ERASMUS UNIVERSITEIT ROTTERDAM

Erasmus School of Economics

Master Thesis Quantitative Finance

A Hybrid Approach to Modular and Gated Neural Networks for Option Pricing and Hedging

Name student:

R.W.M. Duk

Student ID:

432089

Academic supervisor:

Prof. Dr. D.J.C. van Dijk

Second assessor:

Prof. Dr. M. van der Wel

October 11, 2020

Abstract

This paper applies the hybrid neural network approach introduced by Boek et al. (1995), which has been shown to outperform artificial neural networks (Andreou et al., 2008), to the modular neural network from Gradojevic et al. (2009), the multi-criteria modular neural network from Gradojevic (2016) and the gated neural network from Yang et al. (2017) for the pricing of S&P 500 European call options. Parametric option pricing models from Black and Scholes (1973), Corrado and Su (1996), Heston (1993), Kou (2002) and Madan et al. (1998) serve as a benchmark. The option price forecasts of these models are employed to adjust the target function of the hybrid neural network models. Additional input parameters, as first proposed by Palmer (2019), are implemented to extend the neural network models. A single optimal model architecture for all regarded neural network models is utilized to price options. The option pricing performance is evaluated statistically and economically by means of the Diebold and Mariano (1995) test, the model confidence set from Hansen et al. (2011) and a delta-hedged trading strategy. I conclude that introducing a hybrid approach, modularity and additional input parameters significantly improves the option pricing performance of neural networks. Benchmarks are significantly outperformed by hybrid neural network models on the entire moneyness and maturity grid. Finally, I find that the option price forecasts obtained from the hybrid neural networks produce abnormal risk-adjusted returns before transaction costs when implemented in the delta-hedged trading strategy.

Contents

1 Introduction

This paper researches the application of a hybrid approach to modular and gated neural networks for option pricing and hedging. To fully comprehend what this research entails, previously introduced option pricing techniques must first be reviewed. Both parametric and non-parametric option pricing techniques are introduced, as these are the two building blocks of the hybrid approach to neural networks. Subsequently, the modular and gated neural network models are elucidated. Each of these neural network models introduces modularity, a technique to optimize the option pricing performance specifically per moneyness and maturity region. Finally, techniques to improve the option pricing performance of the non-parametric models are introduced, such as an optimal model architecture and additional input parameters.

Non-parametric techniques such as artificial neural networks (ANNs) are promising alternatives to parametric option pricing models, because they do not make as many assumptions about the underlying asset price dynamics. They essentially do not rely on any financial theory. Instead, option prices are inductively estimated from input variables using a multivariate and highly nonlinear option pricing function. The method is thus robust to specification errors. Malliaris and Salchenberger (1993) and Hutchinson et al. (1994) were the first to model option prices with ANNs. They showed that ANNs can be used to estimate an option pricing function with accurate out-of-sample pricing and profitable delta-hedging performance.

While ANNs at times improve upon the parametric models, they struggle to price options on the full moneyness and maturity grid; Lajbcygier (2004) found that standard neural networks "perform erratically for deep-in-the-money options due to a lack of option transactions in the region." Based on the idea that the conventional parametric models output fairly accurate approximations of option prices accross moneyness and maturity regions, Boek et al. (1995) first introduced a hybrid neural network model for option pricing. Because estimating the correct empirical option pricing function is oftentimes troublesome, the hybrid approach uses a parametric model in appropriate combination with a neural network model. Instead of estimating the entire option pricing function from scratch, the neural network is configured to model deviations from outputs of a conventional parametric option pricing model. More specifically, the target function of the hybrid neural network models is the residual between the observed market price and the price estimate of a certain parametric option pricing model. This approach has proved valuable, having succeeded in attaining a more accurate option pricing performance than either parametric models or artificial neural network methods in numerous papers, e.g. Lajbcygier and Conner (1997), Anders et al. (1998), Andreou et al. (2002), Andreou et al. (2008) and Chen and Sutcliffe (2012). This paper is mainly devoted to researching the application of the hybrid approach to neural networks for option pricing.

An interesting avenue to explore is employing more sophisticated parametric option pricing models than used in previous research for the hybrid target function and investigating which hybrid neural network model is able to price options most accurately. Andreou et al. (2008) only used the Black and Scholes (1973) model and the Corrado and Su (1996) model to calculate the option price estimate needed for the target function of the hybrid ANN. However, Lajbcygier (2004) noted that comparisons with modern parametric models may be useful in showing the success of the hybrid neural networks for option pricing. Examples of modern parametric option pricing models are stochastic volatility models and exponential Lévy models. This paper therefore intends to further contribute to existing literature by investigating whether the accuracy of the option price forecasts of hybrid neural networks improves when the target function is adjusted by the output of stochastic volatility models or exponential Lévy models.

The main reason for introducing additional parametric option pricing models is because it has been widely shown that the Black and Scholes (1973) model is misspecified and suffers from systematic biases. These biases are for example demonstrated by Black (1975) and Bakshi et al. (1997). The biases stem from the fact that "the model has been developed under a set of simplified assumptions, such as geometric Brownian motion of stock price movements, constant variance of the underlying returns, continuous trading on the underlying asset and constant interest rates" (Andreou et al., 2006). In recent years, many studies have tried to discover novel option pricing models by relaxing and generalizing these restrictive assumptions. Due to the constant volatility assumption, the Black-Scholes model is unable to describe the complete set of option prices for different moneyness and maturities regions. Heston (1993) proposed a solution to this problem by considering a random process to model the stochastic volatility, thereby mitigating much of the volatility smile bias. The Heston model uses stochastic variables to support the assumption that volatility is not constant but arbitrary. Models based on this assumption are labeled stochastic volatility models. Another approach to solving the volatility smile problem generalizes the Black-Scholes assumptions by allowing the stock prices to jump. Stock price data often exhibits discontinuity, such that continuous-path models are unable to fit the data. Furthermore, the volatility smile shows that distribution of the returns is non-Gaussian and leptokurtic. Jumps are therefore introduced to represent unique events in the underlying process. Models with jumps belong to the class of exponential Lévy models. Examples of such models are the Variance Gamma model introduced by Madan et al. (1998) and the Kou (2002) jump-diffusion model. As stochastic volatility models and exponential Lévy models are designed to improve upon the Black-Scholes model, it is appealing to research whether the hybrid neural networks adjusted by these parametric models are able to outperform hybrid neural networks adjusted by the Black-Scholes model.

A drawback of ANNs and the corresponding hybrid ANNs is that they intend to find a single pricing function for the entire range of traded options. Lajbcygier (2004) noted that in his earlier work (Lajbcygier and Conner, 1997) there existed pricing biases in boundary regions. For instance, the price of options approaching maturity was severely underestimated; a phenomenon observed by Dugas et al. (2001) as well. Gençay and Salih (2003) and Bennell and Sutcliffe (2004) found that estimating option prices separately per moneyness region improved the pricing accuracy of their neural network methods. Inspired by these findings, Gradojevic et al. (2009) implemented an ANN whereby they categorized the options based on their moneyness and time-to-maturity and trained the ANN for each module of options. They referred to this model as a modular neural network (MNN). The MNN consistently outperformed parametric and non-parametric benchmark models in terms of out-of-sample pricing performance, implying that introducing modularity improves the generalization performance of neural networks.

Yang et al. (2017) stated that the categorization of options performed by Gradojevic et al. (2009) may not be consistent with changing market data over time, as it is completely based on fixed manual heuristics. To substantiate their claims, Yang et al. (2017) introduced a new class of neural networks for option pricing, which they labeled the gated neural network (GNN). The GNN also implements a modular approach with the same input variables but with option grouping that is automatic and learned from data. The GNN can thus dynamically adjust the grouping and the pricing function of each group as the market changes over time. Furthermore, Yang et al. (2017) incorporated economic and financial axioms as constraints into their neural network, expanding the work of Dugas et al. (2001). Zheng (2018) referred to this approach as a Bayesian-alike design approach in which prior information is encoded into the neural network. The GNN improves the option grouping of the previously introduced MNN and contributes to the existing literature a neural network that comes with guarantees about the economic and financial rationality of its outputs. Finally, Yang et al. (2017) showed that their GNN approach yields significantly better pricing of S&P 500 index options than other neural networks and alternative econometric methods.

Neither Gradojevic et al. (2009) nor Yang et al. (2017) made use of the hybrid target function that has been shown to outperform the standard target function. This paper intends to further contribute to existing literature by researching whether hybrid MNNs and GNNs can outperform the option pricing performance of standard MNNs and GNNs. In doing so, this paper also researches whether the introduction of modularity based on fixed manual heuristics or automatic option grouping improves the option pricing performance of standard and hybrid neural networks. The range of parametric and non-parametric models included in this research ensures an elaborate analysis of the application of the hybrid approach.

Ruf and Wang (2019) summarized more than 150 papers that use ANNs as a non-parametric option pricing or hedging tool. Contrary to Gradojevic et al. (2009) and Yang et al. (2017), a fair amount of these papers considers more than two input variables. The most commonly used input variables besides moneyness and time-to-maturity are volatility and the risk-free rate, as for example implemented by Culkin and Das (2017). Recently, Palmer (2019) introduced a parameter space reduction technique that reduces the resulting four input parameters to three, without information loss. A reduction of the number of input variables is important as it simplifies inference for the neural network. To expand my research, the ANN, MNN, GNN and their hybrid counterparts are all examined with three input parameters as well as with their original input parameters. The modules of the MNN are adjusted accordingly to include a module for the volatility, resulting in the multi-criteria modular neural network (MCMNN) of Gradojevic (2016).

Instead of assuming a fixed activation function and a fixed number of hidden layers and hidden layer nodes, I consider various activation functions and a range of numbers of hidden layers and hidden layer nodes. Especially the optimal architecture of activation functions is an interesting research topic, as in most previous research on this topic, e.g. by Andreou et al. (2008) and Gradojevic et al. (2009), only the sigmoid and hyperbolic tangent activation functions for the hidden layer have been employed. However, Glorot et al. (2011) and Krizhevsky et al. (2012) found that networks with rectifying neurons in the hidden layer yield better results than networks with hyperbolic tangent neurons and that deep neural networks with rectified linear unit (ReLU) activation functions train much faster. As faster learning greatly influences the performance of large neural network models trained on large data sets, it is interesting to examine whether neural network models with other activation functions in the hidden layer have a superior option pricing performance. An optimal all-encompassing model architecture for all regarded neural network models is inspected and ultimately determined based on the optimal option pricing performance of each model.

To be able to compare option pricing performance in this paper I price S&P 500 European call options. Option price estimates of the parametric models are obtained directly from their respective formulas. The parameters of the Heston, Kou and Variance Gamma model are calibrated daily. The optimal architecture for the neural network models is investigated and thereafter used to obtain option prices. The obtained option prices are first evaluated in terms of mean absolute percentage error (MAPE) and mean squared error (MSE). To assess the statistical significance of the difference in the out-of-sample forecast accuracy the Diebold-Mariano test of Diebold and Mariano (1995) is performed and the model confidence set of Hansen et al. (2011) is regarded. Lastly, the economic significance of the difference in the out-of-sample forecast accuracy is tested by implementing a simple trading strategy based on the predicted option prices.

I find that in terms of MAPE and MSE the Kou jump-diffusion model is the best-performing parametric option pricing model. The optimal all-encompassing model architecture for all neural network models consists of three hidden layers and applies the linear activation function in the output layer and the ReLU activation function in the hidden layers. Most importantly, the option prices from the exponential Lévy models aid the hybrid neural networks in consistently outperforming the respective standard neural networks. The standard ANN, MNN and GNN with two input parameters are all outperformed by their hybrid neural network counterparts in terms of both MAPE and MSE. The additional input parameters further improve the option pricing performance of all standard neural network models. The hybrid neural network models adjusted by exponential Lévy models are however still able to outperform the standard neural network models in terms of MSE, as their option pricing performance also improves.

The MNN with three input parameters attains the best option pricing performance in terms of MAPE, whereas the hybrid GNN with three input parameters adjusted by the Variance Gamma model attains the best option pricing performance in terms of MSE. The Diebold-Mariano test determines that the forecasts of these models significantly outperform the forecasts of all other models as measured by the respective performance metrics. The model confidence sets indicate that the MNN with three input parameters prices out-of-the-money and at-the-money options significantly better than any other model. However, the majority of the options in the data set is comprised of in-the-money options. For these options the model confidence sets consist solely of hybrid neural networks adjusted by exponential Lévy models. Finally, implementing the delta-hedged trading strategy reveals that the option price forecasts of the MNN with three input parameters yield a better mean return per invested dollar and Sharpe ratio than the forecasts of any other model. Hybrid neural networks adjusted by exponential Lévy models are also able to produce abnormal risk-adjusted returns before transaction costs.

The rest of this paper is constructed as follows. First, the non-parametric option pricing models are presented in Section 2. The hybrid approach for each of the non-parametric models is also introduced. Furthermore, the model estimation and evaluation techniques are described. Section 3 then introduces the data sample consisting of S&P 500 index options and discusses the exclusion criteria that are incorporated. In Section 4 the option pricing performance of the parametric and non-parametric models is discussed and the statistical and economic significance of the difference in out-of-sample forecast accuracy is tested. Finally, Section 5 concludes and Section 6 discusses and provides topics for further research.

2 Methodology

2.1 Non-Parametric Models

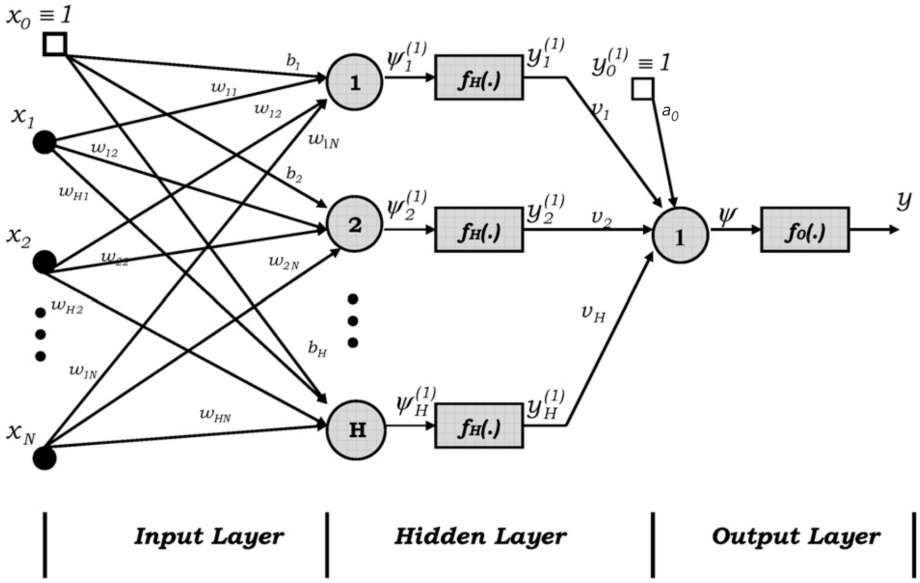
This section introduces the non-parametric models, which consist of several implementations of neural network models. Neural network models are inspired by the characteristics of the biological nervous system that enable experiential learning. Neural networks models are attractive methods for option pricing problems because they are able to effectively approximate non-linear relationships and are not reliant on the restrictive assumptions implicit in parametric approaches. Furthermore, parametric option pricing models in appropriate combination with neural networks are introduced as hybrid neural networks.

2.1.1 Artificial Neural Network

A neural network is a collection of interconnected neurons structured in layers. The neurons can be represented by nodes. The neural network can then be depicted by nodes that are connected by arcs. Hutchinson et al. (1994), who were among the earliest papers to apply neural networks to option pricing, implemented two types of regression networks: the multilayer perceptron and the radial basis function network. Both networks belong to a class of neural networks called feed-forward networks, as the connections between the nodes do not form a cycle. Even though both networks are appropriate to price options, to facilitate the introduction of deep learning and to remain consistent with methods introduced later in this paper I opt to exclusively implement the multilayer perceptron approach, similar to for example Anders et al. (1998). Hereinafter I simply refer to this network as the artificial neural network (ANN).

The first ANN used in this study is depicted in Figure 1. Each neuron is composed of a vector of input signals. For the input layer these are simply the N input variables. The H neurons in the hidden layer also have vector weights and an associated bias term as input variables, represented by w_{ji} and b_j. After computing a weighted sum for each hidden unit, a non-linear activation function, $f_H(\cdot)$, is applied to the result. This non-linearity ensures that the model is more powerful than a linear model. The output signals of the hidden layer are then transferred to the output layer as input signals. Here, the single neuron again sums the product of the input signals with the corresponding bias and weights, now represented by v_j and a_0. Finally, a second activation function, $f_O(\cdot)$, is applied to the result. If this second activation function is disregarded, the output is simply the weighted sum of the input signals from the hidden layer. For notational convenience, the activation functions of the hidden and output layer are hereinafter symbolized by respectively $\sigma(\cdot)$ and $\sigma_0(\cdot)$.

Figure 1: An artificial neural network with a single hidden layer.



Note: In this figure, x_i represent the input variables, b_j and w_{ji} represent weights and biases in the hidden layer, v_j and a_0 represent weights and biases in the output layer, $\psi_j^{(1)}$ and $y_j^{(1)}$ represent intermediate results and y is the output variable. Furthermore, $f_O(\cdot)$ denotes the activation function of the output layer and $f_H(\cdot)$ denotes the activation function of the hidden layer. Source: Andreou et al. (2008)

The output of the ANN with one hidden layer as depicted in Figure 1 is then generated by

$$y = \sigma_0 \left(a_0 + \sum_{j=1}^{H} v_j \sigma \left(b_j + \sum_{i=1}^{N} w_{ji} x_i \right) \right). \tag{1}$$

The Adam optimizer of Kingma and Ba (2015) is utilized to train each neural network. The Adam optimizer uses gradient descent to iteratively update and ultimately optimize network weights and biases. I choose the Adam optimizer over other popular gradient descent optimization algorithms such as the adaptive gradient algorithm (AdaGrad) and root mean square propagation (RMSProp). The Adam optimizer adds bias-correction and momentum to RMSprop, which is an extension of AdaGrad. Kingma and Ba (2015) showed that the Adam optimizer outperforms AdaGrad and RMSProp in terms of speed and accuracy.

The weights and the biases of the network with one hidden layer are updated such that the mean squared error loss function is minimized, i.e. the problem

$$\underset{\boldsymbol{\omega}}{\operatorname{argmin}} \frac{1}{P} \sum_{q=1}^{P} (y_q - t_q)^2 = \underset{\boldsymbol{\omega}}{\operatorname{argmin}} \frac{1}{P} \sum_{q=1}^{P} \left(\sigma_0 \left(a_0 + \sum_{j=1}^{H} v_j \sigma \left(b_j + \sum_{i=1}^{N} w_{ji} x_{iq} \right) \right) - t_q \right)^2 \tag{2}$$

is solved. Here, P is the number of data points in the training sample, $\boldsymbol{\omega} = \{a_0, v, b, W\}$ is the parameter set containing the weights and biases and t_q is the target. The target is defined as the market price of the call option ($c_{M,q}$) divided by the strike price, i.e. $t_q \equiv \frac{c_{M,q}}{K}$. The output that follows from the ANN thus equals $\hat{y}_{\text{ANN},q} = \frac{c_{\text{ANN},q}}{K}$, which must be multiplied by K to arrive at the estimated option price. The target t_q is defined in this way because the moneyness bias described by Garcia and Gençay (2000) is taken into account. Originally, the option pricing formula is assumed to depend on the strike price K, the spot price S_t and the time-to-maturity τ, see e.g. Hutchinson et al. (1994) and Malliaris and Salchenberger (1993). However, Garcia and Gençay (2000) showed that if the return distribution is independent of the stock price, the pricing model is rescalable with respect to K. In other words, homogeneity of degree one of the pricing function with respect to S_t and K is assumed. Garcia and Gençay (2000) therefore called this the homogeneity hint. Incorporating this homogeneity hint results in an ANN consisting of two input variables rather than three. This ANN is solely controlled by the moneyness ratio, defined as $m \equiv \frac{S_t}{K}$, and time-to-maturity, τ. I follow this approach and therefore input two variables into the ANN. Accordingly, N in (1) equals two. To remain consistent across all methods, time-to-maturity is annually normalized following Gradojevic et al. (2009) and Yang et al. (2017). The output, in terms of the option pricing formula $C(\cdot)$, equals

$$y \equiv C(m, 1, \tau) = \frac{C(S_t, K, \tau)}{K}. \tag{3}$$

The main focus of this paper is not on neural networks with the previously introduced standard target function, but on hybrid neural networks with hybrid target functions. The standard neural networks are mainly used as benchmarks to which the performance of the hybrid neural networks is compared. Using ANNs, the hybrid approach has been shown to produce more accurate option pricing than either parametric option pricing models or standard neural network option pricing models, e.g. by Lajbcygier and Conner (1997), Andreou et al. (2008) and Chen and Sutcliffe (2012). First introduced by Boek et al. (1995), the hybrid approach uses a parametric model in combination with a neural network. The basic idea is to use a parametric model as a base and allowing the neural network to augment its performance. More specifically, the target function of hybrid neural networks is the residual between the observed market price and the price estimate of a certain parametric model, i.e. $t_{H,q} \equiv \frac{c_{M,q}}{K} - \frac{c_{j,q}}{K}$, where $c_{j,q}$ is defined as the option price output from a parametric option pricing model and homogeneity is assumed. The output that follows from the hybrid artificial neural network (HANN) equals $\hat{y}_{\text{HANN},q} = \frac{c_{\text{HANN},q}}{K} - \frac{c_{j,q}}{K}$. In total, five parametric option pricing models are employed. The Black and Scholes (1973) (BS) model and the Corrado and Su (1996) (CS) model are chosen following the work of Andreou et al. (2008).

This paper contributes to existing literature by investigating whether using stochastic volatility models, represented by the Heston (1993) (H) model, or exponential Lévy models, represented by the Kou (2002) (K) jump-diffusion model and the Variance Gamma (VG) model of Madan et al. (1998), in the hybrid target function improves the accuracy of the option pricing models. As the Black-Scholes model follows from a set of simplified assumptions, the Heston model is introduced to account for fluctuating volatility and the Kou jump-diffusion model and Variance Gamma model are introduced to account for stock price jumps. The fundamentals of each of these parametric models are described in detail in Appendix A.

Most papers that research hybrid neural networks for option pricing, such as Andreou et al. (2008), only consider hybrid neural networks with one hidden layer. According to the universal approximation theorem of Cybenko (1989) a single hidden layer can be trained to estimate most functions and can attain high accuracy by including enough processing nodes. However, I consider more than one hidden layer to optimize option pricing accuracy. Heaton (2008) stated that "two hidden layers can represent an arbitrary decision boundary to arbitrary accuracy with rational activation functions and can approximate any smooth mapping to any accuracy" and that "additional layers can learn complex representations." Therefore, I consider up to three hidden layers for each non-parametric model and examine which number of hidden layers is optimal for the non-parametric models. In Appendix B, the derivation of the output of the networks with two and three hidden layers is given. The output of the network with two hidden layers is given in (42) and the output of the network with three hidden layers in (43). Networks with multiple hidden layers are created by iteratively inserting an additional hidden layer between the input layer and the adjacent hidden layer, paying careful attention to the dimensionality of the weight and bias terms. The new hidden layer is therefore always inserted to the left of the previous hidden layer(s) in Figure 1. For computational purposes, the same activation function, $\sigma(\cdot)$, is chosen for each hidden layer.

Andreou et al. (2008), Gradojevic et al. (2009) and Gradojevic (2016) all decided only to consider the hyperbolic tangent activation function and the sigmoid activation function for the hidden layer of their neural network models. To extend their research I also examine the softplus, Rectified Linear Unit (ReLU) and Exponential Linear Unit (ELU) activation functions for the hidden layer. For the output layer I also consider the exponential activation function, following Culkin and Das (2017). I examine which activation function architecture works best for each method. All activation functions are defined in Appendix C. Additional activation functions are considered because analyses have shown that the hyperbolic tangent activation function and the sigmoid activation function are flawed by design. In short, gradients in limiting regions are forced to approach zero. A more elaborate discussion on this topic is presented in Appendix C.

Ruf and Wang (2019) summarized more than 150 papers that use ANNs as a non-parametric option pricing or hedging tool. A fair amount of these papers considers more than two input variables. The most commonly used input variables besides moneyness and time-to-maturity are historical volatility and the risk-free interest rate, as for example implemented by Culkin and Das (2017). Adding these two variables increases the number of input variables to four, assuming homogeneity of degree one of the pricing function. Recently, Palmer (2019) introduced a parameter space reduction technique that reduces the number of input variables from four to three. A reduction of the number of input variables is important as it simplifies inference for the neural network. Palmer (2019) found that in this case the option pricing formula can be written as

$$y \equiv C\left(m, 1, 1, \sigma\sqrt{\tau}, r\tau\right) = C\left(S, K, \tau, \sigma, r\right) K. \tag{4}$$

The target function for neural networks with these inputs is then defined as $t_q \equiv K c_{M,q}$. I implement this approach with three input variables to examine whether the increase of input information improves the performance of the ANN and the neural network methods introduced in upcoming sections.

Because of the large range of magnitudes of option prices the target function must be transformed before it can be utilized by the neural network. According to Palmer (2019), in general neural networks work best when all the targets are a similar magnitude in value. Methods later introduced in this paper overcome this problem by dividing the options with modules or gates and pricing the options with separate specialised neural networks for specific options. To resolve this issue for the ANN, a data transform is introduced and applied to the target output: the target function is standardized using the z-score. The neural network then learns to output the transformed value of the target values. The approximated target value can then be retrieved by inverting the applied transform. The data transform is applied to the input variables as well. This way possible neuron saturation is avoided.

2.1.2 Modular Neural Network

A main drawback of the ANN introduced in the previous section is that it tries to find a single pricing function for the entire range of traded options. Lajbcygier (2004) noted that in his earlier work (Lajbcygier and Conner, 1997) there existed pricing biases in boundary regions. The price of options that were approaching maturity were severely underestimated; a phenomenon observed by Dugas et al. (2001) as well. Gençay and Salih (2003) and Bennell and Sutcliffe (2004) found that estimating option prices separately per moneyness region increased the pricing accuracy of their neural network methods. Inspired

by these findings, Gradojevic et al. (2009) implemented an ANN whereby they categorized the options in modules based on their moneyness and time-to-maturity and trained the ANN for each individual group of options. They referred to this model as a modular neural network (MNN). The options are divided into sub-categories, i.e. modules, and are thereupon distinctly priced per module. The MNN introduces modularity to improve the generalization performance of neural networks.

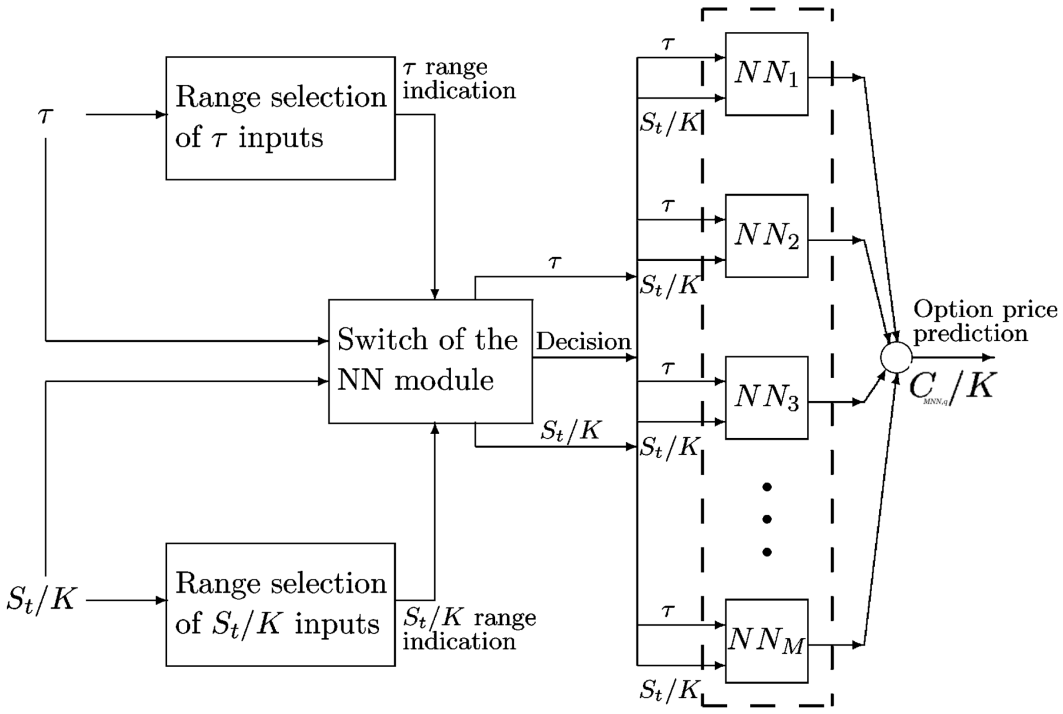
The options are organized conditional on the moneyness ratio m and the annually normalized time-to-maturity τ of the options. Following Gradojevic et al. (2009), the options are divided into three time-to-maturity modules and cut-off points are the following: $\tau < 0.1$ (short term), $0.1 \leq \tau \leq 0.2$ (medium term), and $\tau > 0.2$ (long term). Similarly, for the three moneyness modules the cut-off points are determined as follows: $m < 0.97$ (out-of-the-money), $0.97 \leq m \leq 1.05$ (at-the-money) and $m > 1.05$ (in-the-money). As both criteria are applied concurrently, the option prices are estimated by an MNN with nine modules, thus $M = 9$ in Figure 2. However, at a given time only one module is active given the moneyness and time-to-maturity of the option that is being priced. Seperate modules are thus not interconnected

Each module is defined by a single feed-forward ANN model with two input variables, see Figure 2. Each of these ANNs is trained independently. Similar to the ANN of the previous section, homogeneity of degree one of the pricing function with respect to S_t and K is assumed for each module, such that (3) holds. As the modules are not interconnected, the output of the MNN can be defined as the combination of outputs of nine separate ANNs. Specifically, the output of the MNN equals

$$y_k = \sigma_0 \left(a_{0,k} + \sum_{j=1}^{H_k} v_{jk} \sigma \left(b_{jk} + \sum_{i=1}^{N} w_{jik} x_i \right) \right), \tag{5}$$

where H_k is the number of neurons in the hidden layer for each module $k = 1, \ldots, M$. For computational purposes, H_k is set constant across modules. Furthermore, b_{jk} and w_{jik} represent weights and biases in the hidden layer for each module and v_{jk} and $a_{0,k}$ represent weights and biases in the output layer for each module. With input variables m and τ, $N = 2$. To extend this MNN, I research the case in which each of the M ANNs consists of up to three hidden layers. The derivation of the output of the neural networks with two and three hidden layers is given in Appendix D. The output of the neural network with two hidden layers is given in (44) and the output of the neural network with three hidden layers in (45). The same additional activation functions as for the ANN are regarded for the hidden layer(s).

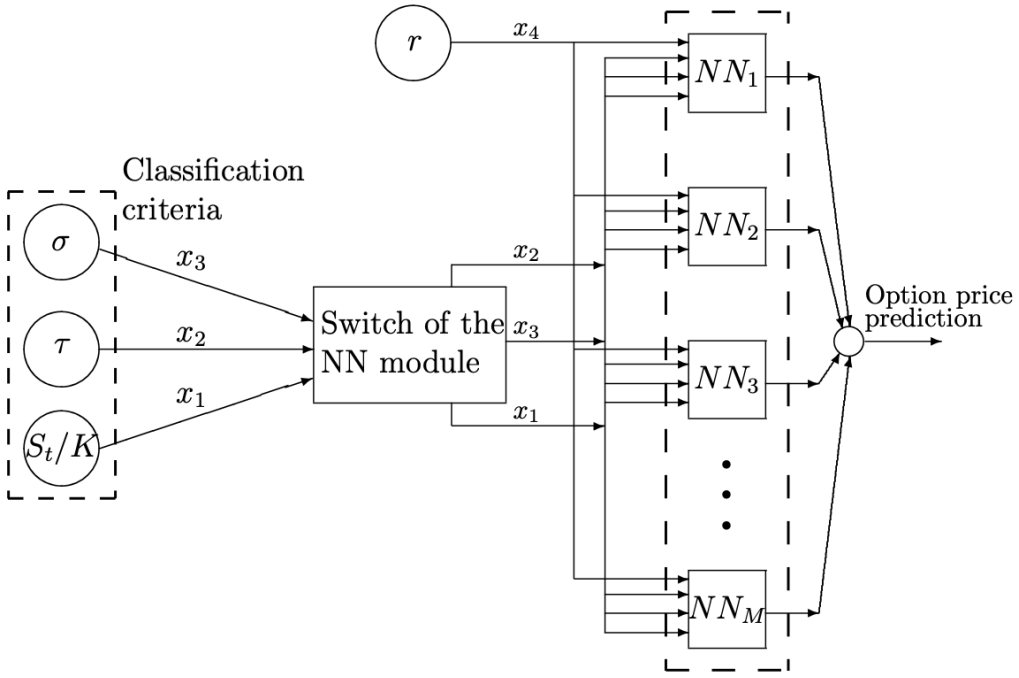
Figure 2: A modular neural network



Note: In this figure, τ and S_t/K represent the input variables of the MNN. Depending on the value of the input variable per option, the switch of the NN module decides which ANN, denoted by $NN_1, \ldots, NN_M$, prices the option. The MNN outputs $c_{\text{MNN},q}/K$. The value of M equals nine. Source: Gradojevic et al. (2009).

To extend the MNN with two input variables, I use the approach of Palmer (2019) to introducing extra input variables to the MNN. This results in an MNN with three input variables. Furthermore, to improve this model, Gradojevic (2016) introduced the multi-criteria modular neural network (MCMNN). The MCMNN extends the standard MNN by adding a switch based on the implied volatility value of the option, see Figure 3. To remain consistent across models and to simplify inference for the neural network, I use the three Palmer (2019) input variables m, $\sigma\sqrt{\tau}$ and $r\tau$ in combination with the MCMNN modules. Historical instead of implied volatility is thus used as input. The time-to-maturity and moneyness ratio cut-off points are similar to those for the MNN. The volatility criteria classifies options across two ranges consisting of low volatility options and high volatility options. The cut-off point is dynamically determined by the median value of volatility across options in the training data. In total, the MCMNN thus consists of 18 modules. The MCMNN is again examined with up to three hidden layers, several combinations of hidden layer nodes and with a range of different activation functions. The output of the MCMNN is equal to that of the MNN. The only difference in (5) is that for the MCMNN it holds that $N = 3$ and $M = 18$.

Figure 3: A multi-criteria modular neural network



Note: In this figure, τ, S_t/K, σ and r represent the input variables of the MCMNN. Depending on the value of the input variables τ, S_t/K and σ per option, the switch of the NN module decides which ANN, denoted by $NN_1, \ldots, NN_M$, prices the option. The MCMNN outputs $c_{\text{MCMNN},q}/K$. The value of M equals 18. Source: Gradojevic (2016).

This paper contributes to existing literature by applying the hybrid approach to the introduced MNN and MCMNN. In their research, Gradojevic et al. (2009) and Gradojevic (2016) only estimated the option-price-to-strike-price ratio, such that similar to the previously introduced standard ANN approach the target was defined as $t_q \equiv \frac{c_{M,q}}{K}$. To extend this research, I also implement the hybrid approach for the MNN and MCMNN, resulting in the hybrid modular neural network (HMNN) and the hybrid multi-criteria modular neural network (HMCMNN). Apart from a change of the target function, nothing changes in terms of input variables or module organization. In Figure 2 and Figure 3 this means that only the output must be modified. The output that follows from the HMNN depends on the specification of the input variables. If two input variables are used, the output equals $\hat{y}_{\text{HMNN},q} = \frac{c_{\text{HMNN},q}}{K} - \frac{c_{j,q}}{K}$. Again, $j \in \{\text{BS}, \text{CS}, \text{H}, \text{K}, \text{VG}\}$. If the three input variables of Palmer (2019) are used, the output equals $\hat{y}_{\text{HMNN},q} = Kc_{\text{HMNN},q} - Kc_{j,q}$. For the HMCMNN, the output equals $\hat{y}_{\text{HMCMNN},q} = Kc_{\text{HMCMNN},q} - Kc_{j,q}$. The HMNN and HMCMNN output must thus be linearly transformed to obtain the estimated option price.

2.1.3 Gated Neural Network

In a recent paper, Yang et al. (2017) stated that the categorization of options performed by Gradojevic et al. (2009) may not be consistent with changing market data over time, as it is based on fixed manual heuristics. To substantiate their claims they introduced a new class of neural networks for option pricing: a gated neural network (GNN). Gated networks contain gating connections in which the outputs of the neurons are multiplied. The GNN basically also applies a modular approach, but with option grouping that is automatic and learned from data rather than performed manually. The GNN can thus dynamically adjust the grouping and the pricing function of each group as the market changes over time. Yang et al. (2017) also incorporated economic and financial axioms implemented as constraints into their neural network, expanding the work of Dugas et al. (2001). Zheng (2018) referred to this approach as a Bayesian-alike design approach in which prior information is encoded into the neural network. The prior neural network is then trained to get the posterior neural network. The GNN thus improves the option grouping, i.e. module organization, of the MNN and contributes to the existing literature a neural network that comes with guarantees about the economic and financial rationality of its outputs.

Similar to the MNN, the GNN implements a modular approach. For the MNN, the ANN is the building block of which each module consists. Yang et al. (2017) first defined a GNN with two inputs but no modules as the building block for the GNN with modules. The GNN model without modules is referred to the single model and the GNN model with modules as the multi model. The single model is to the multi model what the ANN is to the MNN. In the results section I refer to this multi model as the GNN. The single model is portrayed in Figure 4a and the multi model in Figure 4b.

The first step to deriving the single model is defining the theoretical definition of the call option pricing model $C(\cdot)$ of (3), which is given by

$$C(K, S_t, \tau) = \int_0^\infty \max(0, S_T - K) f(S_T|S_t, \tau) \mathrm{d}S_T \tag{6}$$

where $\max(0, S_T - K)$ is the potential revenue that can be obtained from this call option at maturity and $f(S_T|S_t, \tau)$ is the probability density of that revenue. The integral therefore represents the expected revenue at maturity given the current information. To arrive at this, no arbitrage is assumed and no discount term is considered because the risk-free rate is obtained independently. $C(\cdot)$ can be easily obtained from a regression problem. However, this does not necessarily yield a relevant predictive model unless $f(\cdot)$ is a valid probability density function.

Following Föllmer and Schied (2011), Yang et al. (2017) listed six conditions that must hold to ensure a valid probability density function for an option pricing model. To assess the consequences some conditions have for the neural network specification, the standard ANN of (1) is adjusted accordingly. The first condition reads

$$\frac{\partial C}{\partial K} \leq 0. \tag{C1}$$

This ensures that the option price increases when the strike price decreases, ceteris paribus, which is a condition that must hold on a fair market. Mathematically, it must hold because $\frac{\partial C}{\partial K} = \int_0^K f(S_T|S_t,\tau)\mathrm{d}S_T - 1$, and the integral represents a cumulative distribution function $\mathbb{P}(S_T \leq K)$, thus its value cannot exceed one. When the strike price decreases, ceteris paribus, moneyness $m = \frac{S_t}{K}$ increases and inverse moneyness m^{-1} decreases. Yang et al. (2017) input inverse moneyness such that C4 can hold. I therefore also follow this approach. To guarantee that the output of the neural network is monotonically decreasing with its single input variable x, according to Zheng (2018), one can use

$$y = \sigma_0 \left(a_0 + \sum_{j=1}^{H} \mathrm{e}^{v_j} \sigma_1 \left(b_j - \mathrm{e}^{w_j} x \right) \right), \tag{7}$$

where $\sigma_1(x) = \frac{1}{1+\mathrm{e}^{-x}}$ is the sigmoid function, which has the useful property that its derivative is strictly positive. The second condition is given by

$$\frac{\partial^2 C}{\partial K^2} \geq 0 \tag{C2}$$

and ensures convexity of the option price related to the strike price. This condition follows from the fact that $\frac{\partial^2 C}{\partial K^2} = f(S_T|S_t,\tau)$ is a probability density function so its value cannot be smaller than zero. Ensuring C2 holds can be done designing a neural network as

$$y = \sigma_0 \left(a_0 + \sum_{j=1}^{H} \mathrm{e}^{v_j} \sigma_2 \left(b_j + \mathrm{e}^{w_j} x \right) \right), \tag{8}$$

where $\sigma_2(x) = \log(1+\mathrm{e}^x)$ is the softplus function. C2 holds because the derivative of the softplus function with respect to K equals the sigmoid function. The third condition follows from intuition: the option price must be non-decreasing with time-to-maturity, because a longer time-to-maturity corresponds to a larger probability that the underlying asset price becomes greater than the strike price. Mathematically, this gives

$$\frac{\partial C}{\partial \tau} \geq 0. \tag{C3}$$

The consequences of this condition are similar to those of C1, but now concern that the output of the neural network must be monotonically increasing with its single input variable x. This can be attained by specyfying

$$y = \sigma_0 \left(a_0 + \sum_{j=1}^{H} \mathrm{e}^{v_j} \sigma_1 \left(b_j + \mathrm{e}^{w_j} x \right) \right), \tag{9}$$

where $\sigma_1(\cdot)$ again corresponds to the sigmoid function.

When the strike price of an option approaches infinity, there is a no chance of making a profit from this option because the underlying asset price invariably fails to exceed the strike price. The fourth condition states that the price of this option must be zero, i.e.

$$\lim_{K \to \infty} C(K, S_t, \tau) = C(\infty, S_t, \tau) = 0. \tag{C4}$$

This condition can only hold when the input variable x is defined as the inverse moneyness m^{-1}. This way, when K approaches infinity, inverse moneyness also approaches infinity and $\sigma_2 \left(b_j - \mathrm{e}^{w_j} x \right)$ in (7) approaches 0, such that the option price finally approaches 0. Apart from defining inverse moneyness as an input variable, the bias term a_0 for the output layer must be set equal to 0 for this condition to hold. The neural network must therefore be designed as

$$y = \sigma_0 \left(\sum_{j=1}^{H} \mathrm{e}^{v_j} \sigma_2 \left(b_j - \mathrm{e}^{w_j} x \right) \right), \tag{10}$$

where $\sigma_2(\cdot)$ again corresponds to the softplus function.

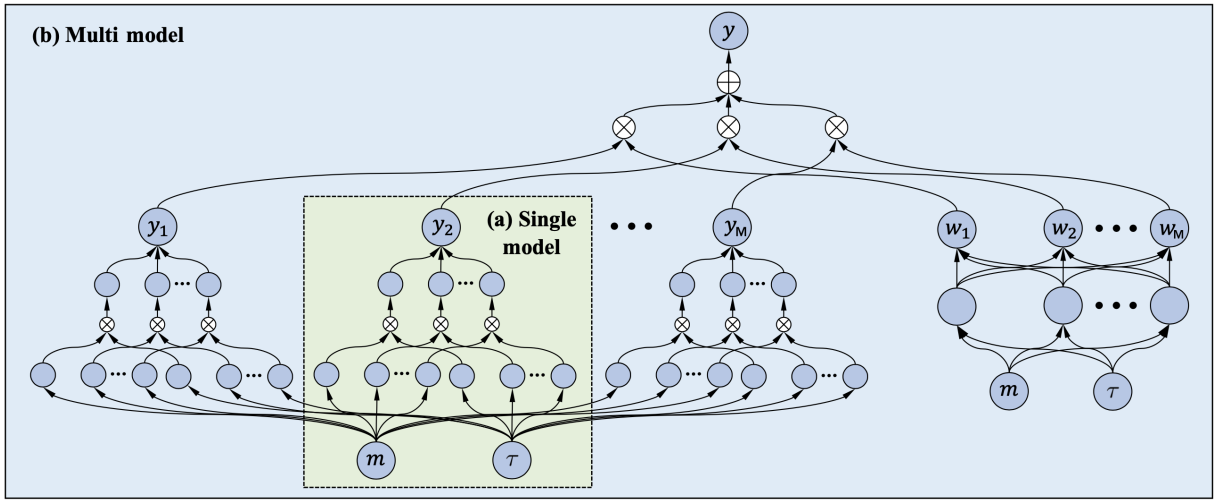
At maturity, when $\tau = 0$, the option price must equal its theoretical value, $\max(0, S_t - K)$, because it is possible for market practitioners to execute the option at once. The fifth condition is therefore defined as

$$C(K, S_t, 0) = \max(0, S_t - K) \quad \text{when} \quad \tau = 0 \tag{C5}$$

The sixth condition provides boundaries for the option price. The option price cannot be smaller than $\max(0, S_t - K)$, because of the time value of the option contract. Furthermore, the option price cannot exceed the underlying stock price, because otherwise exploitable arbitrage opportunities arise. Specifically,

$$\max(0, S_t - K) \leq C(K, S_t, \tau) \leq S_t. \tag{C6}$$

Figure 4: (a) The single model (b) The multi model



Note: This figure shows in (a) the single model with input variables m and τ and in (b) the multi model consisting of M modules of single models and a seperate weighting model to softly switch the single models. Furthermore, $\otimes$ is the multiplication gate that outputs the product of the inputs and $\oplus$ is the addition gate that outputs the sum of the inputs. Source: Zheng et al. (2019)

These last two conditions are hard to achieve by model architecture design and are therefore met by synthesising virtual option contracts in training. These virtual options are appended to the training data of the GNN. How these virtual option contracts are generated is discussed in Section 3.

To arrive at the single model, the consequences in (7), (8), (9) and (10) are regarded. Each of the conditions discusses the case of one input variable, but all the neural networks specified before consisted of more than one input variable. To combine all conditions, the single model is designed as a GNN with two sides as defined by Sigaud et al. (2015). For two input variables, m^{-1} and τ, the single model is specified by the formula

$$y = \sum_{j=1}^{H} \mathrm{e}^{v_j} \sigma_1 \left(b_{1,j} + \mathrm{e}^{w_{j1}} \tau \right) \sigma_2 \left(b_{2,j} - \mathrm{e}^{w_{j2}} m^{-1} \right), \tag{11}$$

where $\sigma_1(x)$ is the sigmoid function, $\sigma_2(x)$ is the softplus function and H is the number of neurons. The multiplication gate, represented by $\otimes$ in Figure 4a, merges the neurons from the different input gates. The activation function of the output layer is linear, such that the output is simply the weighted sum of the input signals from the hidden layer. The output layer produces the output y using weights v_j. The single model provides a simple option pricing model for all options.

The multi model simultaneously trains M single pricing models and a weighting model to switch between the single models, see Figure 4b. Following Yang et al., $M = 9$, as the MNN also has this setting. Each single model can be expressed as

$$y_k = \sum_{j=1}^{H_k} \mathrm{e}^{v_{jk}} \sigma_1 \left(b_{1,jk} + \mathrm{e}^{w_{j1k}} \tau \right) \sigma_2 \left(b_{2,jk} - \mathrm{e}^{w_{j2k}} m^{-1} \right), \tag{12}$$

for $k = 1, \ldots, M$. This gives the output of the left-hand side of the multi model of Figure 4b.

The right-hand side of the multi model of Figure 4b, the side that provides a model selector for the left branch, is a network consisting of one R unit hidden layer and an output layer with an M-way softmax activation function. The softmax activation functions takes as input a vector and enforces the outputs to sum to one. The weights ω_k thus sum to one. The output weights of the right-hand side that follow from the softmax activation function are given by

$$\omega_k = \frac{\mathrm{e}^{\sum_{r=1}^{R} \sigma_1 \left(\psi_{1,r} m^{-1} + \psi_{2,r} \tau + \beta_r \right) v_{r,k} + \gamma_k}}{\sum_{k=1}^{M} \mathrm{e}^{\sum_{r=1}^{R} \sigma_1 (\psi_{1,r} m^{-1} + \psi_{2,r} \tau + \beta_r) v_{r,k} + \gamma_k}}, \tag{13}$$

where ψ (v) and β (γ) denote the weight and bias term for the hidden (output) layer. Finally, the output of the multi model equals the softmax weighted average of the M single models' outputs, i.e.

$$y = \sum_{k=1}^{M} y_k \omega_k. \tag{14}$$

I extend the work of Yang et al. (2017) by introducing the hybrid approach to the GNN. The output of the parametric models are used to construct the hybrid GNN (HGNN) from the standard GNN. For the standard GNN inverse moneyness is examined as input variable, such that the target function of the GNN with two input parameters is defined as $t_q \equiv \frac{c_{M,q}}{S_t}$. This can be deduced from (3). Furthermore, $t_q \equiv S_t c_{M,q}$ when three input parameters are used. Consequently, the output of the GNN with two input variables is given by $\hat{y}_{\mathrm{GNN},q} = \frac{c_{\mathrm{GNN},q}}{S_t}$ and the output of the GNN with three input variables equals $\hat{y}_{\mathrm{GNN},q} = S_t c_{\mathrm{GNN},q}$. Because of this different input variable specification, the target function of the HGNN is also defined contrary to the target function of the HANN and HMNN. The HGNN target functions for two and three input parameters are respectively given by $t_{H,q} \equiv \frac{c_{M,q}}{S_t} - \frac{c_{j,q}}{S_t}$ and $t_{H,q} \equiv S_t c_{M,q} - S_t c_{j,q}$. The outputs of the HGNNs are then respectively given by $\hat{y}_{\mathrm{HGNN},q} = \frac{c_{\mathrm{HGNN},q}}{S_t} - \frac{c_{j,q}}{S_t}$ and $\hat{y}_{\mathrm{HGNN},q} = S_t c_{\mathrm{HGNN},q} - S_t c_{j,q}$. All outputs must thus again be linearly transformed to obtain the estimated option price.

Finally, since the activation functions are chosen such that conditions C1 – C4 hold, I do not consider other activation functions for the GNN. I do expand the GNN from two input variables to three input variables, to remain consistent with the other neural network models. As it is trivially known that $\frac{\partial C}{\partial r} \geq 0$ and $\frac{\partial C}{\partial \sigma} \geq 0$, and it has already been shown that $\frac{\partial C}{\partial \tau} \geq 0$, gates for the Palmer (2019) input variables $\sigma\sqrt{\tau}$ and $r\tau$ can be added to (11) similar to that of τ. Following Yang et al. (2017), inverse moneyness is again used as input variable. Each single model with three input variables can then be expressed as

$$y_k = \sum_{j=1}^{H_k} \mathrm{e}^{v_{jk}} \sigma_1 \left(b_{1,jk} + \mathrm{e}^{w_{j1k}} \sigma\sqrt{\tau}\right) \sigma_1 \left(b_{2,jk} + \mathrm{e}^{w_{j2k}} r\tau\right) \sigma_2 \left(b_{3,jk} - \mathrm{e}^{w_{j3k}} m^{-1}\right). \tag{15}$$

Because of the distinctive multiplicative relationships between the inputs, I disregard multiple hidden layers for the GNN. It is not possible to deduce the historical volatility for the virtual options (see Section 3). Therefore, the virtual options are only generated for the GNN with two input variables. The GNN with virtual options is henceforth denoted as GNN*.

2.2 Model Estimation and Evaluation

For the Black-Scholes model and the Corrado-Su model, the option price can be obtained directly from the option pricing formula of the models using historical estimates for the volatility, skewness and kurtosis parameters. For the parametric models of which the characteristic function is known, options are priced using fractional FFT, as described in Section A.3. For these models, parameters are calibrated on the last training day's data, following Yang et al. (2017). The parameters are calibrated using the Levenberg-Marquardt algorithm. The neural networks are created using Google's TensorFlow package. Furthermore, the neural networks are all trained utilizing the Adam optimizer of Kingma and Ba (2015). The weights and the biases of the network are updated such that the mean squared error loss function is minimized, as described in Section 2.1.1.

For the neural networks the data for each year are divided into four quarters consisting of three months of data, split chronologically. Each set of 3 months of trading days can be used for training, validation or out-of-sample testing using overlapping rolling windows. The training set consists of two blocks of three months of data, the validation set and out-of-sample testing set each consist of one block of three months. In the training set, the neural network is trained to obtain first estimates of weight and bias parameters. In the validation set the optimal number of hidden neurons and the corresponding weights are decided. Overfitting is prevented by ending the training process as soon as the validation set error begins increasing. Finally, in the out-of-sample testing set the pricing capability of the neural network is tested.

This chapter has introduced nine models: five parametric models and four non-parametric models. The five parametric models are used as a benchmark to which the performance of the neural network models is compared. They are also used as inputs for the target function of the hybrid neural network models. Each neural network model thus has six variants: a standard approach and five hybrid approaches. Furthermore, the neural network models are estimated with both two and three input variables. The MCMNN is only examined with three input variables. Also, the GNN with two input variables is examined with and without virtual options. Consequently, eight different neural networks are researched and in total $8 + 5 \times 8 = 48$ non-parametric models are estimated. The optimal all-encompassing model architecture for all neural network models—the number of input variables, the number of hidden layers, the number of nodes in each hidden layer and the activation functions for each layer—is determined by the performance on the full data set. All neural network models are set up with the same model architecture such that it is possible to adequately analyze the contrast between the different approaches.

The statistical performance in this paper is measured by the mean absolute percentage error (MAPE), defined as $\text{MAPE} = \frac{1}{P}\sum_{i=1}^{P}\frac{|y_i - \hat{y}_i|}{y_i}$, and the mean squared error (MSE), defined as $\text{MSE} = \frac{1}{P}\sum_{i=1}^{P}(y_i - \hat{y}_i)$. Here, P is the total number of option price forecasts, y_i is the observed option price and $\hat{y}_i$ is the estimated option price. Two statistical performance metrics are employed to ensure that the models are capable of producing accurate option price forecasts while not making any excessive errors. The accuracy is mostly measured by the MAPE whereas the excessive errors are better detected by the MSE.

Finally, similar to Gradojevic (2016) the training of the neural networks is performed from different random seeds to control for possible sensitivity of the neural networks to the initial parameter values. The average MSE and MAPE values are therefore reported throughout this paper. For computational purposes the number of random seeds is set equal to three.

2.2.1 Diebold-Mariano Test

The statistical significance of the difference in the out-of-sample forecast accuracy is tested using the Diebold-Mariano test of Diebold and Mariano (1995). The Diebold-Mariano test states that two forecasts i are equally accurate if and only if the loss differential $d_q = g(e_{1q}) - g(e_{2q})$ of the forecasts error $e_{iq} = \hat{y}_{iq} - y_q$ for $i = 1, 2$ has zero expectation for all $q = 1, \ldots, P$. Here, P is the total number of option price forecasts. The size of the loss differential depends on the chosen statistical performance metric $g(\cdot)$. Again, both the MAPE and MSE performance metrics are scrutinized. The null hypothesis is that the two forecasts are

equally accurate, i.e. $H_0 : E[d_q] = 0 \; \forall \; q$. The alternative hypothesis is that the two forecasts are not equally accurate, i.e. $H_1 : E[d_q] \neq 0$. Diebold and Mariano (1995) showed that under H_0 it must hold that

$$\text{DM} = \frac{\bar{d}}{\sqrt{\frac{2\pi f_d(0)}{P}}} \rightarrow N(0,1), \tag{16}$$

where $\bar{d} = \sum_{q=1}^{P} d_q$ is the sample mean of the loss differential, $f_d(0) = \frac{1}{2\pi}\left(\sum_{k=-\infty}^{\infty} \gamma_d(k)\right)$ is the spectral density of the loss differential at frequency 0 and $\gamma_d(k)$ is the autocovariance of the loss differential at lag k. Simulation experiments in Diebold and Mariano (1995) showed that the normal distribution is a poor approximation of the finite-sample null distribution. Therefore, Harvey et al. (1997) made a bias correction to the DM test-statistic which improves the small-sample properties of the test. The corrected statistic is obtained as

$$\text{DM}_2 = \sqrt{\frac{P + 1 - 2h + h(h-1)}{P}}\text{DM}, \tag{17}$$

where h is the step-ahead forecast, which is set equal to 1. This test-statistic is compared with a t-distribution with $P - 1$ degrees of freedom. In this paper, the DM test-statistic with bias correction is implemented and the null hypotheses that there is no difference in the MAPE and MSE of the two alternative models are tested.

2.2.2 Model Confidence Set

The statistical significance of the difference in the out-of-sample forecast accuracy is also determined by constructing the model confidence set (MCS) of Hansen et al. (2011). The MCS procedure yields a set of models that contains the best model(s) with a given level of confidence. Because the MCS is not pairwise, as opposed to the Diebold and Mariano (1995) test, MCS can be efficiently used to test the statistical significance of the difference in the out-of-sample forecast accuracy per moneyness and maturity region. The MCS therefore assists in determining which option pricing model simultaneously has the most accurate and consistent overall option pricing performance.

The MCS procedure excludes models unable to pass the null hypothesis of equal predictive ability (EPA) at confidence level α from the MCS. The EPA hypothesis of Hansen et al. (2011) is formulated as $H_0 : E[d_{ijq}] = 0 \; \forall \; i, j = 1, 2, \ldots, m$. The alternative EPA hypothesis is formulated as $H_1 : E[d_{ijq}] \neq 0 \; \forall \; i, j = 1, 2, \ldots, m$. Here, m is the total number of option pricing models and $d_{ijq} = g(e_{iq}) - g(e_{jq})$. Similar to the Diebold-Mariano test, the size of the loss differential depends on the chosen statistical performance

metric $g(\cdot)$, as both the MAPE and MSE performance metrics are scrutinized. In order to test the null hypothesis, the test-statistic

$$t_{ij} = \frac{\bar{d}_{ij}}{\sqrt{\hat{var}\left(\bar{d}_{ij}\right)}} \tag{18}$$

is examined. Here, $\hat{var}\left(\bar{d}_{ij}\right)$ is the bootstrapped variance of the mean loss difference $\bar{d}_{ij}$. The variance is bootstrapped using a block-bootstrap procedure of 1,000 samples in which blocks are of length 2, following Hansen et al. (2011). The null hypothesis can then be tested by the test-statistic $T_M = \max_{i,j \in M} |t_{ij}|$, for which the natural elimination rule defined by

$$e_{\max,M} = \text{argmax}_{i \in M} \sup_{j \in M} t_{ij} \tag{19}$$

is followed. The MCS procedure iteratively excludes models that are unable to pass the EPA hypothesis at a significance level α. The MCS can thus efficiently be used to determine which option pricing models have significant forecast accuracy for each moneyness and maturity region.

2.2.3 Delta-Hedged Trading Strategy

The economic significance of the difference in the out-of-sample forecast accuracy is tested by implementing a simple trading strategy based on the option price forecasts. The trading strategy is loosely based on a trading strategy designed by Bernales and Guidolin (2014). I adapt their trading strategy, which is based on implied volatility forecasts, to a strategy suited for option price forecasts. The strategy consists of purchasing (selling) an option contract on day t if a model forecasts that the price of the option contract is going to increase (decrease) on the next trading day $t+1$. This strategy is also implemented by Andreou et al. (2008). Excessive trading is avoided by requiring the option price to increase or decrease by at least 1%. The purchasing and selling of option contracts is performed simultaneously with the purchasing and selling of the underlying stock. The amount of stocks to be purchased or sold is determined by the option's delta. Adhering to this strategy ensures the trading position does not contain risks caused by stock-price movement of the underlying assets, i.e. the strategy is delta-hedged.

The total value of all delta-hedged positions in the portfolio on day t is given by

$$V_{t,i} = \sum_{m \in Q_{t,i}^{+}} \left(C_{m,t} - S_t \Delta_{m,t}\right) - \sum_{m \in Q_{t,i}^{-}} \left(C_{m,t} - S_t \Delta_{m,t}\right), \tag{20}$$

where S_t is the stock price at time t, $Q^+_{t,i}$ ($Q^-_{t,i}$) is the set of option contracts to be purchased (sold) at time t according to model i, $\Delta_{m,t}$ is the absolute value of delta of option m at time t. Following Bernales and Guidolin (2014), on each day a fixed amount of \$1000 is invested in the delta-hedged portfolio. The portfolio is then rebalanced on a daily basis. Consequently, when the total value of all delta-hedged positions in the portfolio, $V_{t,i}$, is positive, $X_{t,i} = \$1000/V_{t,i}$ units of the delta-hedged portfolio are purchased. The corresponding one-day net gain then equals $G_{t+1,i} = X_{t,i}\,(V_{t+1,i} - V_{t,i})$. When $V_{t,i}$ is negative, $X_{t,i} = \$1000/\,|V_{t,i}|$ units of the delta-hedged portfolio are sold and the obtained \$1000 plus the daily acquired \$1000 are altogether saved. This procedure earns a one-day net gain of $G_{t+1,i} + \$2000 \times (\mathrm{e}^{r_t} - 1)$, where r_t is the risk-free rate. The performance of each model is assessed based on mean return per invested dollar and the Sharpe ratio. Furthermore, the significance of the abnormality of the returns is tested by means of a t-test.

3 Data

The data are daily S&P 500 index European call option prices obtained from OptionMetrics. The data sample covers the period 04/01/1996—31/12/2016, closely following Yang et al. (2017) to make this study comparable. As the data is split in quarters, the results consist of $21 \times 4 - 3 = 81$ quarters. For the market price of the call option the historical end-of-day bid and ask quotes are used. The market price of the call option is approximated by the mid-quote, the mid-point of the bid-ask spread quoted for the call option. The corresponding risk-free rates and index dividend yields are also provided by OptionMetrics. Similar to Andreou et al. (2008), nonlinear cubic spline interpolation is used to match each option contract with a risk-free rate that corresponds to the option's maturity. For the volatility, skewness and kurtosis parameters 60-day historical estimates are obtained from the S&P 500 index quotes.

Following Yang et al. (2017), contracts with maturity less than 2 days are omitted. Furthermore, option quotes smaller than 0.5 are excluded because they are close to tick size. Other than that no exclusion criteria are incorporated, such that a wide range of options is priced. After the procedures 3346668 option quotes are left. Descriptive statistics of the data are given in Table 1. Over the years an increase in option prices and in the number of traded options are observed. Remarkably, more than 50% of all the options is traded between 2014 and 2016. Furthermore, the highest mean implied volatilities are observed around 2000, 2008 and 2011. In these years the dot-com bubble, the global financial crisis and European debt crisis occurred. As option pricing is more complicated during crises, these periods are explicitly featured in the graphs in the results section.

Table 1: Descriptive statistics

	1996—1998	1999—2001	2002—2004	2005—2007	2008—2010	2011—2013	2014—2016
c	109.96	148.95	105.40	147.71	162.83	258.70	299.19
K	849.21	1282.4	995.92	1263.3	1006.2	1247.4	1781.4
S	909.55	1319.4	1031.9	1360.2	1090.3	1470.1	2046.5
T	147.97	168.09	169.51	143.49	138.16	128.65	98.032
r	0.0565	0.0533	0.0167	0.0484	0.0108	0.0030	0.0041
q	0.0187	0.0096	0.0152	0.0193	0.0210	0.0221	0.0198
σ_{IV}	0.2680	0.2774	0.2494	0.2198	0.3496	0.3144	0.2722
σ_{60}	0.1637	0.2046	0.1755	0.1217	0.2599	0.1425	0.1311
Σ	124596	139990	129914	186869	388568	660689	1716042

Note: This table presents the mean market price of the call option (c), mean strike price (K), mean spot price of the underlying S&P 500 asset (S), mean maturity (T), mean risk-free interest rate (r), mean dividend yield (q), mean implied volatility (σ_{IV}), mean 60-day historical volatility (σ_{60}) and total number of options (Σ) for various time periods.

Besides the original S&P 500 option contracts, virtual option contracts are generated for the GNN to meet conditions C5 and C6. To meet condition C5, $\tau = 0$ is fixed, K is sampled uniformly in $[0, S_t]$ and the option price exactly equals $S_t - K$. These virtual options compel the neural network to learn the lower boundary of option prices such that options with small time-to-maturity are not mispriced. To meet condition C6, for every unique τ an option with $K = 0$ is created and S_t is drawn from the historical range. These virtual options correspond to the most expensive options; their existence leads to coincidence of the lower and upper bound, forcing the option price to equal the underlying price. It is not possible to deduce the historical volatility for the virtual options. Therefore, the virtual options are only generated for the GNN with two input variables.

4 Results

I proceed by presenting my findings, which are structured as follows. First, the option pricing performance of the parametric models is discussed. Subsequently, the optimal model architecture is analyzed. Then, the option pricing performance of the non-parametric models is evaluated, both for the hybrid and non-hybrid approaches. The analyses for the ANN, MNN and GNN are performed separately such that a more clear overview is given than when all models are compared simultaneously. Separately discussing the performance of each non-parametric method facilitates clear and concise discussions and in-depth analyses. The statistical significance of the difference in pricing accuracy of each model is tested by means of the Diebold-Mariano test and the model confidence set. Finally, the economic significance of the difference in pricing accuracy of each model is tested by means of a delta-hedged trading strategy.

4.1 Parametric Models

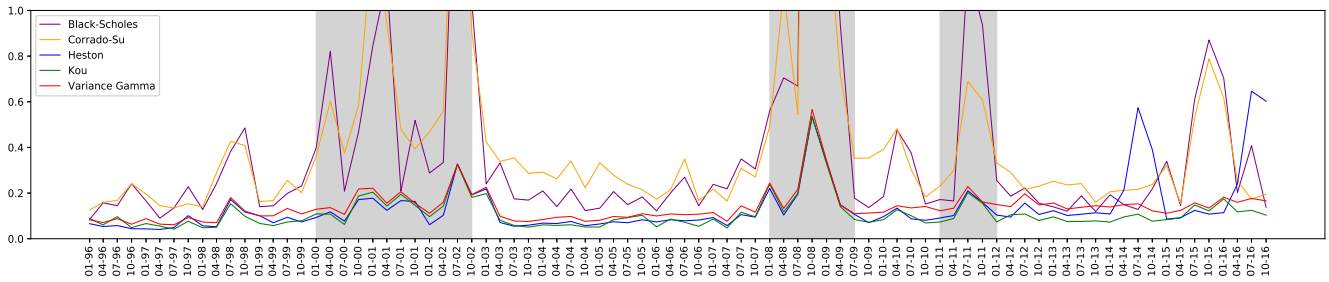
The MAPE and MSE of the option price forecasts of each of the five parametric models are given in Table 2. From Table 2 it can be concluded that the Kou jump-diffusion model outperforms all of the other parametric models in terms of MAPE and MSE. Figure 5 displays that the Kou jump-diffusion model achieves the lowest MAPE by almost constantly outperforming the other models on the whole data set. The Heston model achieves a MAPE almost similar to that of the Kou jump-diffusion model prior to 2012, but its MAPE increases significantly in 2014 and 2016. The Variance Gamma model, similar to the Kou jump-diffusion model, has a fairly constant MAPE. However, its value is almost consistently higher than that of the Kou jump-diffusion model. The Black-Scholes and Corrado-Su models perform worse than the three exponential Lévy models. This is presumably caused by the use of historical (60-day) parameters rather than daily calibrated parameters. Especially in crisis periods, indicated by the shadowed parts in Figure 5, the Black-Scholes and Corrado-Su models perform worse. From the MSE plot, given in Figure 22 in Appendix H, similar conclusions can be drawn. The Kou jump-diffusion model is able to outperform all of the other parametric models by varying its jump intensity, indicated by its calibrated parameters. A more elaborate analysis of the option pricing performance of the parametric models based on the daily calibrated parameters of each model is given in Appendix E.

Table 2: MAPE and MSE of the option price forecasts of the parametric models

	Black-Scholes	Corrado-Su	Heston	Kou	Variance Gamma
MAPE	0.419	0.404	0.203	0.112	0.149
MSE	180.2	183.8	19.32	12.96	16.10

This table presents the MAPE and MSE of the option price forecasts of the Black-Scholes model, the Corrado-Su model, the Heston model, the Kou jump-diffusion model and the Variance Gamma model. The data are daily S&P 500 index European call option prices obtained from OptionMetrics covering the period 04/01/1996–31/12/2016.

Figure 5: MAPE of the option price forecasts of the parametric models over time



Note: This figure shows how the MAPE of the option price forecasts of each of the parametric models evolves over time. The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

4.2 Optimal Model Architecture Analysis of Non-Parametric Models

As described in Section 2.2, the optimal all-encompassing model architecture for all neural network models—the number of hidden layers, the number of nodes in each hidden layer and the activation functions for each layer—is determined based on the performance on the whole data set in terms of the mean absolute percentage errors and the mean squared errors. I first perform a preliminary assessment to determine the combination of hidden layer nodes to test for each method. For the ANN with two input variables I perform a full run on the data set with one hidden layer, two hidden layers and three hidden layers. Also, two activation functions (sigmoid and ReLu) for the hidden layers are examined such that the preliminary decision is not influenced by the activation function. In this case the linear activation function is used for the output layer. For the single hidden layer I examine the case with two, four and eight hidden layer nodes. The second hidden layer either gets 16, 32 or 48 nodes. Finally, the cases with 64, 96 and 128 nodes in the third hidden layer are examined. From the results shown in Table 15 and 16 in Appendix F meaningful insights can already be deduced. The first obvious inference that can be made is that the models with three hidden layers outperform those with one and two hidden layers. The MAPE and MSE from the ANNs with one and two hidden layers are simply much higher than those of the ANN with three hidden layers. Therefore, each model is henceforth examined with three hidden layers. What remains is to decide the number of nodes for each of these three hidden layers per neural network methods.

For the ANN with three hidden layers, poor results are found when using only two nodes in the first hidden layer. The best results when using 64 nodes in the third hidden layer are attained in combination with 32 nodes in the second hidden layer. In the case of 96 nodes in the third hidden layer, either using 16 or 48 nodes in the second hidden layer seems to yield a good pricing performance. When using 128 nodes in the third hidden layer, having 32 nodes in the second hidden layer performs best. For the in-depth analysis of each neural network method the performance of neural networks with combinations of these hidden layer nodes is therefore investigated. Lastly, it can be observed that the results for the sigmoid and ReLU activation functions are similar. Keeping tractability in mind, it therefore suffices to only use one activation function for the following analyses. Therefore, for deciding the optimal number of hidden layer nodes, only neural networks with the ReLU activation function for the hidden layer and linear activation function for the output layer are assumed. Later in this section the activation functions are scrutinized.

Table 3 and 4 show the MAPE and MSE of the option price forecasts of ANNs, MNNs and MCMNNS with various numbers of hidden layer nodes. The choice for eight different combinations of hidden layer nodes for each method is based on performance during the preliminary assessment. It can be observed that

the architecture composed of eight nodes in the first hidden layer, 48 nodes in the second hidden layer and 96 nodes in the third hidden layer yields the best overall ranked pricing performance across all the models. This architecture of nodes attains the lowest MAPE and MSE for the ANN with 2 inputs, the ANN with 3 inputs and MNN with 3 inputs. Furthermore, for the MNN with 2 inputs the lowest MSE is also realized by the neural network configured with this architecture. The difference with the lowest MAPE for the MNN with 2 inputs is 0.007. Lastly, for the MCMNN this architecture attains the second lowest MAPE and MSE with differences of respectively 0.001 and 1.69 with the lowest values. Consequently, I decide to use this architecture of nodes for each of the ANN, MNN and MCMNN configurations, including the hybrid approaches. This way all the neural networks have an architecture of nodes that has been proved to work well for the specific neural network and simultaneously no arguments can be brought up about differences in performance due to a different architecture of nodes. Henceforth I refer to this architecture composed of eight, 48 and 96 nodes in the three hidden layers as the optimal architecture of nodes.

Using the optimal architecture of nodes, which was found using the ReLU and linear activation functions, the optimal architecture of activation functions is researched. I perform additional preliminary analyses to discover if combinations of hidden and output layer activation functions can be disregarded because of poor performance. The preliminary analyses are performed with all five activation functions for the hidden layer introduced in Appendix C. For the activation function of the output layer the linear and exponential activation functions are explored. Table 17 in Appendix F shows that for each activation function of the hidden layer a better performance is achieved when combined with the linear activation function in the output layer. Consequently, the exponential activation function for the output layer is hereinafter disregarded.

Table 3: MAPE from ANNs, MNNs and MCMNNs with various numbers of hidden layer nodes

	ANN				MNN				MCMNN			
	[32,64]	[16,96]	[48,96]	[32,128]	[32,64]	[16,96]	[48,96]	[32,128]	[32,64]	[16,96]	[48,96]	[32,128]
Panel A												
4	0.587	0.599	0.597	0.499	0.307	0.311	0.306	0.286				
8	0.526	0.654	0.522	0.614	0.285	0.283	0.289	0.281				
Panel B												
4	0.374	0.394	0.513	0.411	0.059	0.055	0.060	0.054	0.072	0.060	0.048	0.056
8	0.486	0.459	0.371	0.424	0.060	0.056	0.054	0.055	0.075	0.048	0.047	0.046

This table presents the MAPEs of the option price forecasts of ANNs, MNNs and MCMNNs with various numbers of hidden layer nodes. Panel A (B) displays the MAPEs of the models with two (three) input variables. In all cases, the ReLU activation function is used for the hidden layers and the linear activation function for the output layer. On the vertical axis, the number of hidden layer nodes for the first hidden layer is given. On the horizontal axis the number of nodes for the second and third hidden layer are given; the first number in brackets represents the number of nodes in the second hidden layer.

Table 4: MSE of the option price forecasts of ANNs, MNNs and MCMNNs with various numbers of hidden layer nodes

	ANN				MNN				MCMNN			
	[32,64]	[16,96]	[48,96]	[32,128]	[32,64]	[16,96]	[48,96]	[32,128]	[32,64]	[16,96]	[48,96]	[32,128]
Panel A												
4	153.9	113.8	170.2	131.5	152.1	162.2	272.2	219.5				
8	110.7	184.6	86.77	156.8	113.7	213.5	96.90	148.6				
Panel B												
4	52.34	60.25	94.93	91.40	59.04	56.47	69.67	64.58	110.8	58.51	60.43	62.94
8	78.93	77.66	40.05	146.8	79.24	57.15	53.93	55.75	79.20	44.99	46.68	46.90

This table presents the MSEs of the option price forecasts of ANNs, MNNs and MCMNNs with various numbers of hidden layer nodes. Panel A (B) displays the MSEs of the models with two (three) input variables. In all cases, the ReLU activation function is used for the hidden layers and the linear activation function for the output layer. On the vertical axis, the number of hidden layer nodes for the first hidden layer is given. On the horizontal axis the number of nodes for the second and third hidden layer are given; the first number in brackets represents the number of nodes in the second hidden layer.

Table 5: MAPE of the option price forecasts of ANNs, MNNs and MCMNNs with various activation functions for the hidden layers

	ANN					MNN					MCMNN				
	σ_1	σ_2	σ_3	σ_4	σ_5	σ_1	σ_2	σ_3	σ_4	σ_5	σ_1	σ_2	σ_3	σ_4	σ_5
2	1.005	0.962	1.151	1.393	0.522	0.293	0.290	0.277	0.275	0.289					
3	0.763	1.008	1.820	1.019	0.371	0.058	0.061	0.056	0.050	0.054	0.054	0.064	0.048	0.057	0.047

This table presents the MAPE of the option price forecasts of ANNs, MNNs and MCMNNs with three hidden layers for five hidden layer activation functions. The five activation functions are the sigmoid (σ_1), softplus (σ_2), hyperbolic tangent (σ_3), ELU (σ_4) and ReLU (σ_5) activation functions. The number of input variables is given on the vertical axis. The linear activation function is used for the output layer.

Table 6: MSE of the option price forecasts of ANNs, MNNs and MCMNNs with various activation functions for the hidden layers

	ANN					MNN					MCMNN				
	σ_1	σ_2	σ_3	σ_4	σ_5	σ_1	σ_2	σ_3	σ_4	σ_5	σ_1	σ_2	σ_3	σ_4	σ_5
2	191.6	123.9	358.7	337.3	86.77	231.9	197.6	188.3	96.93	96.90					
3	114.1	155.2	772.7	125.1	40.05	136.0	79.26	202.1	67.58	53.93	257.8	129.7	276.6	164.9	46.68

This table presents the MSE of the option price forecasts of ANNs, MNNs and MCMNNs with three hidden layers for five hidden layer activation functions. The five activation functions are the sigmoid (σ_1), softplus (σ_2), hyperbolic tangent (σ_3), ELU (σ_4) and ReLU (σ_5) activation functions. The number of input variables is given on the vertical axis. The linear activation function is used for the output layer.

Table 5 and 6 show the MAPE and MSE of the option price forecasts of ANNs, MNNs and MCMNNs for five hidden layer activation functions. The poor performance of the hyperbolic tangent activation function stands out; the neural networks with the hyperbolic tangent function have the highest MSE in four out of the five investigated neural network methods. For the MNN with two input variables—the case in which tanh is not the worse performing activation function in terms of MSE—the sigmoid function attains the highest MSE. Overall, the performance of the models with sigmoid activation functions can also be considered poor. The poor performance by the neural networks with tanh and sigmoid activation functions is caused by the vanishing gradient problem, which is described and discussed in detail in Appendix C.

Table 6 confirms that the neural networks with ReLU activation functions outperform all other architectures in terms of MSE. For the MNN, the performance of the ELU activation function is similar to that of the ReLU activation function. Table 5 shows that the networks with ELU activation even slighly outperform the networks with ReLU activation function in terms of MAPE for the MNN with both two and three input variables. However, both Table 5 and 6 reveal that for the ANN and MCMNN the networks with the ELU activation function are not nearly as accurate as the networks with the ReLU activation function. I therefore conclude that for the purpose of option pricing the ReLU activation function has an overall better performance than the ELU activation function across all neural network models studied. Consequently, I decide to use the architecture consisting of the optimal architecture of nodes, the linear activation function for the output layer and the ReLU activation function in each of the three hidden layers of the ANN, MNN and MCMNN. This way, all the neural networks that are examined have an architecture of activation functions that has been proved to work well for the specific neural network and simultaneously no arguments can be brought up about differences in performance due to a different architecture of activation functions. Henceforth, I refer to this architecture as the optimal architecture of the ANN, MNN and MCMNN.

4.3 Non-Parametric Models

In this section the option pricing performance of the non-parametric models is evaluated, both for the hybrid and non-hybrid approaches. A distinction is made for models with two and three inputs; models with two and three input variables are scrutinized similarly but separately. Therefore, for each method the best performing model with two inputs and three inputs is selected in terms of MAPE and MSE. Plots of how the MAPE and MSE of each model behave over time are examined as well. Separately discussing the performance of each non-parametric method facilitates clear and concise discussions and in-depth analyses.

4.3.1 Artificial Neural Network

Table 7 exhibits the MAPE and MSE of the option price forecasts of the standard ANN and each of the hybrid ANN approaches, each with either two or three input variables. The first striking takeaway is that the hybrid approach yields a better option pricing performance than the standard approach for both two and three input variables; the hybrid ANNs adjusted by the Heston, Kou and Variance Gamma achieve MAPEs and MSEs which are reasonably smaller than the MAPE and MSE of the option price forecasts of the standard ANN. In Section 4.1 these three models already showed a superior option pricing performance when compared to the Black-Scholes and Corrado-Su model. Their pricing thus allows the HANN to outperform the standard ANN model. The hybrid approach thus prospers in combination with accurate parametric estimates. For the two input variables, the HANN adjusted by the Kou jump-diffusion model option prices achieves the lowest MAPE and MSE. With three input variables the HANN adjusted by the Variance Gamma model option prices has the lowest MAPE and MSE. The Kou jump-diffusion model is the best performing parametric model, but the HANN with three input variables apparently achieves better results when the target function is adjusted by the Variance Gamma model outputs. A first conclusion that can thus be drawn is that it does not hold that a hybrid neural network adjusted by a parametric model with superior performance in terms of MAPE and MSE necessarily outperforms other hybrid neural networks adjusted by inferiorly performing parametric models. This is related to the pricing of options with high volatilities and is more extensively discussed in Section 4.3.3.

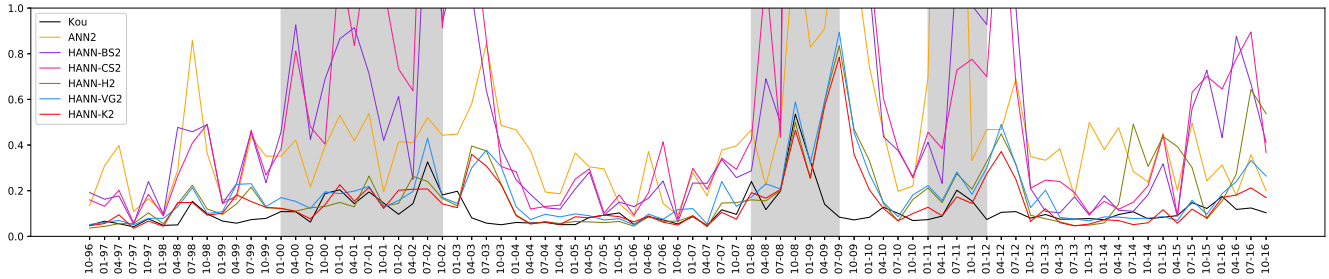
Table 7 further reveals that there is no artificial neural network with two input variables, neither standard nor hybrid, that outperforms the parametric Kou jump-diffusion model in terms of MAPE. The MAPE of the Kou jump-diffusion model output is 0.112 and no ANN or HANN succeeds in attaining a lower MAPE. For this reason the Kou jump-diffusion model is included as a benchmark in the MAPE plots of Figure 6 and 7. However, the HANNs adjusted by the Kou jump-diffsion model and Variance Gamma model with two input parameters do succeed in attaining a lower MSE than the parametric Kou jump-diffusion model. Another interesting observation is that for each model the MAPE decreases when adding extra input variables, whereas the MSE only decreases for the ANN. The additional input variables thus cause the accuracy of the option pricing performance to increase. However, more severe misestimation are observed, such that the consistency of the option pricing performance decreases. Finally, the extra input parameter in general seems to have a bigger influence on the standard model than on the hybrid models. This can be explained by the fact that the volatility information is already encompassed in the information provided by the parametric models.

Table 7: MAPE and MSE of the option price forecasts of the ANN and HANNs with optimal architecture

	ANN	HANN-BS	HANN-CS	HANN-H	HANN-K	HANN-VG
2	0.474 (98.34)	0.599 (143.9)	0.619 (180.7)	0.265 (19.07)	0.143 (10.70)	0.182 (10.80)
3	0.317 (41.60)	0.572 (157.6)	0.600 (196.7)	0.255 (26.61)	0.138 (16.22)	0.123 (15.19)

This table presents MAPEs (MSEs) from the ANN and HANNs with either two (m, τ) or three $(m, \sigma\sqrt{\tau}, r\tau)$ input variables and with optimal architecture. The number of input variables is given on the vertical axis. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG).

Figure 6: MAPE of the option price forecasts of the Kou jump-diffusion model, the standard ANN with two input variables and the HANNs with two input variables over time

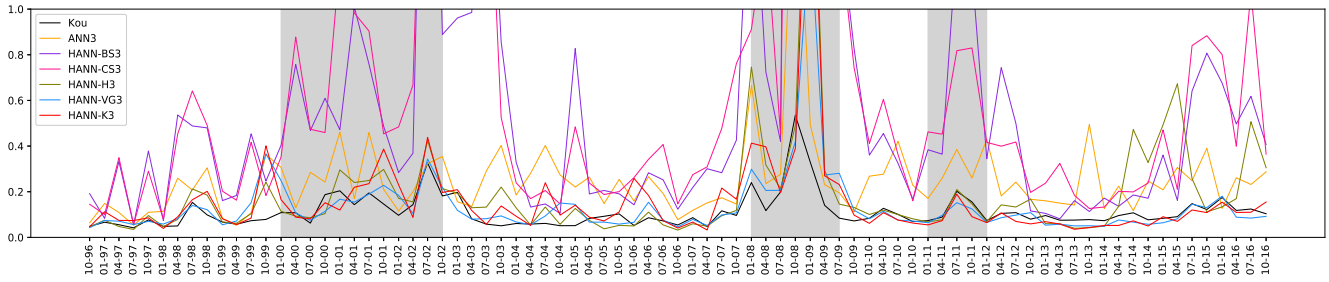


Note: This figure shows how the MAPE from the Kou jump-diffusion model, the standard ANN with two input variables and the HANNs with two input variables evolves over time. The two input variables are m and τ. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

Figure 6 shows that in most of the quarters the Kou jump-diffusion model outperforms the ANN and HANNs with two input variables. Also, the poor performance of the ANN stands out. The ANN is oftentimes outperformed by all other methods. The HANNs adjusted by the Heston, Variance Gamma and Kou models at times outperform the parametric Kou jump-diffusion model, but no persistence is shown. Figure 23 in Appendix H shows that in terms of MSE the HANN-K model with two input parameters succeeds in frequently outperforming the Kou jump-diffusion model.

Figure 7 displays that the HANNs with three input parameters adjusted by the Variance Gamma and Kou model frequently outperform the Kou jump-diffusion model. After 2011 the Kou jump-diffusion model is consistently outperformed by either of the two models. In earlier years, the Kou jump-diffusion model nonetheless has a much better option pricing performance in terms of MAPE. Also, the standard ANN with three inputs seemingly performs much better. In terms of MSE it even outperforms all other models in 1999, as can be observed in Figure 24 in Appendix H. However, the performance is inconsistent. The consistency is key to truly outperform the Kou jump-diffusion model in terms of both MAPE and MSE.

Figure 7: MAPE of the option price forecasts of the Kou jump-diffusion model, the standard ANN with three input variables and the HANNs with three input variables over time



Note: This figure shows how the MAPE of the option price forecasts of the Kou jump-diffusion model, the standard ANN with three input variables and the HANNs with three input variables evolves over time. The three input variables are $m, \sigma\sqrt{\tau}$ and $r\tau$. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

So far, no neural network model that can consistently outperform the exponential Lévy models has been found. However, the introduction of additional information at times improves the option pricing performance of the models, especially for the standard ANN. Furthermore, the hybrid approach improves the performance of the artificial neural network. The conclusions drawn by e.g. Boek et al. (1995), Andreou et al. (2008) and Lajbcygier (2004) that the hybrid approach outperforms the standard approach and parametric models such as the Black-Scholes and Corrado-Su model thus prove to be correct. Nevertheless, no convincing evidence has been found to substantiate the claim that neural networks can consistently outperform more sophisticated parametric models such as exponential Lévy models. The next section researches whether introducing modularity to hybrid neural networks can help succeed in doing so.

4.3.2 Modular Neural Network

Table 8 exhibits the MAPE and MSE from the standard MNN and each of the hybrid MNN approaches, each with either two or three input variables. For the models with two input parameters the same conclusions can be drawn from the results as from the results of the ANN and HANNs with two input variables: the standard approach outperforms the hybrid approaches adjusted by the Black-Scholes and Corrado-Su models, but is outperformed by the hybrid approaches adjusted by the Heston, Kou and Variance Gamma models in terms of MAPE and MSE. Furthermore, the hybrid approach adjusted by the Kou jump-diffusion model is again the best performing neural network with two input variables. In the results of the HMNN models with three input variables a similar tendency can be found as in the results of the HANN models.

Table 8: MAPE and MSE of the option price forecasts of the MNN and HMNNs with optimal architecture

	MNN	HMNN-BS	HMNN-CS	HMNN-H	HMNN-K	HMNN-VG
2	0.282 (92.36)	0.596 (141.2)	0.637 (180.1)	0.256 (18.92)	0.139 (10.64)	0.174 (10.67)
3	0.056 (52.62)	0.588 (154.7)	0.631 (196.0)	0.242 (22.78)	0.139 (14.37)	0.138 (14.37)

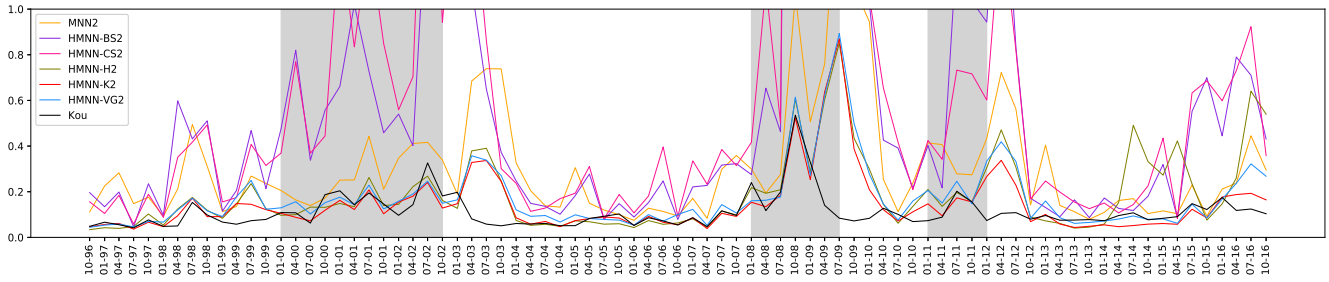
This table presents MAPEs (MSEs) from the MNN and HMNNs with either two (m, τ) or three $(m, \sigma\sqrt{\tau}, r\tau)$ input variables and with optimal architecture. The number of input variables is given on the vertical axis. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG).

The hybrid model adjusted by the Variance Gamma is again the best performing hybrid model in terms of MAPE and MSE. Furthermore, for each of the hybrid models the MAPE has again decreased with the introduction of additional inputs, whereas the MSE has increased. The additional input variables again increase the accuracy but decrease the consistency of the option pricing performance.

A big discrepancy between the results of the artificial and modular neural network models with three input variables is the performance of the standard approach. The standard MNN is the first model that is able to outperform the Kou jump-diffusion model in terms of MAPE. However, in terms of MSE the performance of the standard MNN with three input variables is suboptimal. A closer look into the estimated option prices learns that the high MSE is caused by the severe misestimation of options with high market call prices. While the percentage error of such estimations can be relatively small, the squared error inflates when the absolute estimation error increases. The small MAPE is obtained by accurately pricing options with low market call prices; other models often struggle with pricing these options.

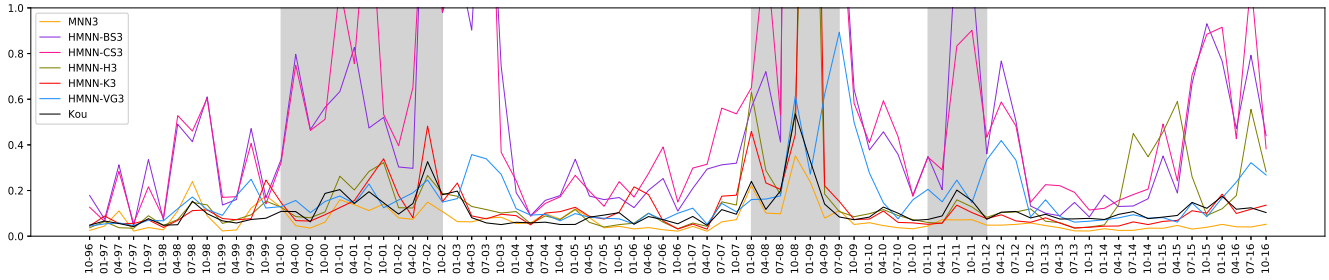
Figure 8 shows that at times the HMNNs with two input parameters adjusted by the Heston, Variance Gamma and Kou models are able to outperform the Kou jump-diffusion model. However, the performance is not consistent enough to obtain a MAPE lower than 0.112. Figure 25 in Appendix H shows that the hybrid MNNs adjusted by the Variance Gamma and Kou models outperform the Kou jump-diffusion model in terms of MSE by constantly realizing a lower MSE from 2012 until 2016 and realizing comparable MSEs in the years prior. Figure 9 depicts that the MNN almost constantly attains a MAPE lower than that of the parametric or hybrid models. The MNN with three input parameters has the lowest MAPE in all but 13 quarters, thereby outperforming all other models on approximately 84% of the data set. The suboptimal option pricing performance of the MNN with three input parameters in terms of MSE is depicted in Figure 26 in Appendix H. While in earlier years the performance is at times paramount, after 2007 it fails to maintain this performance. Overall, none of the MNNs or HMNNs is able to simultaneously outperform the parametric Kou jump-diffusion models in terms of both MAPE and MSE.

Figure 8: MAPE of the option price forecasts of the Kou jump-diffusion model, the standard MNN with two input variables and the HMNNs with two input variables over time



Note: This figure shows how the MAPE of the option price forecasts of the Kou jump-diffusion model, the standard MNN with two input variables and the HMNNs with two input variables evolves over time. The two input variables are m and τ. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

Figure 9: MAPE of the option price forecasts of the Kou jump-diffusion model, the standard MNN with three input variables and the HMNNs with three input variables over time



Note: This figure shows how the MAPE of the option price forecasts of the Kou jump-diffusion model, the standard MNN with three input variables and the HMNNs with three input variables evolves over time. The three input variables are $m, \sigma\sqrt{\tau}$ and $r\tau$. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

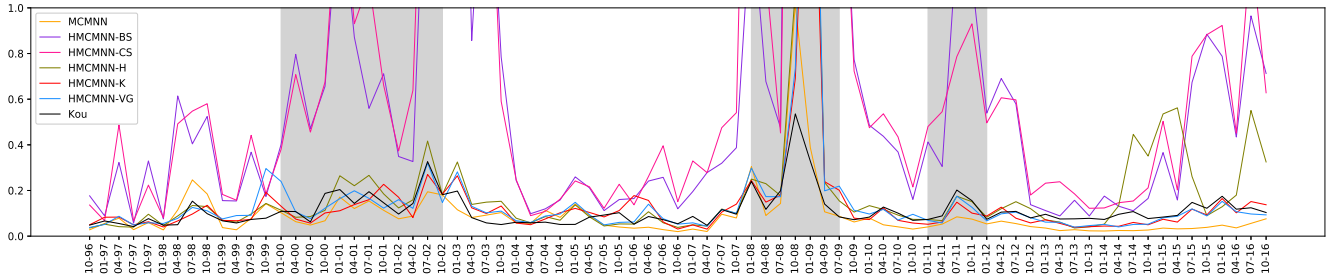
The results of the MCMNN in Table 9 closely resemble the results of the MNN with three input parameters. The introduction of an extra volatility criterion seems not to have a considerable impact. In fact, the performance of the standard MCMNN is inferior to the performance of the standard MNN with three inputs in terms of MAPE and MSE. Similar to the MNN a low MAPE in combination with a high MSE is observed. This can again be attributed to the misestimation of options with high market call prices and the accurate pricing of options with low market call prices. When looking at the HMCMNNs an improvement in option pricing performance is observed when compared to the performance of the HMNNs. The HMCMNN-K attains the lowest MAPE thus far of all the hybrid neural networks.

Table 9: MAPE and MSE of the forecasts of the MCMNN and HMCMNNs with optimal architecture

	MCMNN	HMCMNN-BS	HMCMNN-CS	HMCMNN-H	HMCMNN-K	HMCMNN-VG
3	0.068 (43.64)	0.624 (156.4)	0.729 (207.1)	0.252 (22.02)	0.122 (11.14)	0.130 (11.40)

This table presents MAPEs (MSEs) from the MCMNN and HMCMNNs with two (m, τ) and three $(m, \sigma\sqrt{\tau}, r\tau)$ input variables and with optimal architecture. The number of input variables is given on the vertical axis. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG).

Figure 10: MAPE of the forecasts of the Kou jump-diffusion model, the standard MCMNN with three input variables and the HMCMNNs with three input variables over time

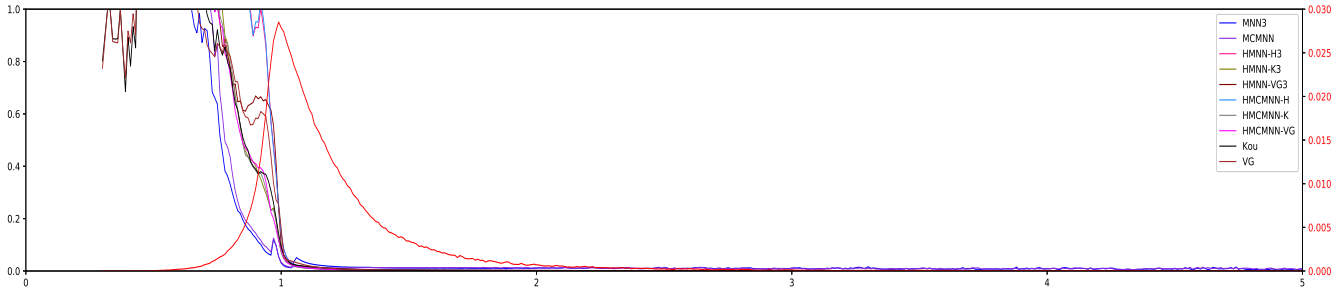


Note: This figure shows how the MAPE of the option price forecasts of the Kou jump-diffusion model, the standard MCMNN with three input variables and the HMCMNNs with three input variables evolves over time. The three input variables are $m, \sigma\sqrt{\tau}$ and $r\tau$. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

Figure 10 exhibits the optimal performance of the MCMNN as measured by the MAPE, especially after 2008. Figure 27 in Appendix H exhibits how the HMCMNNs adjusted by the Variance Gamma and Kou model outperform the standard MCMNN in terms of MSE over time. However, no MCMNN or HMCMNN model succeeds in consistently attaining a MAPE and MSE lower than that of the Kou jump-diffusion model.

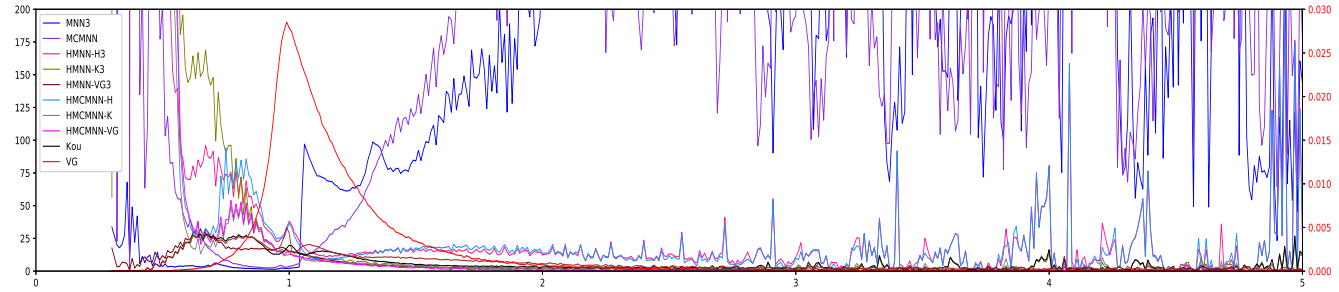
Because of the discrepancy between the MAPE and MSE from the (hybrid) MNNs and MCMNNs, it proves insightful to examine the option pricing performance as a function of moneyness and maturity. Figure 11 and 12 show the MAPE and MSE behavior of standard and hybrid modular neural networks together with two parametric models as a function of moneyness. The hybrid BS and CS models have proved to perform inferiorly and are therefore omitted for visibility purposes. In the figures a red line is included to indicate the percentage of options per moneyness region. The bulk of the options is evidently centered around a moneyness of 1. It is noteworthy that approximately 60% of the options is defined as in-the-money, whereas the at-the-money and out-of-the-money options each constitute approximately 20%.

Figure 11: MAPE of the option price forecasts of standard and hybrid modular neural networks and parametric models as a function of moneyness



Note: This figure shows how the MAPE from several (hybrid) neural networks with three input parameters that implement modularity based on fixed manual heuristics and two parametric benchmark models, Kou and VG, evolves as moneyness increases. The three input variables are $m, \sigma\sqrt{\tau}$ and $r\tau$. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The red line represents the percentage of options per moneyness region. Moneyness regions consist of options whose moneyness is equal at two decimal places. The red line has a separate y-axis on the right.

Figure 12: MSE of the option price forecasts of standard and hybrid modular neural networks and parametric models as a function of moneyness



Note: This figure shows how the MSE from several (hybrid) neural networks with three input parameters that implement modularity based on fixed manual heuristics and two parametric benchmark models, Kou and VG, evolves as moneyness increases. The three input variables are $m, \sigma\sqrt{\tau}$ and $r\tau$. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The red line represents the percentage of options per moneyness region. Moneyness regions consist of options whose moneyness is equal at two decimal places. The red line has a separate y-axis on the right.

Figure 11 indicates that the MNN and MCMNN achieve their supreme performance in terms of MAPE by more accurately pricing out-of-the-money options. Most models are able to accurately price at-the-money and in-the-money options, such that the pricing of out-of-the-money options is decisive. Figure 12 however shows that for deep-in-the-money options the MNN and MCMNN attain a poor performance. The price of deep-in-the-money options on average is very high, due to the big difference between strike and asset price. These results therefore corroborate the claim that the MNN and MCMNN severely misestimate higher priced options but that this is not clearly indicated by the MAPE performance metric. The importance of employing multiple performance metrics is thus emphasized.

Figure 31 and 32 in Appendix I show the MAPE and MSE behavior of standard and hybrid modular neural networks as a function of maturity. As indicated by the red line, the options are more evenly distributed over the maturity regions. The MNN is able to constantly outperform all other models in terms of MAPE. However, in terms of MSE the superior performance of the hybrid modular neural networks adjusted by the Variance Gamma model stands out. Furthermore, the HMCMNNs perform well for short-term maturities, but the MSE increases as maturity increases. Finally, the performance of the hybrid neural networks adjusted by the Heston models deteriorates with maturity, indicating that the hybrid neural networks adjusted by exponential Lévy models are in this case better able to fit the maturity grid than hybrid neural networks adjusted by stochastic volatility models.

Generally, the introduction of simple modularity based on fixed manual heuristics to neural network models improves the pricing performance of the standard approaches in terms of MAPE. This feat is achieved by a more accurate pricing of out-of-the-money options. However, the performance in terms of MSE indicates that the models severely misestimate deep-in-the-money options, such that overall the benefit of introducing simple modularity is questionable. Another interesting result is that introducing the hybrid approach to modular neural networks improves the option pricing performance. Especially the hybrid modular neural networks adjusted by exponential Lévy models are able to accurately fit options on the entire moneyness and maturity grids. However, the hybrid models do not gain much from the introduction of modularity, as the results that are acquired do not differ substantially from the results acquired by the HANNs in the previous section. This suggests that the hybrid approach already partly incorporates simple modularity through the information encompassed in the parametric option price forecasts.

4.3.3 Gated Neural Network

Table 10 exhibits the MAPE and MSE of the option price forecasts of the standard GNN and each of the hybrid GNN approaches, each with either two or three input variables. Furthermore, the MAPE and MSE of the option price forecasts of the GNN* and each of the hybrid GNN* approaches is presented. I observe that adding virtual options to the GNN with two input variables slightly improves the option pricing performance of the model. This statement does however not hold for the hybrid approaches. The goal of the introduction of the virtual options is to encode prior information based on financial axioms as constraints. These constraints compel the GNN* to better price options close to maturity and force boundaries upon the estimated option price, thereby increasing the pricing performance of the GNN*. However, the benefit of these constraints is apparently lost in the implementation of the hybrid approach.

Table 10: MAPE and MSE of the forecasts of the GNN, GNN* and HGNNs with optimal architecture

	GNN	HGNN-BS	HGNN-CS	HGNN-H	HGNN-K	HGNN-VG
2	0.242 (47.30)	0.603 (140.2)	0.639 (173.7)	0.264 (18.37)	0.100 (8.897)	0.115 (8.835)
2*	0.234 (46.42)	0.676 (153.5)	0.848 (200.0)	0.323 (41.01)	0.113 (10.95)	0.125 (13.44)
3	0.108 (9.311)	0.563 (137.9)	0.586 (152.1)	0.258 (19.19)	0.094 (8.791)	0.094 (8.179)

This table presents MAPEs (MSEs) of the option price forecasts of the GNN, GNN* and HMNNs with either two (m, τ) or three $(m, \sigma\sqrt{\tau}, r\tau)$ input variables and with optimal architecture. The number of input variables is given on the vertical axis, where an asterisk (*) implies that virtual options are appended to the training data. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG).

An important takeaway from Table 10 is that both the GNN and HGNN are able to outperform the option pricing performance of the Kou jump-diffusion model in terms of both MAPE and MSE. These models are however not able to improve upon the MAPE of the MNN and MCMNN with three input variables. Nevertheless, it must be noted that these neural network are the only ones able to consistently outperform the introduced exponential Lévy models as measured by all performance metrics. This is done by introducing modularity with option grouping that is automatic and learned from data instead of categorizing options based on fixed manual heuristics, such that the grouping and the pricing function of each group can dynamically be adjusted by the neural network as the market changes over time.

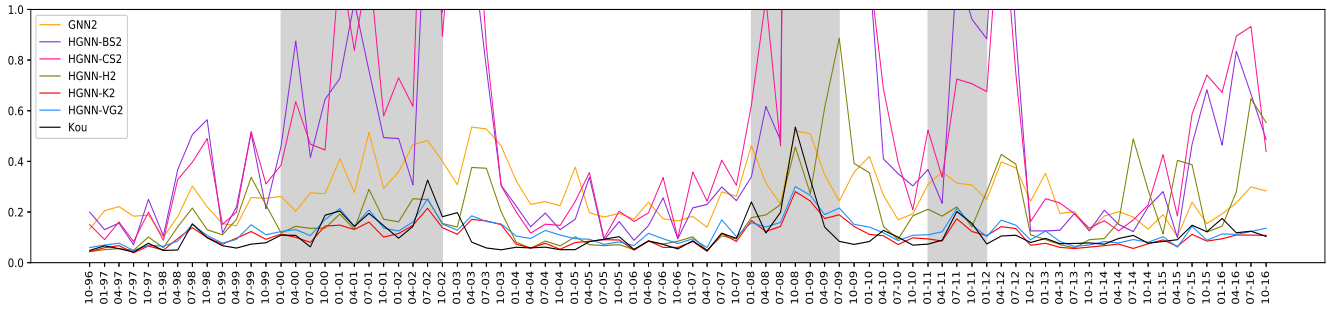
Furthermore, the results in Table 10 confirm the overall superiority of the GNN concerning the hybrid approach; the HGNNs outperform the HANNs, HMNNs and HMCMNNs in terms of both performance metrics for all parametric models except for the Heston model. This can be observed more conveniently in the comprehensive overview of results in Table 18 in Appendix G. For the GNNs with two and three input parameters and for the GNN* the hybrid models are able to outperform the standard model when the target function is adjusted by the option prices obtained from the Variance Gamma or Kou jump-diffusion model. This emphasizes the contribution of the hybrid approach. In general, introducing the hybrid approach and additional input variables improves the option pricing performance of the GNN sufficiently.

The results of the GNN* do not exactly mimic the results obtained by Yang et al. (2017), although approximately the same data set and implementation are adopted. This could be explained by the fact that I do not discard in-the-money option quotes. In-the-money option quotes make up roughly 60% of the data set and disregarding these options ultimately means disregarding three modules of the MNN and six modules of the MCMNN. For the sake of completeness these options are therefore not discarded. This means that the GNN* in this paper is dictated to price a larger array of options than the GNN* in Yang et al. (2017), which leads to more diverse options and larger corresponding prediction errors.

Out of the four models that are able to outperform the Kou jump-diffusion model in terms of both MAPE and MSE the HGNN adjusted by the Variance Gamma model overall attains the best option pricing performance. As previously observed for the HANN and HMNN, the hybrid approach in combination with the Variance Gamma model and three input variables trumps all other hybrid models with three input variables. Although the parametric Variance Gamma model is outperformed by the Kou jump-diffusion model, its performance in combination with the Palmer (2019) input parameters is superior. This can be explained by the fact that the Variance Gamma model prices option with high volatilities more accurately. Figure 35 in Appendix J convincingly depicts that the Variance Gamma models prices high volatility options better than the Kou jump-diffusion model, especially after 2012. This feat, in combination with the additional input parameters and the GNN specification, helps the HGNN adjusted by the Variance Gamma model attain the most consistent option pricing performance.

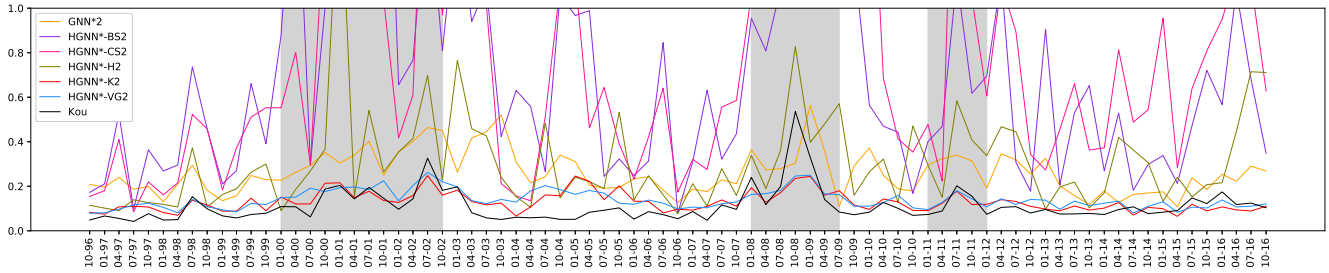
The performance of the GNN, GNN* and HGNNs over time in terms of MAPE is displayed in Figure 13, 14 and 15. The MSE plots of these models are exhibited in Figure 28, 29 and 30 in Appendix H. The first thing that stands out from the plots is the option pricing performance during and after the global financial crisis. Previously examined models mostly struggle with outperforming the Kou jump-diffusion model in times of financial turmoil. Models that succeed in attaining a better option pricing performance than the Kou jump-diffusion model in 2008, such as the ANN and MNN with three input variables, fail to maintain this performance after 2008. The four models that are able to outperform the Kou jump-diffusion model in terms of both MAPE and MSE mainly do so by outperforming the Kou jump-diffusion model in 2008 and thereafter maintaining their performance.

Figure 13: MAPE of the option price forecasts of the Kou jump-diffusion model, the standard GNN with two input variables and the HGNNs with two input variables over time



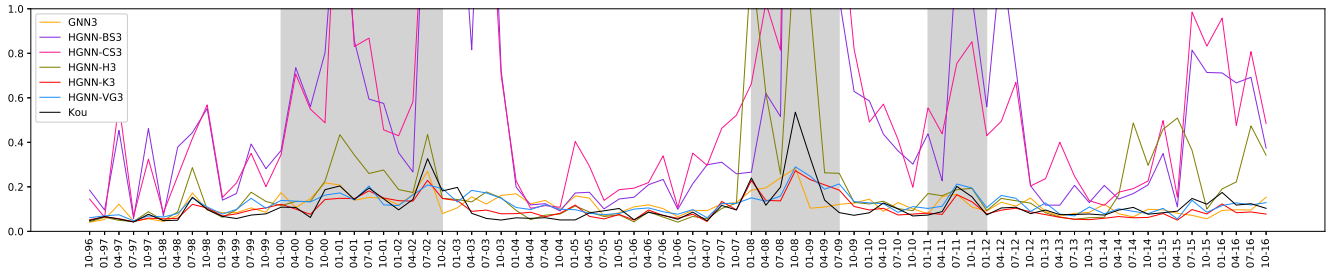
Note: This figure shows how the MAPE of the option price forecasts of the Kou jump-diffusion model, the standard GNN with two input variables and the HGNNs with two input variables evolves over time. The two input variables are m and τ. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

Figure 14: MAPE of the option price forecasts of the Kou jump-diffusion model, the standard GNN* with two input variables and the HGNN*s with two input variables over time



Note: This figure shows how the MAPE of the option price forecasts of the Kou jump-diffusion model, the standard GNN* with two input variables and the HGNN*s with two input variables evolves over time. The two input variables are m and τ. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

Figure 15: MAPE of the option price forecasts of the Kou jump-diffusion model, the standard GNN with three input variables and the HGNNs with three input variables over time

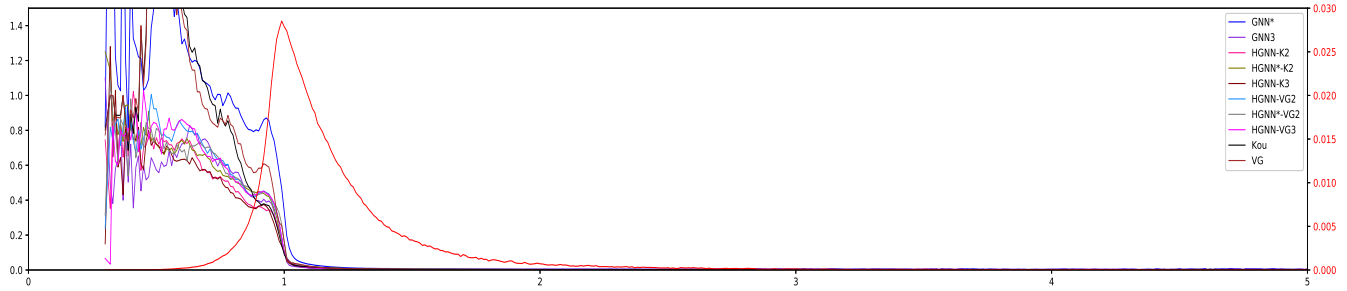


Note: This figure shows how the MSE of the option price forecasts of the Kou jump-diffusion model, the standard GNN with three input variables and the HGNNs with three input variables evolves over time. The three input variables are $m, \sigma\sqrt{\tau}$ and $r\tau$. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

Comparing the MAPE from the GNN in Figure 13 and that from the GNN* in Figure 14 shows that the due to introduction of virtual options the MAPE for each period decreases slightly. This indicates a more consistent and accurate option pricing performance. The performance of the GNN* also contains less excessive errors, as illustrated by the MSE plots over time in Figure 28 and 29. Additionally, applying the hybrid approach to the GNN* further improves the option pricing performance.

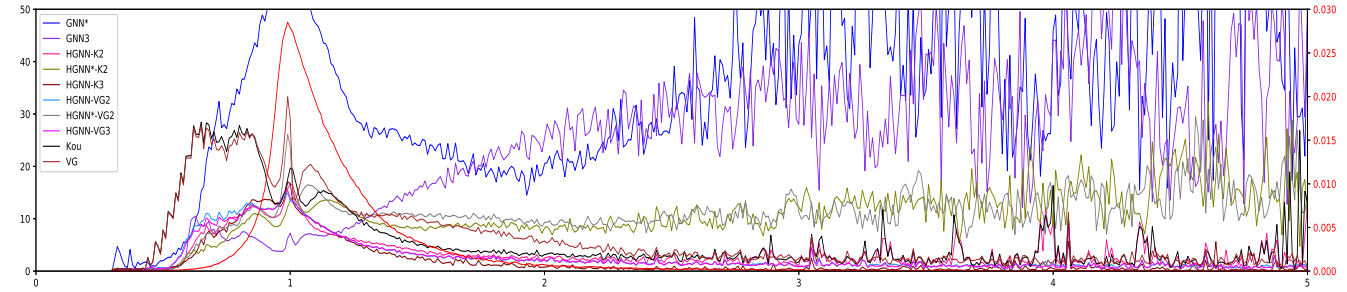
I again examine the option pricing performance as a function of moneyness and maturity. Figure 16 and 17 show the MAPE and MSE behavior of standard and hybrid gated neural networks as a function of moneyness, whereas Figure 33 and 34 in Appendix I show the MAPE and MSE as a function of maturity.

Figure 16: MAPE of the option price forecasts of standard and hybrid gated neural networks and parametric models as a function of moneyness



Note: This figure shows how the MAPE from several (hybrid) neural networks with three input parameters that implement modularity based on dynamic option grouping and two parametric benchmark models, Kou and VG, evolves as moneyness increases. The three input variables are $m, \sigma\sqrt{\tau}$ and $r\tau$. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The red line represents the percentage of options per moneyness region. Moneyness regions consist of options whose moneyness is equal at two decimal places. The red line has a separate y-axis on the right.

Figure 17: MSE of the option price forecasts of standard and hybrid gated neural networks and parametric models as a function of moneyness



Note: This figure shows how the MSE from several (hybrid) neural networks with three input parameters that implement modularity based on dynamic option grouping and two parametric benchmark models, Kou and VG, evolves as moneyness increases. The three input variables are $m, \sigma\sqrt{\tau}$ and $r\tau$. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The red line represents the percentage of options per moneyness region. Moneyness regions consist of options whose moneyness is equal at two decimal places. The red line has a separate y-axis on the right.

In terms of MAPE and MSE, the performance of most neural network models that implement modularity based on dynamic option grouping is comparable for out-of-the-money options. The poor performance of non-hybrid models for deep-in-the-money options in terms of MSE is again noteworthy. The hybrid approach thus seems necessary to properly price the entire moneyness grid. Additionally, all hybrid gated neural networks are able to maintain an accurate and consistent option pricing performance as maturity increases. The performance of most hybrid models is fairly equivalent. To truly differentiate between the models across maturity and moneyness regions the statistical significance of the difference in the out-of-sample forecast accuracy must therefore be scrutinized.

Overall, the results in this section indicate that neural network models for option pricing benefit from introducing modularity with option grouping that is automatic and learned from data. The resulting GNN attains a better option pricing performance than the previously introduced ANN, MNN and MCMNN. Introducing virtual options to the GNN slightly enhances its performance. Most importantly, applying the hybrid approach—specifically an exponential Lévy model in appropriate combination with a GNN—further improves the option pricing performance of the GNN. The option pricing performance of the resulting HGNNs is even more enhanced by introducing additional input parameters. Finally, the HGNNs attain an accurate and consistent option pricing performance across moneyness and maturity regions.

4.4 Diebold-Mariano Test

Based on the MAPE and MSE performance in the previous sections the best-performing model per neural network model approach and number of input parameters is selected. When judged by the MAPE performance metric the MNN with three input variables is the overall best-performing model, whereas judged by the MSE performance metric the HGNN with three input parameters adjusted by the Variance Gamma model is the overall best-performing model. The Diebold-Mariano test assesses if the forecasts accuracy of these models is significantly better than that of other models based on a given loss differential.

The null hypotheses that there is no difference in the MAPE and MSE of these models are tested in Table 11 and 12. A positive test-statistic implies that the performance of the model on the vertical axis is superior, whereas a negative test-statistic implies that the performance of the model on the horizontal axis is superior. The number of option price forecasts equals $P = 3325494$, such that the test-statistic is compared with a t-distribution with $P-1$ degrees of freedom. As the t-distribution approaches the normal distribution for large values of P, the null hypothesis of no significant difference in forecast accuracy is rejected at the 1% significance level if the absolute value of the test-statistic exceeds 2.58.

It follows from the results in Table 11 that the forecasts of the MNN with three input variables are significantly more accurate than the forecasts of all other models if the loss differential is measured by the MAPE performance metric. The results in Table 12 indicate that the forecasts obtained by the HGNN with three input parameters adjusted by the Variance Gamma model are significantly more accurate than the forecasts of all other models if MSE is used as performance metric. Two performance metrics are employed to ensure that the models are capable of producing accurate option price forecast while not making any excessive errors, as mentioned in Section 2.2. Working with two different performance metrics leads to the unavoidable but unsatisfactory conclusion that a different model is favored by each performance metric.

Table 11: Diebold-Mariano test-statistics based on the MAPE loss differential

	Kou	HANN-K2	HANN-VG3	HMNN-K2	MNN3	HMNN-VG3	MCMNN	HMCMNN-K	HGNN-K2	HGNN-VG3
Kou		114.3	15.52	102.7	-271.0	36.15	-82.71	13.24	-92.37	-123.5
HANN-K2	-114.3		-76.68	-10.74	-295.3	-0.397	-130.5	-24.70	-167.8	-185.1
HANN-VG3	-15.52	76.68		72.12	-236.7	38.41	-96.09	9.206	-69.84	-92.6
HMNN-K2	-102.7	10.74	-72.12		-278.8	1.201	-130.4	-23.42	-153.1	-170.5
MNN3	271.0	295.3	236.7	278.8		109.0	35.53	91.03	240.1	213.4
HMNN-VG3	-36.15	0.397	-38.41	-1.201	-109.0		-86.88	-42.06	-53.97	-60.83
MCMNN	82.71	130.5	96.09	130.4	-35.53	86.88		68.33	55.16	44.40
HMCMNN-K	-13.24	24.70	-9.206	23.42	-91.03	42.06	-68.33		-32.31	-39.55
HGNN-K2	92.37	167.8	69.84	153.1	-240.1	53.97	-55.16	32.31		-49.94
HGNN-VG3	123.5	185.1	92.58	170.5	-213.4	60.83	-44.40	39.55	49.94	

This table presents the Diebold-Mariano test statistics for the best-performing models, as determined in Section 4.1, 4.3.1, 4.3.2 and 4.3.3. The parametric model is the Kou jump-diffusion model, denoted by Kou. All other models are non-parametric models. Non-parametric models whose abbreviation starts with an H are hybrid models. For the hybrid models, -K (-VG) denotes that the input of the model is adjusted by the option prices obtained from the Kou (Variance Gamma) model. The integer behind each non-parametric model abbreviation represents the number of input variables. The null hypothesis that there is no difference in the MAPE of the models is tested.

Table 12: Diebold-Mariano test-statistics based on the MSE loss differential

	Kou	HANN-K2	HANN-VG3	HMNN-K2	MNN3	HMNN-VG3	MCMNN	HMCMNN-K	HGNN-K2	HGNN-VG3
Kou		-223.5	20.22	-206.3	300.6	12.37	188.1	-32.33	-180.1	-207.46
HANN-K2	223.5		68.21	40.35	324.4	44.61	207.1	8.775	-87.15	-119.9
HANN-VG3	-20.22	-68.21		-65.51	265.5	-2.033	170.3	-56.45	-98.65	-118.9
HMNN-K2	206.3	-40.35	65.51		322.9	42.84	206.1	6.530	-92.92	-124.3
MNN3	-300.6	-324.4	-265.5	-322.9		-241.2	-41.20	-282.21	-345.5	-353.1
HMNN-VG3	-12.37	-44.61	2.033	-42.84	241.2		159.8	-68.95	-66.52	-78.75
MCMNN	-188.1	-207.1	-170.3	-206.1	41.20	-159.8		-188.2	-221.3	-227.1
HMCMNN-K	32.33	-8.775	56.45	-6.530	282.2	68.95	188.2		-36.67	-51.29
HGNN-K2	180.1	87.15	98.65	92.92	345.5	66.52	221.3	36.67		-72.04
HGNN-VG3	207.5	119.9	118.9	124.3	353.1	78.75	227.1	51.29	72.04	

This table presents the Diebold-Mariano test statistics for the best-performing models, as determined in Section 4.1, 4.3.1, 4.3.2 and 4.3.3. The parametric model is the Kou jump-diffusion model, denoted by Kou. All other models are non-parametric models. Non-parametric models whose abbreviation starts with an H are hybrid models. For the hybrid models, -K (-VG) denotes that the input of the model is adjusted by the option prices obtained from the Kou (Variance Gamma) model. The integer behind each non-parametric model abbreviation represents the number of input variables. The null hypothesis that there is no difference in the MSE of the models is tested.

4.5 Model Confidence Set

Because a comprehensive conclusion does not ensue from the Diebold-Mariano tests, the statistical significance of the difference in the out-of-sample forecast accuracy per moneyness and maturity region is tested by means of the model confidence set. If the entire data set is once more examined the model confidence set comes to the same conclusion as the Diebold-Mariano test: the set consists solely of the MNN with three input parameters if the loss differential is measured by the MAPE and solely of the HGNN-VG with three input parameters if the loss differential is measured by the MSE. However, the model confidence set can also easily be computed per moneyness and maturity region. This leads to more insightful results.

Table 13: Model confidence set per moneyness and maturity region

	OTM			ATM			ITM		
	short	medium	long	short	medium	long	short	medium	long
Panel A: MAPE	MNN3	MNN3	MNN3	MNN3 MCMNN	MNN3	MNN3	HMCMNN-K	HANN-VG3 HMNN-VG3 HMCMNN-K	HANN-K2 HGNN-K2 HMCMNN-K
Panel B: MSE	MNN3	MNN3	MNN3	MNN3	MNN3	MNN3	HMCMNN-K	HANN-VG3 HMNN-VG3 HGNN-VG3	HGNN-K2 HGNN-VG3
Panel C: %	2	5	12	8	6	7	15	16	29

This table presents the model confidence set for each moneyness and maturity region combination. The model confidence set contains the best model(s) with a level of confidence of 90%, as $\alpha = 0.1$. Each moneyness and maturity region is given on the horizontal axis, whereas the models for each region are denoted vertically. The regions are based on the MNN cut-off points. The order in which the models are presented vertically is not of importance. Non-parametric models whose abbreviation starts with an H are hybrid models. For the hybrid models, -K (-VG) denotes that the input of the model is adjusted by the option prices obtained from the Kou (Variance Gamma) model. The integer behind each non-parametric model abbreviation represents the number of input variables. The null hypothesis that there is no difference in the MAPE and MSE is tested in respectively Panel A and Panel B. Panel C indicates the percentage of options belonging to each region.

Table 13 shows the model confidence set for $\alpha = 0.1$ per moneyness and maturity region based on the option price forecasts of all parametric models, neural network models and hybrid neural network models introduced in this paper. First of all, it is evident that the MNN with three input parameters significantly outperforms all other models in terms of pricing out-of-the-money and at-the-money options. The model confidence set for these options consists almost exclusively of this model for all maturity regions. This indicates that implementing modularity based on fixed manual heuristics is sufficient to accurately price out-of-the-money and at-the-money options. Additionally, no parametric model is able to improve the pricing performance of the MNN for these options through the hybrid approach.

However, the out-of-the-money and at-the-money options comprise only roughly 40% of the entire data set, as indicated by Panel C of Table 13. It is therefore essential to scrutinize the model confidence sets of the in-the-money options. All of the model confidence sets for the in-the-money options consist entirely of hybrid neural networks, indicating that the applying the hybrid approach to ANNs, MNNs, MCMNNs and GNNs improves their pricing performance of in-the-money options. The HMCMNN-K model superiorly prices in-the-money options with short-term maturities. For medium-term and long-term maturities the model confidence set consists of multiple hybrid neural network models. For the in-the-money options with medium-term maturities only models with three input parameters are present in the model confidence set based on both performance metrics. For the in-the-money options with long-term maturities, the region to which most options belong, the hybrid GNN is the only model present in both the model confidence sets.

Overall, the model confidence sets reveal that for out-of-the-money and at-the-money options neural network models that introduce modularity based on fixed manual heuristics suffice to attain a superior option pricing performance. For these options no complex hybrid approach or dynamic option grouping is needed. However, these option comprise only roughly 40% of the entire data set. To properly price the majority of the data set, consisting of in-the-money options, applying the hybrid approach has been proved necessary to significantly outperform all introduced parametric and standard neural network benchmark models. For in-the-money options with medium-term maturities hybrids neural network models with additional input parameters attain the best option pricing performance. Finally, for in-the-money options with long-term maturities, hybrid gated neural networks models are preferred. This indicates that for the region to which the majority of options belong hybrid neural networks that introduce modularity based on dynamic option grouping are required to attain the most accurate option price forecasts.

4.6 Delta-Hedged Trading Strategy

In this section the economic significance of the difference in pricing accuracy is tested by means of a delta-hedged trading strategy based on the option price forecasts. After all, options are mainly priced for trading purposes. Furthermore, no all-encompassing conclusion can be drawn from the two methods for testing the statistical significance of the difference in pricing accuracy; the Diebold-Mariano test finds different significantly best-performing models for different performance metrics and the model confidence sets vary by moneyness and maturity region. The trading performance of the same best-performing models as in Section 4.4 is examined.

Table 14 shows that the delta-hedge trading strategy based on the option price forecasts of the MNN with three input parameters attains the highest mean return per invested dollar. In comparison, a buy and hold strategy in which \$1000 is invested daily in the S&P500 index yields a mean return per invested dollar of 0.03%. All models thus outperform this benchmark buy and hold strategy. The MNN and MCMNN attain the highest mean return per invested dollar. This performance can likely be attributed to the accurate pricing of out-of-the-money and at-the-money options; the MNN significantly outperforms all other models in these moneyness regions as indicated by the model confidence sets. Although these options comprise only roughly 40% of the data set it is evident that they are of high importance for determining the trading performance of the models. The poor option pricing performance of the MNN for deep-in-the-money options in terms of MSE apparently does not considerably harm the trading performance of the MNN. This can be due to the fact that the trading strategy is based solely on the direction of the forecasts.

Table 14: Outcomes of the implementation of a delta-hedged trading strategy

	Kou	HANN-K2	HANN-VG3	HMNN-K2	MNN3	HMNN-VG3	MCMNN	HMCMNN-K	HGNN-K2	HGNN-VG3
μ (%)	1.785	1.164	1.685	1.256	4.413	1.229	2.847	1.550	1.716	1.975
σ (%)	22.86	17.67	28.56	17.71	61.38	19.63	35.37	29.44	22.19	27.06
t	5.564	4.693	4.204	5.053	5.123	4.461	5.736	3.753	5.511	5.201
SR	4.092	1.777	2.925	2.293	5.805	1.932	5.647	2.380	3.905	4.159
buy	907823	846538	789488	853874	757143	778231	751441	766645	838137	793729
sell	699957	743199	791424	736014	811385	796567	775192	803909	751437	785522

This table presents the Diebold-Mariano test statistics for the best-performing models, as determined in Section 4.1, 4.3.1, 4.3.2 and 4.3.3. The parametric model is the Kou jump-diffusion model, denoted by Kou. All other models are non-parametric models. Non-parametric models whose abbreviation starts with an H are hybrid models. For the hybrid models, -K (-VG) denotes that the input of the model is adjusted by the option prices obtained from the Kou (Variance Gamma) model. The integer behind each non-parametric model abbreviation represents the number of input variables. On the vertical axis, μ and σ denote the mean and standard deviation of the return per invested dollar. Both are expressed in percentages. Furthermore, t indicates the t-statistic and SP the Sharpe ratio. Finally, buy and sell denote the number of options bought and sold.

The t-statistics confirm the significance of abnormal returns for all models. Although all hybrid models are outperformed by the MNN and MCMNN, the significance of their abnormal returns indicates that their performance must not be neglected. Because the trading strategy runs a considerable risk, the risk-adjusted returns (indicated by the Sharpe ratio) are also examined. As for the mean returns, the MNN with three input parameters records the highest risk-adjusted return. The MCMNN and the HGNN-VG with three input parameters respectively attain the second-highest and third-highest mean return per invested dollar and Sharpe ratio. It is noteworthy that the two best-performing models in terms of mean return and mean risk-adjusted return execute the least number of combined buy and sell orders. This conveys that the MNN and MCMNN price related options within a 1% interval more often than all other models.

Overall, in terms of trading performance the models with higher accuracy, indicated by their MAPE, thus outperform the models with less excessive errors. These models mainly achieve their optimal trading performance by accurately pricing out-of-the-money and at-the-money options and by executing the least number of orders. Finally, it must be remarked that no transaction costs are incorporated. Doing so can potentially influence the magnitude of the returns.

5 Conclusion

The efforts of this paper have focused on improving the option pricing performance of neural networks by introducing a hybrid approach, an optimal model architecture, additional input parameters and modularity to neural network models. To assess the option pricing performance of all the models, daily S&P 500 index European call options are priced. Parametric option pricing models from Black and Scholes (1973), Corrado and Su (1996), Heston (1993), Kou (2002) and Madan et al. (1998) serve as benchmark models.

Simultaneously, the output of these five parametric models is employed to adjust the target function of the hybrid neural network models. Especially the option pricing performance of the exponential Lévy models proves useful. The best performing parametric option pricing model is the Kou jump-diffusion model; this model slightly outperforms the Variance Gamma model in terms of MAPE and MSE. For a data set of 20 years that alternates between regular and crisis periods, the finite jumps of the Kou jump-diffusion model appear to be more adequately fitting than the infinitely many jumps of the Variance Gamma model. However, the performance of the Variance Gamma model trumps the performance of the Kou jump-diffusion model for options with high volatilities specifically.

The optimal model architecture of the non-parametric models is first analyzed based on the option pricing performance of ANNs for several numbers of hidden layers and hidden layer nodes. In general the neural networks with three hidden layers convincingly outperform all other models. Furthermore, the constructed optimal architecture of nodes works best in combination with the linear activation function for the output layer and the ReLU activation function for the hidden layers. The optimal model architecture is implemented by all the neural network models. This way all the neural networks that are examined have an architecture that has been proved to work well for the specific neural network and simultaneously no arguments can be brought up about differences in performance due to a different model architecture.

Most importantly, the obtained option prices from the exponential Lévy models aid the hybrid neural networks in consistently outperforming the standard approach in terms of MSE, as can be observed in Panel B of the comprehensive overview of results in Table 18 in Appendix G. In most cases the hybrid approach in combination with an exponential Lévy model also attains more accurate option pricing forecasts than the standard approach, as for example demonstrated by the MAPEs obtained by the HGNNs adjusted by the Kou and Variance Gamma model option prices. The main contribution of this paper therefore consists of showing that allowing the neural networks to augment the option pricing performance of exponential Lévy models generally decreases the amount and magnitude of excessive option price forecast errors.

Another insightful conclusion from the option price forecasts of the non-parametric models is that each model with three input parameters outperforms its counterpart with two input parameters in terms of MAPE. This statement holds for all neural network models introduced in this paper. The extra information contained in the additional input parameters thus enables the neural networks to achieve a better understanding of the option pricing function. Because the extra information of two variables—the risk-free rate and the historical volatility—is encapsulated in a single additional input parameters, inference is still easily attainable for the neural networks.

Aside from analyzing the benefits of incorporating a hybrid approach and additional input parameters, this paper is devoted to investigating the benefits of introducing modularity to neural network for option pricing. The MNN introduces modularity based on fixed manual heuristics. The MCMNN then extends the standard MNN by adding dynamic volatility modules. Finally, the GNN incorporates modularity with option grouping that is automatic and learned from data. Based on the option price forecasts the MNN has the most accurate option pricing, whereas the option price forecasts of the hybrid GNN contain the least excessive errors. In general, I can conclude from the option price forecasts that introducing modularity to neural network for option pricing is beneficial for the option pricing performance.

The MNN with three input parameters attains the lowest MAPE of all the models. However, due to the severe misestimation of deep-in-the-money options, the MSE of the option price forecasts of the MNN with three input parameters is outperformed by numerous models. Only four models are able to outperform all benchmark models in terms of both MAPE and MSE and all of these models are GNN or HGNN models. In addition to incorporating modularity, the GNNs also encodes prior information based on financial axioms as constraints into its modules. I therefore conclude that incorporating modularity by means of dynamic option grouping in combination with a Bayesian-alike design approach enables the GNN to consistently outperform advanced parametric and non-parametric models as measured by multiple performance metrics. Based on the option pricing performance of the HGNN I also conclude that applying a hybrid approach—specifically an exponential Lévy model in appropriate combination with a GNN—further improves the option pricing performance of the GNN.

In this paper I employ both the MAPE and MSE performance metrics to ensure that models are capable of producing accurate option price forecast while not making any excessive errors. However, working with two different performance metrics leads to the unavoidable but unsatisfactory conclusion that different models are favored by each performance metric. Specifically, the Diebold-Mariano test concludes that based on the MAPE of the option price forecasts the MNN with three input parameters significantly outperforms all other models, whereas based on the MSE of the option price forecasts the hybrid GNN with three input parameters adjusted by the Variance Gamma model significantly outperforms all other models. The model confidence sets reveal that for out-of-the-money and at-the-money options neural network models that introduce modularity based on fixed manual heuristics suffice to attain a superior option pricing performance. To properly price in-the-money options the hybrid approach proves necessary to significantly outperform all introduced parametric and standard neural network benchmark models. Combining multiple neural network methods is thus advised to accurately price the entire moneyness grid.

To be able to draw an all-encompassing conclusion the economic significance of the difference in pricing accuracy is tested by means of a delta-hedged trading strategy based on the option price forecasts. This final test reveals that the accurate forecasts of the MNN with three input parameters yield a better mean return per invested dollar and Sharpe ratio than the forecasts of any other model. From the trading performance I conclude that forecasts from neural networks that augment the option pricing performance of exponential Lévy models and implement modularity add significant economic value. For trading purposes modularity based on fixed manual heuristics is favored over modularity based on dynamic option grouping.

6 Discussion and Further Research

Although I conclude that introducing a hybrid approach and modularity significantly improves the option pricing and hedging performance of neural networks, certain remarks must be made regarding the analyses presented. First of all, for the Black-Scholes model and the Corrado-Su model 60-day historical estimates for the volatility, skewness and kurtosis parameters are used. The poor performance of these two models and the hybrid models adjusted by these models can be explained by the relatively long 60-day horizon which causes a significant delay in information processing. Option pricing of these models can possibly be improved by introducing implied parameters, as shown by Andreou et al. (2008). For the parametric models of which the characteristic function is known options are priced using fractional FFT. For these models parameters are calibrated on the last training day's data using the Levenberg-Marquardt algorithm. These models are thus given an unfair advantage by being optimized daily. Furthermore, the parameters of the parametric models at times rise or fall abruptly and rapidly because few parameters restrictions are implemented. Further research into an optimal and fair fitting of the parametric model parameters and the corresponding option pricing is therefore recommended.

Several comments must be made concerning the employed neural network methods. First of all, I exclusively implement the multilayer perceptron approach. Generally, the multilayer perceptron approach is considered to be a simple deep learning technique. More complex techniques that can be implemented for option pricing are convolutional neural networks and recurrent neural networks. Recent analyses have also demonstrated that deep reinforcement learning is a suitable deep learning technique for option pricing, see e.g. Kolm and Ritter (2019) and Carbonneau and Godin (2020). I recommend researching introducing modularity and the hybrid approach to models that incorporate more complex deep learning techniques.

The optimal model architecture of the neural network models is determined based on the performance of the ANN with two input parameters for various numbers of hidden layers, hidden layer nodes and acti-

vation functions. I then implement this optimal model architecture for the ANNs, MNNs and MCMNNs and their hybrid counterparts. This is done such that the contrast between the two approaches can be adequately analyzed. I thus assume that the optimal model architecture found for the ANN with two input parameters holds for all other models. Because this simplifying assumption can most likely be easily debunked, further research into the optimal model architecture of neural networks that introduce modularity and neural networks with additional input parameters is therefore recommended. Additionally, more sophisticated techniques for conducting research into the optimal model architecture are recommended, such as Bayesian optimization techniques to optimize model architectures. Finally, the fact that the optimal model architecture is based on the performance on the whole data set possibly induces a data mining bias.

For the neural networks the data for each year are divided into four quarters consisting of three months of data, split chronologically. The training set consists of two quarters of data, whereas the validation set and the out-of-sample testing set consist of one quarter of data. However, Yang et al. (2017) only utilized five days of data to train the neural networks. They stated that "feeding multiple days' data implicitly assumes the market structure is stable in those days, but that this is likely to be violated as the number of days grows, thereby introducing a domain-shift problem." For computational purposes I do not implement this approach, but I recommend future researchers to do so. This approach also possibly solves the unfair advantage of the parametric models, whose parameters are calibrated daily.

Finally, I implement a simplified version of a delta-hedged trading strategy in this paper. The analysis of the economic significance of the difference in pricing accuracy can be improved by incorporating transaction costs. Furthermore, the trading strategy can be expanded by introducing modularity. This can for example be implemented by applying a different trading strategy per moneyness and time-to-maturity region.

Although this section discusses several limitations of the undertaken research I deem the main results relevant for practitioners seeking to introduce the hybrid approach and modularity to neural networks for option pricing and hedging. My findings indicate the importance of considering exponential Lévy models for the hybrid neural networks and considering modularity with dynamic option grouping to attain an accurate and consistent option pricing performance. Given the vast amount of unexplored possibilities of refinement and extensions I believe there exist ways for further research and improvements in implementing the hybrid approach and modularity to forecast option prices with neural networks. I highly recommend further researching neural networks that implement different modularity approaches and hybrid approaches based on the features of the option contract.

References

Albrecher, H., Mayer, P., Schoutens, W., and Tistaert, J. (2007). The little Heston trap. *Wilmott Magazine*, (1):83–92.

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. (2002). Critical assessment of option pricing methods using artificial neural networks. In *Lecture Notes in Computer Science*, volume 2415 LNCS, pages 1131–1136.

Andreou, P. C., Charalambous, C., and Martzoukos, S. H. (2006). Robust artificial neural networks for pricing of European options. *Computational Economics*, 27(2-3):329–351.

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.

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

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

Bernales, A. and Guidolin, M. (2014). Can we forecast the implied volatility surface dynamics of equity options? Predictability and economic value tests. *Journal of Banking & Finance*, 46:326–342.

Black, F. (1975). Fact and fantasy in the use of options. *Financial Analysts Journal*, 31(4):36–41.

Black, F. and Scholes, M. (1973). The pricing of options and corporate liabilities. *Journal of Political Economy*, 81(3):637–657.

Boek, C., Lajbcygier, P., Palaniswami, M., and Flitman, A. (1995). Hybrid neural network approach to the pricing of options. In *IEEE International Conference on Neural Networks - Conference Proceedings*, volume 2, pages 813–817.

Brown, C. A. and Robinson, D. M. (2002). Skewness and kurtosis implied by option prices: A correction. *Journal of Financial Research*, 25(2):279–282.

Carbonneau, A. and Godin, F. (2020). Equal risk option pricing with deep reinforcement learning. *arXiv preprint arXiv:2002.08492*.

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

Chen, F. and Sutcliffe, C. (2012). Pricing and hedging short sterling options using neural networks. *Intelligent Systems in Accounting, Finance and Management*, 19(2):128–149.

Chourdakis, K. (2004). Option pricing using the fractional FFT. *The Journal of Computational Finance*, 8(2):1–18.

Clevert, D.-A., Unterthiner, T., and Hochreiter, S. (2015). Fast and accurate deep network learning by exponential linear units (elus). *arXiv preprint arXiv:1511.07289*.

Cont, R. and Tankov, P. (2002). Calibration of jump-diffusion option pricing models: a robust non-parametric approach.

Cooley, J. W. and Tukey, J. W. (1965). An algorithm for the machine calculation of complex Fourier series. *Mathematics of computation*, 19(90):297–301.

Corrado, C. J. and Su, T. (1996). Skewness and kurtosis in SP 500 index returns implied by option prices. *Journal of Financial Research*, 19(2):175–192.

Cox, J. C., Ingersoll, J. E., and Ross, S. A. (1985). A Theory of the Term Structure of Interest Rates. *Econometrica*, 53(2):385–407.

Crisóstomo, R. (2018). Speed and biases of Fourier-based pricing choices: a numerical analysis. *International Journal of Computer Mathematics*, 95(8):1565–1582.

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

Cybenko, G. (1989). Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*, 2(4):303–314.

Diebold, F. and Mariano, R. (1995). Comparing Predictive Accuracy. *Journal of Business & Economic Statistics*, 13(3):253–263.

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

Föllmer, H. and Schied, A. (2011). *Stochastic finance: an introduction in discrete time*. Walter de Gruyter.

Garcia, R. and Gençay, 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 Salih, A. (2003). Degree of mispricing with the Black-Scholes model and nonparametric cures. *Economics and Finance. Annals*, 4:73–101.

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

Gradojevic, N. (2016). Multi-criteria classification for pricing European options. *Studies in Nonlinear Dynamics & Econometrics*, 20(2):123–139.

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

Hansen, P. R., Lunde, A., and Nason, J. M. (2011). The model confidence set. *Econometrica*, 79(2):453–497.

Harvey, D., Leybourne, S., and Newbold, P. (1997). Testing the equality of prediction mean squared errors. *International Journal of Forecasting*, 13(2):281–291.

Heaton, J. (2008). *Introduction to neural networks with Java*. Heaton Research, Inc.

Heston, S. L. (1993). A closed-form solution for options with stochastic volatility with applications to bond and currency options. *Review of Financial Studies*, 6(2):327–343.

Hochreiter, S., Bengio, Y., Frasconi, P., and Schmidhuber, J. (2001). Gradient flow in recurrent nets: the difficulty of learning long-term dependencies.

Hull, John, C. (2008). *Options, Futures, and other Derivatives*. Pearson Education, USA., 7th edition.

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

Jeanblanc, M., Yor, M., and Chesney, M. (2009). *Mathematical methods for financial markets*. Springer.

Kahl, C. and Jäckel, P. (2005). Not-so-complex logarithms in the Heston model. *Wilmott magazine*, 19(9):94–103.

Kingma, D. P. and Ba, J. L. (2015). Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*. ICLR.

Kolm, P. N. and Ritter, G. (2019). Dynamic replication and hedging: A reinforcement learning approach. *The Journal of Financial Data Science*, 1(1):159–171.

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

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Lajbcygier, P. (2004). Improving option pricing with the product constrained hybrid neural network. *IEEE Transactions on Neural Networks*, 15(2):465–476.

Lajbcygier, P. R. and Conner, J. T. (1997). Improved option pricing using bootstrap methods. In *IEEE International Conference on Neural Networks - Conference Proceedings*, volume 4, pages 2193–2197.

Madan, D. B., Carr, P. P., and Chang, E. C. (1998). The variance Gamma process and option pricing. *Review of Finance*, 2(1):79–105.

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

Merton, R. C. (1976). Option pricing when underlying stock returns are discontinuous. *Journal of Financial Economics*, 3(1-2):125–144.

Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814.

Palmer, S. (2019). Evolutionary Algorithms and Computational Methods for Derivatives Pricing. *Doctoral thesis, UCL (University College London)*.

Ruf, J. and Wang, W. (2019). Neural Networks for Option Pricing and Hedging: A Literature Review. *SSRN Electronic Journal*.

Schoutens, W., Simons, E., and Tistaert, J. (2004). A perfect calibration! Now what? *Wilmott Magazine*, pages 66–78.

Sigaud, O., Masson, C., Filliat, D., and Stulp, F. (2015). Gated networks: an inventory. *arXiv preprint arXiv:1512.03201*.

Tankov, P. and Voltchkova, E. (2009). Jump-diffusion models: a practitioner's guide. *Banque et Marchés*, 99(1):24.

Yang, Y., Zheng, Y., and Hospedales, T. M. (2017). Gated neural networks for option pricing: rationality by design. In *Association for the Advancement of Artificial Intelligence*, pages 52–58.

Zheng, Y. (2018). Machine Learning and Option Implied Information. *Doctoral Thesis*.

Zheng, Y., Yang, Y., and Chen, B. (2019). Gated deep neural networks for implied volatility surfaces. *arXiv preprint arXiv:1904.12834*.

Appendices

A Parametric Models

This section introduces the parametric models. The contribution of the parametric models is twofold. First of all, they serve as benchmark models to which the performance of the neural network models is compared. Additionally, they are used as inputs for the target function of the hybrid neural network models.

A.1 The Black-Scholes Model

The price of a European call option, using the notation by Hull (2008), that follows from the Black and Scholes (1973) model, is

$$c^{\text{BS}} = S_0 \mathrm{e}^{-q\tau} N(d_1) - K \mathrm{e}^{-r\tau} N(d_2), \tag{21}$$

with

$$d_1 = \frac{\log \frac{S_0}{K} + (r - q + \frac{\sigma^2}{2})\tau}{\sigma\sqrt{\tau}} \tag{22}$$

and

$$d_2 = d_1 - \sigma\sqrt{\tau}. \tag{23}$$

Here, c^{BS} is the market price to be charged for the European call option, S_0 is the current spot price of the underlying asset, $N(\cdot)$ is the cumulative distribution function of the standard normal distribution, τ is the time-to-maturity, K is the strike price of the option, r is the risk-free interest rate, q is the dividend yield paid by the underlying asset and σ is the volatility of the stock. This option pricing formula relies on the assumption that the underlying asset follows a geometric Brownian motion with constant drift and volatility, that is

$$\mathrm{d}S = \mu S \mathrm{d}t + \sigma S \mathrm{d}W, \tag{24}$$

where W is a Wiener process or Brownian motion, and μ and σ are constants. Using Itô's lemma, considering a delta-hedged portfolio and simplifying equations, Black and Scholes arrive at their partial differential equation, given by

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} = rV - (r - q)S\frac{\partial V}{\partial S}, \tag{25}$$

where V is the price of the derivative. Solving this partial differential equation numerically gives (21).

Since (25), and thus (21), follows from a set of simplified assumptions, such as efficient markets, constant volatility and normally distributed asset prices, the Black-Scholes model has several well-documented biases, see e.g. Black (1975) and Bakshi et al. (1997). Each of the following parametric models that is introduced extends the Black-Scholes model by accounting for some of these biases.

A.2 The Corrado-Su Model

Corrado and Su (1996) stated that the Black-Scholes biases are caused by the violation of the assumption that asset prices are normally distributed. They accounted for this bias by expanding the normal density function with third and fourth moments. This way they allowed for non-normal skewness and kurtosis in the returns distribution, thereby removing systematic strike price biases. Brown and Robinson (2002) noted that the proposed model contained an economically significant error and proposed a correction. The price of a European call option that follows from Corrado-Su the model, with correction, is

$$c^{\mathrm{CS}} = c^{\mathrm{BS}} + \mu_3 Q_3 + (\mu_4 - 3)Q_4, \tag{26}$$

with

$$\begin{aligned} Q_3 &= \frac{1}{3!} S_0 \mathrm{e}^{-q\tau} \sigma\sqrt{\tau} \left((2\sigma\sqrt{\tau} - d_1)\phi(d_1) + \sigma^2 \tau N(d_1) \right), \\ Q_4 &= \frac{1}{4!} S_0 \mathrm{e}^{-q\tau} \sigma\sqrt{\tau} \left(\left(d_1^2 - 1 - 3\sigma\sqrt{\tau}(d_1 - \sigma\sqrt{\tau}) \right) \phi(d_1) + \sigma^3 \tau^{\frac{3}{2}} N(d_1) \right), \\ \phi(z) &= \frac{1}{\sqrt{2\pi}} \mathrm{e}^{\frac{-z^2}{2}}. \end{aligned} \tag{27}$$

Here, μ_3 and μ_4 are the coefficients of skewness and kurtosis implied by option prices.

A.3 The Heston Model

Heston (1993) introduced a second Brownian motion and revised (24) to account for the fact that the Black-Scholes model cannot adequately capture the observed fact that volatility starts to fluctuate when new information enters the market. The Black-Scholes model cannot capture this feat because it relies on a constant volatility assumption. The Heston model assumes that volatility changes stochastically over time. Models with a second Brownian motion to describe the volatility fluctuation are therefore called stochastic volatility models. The Heston model, one of the most notable stochastic volatility models, also allows for correlation between the underlying and volatility. Finally, the volatility follows a mean-reverting

square root process to model the interest rate, as first proposed by Cox et al. (1985). The model is defined by the system of stochastic differential equations

$$\begin{aligned} \mathrm{d}S_t &= \mu S_t \mathrm{d}t + \sqrt{v_t} S_t \mathrm{d}W_t^{(1)}, \\ \mathrm{d}v_t &= \kappa(\theta - v_t)\mathrm{d}t + \sigma\sqrt{v_t}\mathrm{d}W_t^{(2)}, \\ \mathrm{d}W_t^{(1)}\mathrm{d}W_t^{(2)} &= \rho \mathrm{d}t, \end{aligned} \tag{28}$$

where S_t is the underlying price, μ is the rate of return, v_t is the variance of the underlying price, θ is the long-term variance, κ is the mean-reversion rate, σ is the volatility of the volatility and ρ is the correlation between the Brownian motions that drive the underlying price and its variance. Henceforth, $\boldsymbol{\xi} := \begin{bmatrix} \theta & \kappa & \sigma & \rho \end{bmatrix}$.

For a spot price S_0, interest rate r and dividend yield q, the formula for pricing of a European call option that follows from the stochastic differential equations becomes

$$\begin{aligned} C(\boldsymbol{\xi}; K, \tau) =& \frac{1}{2}(S_0 - \mathrm{e}^{-(r-q)\tau}K) + \frac{\mathrm{e}^{-r\tau}}{\pi}\int_0^\infty \mathrm{Re}\left(\frac{\mathrm{e}^{iu\log K}}{iu}\phi(\boldsymbol{\xi}; u-i, \tau)\right)\mathrm{d}u \\ &- \frac{\mathrm{e}^{-r\tau}K}{\pi}\int_0^\infty \mathrm{Re}\left(\frac{\mathrm{e}^{iu\log K}}{iu}\phi(\boldsymbol{\xi}; u, \tau)\right)\mathrm{d}u, \end{aligned} \tag{29}$$

where i is the imaginary unit and $\phi(\cdot)$ is the characteristic function. The characteristic function of a random variable X is defined as $\phi_X(u,t) \equiv \mathrm{E}\left[\mathrm{e}^{iuX}\right]$. The characteristic function of the logarithm of the stock price process, $\phi_S(\boldsymbol{\theta}; u, t)$, proposed by Heston causes discontinuities to appear for moderate to long maturities because of the branch switching of the complex power function (Kahl and Jäckel, 2005). A slight alteration, originally proposed by Schoutens et al. (2004), has been shown by Albrecher et al. (2007) to be continuous and to give numerically stable prices. This characteristic function is given by

$$\phi_S(\boldsymbol{\theta}; u, t) = \exp\left\{iu(\log S_0 + (r-q)t) + \frac{\kappa\theta}{\sigma^2}\left[(\xi - d)t - 2\log\frac{1 - g_2\mathrm{e}^{-dt}}{1-g_2}\right] + \frac{v_0}{\sigma^2}(\xi - d)\frac{1-\mathrm{e}^{-dt}}{1-g_2\mathrm{e}^{-dt}}\right\}, \tag{30}$$

where

$$\begin{aligned} \xi &:= \kappa - \sigma\rho i u, \\ d &:= \sqrt{\xi^2 + \sigma^2(u^2 + iu)}, \\ g_2 &:= \frac{\xi - d}{\xi + d}. \end{aligned} \tag{31}$$

Although the Heston formula for pricing a European call option is not as analytically tractable as those previously seen for the Black-Scholes or Corrado-Su models, knowing the characteristic function of the logarithm of the stock price process is sufficient to calculate option prices for the Heston model. Aside from his seminal stochastic volatility model, Heston (1993) also introduced the Fourier transform approach to option pricing. The Fourier transform approach is applicable when only the characteristic function is known. This is explained by the fact that the probability density function of a distribution can be computed by taking the inverse Fourier transform of this characteristic function. This approach has been shown to be substantially more efficient than other methods, such as finite difference solutions to partial differential equations or integro-differential equations, for pricing options under general stochastic volatility processes. Therefore, Heston refers to solutions of this approach as closed-form solutions.

The fast Fourier transform (FFT) algorithm, introduced by Cooley and Tukey (1965) is an algorithm designed to efficiently compute Fourier transforms. Carr and Madan (1999) were the first to develop an application of the FFT for option pricing. An advantage of this algorithm is that it "exploits periodicities and symmetries in the characteristic function evaluations to reduce the number of operations" (Crisóstomo, 2018). Furthermore, it enables the simultaneous calculation of prices for a range of strikes. Carr and Madan (1999) concluded that using FFT is considerably faster and more accurate than most available methods, such as the approach described by Heston. Chourdakis (2004) showed that the fractional FFT algorithm can also be used to price options using solely the characteristic function information. Fractional FFT uses the information of the characteristic function in an even more efficient manner than regular FFT, thus needing fewer function evaluations. Consequently, fractional FFT generally outperforms regular FFT in terms of computation time. Similar to Yang et al. (2017), I therefore opt to use fractional FFT for the parametric option pricing models of which the characteristic function is known.

A.4 The Kou Jump-Diffusion Model

As can be observed in stock price data, jumps often occur in stock prices due to unpredictable events. Stock price data therefore often exhibits discontinuity. Consequently, another source of bias of the Black-Scholes model is the assumption that the underlying asset follows a geometric Brownian motion, as the geometric Brownian motion path is continuous. Exponential Lévy models generalize the Black-Scholes model by allowing stock prices to jump.

There are two categories of parametric exponential Lévy models. The first category consists of jump-diffusion models. In jump-diffusion models the price evolution is given by a diffusion process, disrupted

by a finite number of jumps at random intervals. The jumps represent events such as breaking news and corresponding crashes. The two building blocks of a jump-diffusion model are the Poisson process (for the jumps) and the Brownian motion (for the diffusion). The price evolution is given by a Lévy process consisting of a Gaussian part and a jump part with a finite number of jumps, i.e.

$$X_t = \mu t + \sigma W_t + \sum_{i=1}^{N(t)} Y_i \tag{32}$$

where W_t is a standard Brownian motion and $N(t)$ is a Poisson process with jump intensity λ. The set $\{Y_i\}$ is a sequence of independent identically distributed non-negative random variables.

The Lévy process is used to model stock prices, according to

$$S_t = S_0 \mathrm{e}^{X_t}, \tag{33}$$

to ensure positive, independent and stationary log-returns. The process evolves like a geometric Brownian motion in between jumps. Equations (32) and (33) can be merged as

$$\frac{\mathrm{d}S_t}{S_{t^-}} = \mu \mathrm{d}t + \sigma \mathrm{d}W_t + \mathrm{d}\left(\sum_{j=1}^{N(t)} \mathrm{e}^{Y_i} - 1 \right), \tag{34}$$

where the notation S_{t^-} implies that the process's value prior to the jump is employed in case of a jump. To simplify notation for option pricing, similar to Cont and Tankov (2002) and Tankov and Voltchkova (2009) the interest rate is contained in the construction of the exponential Lévy model:

$$S_t = S_0 \mathrm{e}^{rt + X_t}. \tag{35}$$

Merton (1976), who was the first to propose a jump-diffusion model for option pricing, assumed the jumps Y_i to be normally distributed. Kou (2002) extended this model by suggesting the jumps to follow a double exponential distribution. The density of a double exponential distribution is

$$f_Y(y) = p\eta_1 \mathrm{e}^{-\eta_1 y} \mathbb{1}_{y \geq 0} + q\eta_2 \mathrm{e}^{\eta_2 y} \mathbb{1}_{y<0}, \tag{36}$$

where η_1 and η_2 are the expected positive and negative jump sizes respectively, $\eta_1 > 1$, $\eta_2 > 0$, $p, q \geq 0$, $p + q = 1$. In (36), p and q represent the probabilities of upward and downward jumps. In the model, all

sources of randomness are assumed to be independent. Henceforth, $\boldsymbol{\gamma} := \begin{bmatrix}\sigma & \lambda & \eta_1 & \eta_2 & p\end{bmatrix}$.

For jump-diffusion models, the characteristic function of X_t is given by the Lévy–Khintchine formula:

$$\phi_X(u,t) = \exp\left\{t\left(i\mu u - \frac{\sigma^2 u^2}{2} + \int_{-\infty}^{\infty}(\mathrm{e}^{iuy}-1)\lambda f_Y(y)\mathrm{d}y\right)\right\} \tag{37}$$

For the Kou jump-diffusion model, the characteristic function of X_t can be derived analytically. The characteristic function that emerges, as for example shown in Jeanblanc et al. (2009), is given by

$$\phi_X(\boldsymbol{\gamma};u,t) = \exp\left\{t\left(i\mu u - \frac{\sigma^2 u^2}{2} + \lambda\left(\frac{p\eta_1}{\eta_1 - iu} - \frac{q\eta_2}{\eta_2 + iu} - 1\right)\right)\right\}. \tag{38}$$

Because the characteristic function of X_t is known, the characteristic function of the logarithm of the stock price process can be deduced from (35). Fractional FFT can then be used to retrieve option prices.

A.5 The Variance Gamma Model

In the previous section, the existence of two categories of parametric exponential Lévy models was mentioned and the first category was described. The second category consists of Lévy processes with infinitely many jumps in every interval, called infinite activity models. In general, the jumps are smaller than the jumps in jump-diffusion models, but the jumps occur much more frequently.

The simplest example of an infinite activity model is the gamma process. Like the Poisson process and the Brownian motion for the jump-diffusion models, the gamma process is the building block for infinite activity models. The gamma process $\gamma_t(\mu,\nu)$ with mean rate μ and variance rate ν is a process with independent and stationary gamma distributed increments over non-overlapping intervals of time. The marginal distribution of the increment $x = \gamma_{t+h}(\mu,\nu) - \gamma_t(\mu,\nu)$ is the gamma density function with mean μx and variance νx, given by

$$f_h(x) = \frac{x^{\frac{\mu^2 h}{\nu}-1}\left(\frac{\mu}{\nu}\right)^{\frac{\mu^2 h}{\nu}}\exp\left(\frac{-\mu x}{\nu}\right)}{\Gamma\left(\frac{\mu^2 h}{\nu}\right)}, \tag{39}$$

where $\Gamma(\cdot)$ is the gamma function. The gamma process is a pure-jump increasing Lévy process.

A more complex infinite activity model, the Variance Gamma (VG) model of Madan et al. (1998), is constructed by time-changing a Brownian motion with drift with a gamma process with unit mean rate, $\gamma_t(1,\nu)$. Specifically,

$$X_t = \theta\gamma_t(1,\nu) + \sigma W_{\gamma_t(1,\nu)}, \tag{40}$$

where θ is the drift of the Brownian motion, σ is the volatility of the Brownian motion and ν is the variance rate of the gamma time change. Madan et al. (1998) asserted that the process "provides two dimensions of control on the distribution over and above that of the volatility", as control over the skewness is attained via θ and control over the kurtosis via ν. The VG process acquires the feature of an infinite arrival rate of stock price jumps from the gamma process and is therefore also regarded as an infinite activity model.

Similar to the Kou jump-diffusion model, the characteristic function of the log of the price process can be deduced from (35) if the characteristic function of X_t is known. According to Madan et al. (1998) the characteristic function of X_t is given by

$$\phi_X(\boldsymbol{\zeta}; u, t) = \left(1 - i\theta\nu u + \frac{1}{2}\sigma^2\nu u^2\right)^{-\frac{t}{\nu}} \tag{41}$$

where $\boldsymbol{\zeta} := \begin{bmatrix}\theta & \sigma & \nu\end{bmatrix}$. Again, fractional FFT is used to retrieve option prices.

B ANN With Multiple Hidden Layers

The ANN with two hidden layers is created by adding a hidden layer of $\dot{H}$ nodes to the left of the hidden layer in Figure 1. To obtain the output of this network, $\sigma\left(b_j + \sum_{i=1}^{N} w_{ji}x_i\right)$, which represents the contribution of the input layer and the hidden layer, is replaced by $\sigma\left(c_j + \sum_{k=1}^{\dot{H}} \dot{w}_{jk}\sigma\left(b_k + \sum_{i=1}^{N} w_{ki}x_i\right)\right)$ in (1). This term consists of a weight and bias term for the new hidden layer, an activation function for the new hidden layer, as well as the contribution of the input layer and the original hidden layer. The output of the network with two hidden layers is then given by

$$y = \sigma_0\left(a_0 + \sum_{j=1}^{H} v_j\sigma\left(c_j + \sum_{k=1}^{\dot{H}} \dot{w}_{jk}\sigma\left(b_k + \sum_{i=1}^{N} w_{ki}x_i\right)\right)\right), \tag{42}$$

where $\dot{w}$ and c are the weight and bias term for the second hidden layer. Similarly, the ANN with three hidden layers is created by adding another hidden layer to (42), now with $\ddot{H}$ nodes. Again, the hidden layer is inserted to the left of the existing hidden layers. In this case, $\sigma\left(b_k + \sum_{i=1}^{N} w_{ki}x_i\right)$, which represents the contribution of the input layer and the adjacent hidden layer, is replaced by $\sigma\left(d_k + \sum_{l=1}^{\ddot{H}} \ddot{w}_{kl}\sigma\left(b_l + \sum_{i=1}^{N} w_{li}x_i\right)\right)$. The output of the network with three hidden layers therefore reads

$$y = \sigma_0\left(a_0 + \sum_{j=1}^{H} v_j\sigma\left(c_j + \sum_{k=1}^{\dot{H}} \dot{w}_{jk}\sigma\left(d_k + \sum_{l=1}^{\ddot{H}} \ddot{w}_{kl}\sigma\left(b_l + \sum_{i=1}^{N} w_{li}x_i\right)\right)\right)\right), \tag{43}$$

where $\ddot{w}$ and d are the weight and bias term for the third hidden layer. For computational purposes, the same activation function, $\sigma(\cdot)$, is chosen for each hidden layer.

C Activation Functions

I investigate five different activation functions for the hidden layer(s), $\sigma(\cdot)$, denoted as $\sigma_1(\cdot)\ldots\sigma_5(\cdot)$. The sigmoid activation function is defined as $\sigma_1(x) = \dfrac{1}{1+\mathrm{e}^{-x}}$ and the softplus activation function as $\sigma_2(x) = \log(1+\mathrm{e}^{x})$, as introduced in Section 2.1.3. The hyperbolic tangent activation function is defined as $\sigma_3(x) = \tanh(x)$. The ReLU activation function is defined as $\sigma_4(x) = \max(0, x)$ and the ELU activation function is defined as $\sigma_5(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha(e^{x}-1) & \text{otherwise} \end{cases}$, where $\alpha > 0$.

A closer look into the workings of these activation functions reveals why tanh and sigmoid as activation function for hidden layers of deep neural networks should be considered with caution. Both functions take a real-valued input and compress it into a predefined range. The definition of the sigmoid causes the output to range between 0 and 1, as $\lim_{x\to-\infty} \frac{1}{1+\mathrm{e}^{-x}} = 0$ and $\lim_{x\to\infty} \frac{1}{1+\mathrm{e}^{-x}} = 1$. Similarly, the output of the tanh function is compressed between -1 and 1, for it holds that $\lim_{x\to-\infty} \tanh(x) = \lim_{x\to-\infty} \frac{\mathrm{e}^{x}-\mathrm{e}^{-x}}{\mathrm{e}^{x}+\mathrm{e}^{-x}} = -1$ and $\lim_{x\to\infty} \frac{\mathrm{e}^{x}-\mathrm{e}^{-x}}{\mathrm{e}^{x}+\mathrm{e}^{-x}} = 1$. The outputs of the activation functions thus saturate, thereby forcing the gradients in limiting regions to approach zero. The gradients, however, are of upmost importance during the training of the neural networks. The gradient for each neuron is the rate of change of the loss function with respect to the neuron's bias, which controls the changes in weight and biases of the network. The Adam optimizer uses gradient-based optimization, i.e. each gradient is multiplied with the gradient of other output gates. Gradients close to zero then effectively shut down a signal, such that no signal flows through the neurons and inescapably much less information reaches deeper layers. This is lethal to the performance of deeper neural networks, through which lots of information flows to build up multiple layers of abstraction.

Consequently, this problem—first dubbed the vanishing gradient problem by Hochreiter et al. (2001) and later extensively researched by many others—requires a solution. In search thereof, Nair and Hinton (2010) first introduced the rectified linear units for restricted Boltzmann machines to replace sigmoid units. Then, Glorot et al. (2011) asserted that for neural networks it also holds that networks with rectifying neurons yield better results than networks with hyperbolic tangent neurons. Krizhevsky et al. (2012) further found that deep neural networks with ReLU activation functions train much faster than equivalent deep neural network with tanh activation functions. Importantly, faster learning greatly influences the performance of large neural network models trained on large data sets. This faster learning is mainly achieved by the simplicity of the ReLU activation function, as ReLU can simply be implemented by performing a threshold at zero for all inputs. Also, ReLUs partly alleviate the vanishing gradient problem by not saturating the inputs for both limiting cases. To fully alleviate the vanishing gradient problem, Clevert et al. (2015) introduced the ELU activation function, which enables negative values. However, the introduction of negative values comes at the expense of speed. It is therefore interesting to research which of these two solutions to the vanishing gradient problem best fits hybrid neural networks for option pricing. Furthermore, it is interesting to research whether or not neural networks with these activation functions indeed outperform the neural networks with tanh and sigmoid activation functions.

D MNN and MCMNN With Multiple Hidden Layers

Following the approach outlined in Appendix B, the MNN and MCMNN with two hidden layers are created by adding a hidden layer of $\dot{H}_k$ nodes to the left of the hidden layer in each of the $k = 1, \ldots, M$ modules in Figure 2. To obtain the output of this network, $\sigma\left(b_{jk} + \sum_{i=1}^{N} w_{jik} x_i\right)$, which represents the contribution of the input layer and the hidden layer, is replaced by $\sigma\left(c_{jk} + \sum_{l=1}^{\dot{H}_k} \dot{w}_{jlk}\sigma\left(b_{lk} + \sum_{i=1}^{N} w_{lik} x_i\right)\right)$ in (5). This term consists of a weight and bias term for the new hidden layer, an activation function for the new hidden layer, as well as the contribution of the input layer and the original hidden layer. The output of the network with two hidden layers is then given by

$$y = \sigma_0\left(a_{0,k} + \sum_{j=1}^{H_k} v_{jk}\sigma\left(c_{jk} + \sum_{l=1}^{\dot{H}_k} \dot{w}_{jlk}\sigma\left(b_{lk} + \sum_{i=1}^{N} w_{lik} x_i\right)\right)\right), \tag{44}$$

where $\dot{w}$ and c are the weight and bias term for the second hidden layer.

Following the same approach, the MNN and MCMNN with three hidden layers are created by adding another hidden layer to (44), now with $\ddot{H}_k$ nodes. Again, the hidden layer is inserted to the left of the existing hidden layers. In this case, $\sigma\left(b_{lk} + \sum_{i=1}^{N} w_{lik} x_i\right)$, which represents the contribution of the input layer and the adjacent hidden layer, is replaced by $\sigma\left(d_{lk} + \sum_{m=1}^{\ddot{H}_k} \ddot{w}_{lmk}\sigma\left(b_{mk} + \sum_{i=1}^{N} w_{mik} x_i\right)\right)$. The output of the network with three hidden layers therefore reads

$$y = \sigma_0\left(a_{0,k} + \sum_{j=1}^{H_k} v_{jk}\sigma\left(c_{jk} + \sum_{l=1}^{\dot{H}_k} \dot{w}_{jlk}\sigma\left(d_{lk} + \sum_{m=1}^{\ddot{H}_k} \ddot{w}_{lmk}\sigma\left(b_{mk} + \sum_{i=1}^{N} w_{mik} x_i\right)\right)\right)\right), \tag{45}$$

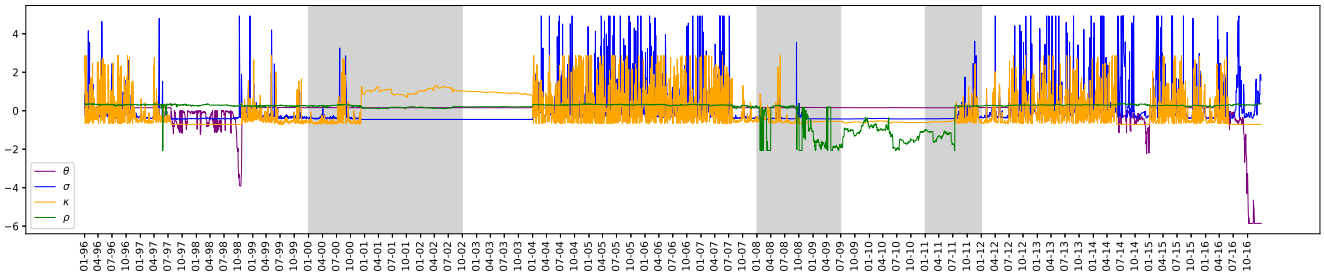
where $\ddot{w}$ and d are the weight and bias term for the third hidden layer. For computational purposes, the same activation function, $\sigma(\cdot)$, is chosen for each hidden layer. Evidently, the number of nodes per hidden layer, respectively H, $\dot{H}$ and $\ddot{H}$, does differ. However, for each model, the number of nodes across modules, respectively H_k, $\dot{H}_k$ and $\ddot{H}_k$ is set constant across modules, again for computational purposes. Finally, for the MNN, $M = 9$ and $N = 2$ and for the MCMNN, $M = 18$ and $N = 3$.

E Analysis of the Daily Calibrated Parameters of the Parametric Models

To better understand the optimal performance of the three parametric models, the daily calibrated parameters of each model are scrutinized. In Figure 18, 19 and 20 parameters are standardized for visibility purposes such that it is more easily observable in which periods the parameters need to fluctuate most for the model to have an optimal pricing performance.

Figure 18 shows the standardized parameters of the Heston model. For the Heston model, the most variation is observed in the mean reversion rate, κ, and the volatility of the volatility, σ. These parameters respectively control the volatility smile and skew. The parameters fluctuate more after 2004; this can possibly be linked to the fact that starting in 2004 more options must be priced, see Table 1, such that more complex volatility surfaces are fit. Furthermore, in the crisis periods of 2008 and 2011 the optimal value of ρ is much smaller than in other periods. This can be explained by the fact that in this period the underlying price of the S&P500 underlying price is strongly decreasing, whereas the volatility in this period is extremely high, leading to a large negative correlation between the underlying price and the corresponding variance. Finally, starting in 2014 the Heston model performs much worse than prior to 2014, as can be observed in Figure 5. Prior to 2014, the Heston model is among the top-performing parametric models, but fails to do so in 2014 and 2016. This poor performance can possibly be due to large negative values of θ, the long-term variance, which accounts for the overall level of volatility skew. This poor performance has a big impact on the overall performance of the Heston model, as approximately half of the options is traded between 2014 and 2016.

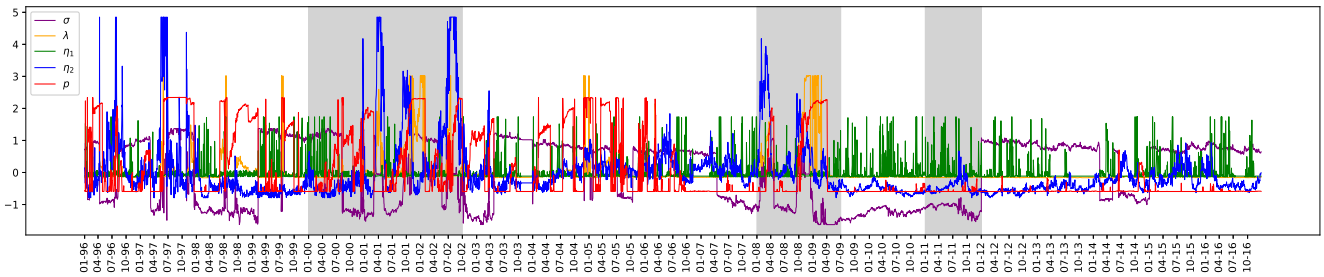
Figure 18: Standardized parameters of the Heston model over time



Note: This figure shows how the standardized parameters θ, σ, κ and ρ of the Heston model evolve over time. Here, κ is the mean-reversion rate, θ is the long-term variance, σ is the volatility of the volatility and ρ is the correlation between the underlying price and its variance. Parameters are standardized for visibility purposes. The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

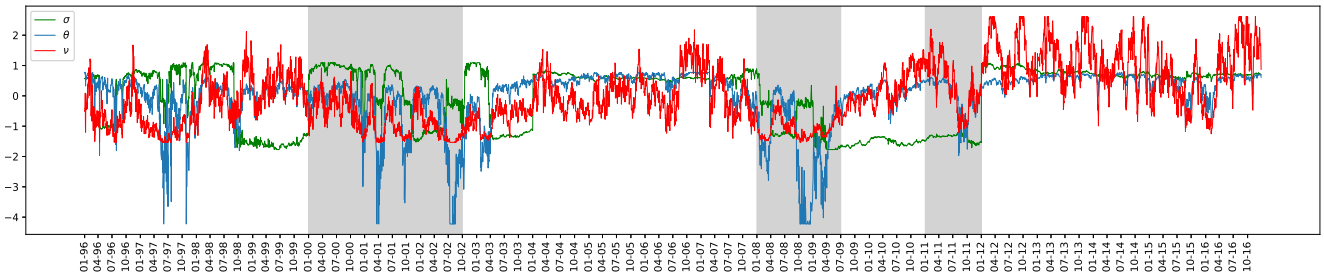
The standardized parameters of the Kou jump-diffusion model are displayed in Figure 19. First of all, in crisis periods low values for the volatility of the Brownian motion, σ, are observed. The two building blocks of a jump-diffusion model are the Poisson process (for the jumps) and the Brownian motion (for the diffusion). For the Kou jump-diffusion models the low values of σ in crisis periods therefore indicate that the Lévy process of (32) is then mostly influenced by the Poisson process. As the jumps represent events such as breaking news and corresponding crashes, it appears to be valid that prices are greatly influenced by jumps in periods of distress. Accordingly, in these periods the highest values for λ and η_2 are observed, respectively representing the jump intensity of the Poisson process and the expected negative jump size. Furthermore, for high values of η_2, low probabilities of upward jumps, p, are observed. After 2011, high values of σ and relatively low values of η_2 are observed, indicating that that the Lévy process of (32) is mostly influenced by the Brownian motion.

Figure 19: Standardized parameters of the Kou jump-diffusion model over time



Note: This figure shows how the standardized parameters σ, λ, η_1, η_2 and p of the Kou jump-diffusion model evolve over time. Here, σ is the volatility of the Brownian motion, λ is jump intensity of the Poisson process, η_1 and η_2 are the expected positive and negative jump sizes respectively and p represents the probability of upward jumps. Parameters are standardized for visibility purposes. The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

Figure 20: Standardized parameters of the Variance Gamma model over time

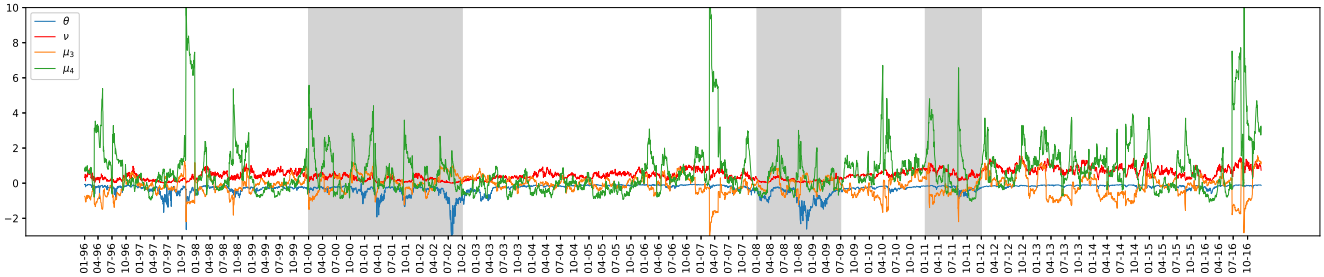


Note: This figure shows how the standardized parameters σ, θ and ν of the Variance Gamma model evolve over time. Here, σ is the volatility of the Brownian motion, θ is the drift of the Brownian motion and ν is the variance rate of the gamma time change. Parameters are standardized for visibility purposes. The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

The standardized parameters of the Variance Gamma model are shown in Figure 20. The drift of the Brownian motion, θ, is the parameter that fluctuates the most, whereas the volatility of the Brownian motion, σ, and the variance rate of the gamma time change, ν, are fairly constant. The Variance Gamma model thus mostly fits option prices by varying the $\theta\gamma_t(1,\nu)$ term in (40). Furthermore, especially in crisis periods each of the three parameters seems to be smaller than in other periods. Similar to the Kou jump-diffusion model, the low values of σ between 2008 and 2011 stand out, indicating that this exponential Lévy model also neglects the Brownian motion in periods of financial turmoil. After the crises of 2008 and 2011 the mean market price of call options starts increasing, see Table 1. The parameters of the VG model all increase to account for the higher market prices. The interpretation of the Variance Gamma parameters is an interesting topic to examine. According to Madan et al. (1998) one of the paramount features of the VG model is that there are parameters that control for skewness and kurtosis, which is done by the parameters θ and ν. In Figure 21 it can be observed that when both skewness and kurtosis are high θ is negative. When skewness and kurtosis are low, ν increases and when skewness and kurtosis are opposite θ and ν are centered around 0. This shows that θ and ν together control for skewness and kurtosis.

Overall, the best performing parametric models in terms of MAPE and MSE mainly achieve their optimal performance by implementing jumps. The Kou jump-diffusion model slightly outperforms the Variance Gamma model. For a data set that alternates between regular and crisis periods, the finite jumps of the Kou jump-diffusion model appear to be more adequately fitting than the infinitely many jumps of the Variance Gamma model. The stochastic volatility of the Heston model also has its advantages and at times outperforms both jump models. It is therefore interesting to research which parametric model can help hybrid neural networks achieve the best option pricing performance.

Figure 21: Parameters of the Variance Gamma model over time plotted together with skewness and kurtosis



Note: This figure shows how the parameters θ and ν of the Variance Gamma model evolve over time. Here, θ is the drift of the Brownian motion and ν is the variance rate of the gamma time change. Skewness, μ_3 and kurtosis, μ_4 of the S&P500 index are also shown in this figure. Parameters are not standardized. The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

F Preliminary Hidden Layer and Activation Function Analyses

Table 15: Preliminary analysis of MAPE of the option price forecasts of an ANN with two input variables for various numbers of hidden layers, hidden layer nodes and activation functions

	0				64			96			128		
	0	16	32	48	16	32	48	16	32	48	16	32	48
Panel A													
2	4.668	1.943	1.558	1.156	0.717	3.006	3.691	2.812	3.470	4.471	0.769	3.236	4.596
4	3.288	1.269	1.368	1.074	0.661	0.587	0.713	0.599	0.765	0.597	3.063	0.499	2.077
8	2.473	1.054	1.235	1.021	0.706	0.526	0.623	0.654	0.685	0.522	0.624	0.614	0.771
Panel B													
2	5.276	0.963	1.201	0.922	1.110	6.100	3.307	2.812	1.669	2.386	3.211	1.423	1.355
4	5.001	1.087	1.164	1.107	0.860	0.597	0.642	0.549	0.715	0.668	1.110	0.487	0.616
8	4.619	1.295	1.205	1.214	0.601	0.605	0.635	0.498	0.680	0.649	0.606	0.555	0.525

This table presents the MAPE of the option price forecasts of an ANN with two input variables, m and τ, for various numbers of hidden layers, hidden layer nodes and activation functions. Panel A displays the MAPEs from the ANN with the ReLU activation function, Panel B from the ANN with sigmoid activation function. In both panels the linear activation is used for the output layer. On the vertical axis, the number of hidden layer nodes for the first hidden layer is given. The horizontal axis is split; the lower number is the number of nodes in the second hidden layer and the upper number is the number of nodes in the third hidden layer.

Table 16: Preliminary analysis of MSE of the option price forecasts of an ANN with two input variables for various numbers of hidden layers, hidden layer nodes and activation functions

	0				64			96			128		
	0	16	32	48	16	32	48	16	32	48	16	32	48
Panel A													
2	6310	3500	2980	1791	376.7	8409	12259	10357	9914	18488	421.9	11813	15413
4	3439	1825	2410	1579	144.9	153.9	332.6	113.8	382.3	170.2	8084	131.5	7397
8	2336	1249	2025	1419	263.8	110.7	121.7	184.6	466.3	86.77	153.8	156.8	261.8
Panel B													
2	1676	202.4	221.5	204.7	604.7	19805	8955	10236	1980	8227	10307	1961	3449
4	1468	199.9	212.1	187.8	401.9	138.5	222.2	108.4	236.0	211.4	116.9	309.1	136.9
8	1227	227.0	208.3	168.9	180.4	139.0	139.5	94.86	241.0	143.7	110.1	103.0	99.69

This table presents the MSE from an ANN with two input variables, m and τ, for various number of hidden layers, hidden layer nodes and activation functions. Panel A displays the MSEs for the ANN with the ReLU activation function, Panel B for the ANN with sigmoid activation function. In both panels the linear activation is used for the output layer. On the vertical axis, the number of hidden layer nodes for the first hidden layer is given. The horizontal axis is split; the lower number is the number of nodes in the second hidden layer and the upper number is the number of nodes in the third hidden layer.

Table 17: Preliminary analysis of MAPE and MSE of the option price forecasts of an ANN with two input variables for various activation functions

	σ_1	σ_2	σ_3	σ_4	σ_5
linear	1.005 (191.6)	0.962 (123.9)	1.151 (358.7)	1.393 (337.3)	0.522 (86.77)
exponential	1.205 (193.7)	1.397 (21934)	4.281 (1250)	3.510 (17595)	1.299 (33649)

This table presents the MAPE (MSE) of the option price forecasts of an ANN with two input variables, m and τ, and three hidden layers with respectively 96, 48 and 8 hidden layer nodes for five activation functions. The five activation functions are the sigmoid (σ_1), softplus (σ_2), hyperbolic tangent (σ_3), ELU (σ_4) and ReLU (σ_5) activation functions. On the vertical axis the activation function of the output layer is displayed.

G Comprehensive Overview of Results of the Non-Parametric Models

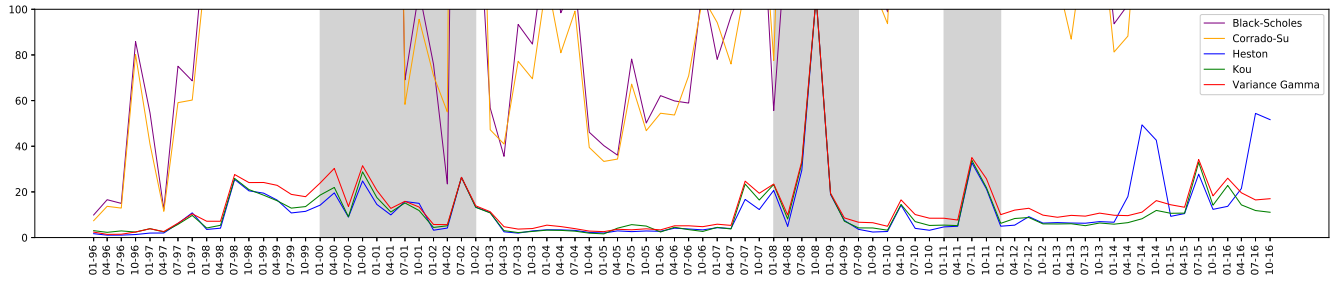
Table 18: Comprehensive overview of results of the non-parametric models

	ANN		MNN		MCMNN	GNN		GNN*
	2 inputs	3 inputs	2 inputs	3 inputs	3 inputs	2 inputs	3 inputs	2 inputs
Panel A: MAPE								
Standard	0.474	0.317	0.282	0.056	0.068	0.242	0.108	0.234
Hybrid-BS	0.599	0.572	0.596	0.588	0.624	0.603	0.563	0.676
Hybrid-CS	0.619	0.600	0.637	0.631	0.729	0.639	0.586	0.848
Hybrid-H	0.265	0.255	0.256	0.242	0.252	0.264	0.258	0.323
Hybrid-K	0.143	0.138	0.139	0.139	0.122	0.100	0.094	0.113
Hybrid-VG	0.182	0.123	0.174	0.138	0.130	0.115	0.094	0.125
Panel B: MSE								
Standard	98.34	41.60	92.36	52.62	43.64	47.30	9.311	46.42
Hybrid-BS	143.9	157.6	141.2	154.7	156.4	140.2	137.9	153.5
Hybrid-CS	180.7	196.7	180.1	196.0	207.1	173.7	152.1	200.0
Hybrid-H	19.07	26.61	18.92	22.78	22.02	18.37	19.19	41.01
Hybrid-K	10.70	16.22	10.64	14.37	11.14	8.897	8.791	10.95
Hybrid-VG	10.80	15.19	10.67	14.37	11.40	8.835	8.179	13.44

This table presents MAPEs (in Panel A) and MSEs (in Panel B) of the option price forecasts of the ANN, MNN, MCMNN GNN, GNN* and their hybrid counterparts with either two (m, τ) or three $(m, \sigma\sqrt{\tau}, r\tau)$ input variables and with optimal architecture. The number of input variables is given below each model on the horizontal axis. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG).

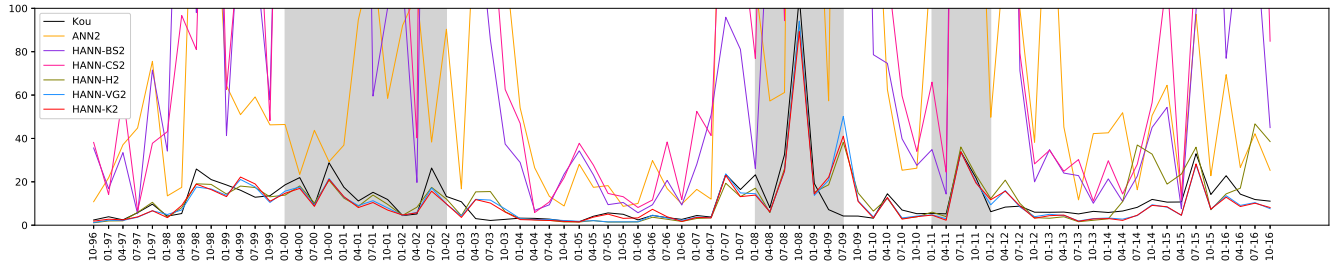
H MSE Plots of All Parametric and Non-Parametric Models

Figure 22: MSE of the parametric models over time



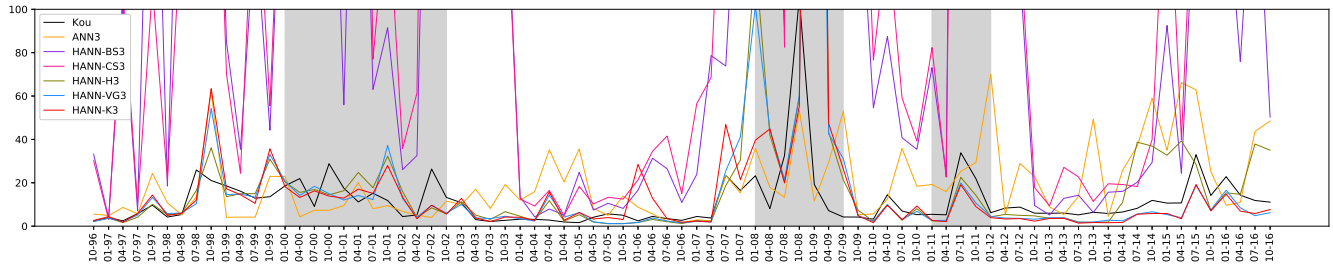
Note: This figure shows how the MSE of the option price forecasts of each of the parametric models evolves over time. The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

Figure 23: MSE of the option price forecasts of the Kou jump-diffusion model, the standard ANN with two input variables and the HANNs with two input variables over time



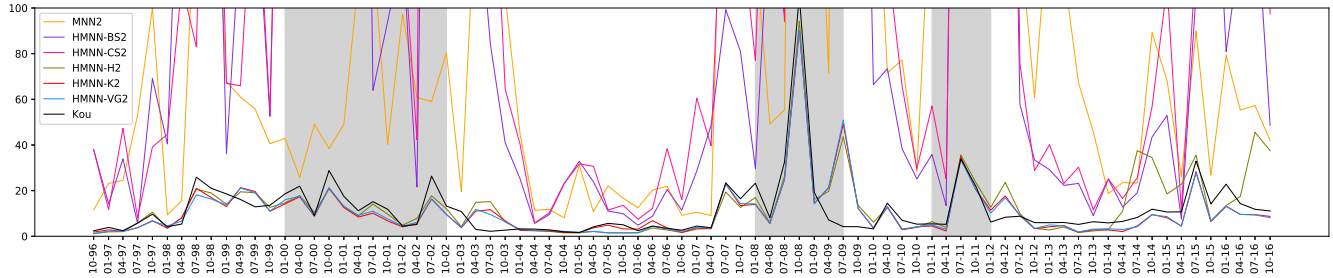
Note: This figure shows how the MSE from the Kou jump-diffusion model, the standard ANN with two input variables and the HANNs with two input variables evolves over time. The two input variables are m and τ. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

Figure 24: MSE of the option price forecasts of the Kou jump-diffusion model, the standard ANN with three input variables and the HANNs with three input variables over time



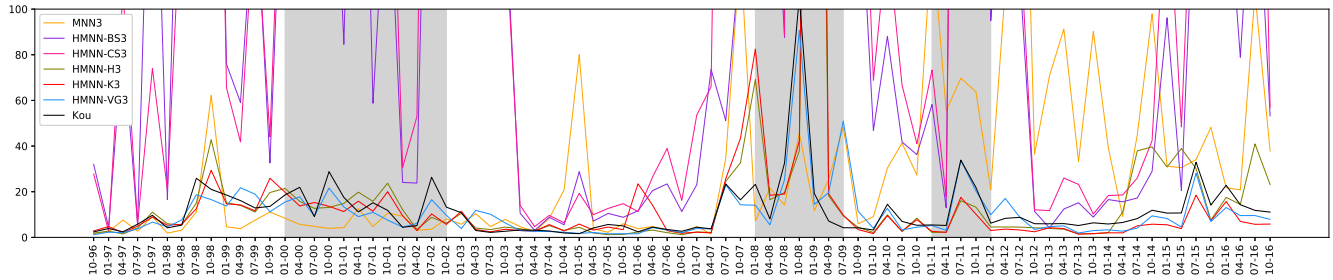
Note: This figure shows how the MSE from the Kou jump-diffusion model, the standard ANN with three input variables and the HANNs with three input variables evolves over time. The three input variables are $m, \sigma\sqrt{\tau}$ and $r\tau$. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

Figure 25: MSE of the option price forecasts of the Kou jump-diffusion model, the standard MNN with two input variables and the HMNNs with two input variables over time



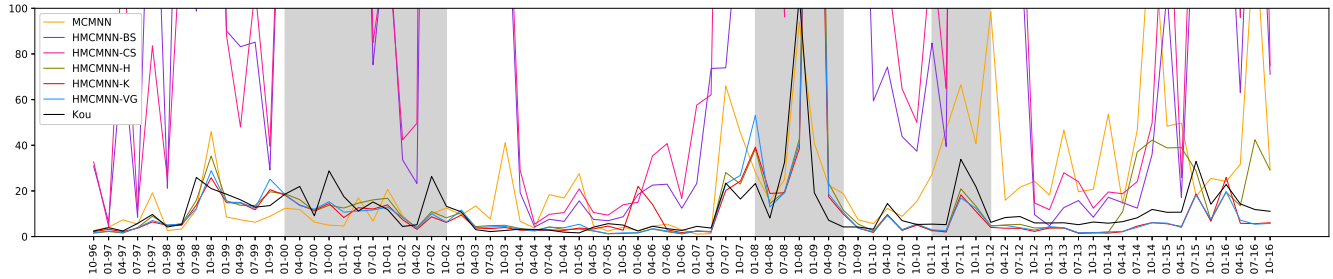
Note: This figure shows how the MSE from the Kou jump-diffusion model, the standard MNN with two input variables and the HMNNs with two input variables evolves over time. The two input variables are m and τ. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

Figure 26: MSE of the option price forecasts of the Kou jump-diffusion model, the standard MNN with three input variables and the HMNNs with three input variables over time



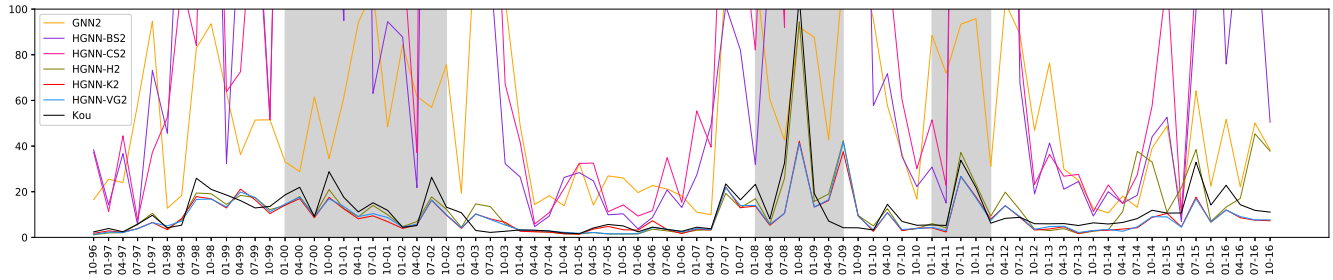
Note: This figure shows how the MSE from the Kou jump-diffusion model, the standard MNN with three input variables and the HMNNs with three input variables evolves over time. The three input variables are $m, \sigma\sqrt{\tau}$ and $r\tau$. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

Figure 27: MSE of the option price forecasts of the Kou jump-diffusion model, the standard MCMNN with three input variables and the HMCMNNs with three input variables over time



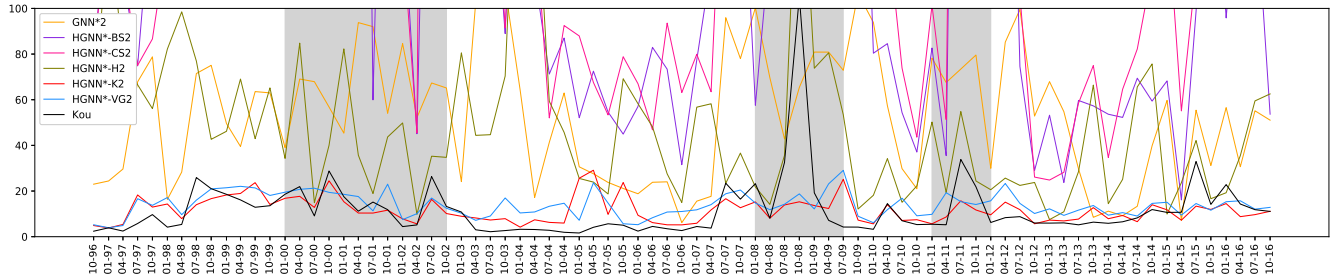
Note: This figure shows how the MSE from the Kou jump-diffusion model, the standard MCMNN with three input variables and the HMCMNNs with three input variables evolves over time. The three input variables are $m, \sigma\sqrt{\tau}$ and $r\tau$. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

Figure 28: MSE of the option price forecasts of the Kou jump-diffusion model, the standard GNN with two input variables and the HGNNs with two input variables over time



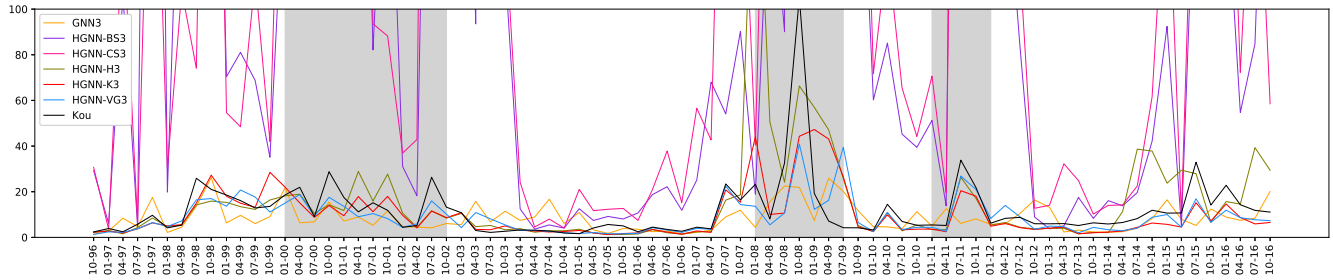
Note: This figure shows how the MSE from the Kou jump-diffusion model, the standard GNN with two input variables and the HGNNs with two input variables evolves over time. The two input variables are m and τ. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

Figure 29: MSE of the option price forecasts of the Kou jump-diffusion model, the standard GNN* with two input variables and the HGNN*s with two input variables over time



Note: This figure shows how the MSE from the Kou jump-diffusion model, the standard GNN* with two input variables and the HGNN*s with two input variables evolves over time. The three input variables are m and τ. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

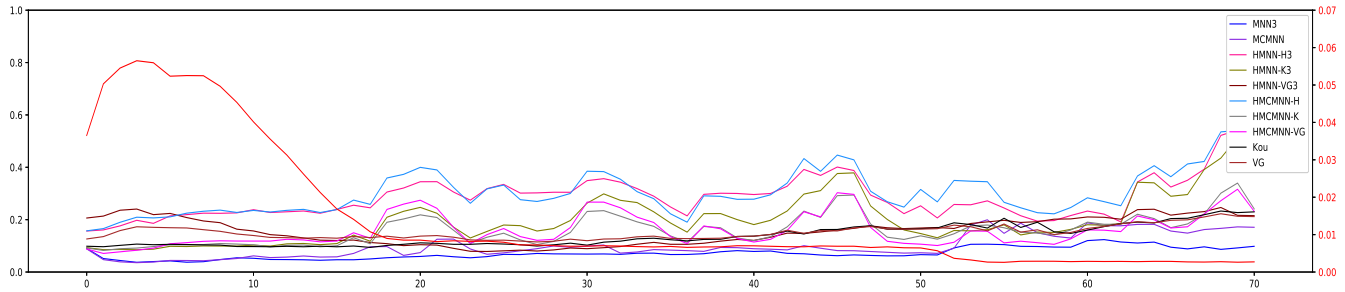
Figure 30: MSE of the option price forecasts of the Kou jump-diffusion model, the standard GNN with three input variables and the HGNNs with three input variables over time



Note: This figure shows how the MSE from the Kou jump-diffusion model, the standard GNN with three input variables and the HGNNs with three input variables evolves over time. The three input variables are $m, \sigma\sqrt{\tau}$ and $r\tau$. The employed parametric models are the Black-Scholes model (BS), Corrado-Su model (CS), Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).

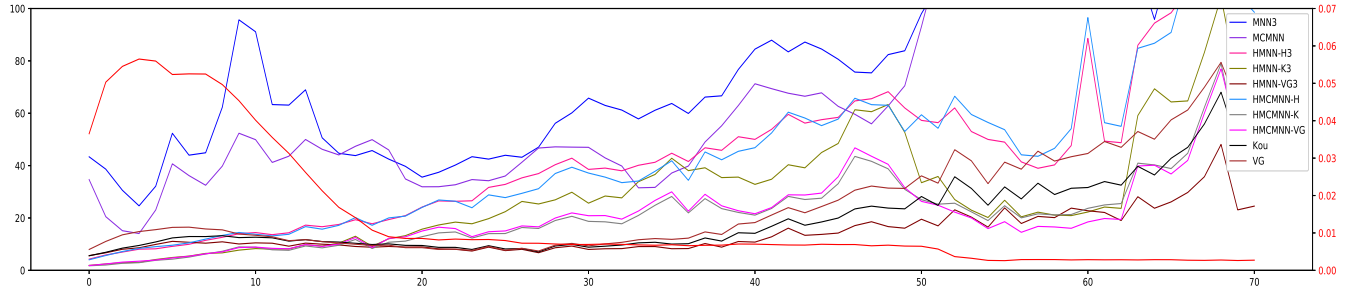
I Performance Plots of Modular and Gated Neural Networks as a Function of Maturity

Figure 31: MAPE of the option price forecasts of standard and hybrid modular neural networks and parametric models as a function of maturity



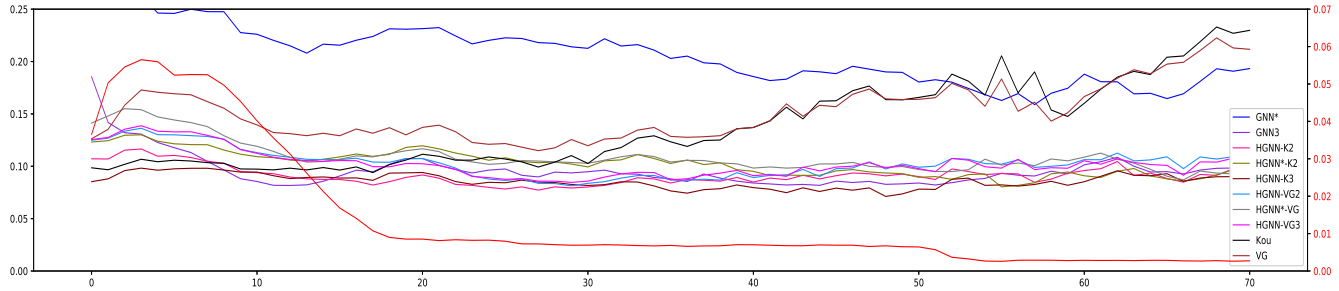
Note: This figure shows how the MAPE from several (hybrid) neural networks with three input parameters that implement modularity based on fixed manual heuristics and two parametric benchmark models, Kou and VG, evolves as maturity increases. The three input variables are $m, \sigma\sqrt{\tau}$ and $r\tau$. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The red line represents the percentage of options per maturity region. Maturity regions are measured in weeks. Maturity is divided by 7 and rounded up to obtain the maturity in weeks. The red line has a separate y-axis on the right.

Figure 32: MSE of the option price forecasts of standard and hybrid modular neural networks and parametric models as a function of maturity



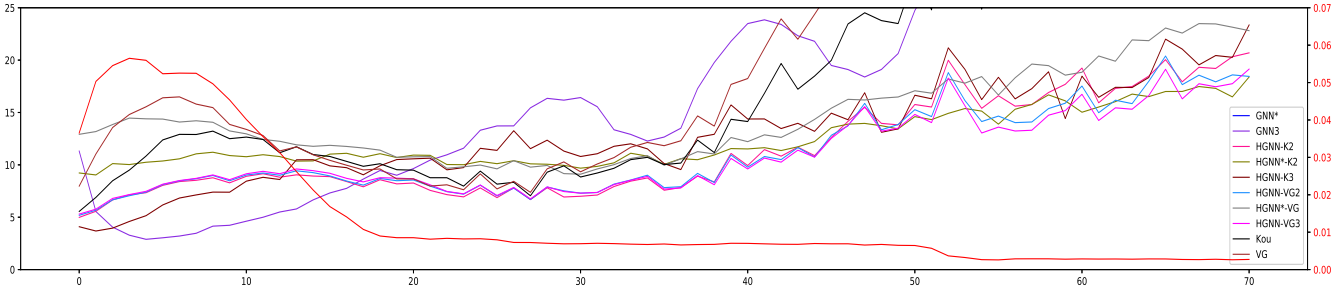
Note: This figure shows how the MSE from several (hybrid) neural networks with three input parameters that implement modularity based on fixed manual heuristics and two parametric benchmark models, Kou and VG, evolves as maturity increases. The three input variables are $m, \sigma\sqrt{\tau}$ and $r\tau$. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The red line represents the percentage of options per maturity region. Maturity regions are measured in weeks. Maturity is divided by 7 and rounded up to obtain the maturity in weeks. The red line has a separate y-axis on the right.

Figure 33: MAPE of the option price forecasts of standard and hybrid gated neural networks and parametric models as a function of maturity



Note: This figure shows how the MAPE from several (hybrid) neural networks with three input parameters that implement modularity based on dynamic option grouping and two parametric benchmark models, Kou and VG, evolves as maturity increases. The three input variables are $m, \sigma\sqrt{\tau}$ and $r\tau$. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The red line represents the percentage of options per maturity region. Maturity regions are measured in weeks. Maturity is divided by 7 and rounded up to obtain the maturity in weeks. The red line has a separate y-axis on the right.

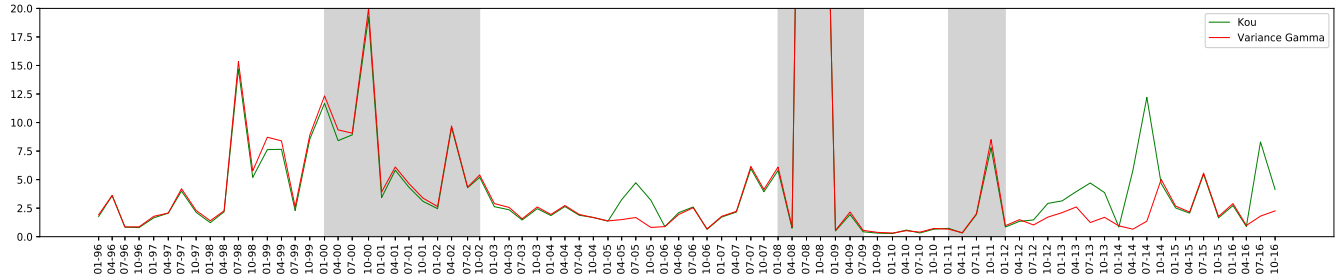
Figure 34: MSE of the option price forecasts of standard and hybrid gated neural networks and parametric models as a function of maturity



Note: This figure shows how the MSE from several (hybrid) neural networks with three input parameters that implement modularity based on dynamic option grouping and two parametric benchmark models, Kou and VG, evolves as maturity increases. The three input variables are $m, \sigma\sqrt{\tau}$ and $r\tau$. For the hybrid neural networks, the target function is adjusted by the output of a parametric model. The employed parametric models are the Heston model (H), Kou jump-diffusion model (K) and Variance Gamma model (VG). The red line represents the percentage of options per maturity region. Maturity regions are measured in weeks. Maturity is divided by 7 and rounded up to obtain the maturity in weeks. The red line has a separate y-axis on the right.

J Performance Plot of Parametric Models for High Volatility Options

Figure 35: MSE of the option price forecasts of the Variance Gamma model and the Kou jump-diffusion model over time for options with high volatilties



Note: This figure shows how the MSE from the Variance Gamma model and Kou jump-diffusion model evolves over time. Included in this analysis are only options with high volatilties, i.e. options for which the volatility exceeds 0.9. The shadowed parts correspond to the dot-com bubble (2000), global financial crisis (2008) and European debt crisis (2011).