
10-day (%)	45.47	43.23	38.47	54.56	62.26	**65.95**

5.3 Trading performance

Table 5.12 shows the average yearly trading performance by each of the option pricing models studied across the full out-of-sample test period from 2015 to 2022. The yearly trading performance used to calculate these averages can be found in Appendix E. It can be observed that the LSTM-MLP model outperforms all benchmark models measured by Sharpe, Sortino, and percentage return. However, the MLP and BS GARCH performs best in terms of maximum drawdown. Our finding that the overall best trading performance is achieved by a neural network-based pricing model is consistent with Andreou et al. (2008)[17].

The combination of high values for percentage return and simultaneously higher Sharpe and Sortino ratios emphasizes the ability of the LSTM-MLP to achieve better risk-adjusted returns than all

[17]A fair comparison to other trading implementations in the literature on neural network-based option pricing models is not possible due to the unavailability of similar model architectures and trading performance metrics. However, comparisons can be made on a rather generalized basis to provide some more context about how our results compare to previous research.

benchmark models. This suggests that the LSTM-MLP is better at accurately pricing options in a way that aligns more closely with the true option value. This ability can in turn be attributed to the LSTM-MLP architecture. By enabling the model to combine insights from time series and options data, it is better equipped to accurately calculate the true option value with regard to prevailing market conditions and option characteristics.

Table 5.12
Average yearly trading performance for each option pricing model from 2015 to 2022. Bolded values indicate the best-performing model based on the specific metric.

Metric	BS 30-day	BS GARCH	BS IV	Heston	MLP	LSTM-MLP
Sharpe	0.16	-0.52	-0.21	0.38	0.83	**1.28**
Sortino	0.31	-0.71	-0.31	0.53	1.43	**1.90**
Return (%)	2.86	-1.87	-2.31	4.70	9.52	**18.47**
MDD (%)	-10.19	-8.02	-9.92	-12.98	**-7.74**	-9.26

Table 5.13 shows the trading results in terms of alpha and beta values using the CAPM. It should be noted that even though the CAPM model is used in this research, it should not be seen as an endorsement of the model for options trading. As stated in Leland (1999), it is not ideal to apply the CAPM to strategies with positively skewed returns, for which strategies that limit downside risk will be mismeasured. Despite these potential shortcomings, CAPM is still utilized in our study as it facilitates a certain degree of performance comparison between the models and because within the hedge fund industry, α is frequently used as the go-to reference model for benchmarking trading strategies (Ewald et al., 2022).

The table shows that of all models, only the LSTM-MLP has an alpha significant at the 1% level. This implies strong statistical confidence that the excess return generated by the LSTM-MLP model is not due to chance or by taking on systematic risk, but rather the proficiency of the model. The MLP model is the only benchmark model to achieve a statistically significant alpha at the 5% level. The superior alpha for the LSTM-MLP model provides further support for the improved risk-adjusted performance of the proposed model. Analogous to how alpha traditionally represents the value added by investment managers, the superior alpha of the LSTM-MLP could be attributed to its ability to use available market information in a better way to accurately calculate the true value of options and identify truly mispriced options. As for the beta of the portfolio, its value will depend on the portfolio delta[18]. A negative beta portfolio, as observed for the LSTM-MLP, can easily be constructed by simply selling more options than you buy.

Table 5.13
Alpha and Beta values based on the CAPM for each of the option pricing models studied. The statistical significance of the results is denoted by asterisks: *** indicates significance at the 1% level, ** indicates the 5% level, and * indicates the 10% level. A model with alphas and betas that are statistically significantly different from zero is able to provide distinct predictions of asset pricing.

	BS 30-day	BS GARCH	BS IV	Heston	MLP	LSTM-MLP
α (%)	3.33	0.16	0.08	8.30	10.86	21.14
SE α	4.92	3.56	4.03	5.66	4.25	5.35
α p-value	0.4985	0.9639	0.9836	0.1430	0.0107**	8.108e-05***
β	0.0802	0.1510	0.3060	0.0527	-0.1329	-0.2564
SE β	0.0164	0.0119	0.0134	0.0189	0.0142	0.0178
β p-value	1.12e-06***	9.98e-36***	4.59e-102***	0.0053***	1.73e-20***	1.33e-44***

Figure 5.18 illustrates the development in the cash position, net value of the options portfolio[19], and total value[20] for each trading year. The graphs clearly demonstrate a consistent upward trajectory for the total value, depicted by the blue line. We also observe that in some periods,

[18]The delta of an options portfolio is a measure of the sensitivity of the portfolio's value to changes in the underlying asset's price, often interpreted as the expected change in the portfolio's value for a small change in the underlying's price

[19]Value of long options minus the value of short options.

[20]Sum of cash and net value of options portfolio.

few trades are made. This is either due to the fact that the LSTM-MLP identifies few sufficiently mispriced options at the time, or that it only identifies overpriced or underpriced options, thus not being able to create a balanced portfolio satisfying the trading rules.

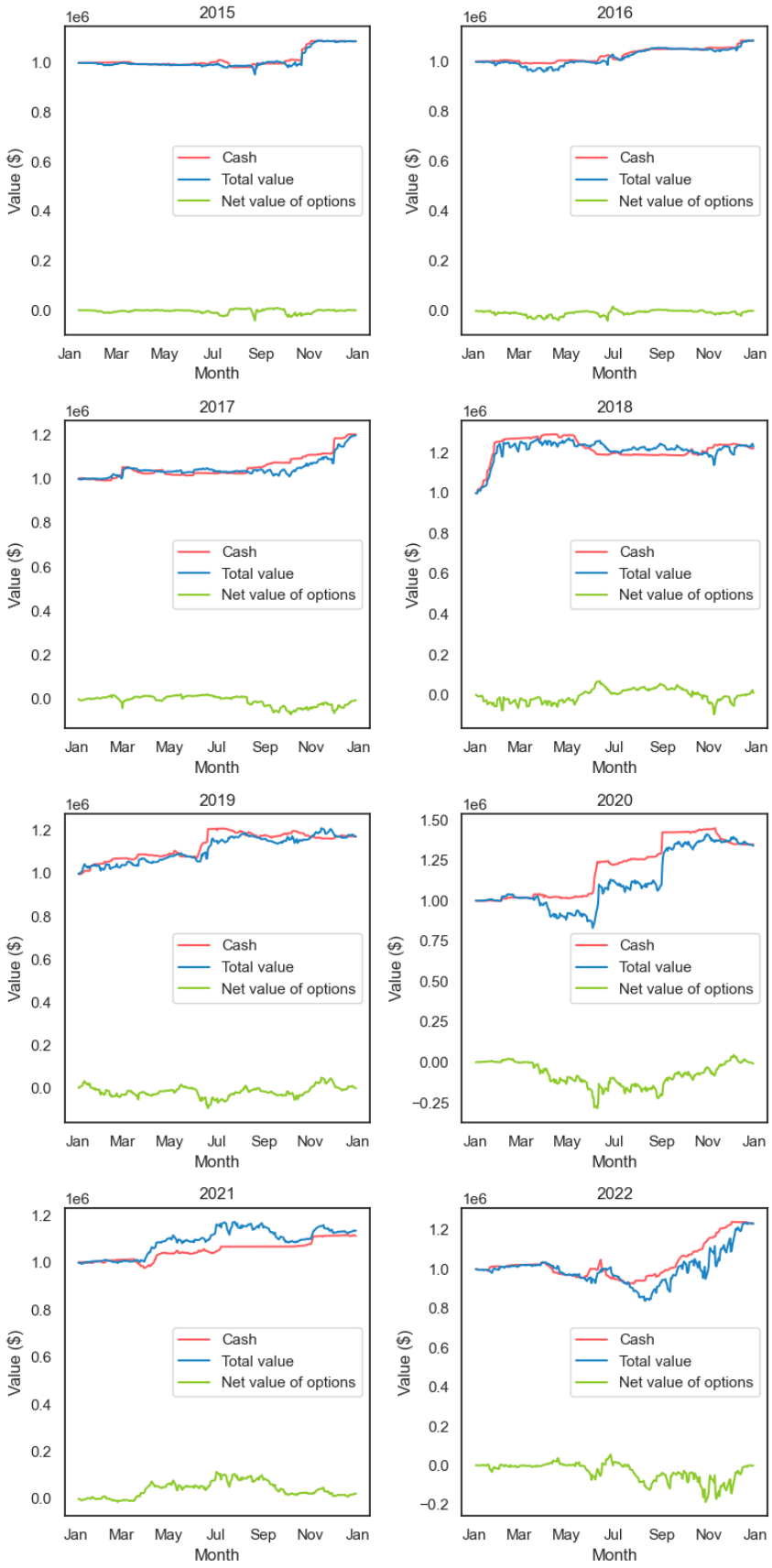


Figure 5.18 Annual progression of cash balance, net value of the options portfolio, and total portfolio value for the LSTM-MLP trading algorithm from 2015 to 2022

To better understand the trading behavior of the LSTM-MLP model, Figure 5.19 shows the distribution of options bought and sold during the period from 2015 to 2022. The median moneyness is 0.994 for the options bought, and 0.960 for options sold. Accumulated, this bears resemblance to the options trading strategy known as a bull spread in which you buy a call option at a lower strike and sell a call option on the same underlying asset at a higher strike Hull (2022). Such a strategy limits the upside as well as the downside risk, with the maximum profit being made when the strike price of the option sold is reached (Vejendla and Enke, 2013). This is an optimistic option strategy that takes advantage of a moderate increase in the price of the underlying asset (Djeutcha and Kamdem, 2022). Interestingly, the standard deviation of the moneyness distribution of options sold is 0.077 compared to 0.037 for the options bought. Speculating on why the options the market overvalues compared to the LSTM-MLP are more diverse than the ones the market undervalues is challenging due to the black-box nature of the pricing model.

Another observation from Figure 5.19 is that the longer the maturity, the more out of the money the options bought are.

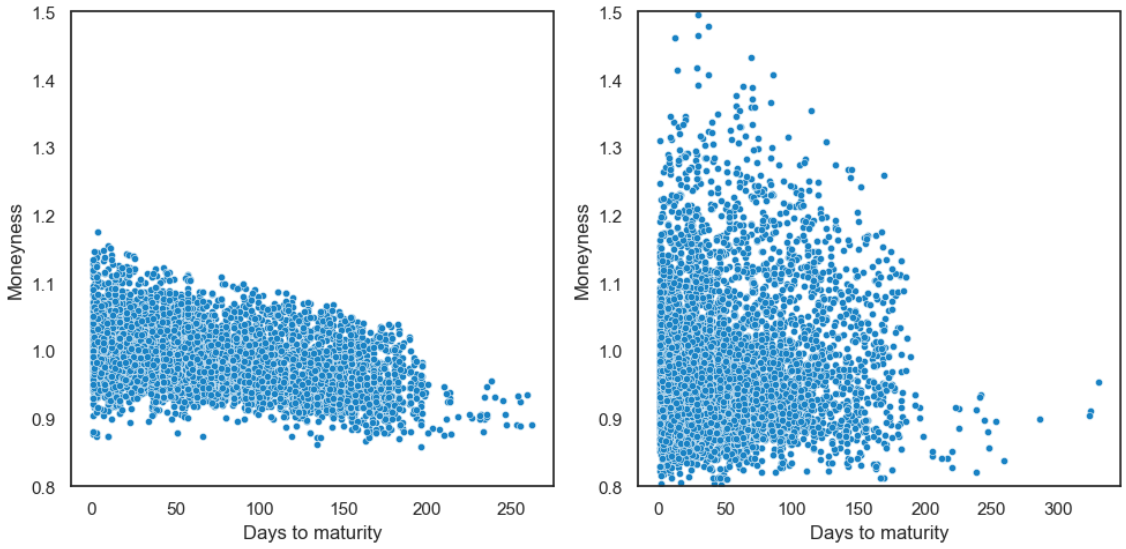


Figure 5.19 Distributions of options bought and sold at the time of the transaction. The graph to left is options bought and the graph to the right is options sold.

6 Conclusion

This paper presents a deep learning approach to option pricing using an LSTM-MLP model. The proposed model uses both time series market data and option characteristics to create a robust framework for option pricing. To unlock the value of the time series market data, historical returns for the underlying asset are given as input to the LSTM component. Given the ability of LSTM networks to learn and model temporal patterns embedded in time series data, we hypothesize that the information extracted by the LSTM component is more useful for option pricing than the explicit volatility input used by traditional option pricing models. We complement the pricing analysis of our model with an interpretability analysis using XAI frameworks to better understand the behavior of the LSTM-MLP model. Furthermore, the practical significance of the proposed pricing model is investigated using trading strategies based on mispricings identified by comparing the model price to the market price.

In terms of pricing performance, the LSTM-MLP model exhibits enhanced accuracy compared to the benchmark models. It outperforms both a standard MLP model with an explicit volatility input and traditional option pricing methods. The improved performance is consistently observed across most years and fluctuating market conditions, as characterized by VIX levels, market directional movements and the impact of Covid-19. In comparison to the benchmark MLP model, the integration of the LSTM component in the LSTM-MLP proves particularly advantageous during larger market movements and elevated volatility. The LSTM-MLP model's strong performance is most noticeable for out-of-the-money and at-the-money options across all examined maturities, as well as for in-the-money options with longer time to maturity. The environment for option pricing is perceived to be more challenging during and after the Covid-19 period for all models under consideration.

By employing the XAI frameworks of ALE and SHAP, we delve deeper into the workings of the LSTM-MLP model and discover compelling evidence for the efficacy of substituting explicit volatility input with the LSTM component. We find that the model exhibits reasonable behavior in terms of higher and lower feature values for the option characteristics. While the model pays attention to the entire return series, it attaches more significance to recent returns. Market shocks, particularly negative ones, appear to be the most influential components in the return series. The model tends to increase option prices the most in response to negative returns, suggesting that the LSTM-MLP successfully captures the leverage effect. Furthermore, the LSTM-MLP model assigns a considerably higher degree of feature importance to the provided return series compared to the explicit volatility input used in the benchmark MLP model.

Regarding trading performance, the LSTM-MLP demonstrates better risk-adjusted returns compared to the benchmark models. The LSTM-MLP model achieved an average annual return of 18.47%, outperforming the rest of the models by a considerable margin. The highest risk-adjusted returns, as measured by the Sharpe and Sortino ratios, further validate the robust performance of the LSTM-MLP. The only metric where it did not lead was the maximum drawdown, where the MLP and BS GARCH models exhibited marginally less severe drawdowns. The LSTM-MLP model also stands out as the only model to deliver a statistically significant alpha at the 1% level, reinforcing its ability to generate statistically significant excess returns.

For further research, given the demonstrated ability of the LSTM component to extract valuable time series information in the option pricing context, it would be intriguing to experiment with different sequence models such as transformers or alternative recurrent neural network architectures in combination with an MLP. To delve deeper into the discrepancies between the market price and the LSTM-MLP predicted price, we have started using these price deviations as a tool for price direction predictions and generating trading signals, but other testing methods could add to these valuable insights. Further, enhancing our trading algorithm with more realistic assumptions could provide additional evidence of the LSTM-MLP model's usefulness to generate trading signals. For example, we could examine its performance under margin requirements or implement it within the framework of delta hedging. The LSTM-MLP model could be explored directly in hedging strategies as well. It could also be interesting to dive even deeper into the interpretability analysis to further understand the deeper and more complicated time series patterns learned by the LSTM-MLP model and their changing dynamics over time. Lastly, it could be interesting to explore the

use of the proposed LSTM-MLP model architecture in other applications within finance where one could benefit from its dual design. Such applications could include forecasting stock price predictions, managing risk, optimizing portfolios, or predicting economic indicators.

As a closing note, our research motivates the use of alternative methods to incorporate time series data into option pricing. We do so by demonstrating enhanced pricing and trading performance with the proposed LSTM-MLP model relative to established models. This encourages us, and should encourage others, to explore how much more we can achieve in option pricing using deep learning techniques.

Bibliography

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

Anders, U., Korn, O., and Schmitt, C. (1998). Improving the pricing of options: A neural network approach. *Journal of Forecasting*, 17(5-6):369–388.

Andreou, P. C., Charalambous, C., and Martzoukos, S. H. (2008). Pricing and trading European options by combining artificial neural networks and parametric models with implied parameters. *European Journal of Operational Research*, 185(3):1415–1433.

Apley, D. W. and Zhu, J. Y. (2016). Visualizing the effects of predictor variables in black box supervised learning models.

Bakshi, G., Cao, C., and Chen, Z. (1997). Empirical performance of alternative option pricing models. *The Journal of Finance*, 52:2003–2049.

Bennell, J. and Sutcliffe, C. (2004). Black–Scholes versus artificial neural networks in pricing FTSE 100 options. *Intelligent Systems in Accounting, Finance and Management*, 12(4):243–260.

Black, F. (1976). Studies of stock price volatility changes. *Proceedings of the 1976 Meeting of the Business and Economic Statistics Section*, pages 177–181.

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

Bloch, D. A. (2023). A review of 'The pricing of options and corporate liabilities'. *SSRN Electronic Journal*.

Blyth, S. (2014). *An Introduction to Quantitative Finance*. Oxford University Press.

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

Boness, A. J. (1964). Elements of a theory of stock-option value. *Journal of Political Economy*, 72(2).

Bouchaud, J.-P., Matacz, A., and Potters, M. (2001). Leverage effect in financial markets: The retarded volatility model. *Physical Review Letters*, 87:228701.

Cao, Y., Liu, X., and Zhai, J. (2021). Option valuation under no-arbitrage constraints with neural networks. *European Journal of Operational Research*, 293:361–374.

Christie, A. A. (1982). The stochastic behavior of common stock variances: Value, leverage and interest rate effects. *Journal of Financial Economics*, 10(4):407–432.

Culkin, R. and Das, R. (2017). Machine learning in finance : The case of deep learning for option pricing. *Journal Of Investment Management*, 15(4):1–9.

Djeutcha, E. and Kamdem, J. S. (2022). Pricing for a vulnerable bull spread options using a mixed modified fractional Hull–White–Vasicek model. *Annals of Operations Research*.

Egelkraut, T. M. and Garcia, P. (2006). Intermediate volatility forecasts using implied forward volatility: The performance of selected agricultural commodity options. *Journal of Agricultural and Resource Economics*, 31(3):508–528.

Ewald, C.-O., Haugom, E., Lien, G., Størdal, S., and Wu, Y. (2022). Trading time seasonality in commodity futures: An opportunity for arbitrage in the natural gas and crude oil markets? *Energy Economics*, 115:106324.

Fadda, S. (2020). Pricing options with dual volatility input to modular neural networks. *Borsa Istanbul Review*, 20:269–278.

Fan, R., Taylor, S. J., and Sandri, M. (2017). Density forecast comparisons for stock prices, obtained from high-frequency returns and daily option prices. *Journal of Futures Markets*, 38(1).

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *The Annals of Statistics*, 29(5).

Fukasawa, M. and Gatheral, J. (2021). A rough SABR formula. *Frontiers of Mathematical Finance*.

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

Gençay, R. and Gibson, R. (2007). Model risk for European-style stock index options. *IEEE Transactions on Neural Networks*, 18:193–202.

Gradojevic, N. (2016). Multi-criteria classification for pricing European options. *Studies in Nonlinear Dynamics and Econometrics*, 20:123–139.

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

Gradojevic, N. and Kukolj, D. (2022). Unlocking the black box: Non-parametric option pricing before and during Covid-19. *Annals of Operations Research*.

Hagan, P. S., Kumar, D., Lesniewski, A., and Woodward, D. E. (2002). Managing smile risk. *Wilmott*, 1:84–108.

Haug, E. and Taleb, N. (2011). Option traders use (very) sophisticated heuristics, never the Black-Scholes–Merton formula. *Journal of Economic Behavior Organization*, 77(2):97–106.

Heston, S. (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.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Hornik, K., Stinchcombe, M., and White, H. (1989). Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366.

Hsu, P. Y., Chou, C., Huang, S. H., and Chen, A. P. (2018). A market-making quotation strategy based on dual deep learning agents for option pricing and bid-ask spread estimation. *Proceedings - 2018 IEEE International Conference on Agents, ICA 2018*, pages 99–104.

Hull, J. (2022). *Options, Futures and Other Derivatives*. Pearson.

Hull, J. (2023). Why Has Black-Scholes-Merton Been So Successful? *Wilmott*, 2023(125).

Hull, J. and White, A. (1987). The pricing of options on assets with stochastic volatilities. *The Journal of Finance*, 42(2):281–300.

Hull, J. and White, A. (1988). An analysis of the bias in option pricing caused by a stochastic volatility. *Advances in Futures and Options Research*, 2.

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*, 49(3):851–889.

Iltüzer, Z. (2022). Option pricing with neural networks vs. Black-Scholes under different volatility forecasting approaches for BIST 30 index options. *Borsa Istanbul Review*, 22:725–742.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift.

Ivaşcu, C. (2021). Option pricing using machine learning. *Expert Systems with Applications*, 163.

Jerbi, Y. and Chaabene, S. (2020). European call price modeling using neural networks in considering volatility as stochastic with comparison to the Heston model. 90:1793–1810.

Jorion, P. (1995). Predicting volatility in the foreign exchange market. *The Journal of Finance*, 50(2):507–528.

Ke, A. and Yang, A. (2019). *Option Pricing with Deep Learning*. Research paper for CS230: Deep learning, Stanford University.

Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization.

Larikka, M. and Kanniainen, J. (2012). Calibration strategies of stochastic volatility models for option pricing. *Applied Financial Economics*, 22(23):1979–1992.

Leland, H. E. (1999). Beyond mean-variance: Performance measurement in a nonsymmetrical world. *Financial Analysts Journal*, 55(1):27–36.

Levenberg, K. (1944). A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, 2(2):164–168.

Liang, L. and Cai, X. (2022). Time-sequencing european options and pricing with deep learning – analyzing based on interpretable ALE method. *Expert Systems with Applications*, 187.

Lin, C. T. and Yeh, H. Y. (2009). Empirical of the Taiwan stock index option price forecasting model–applied artificial neural network. *Applied Economics*, 41:1965–1972.

Liu, D. and Wei, A. (2022). Regulated LSTM artificial neural networks for option risks. *FinTech*, 1(2):180–190.

Liu, S., Oosterlee, C., and Bohte, S. (2019). Pricing options and computing implied volatilities using neural networks. *Risks*, 7(1):16.

Ludkovski, M. (2023). Statistical machine learning for quantitative finance. *Annual Review of Statistics and Its Application*, 10:271–295.

Lundberg, S. and Lee, S. I. (2017). A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems 30 (NIPS 2017)*.

Maas, A. L., Hannun, A. Y., and Ng, A. Y. (2013). Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of the 30th International Conference on Machine Learning*, volume 30.

Marquardt, D. W. (1963). An algorithm for least-squares estimation of nonlinear parameters. *Journal of the society for Industrial and Applied Mathematics*, 11(2):431–441.

Merton, R. (1973). Theory of rational option pricing. *The Bell Journal of Economics and Management Science*, 4(1):141–183.

Mikhailov, S. and Nögel, U. (2004). *Heston's stochastic volatility model: Implementation, calibration and some extensions*. John Wiley and Sons.

Molnar, C. (2022). *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*. 2nd edition.

Natenberg, S. (2015). *Option Volatility and Trading*. McGraw-Hill Education.

Olah, C. (2015). Understanding LSTM networks. `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`. Accessed: 2022-11-24.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). Why should i trust you?: Explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*.

Ruf, J. and Wang, W. (2020). Neural networks for option pricing and hedging: A literature review. *Journal of Computational Finance*, 24(1):1–46.

Samuelson, P. and Merton, R. C. (1969). A complete model of warrant pricing that maximizes utility. *Industrial Management Review*.

Sharpe, W. F. (1966). Mutual fund performance. *The Journal of Business*, 39(1):119–138.

Sood, S., Jain, T., Batra, N., and Taneja, H. C. (2022). Black–Scholes option pricing using machine learning. *Proceedings of International Conference on Data Science and Applications*, 1.

Sortino, F. A. (1994). Performance measurement in a downside risk framework. *The Journal of Investing*, 3(3):59–65.

Sprenkle, C. M. (1961). Warrant prices as indicators of expectations and preferences. *Yale Economics Essays*, 1(2).

Tseng, C. H., Cheng, S. T., Wang, Y. H., and Peng, J. T. (2008). Artificial neural network model of the hybrid EGARCH volatility of the Taiwan stock index option prices. *Physica A: Statistical Mechanics and its Applications*, 387:3192–3200.

Vejendla, A. and Enke, D. (2013). Performance evaluation of neural networks and GARCH models for forecasting volatility and option strike prices in a bull call spread strategy. *Journal of Economic Policy and Research*, 8(2).

Wang, C. P., Lin, S. H., Huang, H. H., and Wu, P. C. (2012). Using neural network for forecasting TXO price under different volatility models. *Expert Systems with Applications*, 39:5025–5032.

Wang, M., Zhang, Y., Qin, C., Liu, P., and Zhang, Q. (2022). Option pricing model combining ensemble learning methods and network learning structure. *Mathematical Problems in Engineering*, 2022.

Wang, Y. H. (2009a). Nonlinear neural network forecasting model for stock index option price: Hybrid GJR–GARCH approach. *Expert Systems with Applications*, 36:564–570.

Wang, Y. H. (2009b). Using neural network to forecast stock index option price: A new hybrid garch approach. *Quality and Quantity*, 43:833–843.

Wei, X., Xie, Z., Cheng, R., Zhang, D., and Qing, L. (2021). An intelligent learning and ensembling framework for predicting option prices. *Emerging Markets Finance and Trade*, 57(15):4237–4260.

Yang, Y., Zheng, Y., and Hospedales, T. (2017). Gated neural networks for option pricing: Rationality by design. *31st AAAI Conference on Artificial Intelligence*.

Yao, J., Li, Y., and Tan, C. (2000). Option price forecasting using neural networks. *Omega*, 28(4):455–466.

Appendix

A Nelson–Siegel–Svensson for generating rate curves

NSS is a method to fit discrete data points to a smooth and continuous curve. The resulting yield curve is matched with the observed interest rates at different maturities.

The NSS model is defined as:

$$y(\tau) = \beta_0 + \beta_1 \frac{1 - e^{-\lambda\tau}}{\lambda\tau} + \beta_2 \left(\frac{1 - e^{-\lambda\tau}}{\lambda\tau} - e^{-\lambda\tau} \right) + \beta_3 \left(\frac{1 - e^{-\lambda_2\tau}}{\lambda_2\tau} - e^{-\lambda_2\tau} \right) \tag{A.1}$$

where $y(\tau)$ is the yield for a given time to maturity τ, β_0, β_1, β_2, and β_3 are the model parameters, and λ and λ_2 are the exponential decay factors. The parameters of the NSS model are estimated by minimizing the sum of squared differences between the observed yields and the estimated yields from the model.

We apply the NSS model to the discrete interest rate data for each day in order to obtain the yield curve on that day. We then match each specific option's TTM with the maturity on the yield curve in order to obtain the interest rate to use for pricing.

In cases where the optimization of the NSS model fails or the TTM is less than one month, we use linear interpolation as a fallback method to estimate the risk-free rate. This ensures that we have a consistent and accurate estimate of the risk-free rate for each time to maturity in our data set.

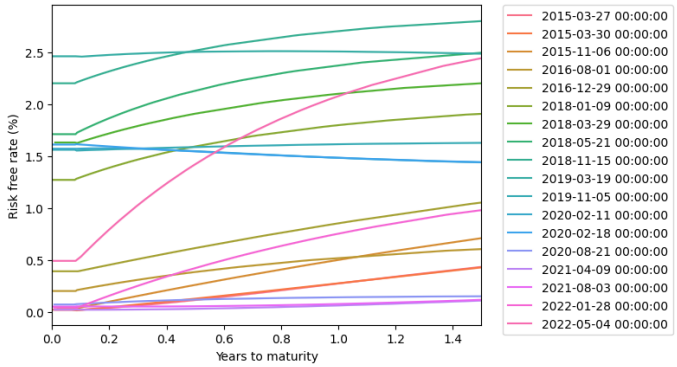


Figure A.1 18 Random interest rate curves generated using the NSS method

B MLP model hyperparameter search

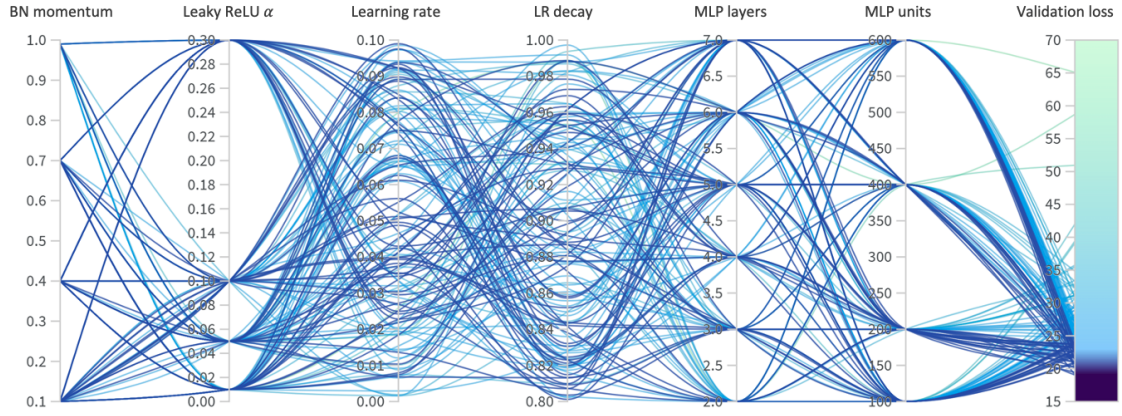


Figure B.1 Hyperparameter search for the MLP model in terms of hyperparameter combinations, hyperparameter value ranges and MSE validation loss. Results are shown for model runs testing 150 distinct combinations of hyperparameter values. The data set used consists of training data for the three years from November 2011 to October 2014, and validation data for the two months from November to December 2014.

Table B.1
Configuration for the MLP model grouped by model component. The parameters layers, units, BN momentum, alpha constant, learning rate and learning rate decay were found during the hyperparameter search using Wandb while the others were determined manually beforehand to reduce the hyperparameter space

	Parameter	Value
MLP	Layers	4
	Units per layer	200
	Batch normalization momentum	0.70
	Activation function	Leaky ReLu
	Alpha constant for Leaky ReLu	0.01
Training	Training set samples	1,000,000
	Learning rate	4.9×10^{-2}
	Learning rate decay	0.82
	Minibatch size	4 096
	Epochs	Early stopping
	Optimizer	Adam

C Interpretability analysis

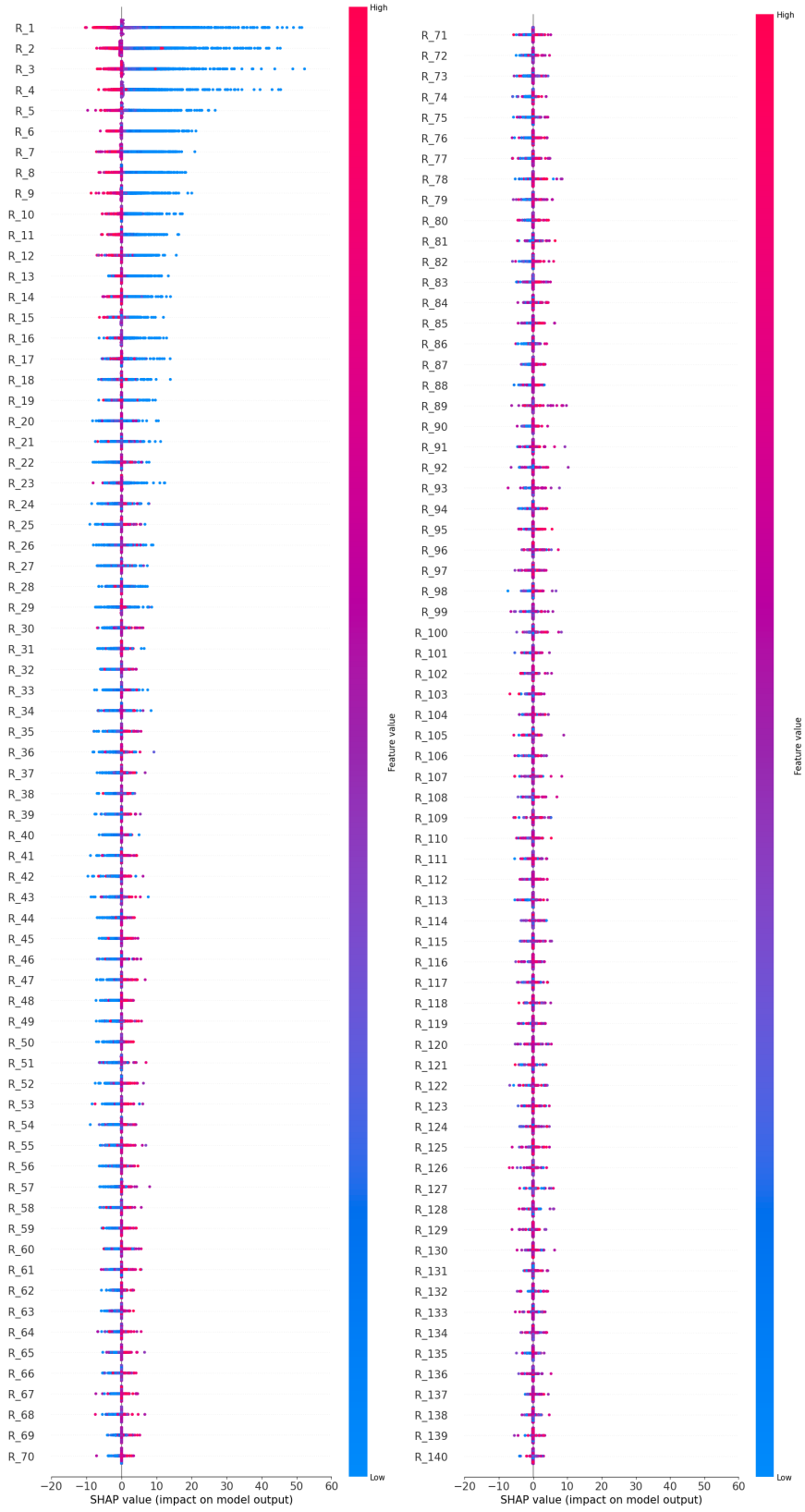


Figure C.1 SHAP summary plot of feature importance and positive or negative impact on model output. The results shown are for the latest LSTM-MLP model trained on February 2020 - January 2022. SHAP feature values are evaluated as high or low relative to a baseline.

D Market conditions

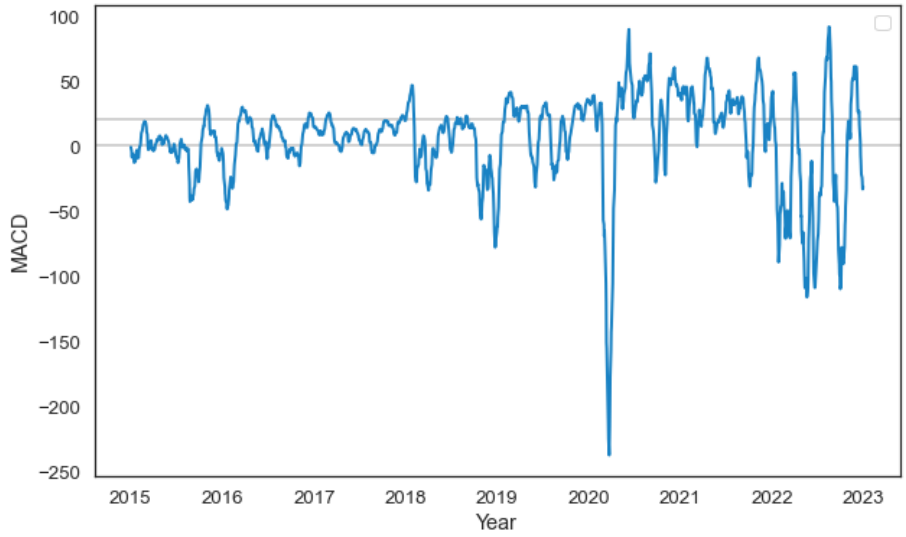


Figure C.1 MACD during the testing period. Grey lines indicate threshold values of 0.93 and 21.12 used during the analysis in section 5.1.2

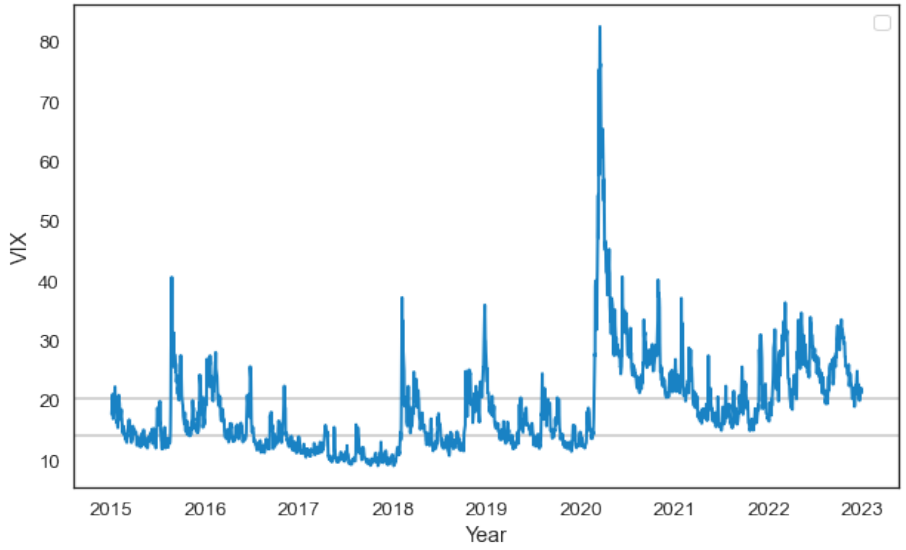


Figure C.2 VIX during the testing period. Grey lines indicate threshold values of 14.22 and 20.48 used during the analysis in section 5.1.2

E Trading performance by year

Note the discrepancies between Table 5.13 and the α values reported in Appendix E, due to the former being based on a regression performed on returns in the entire 2015-2022 period, while the latter is the average of the coefficients from running the regression on each year individually.

Table E.1
BS 30-day trading performance by year

Year	Sharpe	Sortino	Return (%)	MDD (%)	α(%)
2015	-0.33	-0.55	-1.09	-2.91	-0.28
2016	0.29	0.14	2.37	-9.61	2.37
2017	0.89	1.40	7.56	-4.85	6.43
2018	-0.09	0.01	2.18	-6.62	6.06
2019	-0.13	-0.25	2.26	-11.19	0.16
2020	0.39	1.11	6.34	-9.09	7.10
2021	0.33	0.52	6.86	-19.50	24.60
2022	-0.05	0.12	-3.62	-17.76	6.82
Average	0.16	0.31	2.86	-10.19	6.66

Table E.2
BS GARCH trading performance by year

Year	Sharpe	Sortino	Return (%)	MDD (%)	α(%)
2015	0.16	0.28	0.77	-7.33	2.39
2016	-0.11	-0.14	-0.61	-4.27	-0.34
2017	-1.00	-1.40	-0.17	-1.20	-0.50
2018	-0.99	-1.29	-1.37	-2.12	0.49
2019	-1.17	-1.63	-1.17	-3.10	-3.42
2020	0.34	0.50	5.89	-10.09	6.67
2021	-1.02	-1.64	-11.07	-18.06	-16.96
2022	-0.41	-0.37	-7.26	-17.98	4.71
Average	-0.52	-0.71	-1.87	-8.02	-0.87

Table E.3
BS IV trading performance by year

Year	Sharpe	Sortino	Return (%)	MDD (%)	α(%)
2015	-0.01	0.22	-0.47	-7.37	0.83
2016	-0.14	-0.27	-1.01	-6.61	-1.81
2017	-0.40	-0.63	-0.60	-4.20	-3.65
2018	-0.60	-0.77	0.50	-4.05	2.79
2019	-0.29	-0.35	-2.51	-9.39	-6.54
2020	0.09	0.22	-5.31	-27.62	5.64
2021	0.02	-0.49	-0.30	-0.55	-0.43
2022	-0.35	-0.42	-8.77	-19.60	7.39
Average	-0.21	-0.31	-2.31	-9.92	0.53

Table E.4
Heston trading performance by year

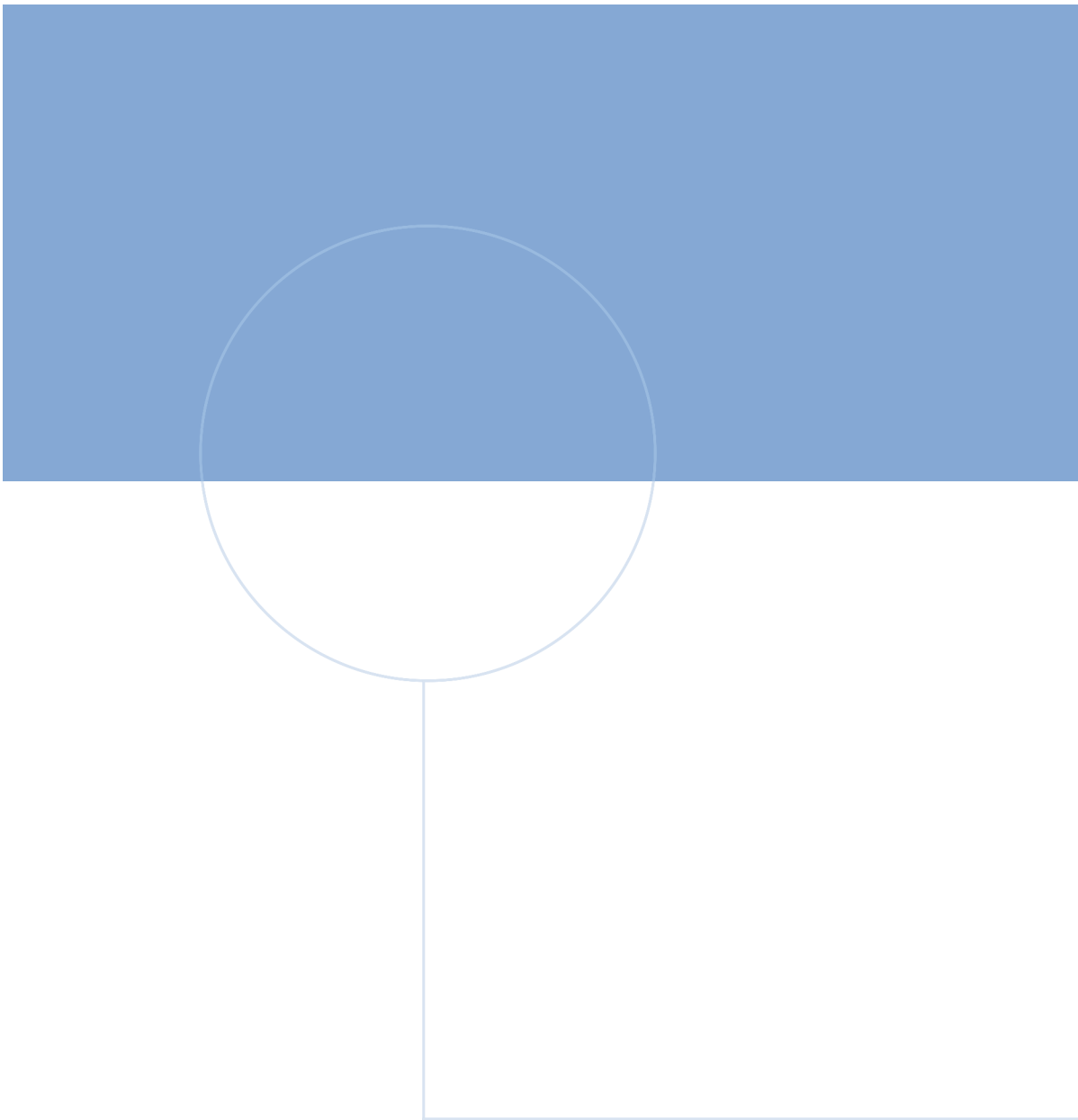
Year	Sharpe	Sortino	Return (%)	MDD (%)	α(%)
2015	0.18	0.65	0.73	-2.67	1.73
2016	0.43	-0.29	-0.08	-5.70	0.98
2017	0.66	1.21	3.03	-4.31	-6.10
2018	0.60	0.83	12.31	-11.07	16.64
2019	-0.03	0.05	-1.99	-23.42	-1.66
2020	0.58	0.94	7.59	-13.95	17.82
2021	-0.15	-0.07	-2.18	-19.04	0.39
2022	0.73	0.90	18.15	-23.68	22.80
Average	0.38	0.53	4.70	-12.98	6.58

Table E.5
MLP trading performance by year

Year	Sharpe	Sortino	Return (%)	MDD (%)	α(%)
2015	0.42	0.52	2.29	-2.93	3.94
2016	0.56	1.21	8.12	-5.15	11.34
2017	1.40	2.86	16.36	-3.67	13.32
2018	1.71	2.26	22.20	-9.79	25.88
2019	0.40	0.45	8.41	-7.68	18.25
2020	0.96	2.18	14.73	-9.62	21.06
2021	1.73	2.68	13.83	-3.03	14.56
2022	-0.52	-0.72	-9.79	-20.06	-8.96
Average	0.83	1.43	9.52	-7.74	12.42

Table E.6
LSTM-MLP trading performance by year

Year	Sharpe	Sortino	Return (%)	MDD (%)	α(%)
2015	1.16	1.61	8.82	-4.78	11.95
2016	0.99	1.29	8.56	-4.10	12.73
2017	1.66	2.54	20.50	-3.93	23.29
2018	1.44	1.97	22.34	-10.46	25.78
2019	1.44	2.19	17.46	-4.28	24.49
2020	1.29	2.72	35.07	-20.12	43.11
2021	1.44	1.93	11.49	-7.47	5.61
2022	0.83	0.98	23.55	-18.92	23.14
Average	1.28	1.90	18.47	-9.26	21.26



NTNU

Norwegian University of
Science and Technology