sciendo Economics and Business Review https://www.ebr.edu.pl Vol. 9 (2), 2023: 133–159 https://doi.org/10.18559/ebr.2023.2.737

Forecasting realized volatility through financial turbulence and neural networks

Hugo Gobato Souto[1] Amir Moradi[2]

Abstract

This paper introduces and examines a novel realized volatility forecasting model that makes use of Long Short-Term Memory (LSTM) neural networks and the risk metric financial turbulence (FT). The proposed model is compared to five alternative models, of which two incorporate LSTM neural networks and the remaining three include GARCH(1,1), EGARCH(1,1), and HAR models. The results of this paper demonstrate that the proposed model yields statistically significantly more accurate and robust forecasts than all other studied models when applied to stocks with middle-to-high volatility. Yet, considering low-volatility stocks, it can only be confidently affirmed that the proposed model yields statistically significantly more robust forecasts relative to all other models considered.

Keywords

- neural networks
- LSTM neural networks
- realized volatility prediction
- financial turbulence

JEL codes: C45, C53, G19

Article received 27 February 2023, accepted 16 June 2023.

Suggested citation: Souto, H. G., & Moradi, A. (2023). Forecasting realized volatility through financial turbulence and neural networks. *Economics and Business Review*, *9*(2), 133–159. https://doi.org/10.18559/ebr.2023.2.737

[1] International School of Business at HAN University of Applied Sciences, Ruitenberglaan 31, 6826 CC Arnhem, the Netherlands, H.GobatoSouto@han.nl; https://orcid.org/0000-0002-7039-0572.

[2] International School of Business at HAN University of Applied Sciences, Ruitenberglaan 31, 6826 CC Arnhem, the Netherlands, corresponding author: amir.moradi@han.nl; https://orcid.org/0000-0003-1169-7192.

Introduction

Properly determining and forecasting the volatility of securities is crucial for investment institutions. Such a risk parameter is often used in portfolio risk management, asset pricing, and portfolio construction (Gajdka & Pietraszewski, 2017; Latoszek & Ślepaczuk, 2020; Loang & Ahmad, 2021). Short-term volatility forecasts, such as on a monthly or ideally daily basis, is particularly advantageous for active portfolio managers. Though volatility is not observable during short periods, the so-called realized volatility is commonly used as a proxy since it is considered a reliable estimator of volatility (Andersen & Bollerslev, 1998). Thus, a great part of financial literature has been devoted to finding a proper way of measuring realized volatility and accurately predicting it.

Many methods of measuring daily realized volatility have already been devised. The best known examples might be Parkinson's realized volatility (Parkinson, 1980), Garman and Klass' realized volatility (Garman & Klass, 1980), Rogers and Satchell's realized volatility (Rogers & Satchell, 1991), and Yang and Zhang's realized volatility (Yang & Zhang, 2000). Among these, Yang and Zhang's realized volatility stands out due to its unbiasedness in the continuous limit, drift independence, and consistency in addressing opening price jumps (Yang & Zhang, 2000).

Regardless of the choice of realized volatility proxy, accurately forecasting daily realized volatility is challenging due to its asymmetrical reaction to unexpected news and heteroscedasticity (Black, 1986; Bollerslev, 1986; Engle, 1982). To address this issue, a wide variety of time series models, primarily known as GARCH models, have been developed (Bauwens et al., 2006). Some examples of these models include GARCH, EGARCH, ARCH-M, APARCH, and T-GARCH. The efficiency of these models in daily realized volatility forecasting has been extensively studied (see Borup & Jakobsen, 2019; Brandt & Jones, 2006; Haugom et al., 2010; Kambouroudis et al., 2016). Furthermore, it has been proven that they accurately capture short-term variations in the daily realized volatility of various stocks, yet this is not the case for long-term and nonlinear variations (Borup & Jakobsen, 2019; Brandt & Jones, 2006; Kambouroudis et al., 2016). Consequently, none of them yield a perfect forecasting method for daily realized volatility (Bauwens et al., 2006).

In addition to GARCH models, other linear models such as HAR models have also been devised to attempt to correctly forecast daily realized volatility. However, their shared drawback with GARCH models of not accurately capturing nonlinear and long-term trends is still present (Engle et al., 2013; Liu, Demirer et al., 2020). As a result, researchers have devised and employed a variety of nonlinear models to attempt to capture these nonlinear and long-term patterns. Presumably, the most known and effective model of these models is artificial neural networks (ANNs). ANNs are excellent at forecasting

realized volatility as without being aware of the data generation process; this technique enables the approximation of a large class of linear and nonlinear functions arbitrarily well (Bucci, 2020).

ANNs are widely used in the financial world and literature. Their efficacy has already been demonstrated for forecasting many financial variables, such as exchange rates, probability of default, stock price, and realized volatility (Donaldson & Kamstra, 1996a,b; Kamijo & Tanigawa, 1990; Khan, 2011; Naidu & Govinda, 2018; White, 1988; Wilson & Sharda, 1994; Yan & Yang, 2021; Zhu et al., 2008). However, it is worth mentioning that relative to exchange rates and stock price predictions, research about the use of ANNs to predict realized volatility has been a bit less developed (Bucci, 2020). Furthermore, when ANNs are employed to predict realized volatility, they usually are used in conjunction with GARCH models (Donaldson & Kamstra, 1997; Hajizadeh et al., 2012; Maciel et al., 2016).

However, there are also some research papers that solely make use of ANNs. For instance, Hamid & Iqbal (2004) prove that by only using ANNs, one can outperform implied volatility forecasts. Chen and Robert (2022), on the other hand, make use of a Graph Neural Network (GNN), a type of ANN, to predict realized volatility for the next 600 seconds with a precision of 77.13%. Another example is the study performed by Bucci (2020) in which he shows that various types of ANNs outperformed the famous time series models ARFIMA and ARFIMAX in the prediction of the monthly realized volatility of the S&P 500 using a sample from August 1997 until December 2017.

Arguably, the most effective types of ANN in predicting realized volatility are Long Short-Term Memory neural network (LSTM) (Hu et al., 2020; Li, 2022; Lin et al., 2022; Rodikov & Antulov-Fantulin; 2022; Vidal & Kristjanpoller, 2020), Nonlinear Autoregressive model process with eXogenous input (NARX) (Aaltio, 2022; Baffour et al., 2019; D'Ecclesia & Clementi, 2021), and Convolutional Neural Network (CNN) (Chen et al., 2022; Li, 2022; Vidal & Kristjanpoller, 2020).

Incidentally, volatility is not the sole measure of stock risk (Kritzman & Li, 2010). One recent risk parameter that is an alternative to volatility is financial turbulence (FT) (Kritzman & Li, 2010). Its application in portfolio management has yielded promising results (Liu, Yang et al., 2021; Nystrup, Boyd et al., 2019; Nystrup, Madsen et al., 2018). FT is defined by Equation (1):

$$d_t = (y_t - \mu)\, \Sigma^{-1}\, (y_t - \mu)' \tag{1}$$

where d_t = turbulence for a particular time period t, $y_t = 1 \times n$ vector of asset returns for period t, μ = sample average $1 \times n$ vector of historical returns, and Σ = sample covariance $n \times n$ matrix of historical returns.

Similar to realized volatility, predicting FT is also challenging due to its nonlinear and long-term patterns, and its results are not as interpretable as real-

ized volatility (Souto, 2023a,b). Nonetheless, FT can be used in the prediction of realized volatility. For example, Salisu et al. (2022) show that the use of FT can improve the out-of-sample predictive performance of stock market volatility linear models over both the short and long time horizon.

Nonetheless, there is a lack of literature on the application of FT in the prediction of realized volatility through the use of ANNs. Thus, this paper aims to investigate the use of FT in predicting realized volatility through the use of ANNs. The selected ANN type is LSTM with the rationale for this choice discussed in Section 1. Furthermore, this research contributes to the existing literature in three main ways. Firstly, to the best of our knowledge, this is the first paper that explores and shows that the use of FT in predicting realized volatility through the use of LSTM yield statistically more accurate and robust forecasts. Secondly, this paper employs Yang and Zhang's realized volatility as the realized volatility proxy, a robust yet accessible proxy. This proxy is considered to be accessible as it only requires high, low, close and open prices to be estimated, which is a type of data that is easily accessible by practitioners and researchers. Thirdly, this research provides the code for the novel neural network model as open source, allowing access to practitioners and researchers.

The rest of this paper is structured as follows: Section 1 briefly introduces ANNs and LSTM. The research design used in this study and sample are described in Section 2. Section 3 evaluates the models' success in terms of forecasting accuracy. In Section 4, a robustness check is performed to test the results of Section 3, and in Section 5 the conclusion and limitations of this paper can be found.

1. ANNs and LSTM

ANNs are a mathematical system that aims to reproduce the human brain in order to take one (or many) input(s) and yield one (or many) estimated output(s) (Donaldson & Kamstra, 1996a). ANNs are composed of interconnected neurons (or nodes) arranged in ties. Their neurons can be divided into three types of layers: the input layer (where the inputs come), the hidden layer (where the calculations occur), and the output layer (where the output comes).

Such a mathematical system model makes use of weights and intercepts (as a linear regression), and activation functions to allow nonlinearity modelling. The weights and intercepts (also commonly named as biases in the ANNs field) are updated at each iteration through an algorithm based on the gradient descent rule that has the aim of minimizing a selected error function. Generally, the chosen error function is the mean squared error (MSE)

of the actual and predicted values, yet other error functions are also possible (Donaldson & Kamstra, 1996a).

In simple words, one can think of an ANN as an aggregation of many linear regressions with the addition of activation functions to capture nonlinear patterns, and the use of an arbitrary error function that will be the basis of parameters optimization. As one may already expect, the bigger the number of hidden layers and nodes, the better ANNs can learn complex patterns, but also the more time it takes to estimate the ANN's optimal weights and biases (Donaldson & Kamstra, 1996a). Currently, there is some research (Sheela & Deepa, 2013) about possible procedures to determine the ideal quantity of hidden layers and nodes. Nonetheless, no widely accepted procedure currently exists (Bucci, 2020; Vujičić et al., 2016). Fortunately, a single hidden neural network is a universal approximator, indicating that if enough hidden nodes are present, the network can approximate a variety of linear and nonlinear functions (Donaldson & Kamstra, 1996b). As a consequence, this paper makes use of a single hidden layer network for its LSTM models.

To further understand how an ANN works, consider a three-layer neural network and a single output variable. This ANN example is depicted in Figure 1.

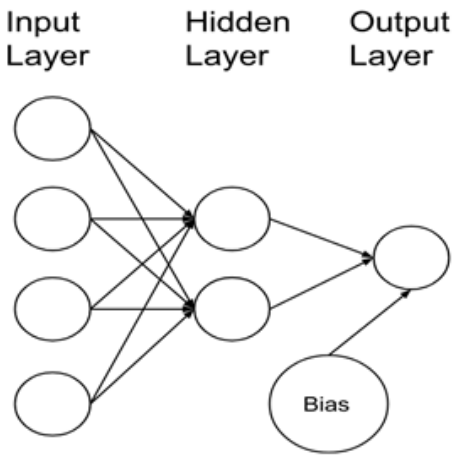

Figure 1. ANN with a single hidden layer

Source: Own work.

The output function of such an ANN is given as:

$$f_t(x_t, \theta) = F\left(\beta_0 + \sum_{j=1}^{q} G(x_t \gamma_j') \beta_j \right) \tag{2}$$

where F is the initial activation function, G is the hidden node activation function, β_j is the weights from hidden node j to the output unit, $x_t = \{x_{1,t}, \ldots, x_{h,t}\}$

is the $1 \times n$ vector of input variables at time t (with $n = h + 1$), β_0 is the bias of the final output, $\gamma_t = \{\gamma_{1,j}, \ldots, \gamma_{n,j}\}$ is the $1 \times n$ vector of weights for the links between the inputs and the hidden neuron j, $\theta = \{\beta_0, \ldots, \beta_q, \gamma_1', \ldots, \gamma_q'\}$ is the vector of all network weights, and finally q is the number of hidden nodes.

Nowadays, there is a wide variety of functions to choose for F and G. Generally, is an identity function and G is a logistic function (Bucci, 2020). However, for this paper, F is chosen to be the hyperbolic tangent function (tanh) in order to support the use of GPUs under the Keras framework (Keras Team, n.d.). G, on the other hand, is chosen to be a sigmoid function as it allows the network to learn more complex decision boundaries (Sheela & Deepa, 2013). Lastly, the optimizer algorithm used in this paper to determine the weights and biases is the Adam algorithm, a stochastic gradient descent method that is based on adaptive estimation of first-order and second-order moments.

The neural network exhibited in Figure 1 is usually referred to as a static network as such a neural network does not show any memory, even when sample information contains time dependence. That is why the so called Recurrent Neural Networks (RNN) are commonly used with time series data. RNNs allow internal feedback by propagating data from input to output, but also from later layers to earlier layers. One type of RNN is LSTM neural networks. The LSTM neural network is an extension of the RNN architecture by replacing each hidden node with a memory cell. Such a memory cell is depicted in Figure 2.

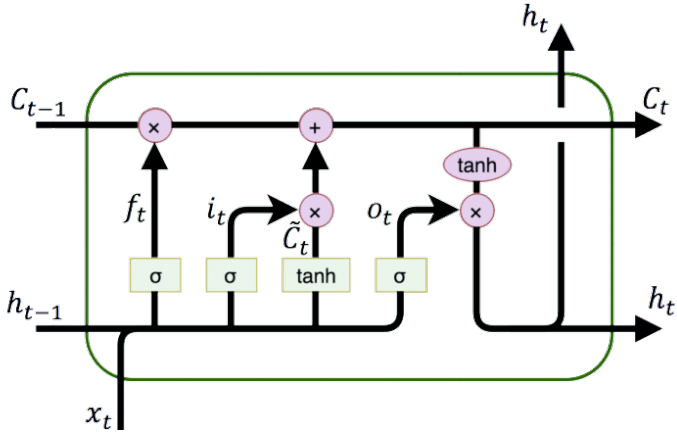

Figure 2. Basic LSTM memory cell

Source: Own work.

Each block has three multiplicative units termed input (i_t), forget (f_t), and output (o_t) gates as well as one cell input activation vector ($\tilde{c}_t$). Such gates enable the memory cells to store and retrieve data in order to choose which

data need to be permanently stored. The forget gates determine the amount of information from the earlier time step that will be retained and passed, whereas the input gates determine the amount of information from the current time step that will be retained and passed. The outputs gates, on the other hand, determine the information used to estimate the prediction (h_t) of the considered random variable (y_t).

In simple words, LSTM neural networks are almost equal to the neural network depicted in Figure 1, but they sequence the inputs in a time series manner ($x_{t-n}, \ldots, x_t$), and estimate the weights and biases through the parameters of the self-contained memory cell, and input, forget and output gates.

This system allows LSTM neural networks to keep important information from input signals while ignoring the pointless details. Thanks to this memory present in LSTM neural networks, they are the most commonly used type of ANN for time-series prediction (Bucci, 2020), and that is the primary reason for their choice in this paper. Besides their use for time-series prediction, they are also used in handwriting recognition (Graves et al., 2009), speech recognition (Li & Wu, 2015), machine translation (Wu et al., 2016), speech activity detection (Sahidullah et al., 2019), robot control (Mayer et al., 2006), video games (Rodriguez, 2018) and healthcare (Awais et al., 2021). LSTM neural networks can be mathematically represented as:

$$f_t = \sigma(W_f h_{t-1} + U_f x_t + b_f) \tag{3}$$

$$i_t = \sigma(W_i h_{t-1} + U_i x_t + b_i) \tag{4}$$

$$o_t = \sigma(W_o h_{t-1} + U_o x_t + V_o c_t + b_o) \tag{5}$$

$$\tilde{c}_t = tanh(W_c h_{t-1} + U_c x_t + b_c) \tag{6}$$

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t \tag{7}$$

$$h_t = o_t \odot \tanh(c_t) \tag{8}$$

$$\hat{y}_t = h_t \tag{9}$$

where W_f, W_i, W_c, W_o, U_f, U_i, U_c, and U_o respectively contain the weights of the input and recurrent connections. $x_t \in \mathbb{R}$ is the input vector to the LSTM unit, $f_t \in (0, 1)^h$ is the forget gate's activation vector, $i_t \in (0, 1)^h$ is the input gate's activation vector, and $o_t \in (0, 1)^h$ is the output gate's activation vector. Additionally, $h_t \in (-1, 1)^h$ is the output vector of the LSTM unit, $c_t \in \mathbb{R}^h$ is the cell state vector, and $\tilde{c}_t \in (-1, 1)^h$ is the cell input activation vector. Finally, σ is the sigmoid function, tanh is the hyperbolic tangent function, and $\odot$ is the Hadamard product.

2. Research design

In this chapter the sample choice, realized volatility and financial turbulence proxies, models selection, LSTM hyperparameters search space and the forecast assessment used in this paper are discussed.

2.1. Sample

The data set used in this research is composed of daily observations of the S&P 500 realized volatility (RV) from 01 November 2017 until 01 November 2022, five years in total. In the chosen timeframe there exist 1,257 daily observations. The data set is divided into training data, 80% of the total data, and test data, 20% of the total data. The training data is used to determine the model parameters while the test data is used to evaluate the model performance. Further, 12.5% of the training data is used as validation data to determine the hyperparameters of the LSTM model (see Section 2.4 for more details).

Additionally, two data sets composed of daily RV observations of the DJIA and NASDAQ from 01 November 2017 until 01 November 2022 are used in the robustness check (see Section 4 for more details) to test the results of Section 5. Once again the data set is divided into training data, 80% of the total data, and test data, 20% of the total data. Moreover, 12.5% of the training data is used as validation data.

Lastly, the data sources used to retrieve data for the input variables of the studied models can be found in Appendix 1.

2.2. Realized volatility and financial turbulence proxies

As already stated, Yang and Zhang's realized volatility is chosen as an RV proxy due to its robustness and accessibility. As proposed by Yang and Zhang (2000), RV is estimated by Equations (10), (11), (12), (13):

$$\sigma^2 = \sigma_O^2 + k\,\sigma_C^2 + (1-k)\,\sigma_{RS}^2 \tag{10}$$

$$RV = \sqrt{\sigma^2} \tag{11}$$

where:

$$\sigma_O^2 = \frac{1}{n-1}\sum_{i=1}^{n}(o_i - \bar{o}) \tag{12}$$

$$\sigma_C^2 = \frac{1}{n-1}\sum_{i=1}^{n}(c_i - \overline{c}) \tag{13}$$

with o_i = opening price at time i, $\overline{o}$ = opening price mean, c_i = close price at time i, $\overline{c}$ = close price mean, k = parameter, and σ_{RS}^2 = Rogers et al. (1994) variance estimation.

Yang & Zhang's (2000) empirical research indicates that the best k value is given as:

$$k = \frac{0.34}{1.34 + \frac{n+1}{n-1}} \tag{14}$$

Regarding financial turbulence, the proxy based on Mahalanobis distance as proposed by Kritzman and Li (2010) is used and is defined in Equation (1) in Section 1. This choice is motivated by Salisu et al. (2022) as they show that the use of FT can improve the out-of-sample predictive performance of stock market volatility linear models over both the short and long-time horizon.

2.3. Models selection

The time series of S&P 500 RV training data can be seen in Figure 3. It can be observed that this time series is stationary, and this is confirmed in Table 1 with the results of Augmented Dickey Fuller (ADF) test. Table 1 also shows the results of Breush-Godfrey (BG) test and Ljung-Box (LB) test considering

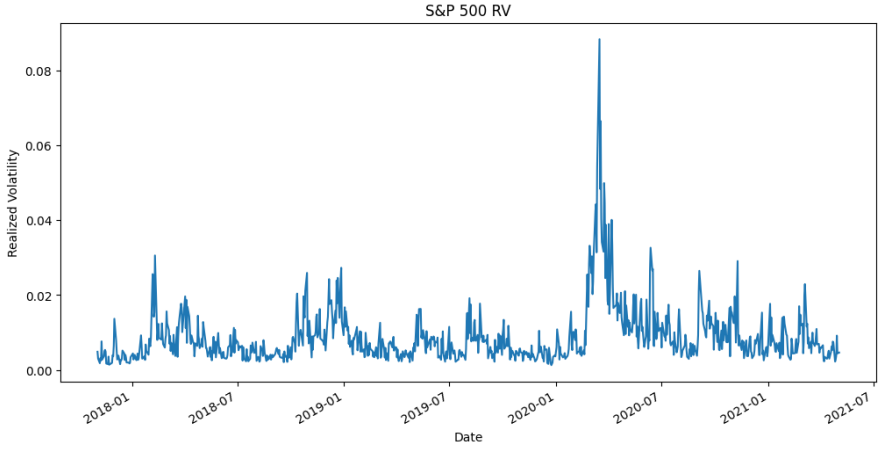

Figure 3. S&P 500 realized volatility time series

Source: Own work.

Table 1. Time series statistical tests

***p*-values**
AUF: $5.92E^{-6***}$
BG: lower than $2.03E^{-17***}$ for the first 25 lags
LB: lower than $1.00E^{-7***}$ for the first 25 lags

Source: Own work.

25 lags. These results prove that there exists a high autocorrelation persistence in the time series. Hence, this high persistence indicates that a long memory detecting model ought to be implemented (Rossi & Santucci de Magistris, 2014). That is another reason for the choice of LSTM neural networks for this research. This high persistence and stationarity also mean that common volatility models, e.g. GARCH and HAR models, can be used. Therefore, besides the studied LSTM models, GARCH(1,1), EGARCH(1,1) and HAR models are used in this study as comparison to the LSTM models.

Three different LSTM models are considered in this study. Their inputs are summarized below:

- Model 1: Past values of RV;
- Model 2: Past values of RV and FT;
- Model 3 (Bucci's model): Past values of RV, Dividend Yield Ratio S&P 500 (DP), Fama-French's Market Excess Return (MKT), Fama-French's Short-Term Reversal Factor (STR), and BAA and AAA bond yields Default Spread (DEF).

Model 1 is the simplest model that can be used to predict future values of RV, whereas Model 2 has additional information on FT. Model 3 (Bucci's model) is devised by Bucci (2020) through the use of the Least Absolute Shrinkage and Selection Operator (LASSO) regression. Bucci's model is chosen due to its high accuracy forecast performance for the S&P 500 realized volatility in Bucci's subsample from September 2007 until June 2009 (Bucci, 2020). It is important to notice that for DP and DEF, linear interpolation is used to estimate their daily development given their monthly periodicity. Incidentally, further on in this paper, Model 4 refers to GARCH(1,1), Model 5 refers to EGARCH(1,1), and Model 6 refers to HAR model.

2.4. LSTM hyperparameters search space

All ANNs have certain parameters that need to be chosen by humans, which are known as hyperparameters. Some examples of these hyperparameters are the number of hidden neurons, the error function, number of train-

ing rounds (known as epochs), etc. Currently, the best procedure for finding the optimal hyperparameters is separating a part of the training data, called the validation data, and training various ANNs with randomly chosen different hyperparameters within the hyperparameter search space, testing them with the validation data, and finally choosing the hyperparameters based on the best forecasting accuracy results. Table 2 shows the considered hyperparameters and their respective search space for the LSTM models of this study.

Table 2. Hyperparameters search space

Hyperparameters	Search space
Number of inputs	[21, 63, 84, 126, 189, 252]
Number of neurons	[14, 42, 56, 84, 126, 168]
Epochs	[3, 5, 10, 15]
Dropouts	[0, 0.2, 0.4]
Error Functions	Mean Squared Errors (MSE), Root Mean Squared Errors (RMSE), Huber Loss

Source: Own work.

2.5. Forecast assessment

In order to assess the model forecast accuracy in the test data, RMSE and mean percent error (MPE) are used. RMSE is explained by Equation (15), while MPE is defined by Equation (16):

$$RMSE = \sqrt{\frac{\sum_{t=1}^{T} (\hat{y}_t - y_t)^2}{N}} \tag{15}$$

$$MPE = \frac{\sum_{t=1}^{T} \frac{\left|\hat{y}_t - y_t\right|}{y_t}}{N} \tag{16}$$

Furthermore, the model's accuracy power of a certain model is assumed to be equal to Equation (16):

$$Model\ Accuracy = 1 - MPE \tag{17}$$

RMSE and Model Accuracy (MA) are chosen because they give more readily interpretable results than the commonly used error measure MSE. In addition, a brief statistical analysis of the models' residuals is performed.

Since ANNs can yield different forecast accuracy depending on where they start their gradient descent algorithm (Karsoliya & Azad, 2012; Vujičić et al., 2016), ten trials are performed for each LSTM model. Further, the average and standard deviation of RMSE and MA are taken to compare the LSTM models' forecast power. Moreover, Confidence Intervals (CI) of 95% are taken, under the assumption that ten means are enough to transform the means of the underlying random variables' distributions into normal distributions (i.e., enough to fulfill the Central Limit Theorem (CLT)).

Additionally, Mann-Whitney U (MWU) tests, T-tests and F-tests with the RMSE and MA trials results are performed for LSTM models. MWU tests and T-tests are used to determine whether a pair of LSTM models yields statistically significantly different forecast accuracy, with the null hypothesis (H_0) being that there is not statistically significant difference and the alternative hypothesis (H_1) being that there is statistically significant difference. F-tests are used to determine whether the variance in the forecast accuracy of a certain LSTM model is statistically significantly different to another LSTM model, with H_0 and H_1 being the same as for the aforementioned tests.

Finally, Diebold-Mariano (DM) tests are performed with the forecasts of the proposed LSTM model in this study (Model 2) and all other five models. This test is used to determine whether there exists a statistically significant difference in the forecasts of a pair of models (and this being H_1) or that this difference is simply due to the specific sample choice (and this being H_0). The choice of this test is motivated by the fact that this test makes use of fewer assumptions and is more robust than other commonly used statistical forecast comparison tests (Diebold & Mariano, 1995; Harvey et al., 1997). Lastly, for DM tests, the forecasts of the first trial of the LSTM models will be used.

3. Results and discussion

After the hyperparameters search through the use of the validation data, the optimal hyperparameters shown in Table 3 were chosen.

Table 4 shows the error measures results of all models. It can be seen that Model 2 outperforms all models regarding both RMSE and MA. Model 2 also yields less variability in its forecasts than Model 3 for all error measures and Model 2 for RMSE. Additionally, considering the 95% CI, even for the upper bound of RMSE and lower bound of MA, Model 2 outperforms all models. This outperformance can be graphically seen in Figure 4. Although it seems that Model 4 and Model 5 are the best models when looking at Figure 4, they are actually almost always predicting the next time-step value as being the current time-step value. As a result, they have the worst RMSE and MA results.

Table 3. Optimal hyperparameters

Hyperparameters	Model 1	Model 2	Model 3
Number of inputs	252	21	63
Number of neurons	56	56	56
Epochs	15	15	15
Dropouts	0.4	0.2	0.2
Error functions	RMSE	MSE	RMSE

Source: Own work.

Table 4. S&P 500 error measures results

Measures	Model 1 (%)	Model 2 (%)	Model 3 (%)	Model 4 (%)	Model 5 (%)	Model 6 (%)
RMSE average	0.339	0.318	0.339	0.401	0.418	0.340
RMSE standard deviation	0.009	0.004	0.025	–	–	–
RMSE 95% CI	0.321 ≤ X ≤ 0.356	0.310 ≤ X ≤ 0.327	0.290 ≤ X ≤ 0.389	–	–	–
MA average	70.10	72.12	68.97	61.14	58.81	66.11
MA standard deviation	0.21	0.60	2.76	–	–	–
MA 95% CI	69.67 ≤ X ≤ 70.53	70.92 ≤ X ≤ 73.32	63.45 ≤ X ≤ 74.48	–	–	–

Source: Own work.

On the other hand, Table 5 shows the results of the statistical tests for the LSTM models. Given the *p*-values in Table 5, it can be concluded that Model 2 yields on average better forecast accuracy than Model 1 and Model 3, though the H_0 between Model 2 and Model 3 considering RMSE cannot be rejected using the conservative *p*-value of 1%. Furthermore, the *p*-values of the F-tests indicate that Model 2 yields more robust forecasts than Model 1 and Model 3, besides between Model 1 and Model 2 when considering RMSE.

Table 6 and Table 7 respectively show the statistical analysis of the forecast residuals and the DM test results. It can be seen in Table 6 that Model 2 again outperforms the other models when considering the residuals 95% CI, though Model 2 had a slightly more extreme minimum residual value than Model 6. The p-values of Table 7 indicate that there exists a statistically significant difference in the forecasts of Model 2 in respect to all other models. This presumably means that Model 2 yields statistically better forecasts than all other considered models.

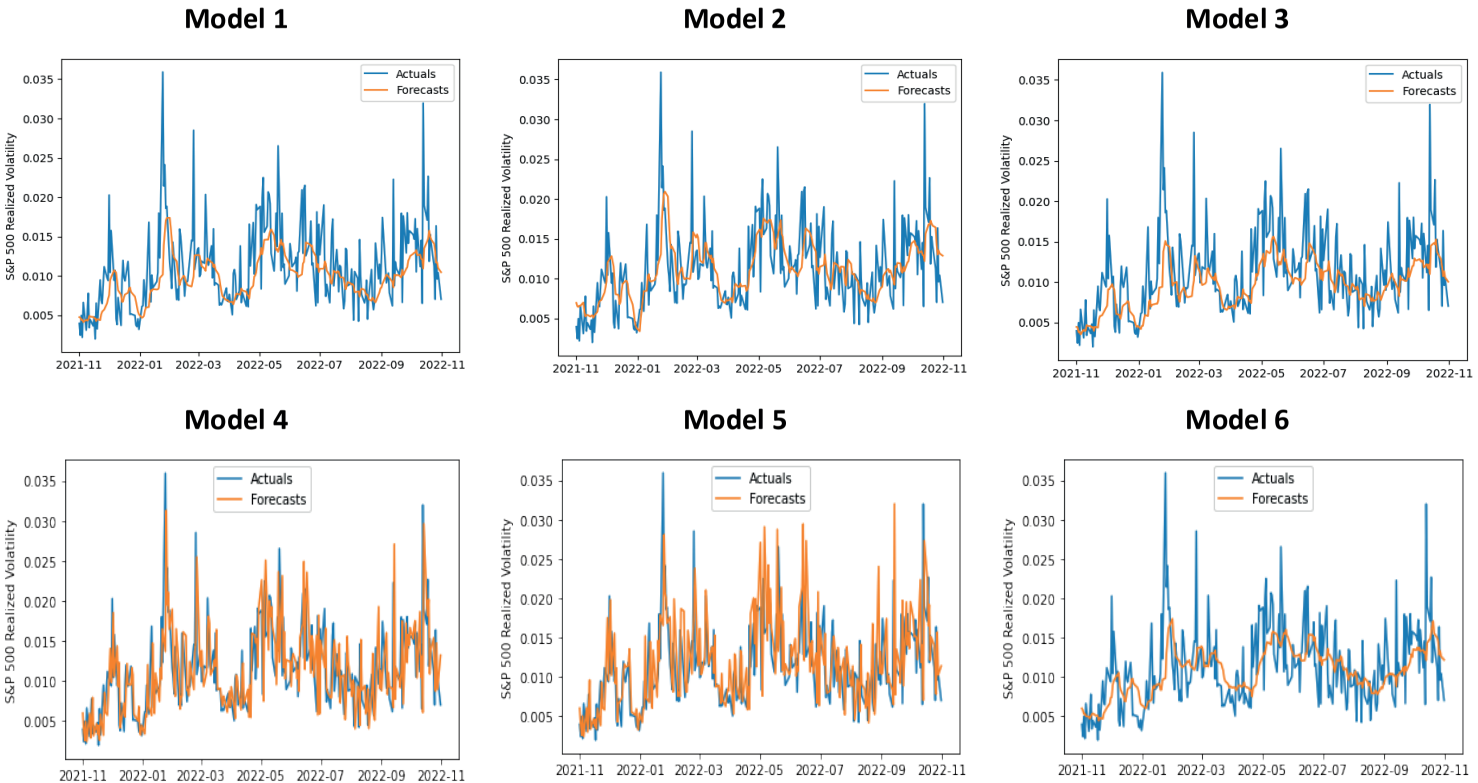

Figure 4. Out-sample forecasts performance

Source: Own work

Table 5. S&P 500 statistical tests for LSTM models

	Model 2	**Model 3**
Model 1	**RMSE** MWU p-value: 1.57E^{-4***} T-test p-value: 1.36E^{-5***} F-test p-value: 0.053* **MA** MWU p-value: 1.57E^{-4***} T-test p-value: 6.01E^{-7***} F-test p-value: 0.005***	RMSE MWU p-value: 0.449 T-test p-value: 0.949 F-test p-value: 0.004*** **MA** MWU p-value: 0.762 T-test p-value: 0.226 F-test p-value: 1.29E^{-8***}
Model 2	-	**RMSE** MWU p-value: 0.015** T-test p-value: 0.027** F-test p-value: 1.51E^{-5***} **MA** MWU p-value:0.001*** T-test p-value: 0.006*** F-test p-value: 1.02E^{-4***}

Source: Own work.

Table 6. S&P 500 residuals analysis

	Model 1 (%)	**Model 2 (%)**	**Model 3 (%)**	**Model 4 (%)**	**Model 5 (%)**	**Model 6 (%)**
Residuals Mean	0.15	0.02	0.29	–0.01	–0.07	0.05
Residuals Standard Deviation	0.45	0.43	0.43	0.55	0.57	0.46
Residuals Skew	1.50	1.39	1.65	0.61	0.21	1.49
Residuals Kurtosis	4.64	4.40	5.25	2.97	2.81	4.46
Residuals max	2.56	2.29	2.72	2.59	2.49	2.61
Residuals min	–0.59	–0.87	–0.48	–1.83	–2.32	–0.75
Residuals 95% (upper)	1.05	0.88	1.16	1.10	1.07	0.97
Residuals 95% (lower)	–0.74	–0.84	–0.58	–1.11	–1.22	–0.87

Source: Own work.

Table 7. S&P 500 Model 2 DM Tests

Model	**Model 1**	**Model 3**	**Model 4**	**Model 5**	**Model 6**
***p*-values**	0.001***	4.79E-6***	1.09E-5***	7.73E-5***	0.003***

Source: Own work.

4. Robustness check

Despite the promising results of Section 4, robust tests are still needed to confirm these results. Therefore, the same procedure is performed in this section as explained in Section 3 with different stocks: DJIA and NASDAQ. DJIA is chosen to reflect stocks with a lower volatility than the S&P 500, whilst NASDAQ is chosen to reflect stocks with a higher volatility than the S&P 500.

Table 8 shows the error measures results for NASDAQ and DJIA. Regarding NASDAQ, the results are similar to Section 4; that is, Model 2 outperforms all models considering averages and the 95% CI whilst yielding less forecast variability than Model 3 for both measures and than Model 1 for RMSE. Regarding DJIA, Model 2 outperforms all models considering RMSE, apart from Model 1 and Model 6. On the other hand, considering MA, Model 2 outperforms all models, albeit Model 2 yields more forecast variability than Model 1. Additionally, Model 2 does not outperform Model 1 regarding the 95% CI of MA.

In addition, the p-values for the statistical tests for the LSTM models are found in Table 9. It can be observed that the results for NASDAQ are convergent to the results in Section 3. The only exception is the F-test results between Model 1 and Model 2. The results for DJIA, on the other hand, diverge from Section 3. The only similarity is the F-test results between Model 2 and Model 3. It could be thus hypothesized that Model 2 performs relatively better than the other considered models only when applied stocks with middle-to-high volatility

Finally, Table 10 and Table 11 respectively show the statistical analysis of the forecast residuals and the DM test results. Once more, it can be seen in Table 10 that Model 2 outperforms the other models for both NASDAQ and DJIA when considering the residuals 95% CI and extreme residual values. Moreover, the p-values in Table 11 indicate that there exists a statistically significant difference in the forecasts of Model 2 in respect to all other models besides to Model 3 when considering NASDAQ. This deviation from Section 3 results presumably comes from the fact that only the first trial of the LSTM models was used for the DM tests. Hence, there exists the possibility that though generally Model 2 and Model 3 yield statistically significantly different forecasts, an exception was selected and used for DM tests by chance. This hypothesizes is likely to be true given the results in Table 8 and Table 9.

Table 8. Error measures results

	Measures	Model 1 (%)	Model 2 (%)	Model 3 (%)	Model 4 (%)	Model 5 (%)	Model 6 (%)
NASDAQ	RMSE average	0.494	0.479	0.495	0.589	0.606	0.487
	RMSE standard deviation	0.010	0.003	0.014	–	–	–
	RMSE 95% CI	0.475 ≤ X ≤ 0.513	0.472 ≤ X ≤ 0.486	0.468 ≤ X ≤ 0.522	–	–	–
	MA average	69.75	72.19	69.17	60.25	58.47	66.32
	MA standard deviation	0.45	0.94	3.17	–	–	–
	MA 95% CI	68.84 ≤ X ≤ 70.65	70.31 ≤ X ≤ 74.07	62.84 ≤ X ≤ 75.51	–	–	–
DJIA	RMSE average	0.283	0.283	0.296	0.327	0.340	0.282
	RMSE standard deviation	0.011	0.007	0.041	–	–	–
	RMSE 95% CI	0.261 ≤ X ≤ 0.305	0.270 ≤ X ≤ 0.297	0.213 ≤ X ≤ 0.379	–	–	–
	MA average	72.62	73.01	67.85	65.58	63.29	69.27
	MA standard deviation	0.80	1.22	9.11	–	–	–
	MA 95% CI	71.03 ≤ X ≤ 74.21	70.58 ≤ X ≤ 75.45	49.63 ≤ X ≤ 86.06	–	–	–

Source: Own work.

Table 9. Statistical tests for LSTM models

NASDAQ	Model 2	Model 3
Model 1	**RMSE** MWU p-value: 5.07E^{-4***} T-test p-value: 9.13E^{-4***} F-test p-value: 0.005*** **MA** MWU p-value: 3.81E^{-4***} T-test p-value: 5.35E^{-6***} F-test p-value: 0.040**	**RMSE** MWU p-value: 0.545 T-test p-value: 0.802 F-test p-value: 0.299 **MA** MWU p-value: 0.174 T-test p-value: 0.585 F-test p-value: 2.89E^{-6***}
Model 2	-	**RMSE** MWU p-value: 0.006*** T-test p-value: 0.005*** F-test p-value: 3.29E^{-4***} **MA** MWU p-value: 8.81E^{-4***} T-test p-value: 0.015** F-test p-value: 0.001***

DJIA	Model 2	Model 3
Model 1	**RMSE** MWU p-value: 0.545 T-test p-value: 0.854 F-test p-value: 0.163 **MA** MWU p-value: 0.096* T-test p-value: 0.403 F-test p-value: 0.220	RMSE MWU p-value: 0.496 T-test p-value: 0.361 F-test p-value: 4.92E^{-4***} **MA** MWU p-value: 0.406 T-test p-value: 0.133 F-test p-value: 3.70E^{-8***}
Model 2	-	**RMSE** MWU p-value: 0.597 T-test p-value: 0.381 F-test p-value: 8.41E^{-6***} **MA** MWU p-value:0.082* T-test p-value: 0.108 F-test p-value: 1.59E^{-6***}

Source: Own work.

Table 10. S&P 500 residuals analysis

		Model 1 (%)	Model 2 (%)	Model 3 (%)	Model 4 (%)	Model 5 (%)	Model 6 (%)
NASDAQ	Residuals Mean	0.19	0.14	0.39	0.01	–0.05	0.09
	Residuals Standard Deviation	0.64	0.44	0.62	0.82	0.84	0.65
	Residuals Skew	1.32	1.33	1.43	0.25	0.06	1.32
	Residuals Kurtosis	3.05	3.08	3.28	1.88	2.02	2.88
	Residuals max	3.05	2.09	3.24	3.23	3.18	3.06
	Residuals min	–1.04	–0.69	–0.73	–2.75	–3.21	–1.10
	Residuals 95% (upper)	1.47	1.02	1.63	1.65	1.63	1.38
	Residuals 95% (lower)	–1.08	–0.75	–0.85	–1.63	–1.72	–1.21
DJIA	Residuals Mean	0.06	0.10	0.04	0.01	–0.05	0.03
	Residuals Standard Deviation	0.38	0.29	0.39	0.46	0.48	0.39
	Residuals Skew	1.44	1.45	1.26	0.58	0.15	1.42
	Residuals Kurtosis	4.15	3.89	4.02	3.03	3.08	4.12
	Residuals max	1.97	1.55	2.15	2.20	2.16	2.07
	Residuals min	–0.61	–0.39	–0.76	–1.46	–1.95	–0.63
	Residuals 95% (upper)	0.81	0.68	0.82	0.93	0.91	0.81
	Residuals 95% (lower)	–0.70	–0.48	–0.75	–0.92	–1.02	–0.75

Source: Own work.

Table 11. S&P 500 Model 2 DM tests

Model	Model 1	Model 3	Model 4	Model 5	Model 6
NASDAQ *p*-values	$9.00E^{-7***}$	0.913	$4.50E^{-11***}$	$3.23E^{-11***}$	$4.64E^{-17***}$
DJIA *p*-values	$5.80E^{-5***}$	$6.39E^{-4***}$	$6.80E^{-6***}$	$1.35E^{-4***}$	$5.89E^{-5***}$

Source: Own work.

Conclusions

The aim of this paper is to compare the performance of a novel volatility forecasting model with current commonly used models. The novel volatility forecasting model makes use of Long Short-Term Memory (LSTM) neural networks and the risk parameter financial turbulence (FT). This paper utilizes the robust yet simple Yang and Zhang's realized volatility (RV) proxy and considers a total of six different models. These models and their respective inputs can be seen below:

- LSTM Model 1: Inputs = past values of RV;
- LSTM Model 2: Inputs = past values of RV and FT;
- LSTM Model 3 (Bucci's model): Inputs = past values of RV, Dividend Yield Ratio S&P 500 (DP), Fama-French's Market Excess Return (MKT), Fama-French's Short-Term Reversal Factor (STR), and BAA and AAA bond yields Default Spread (DEF);
- Model 4: GARCH(1,1);
- Model 5: EGARCH(1,1);
- Model 6: HAR.

The sample period used in this research is from 01 November 2017 until 01 November 2022. S&P 500 daily RV observations are used as the main sample, yet NASDAQ and DJIA daily RV observations are also used in a robustness check to determine the robustness of the models' results achieved with S&P 500 RV.

The error measures used to assess the models' forecast accuracy are Root Mean Squared Error (RMSE) and Model Accuracy (MA). MA is defined as one minus Mean Percentage Error (MPE). Further, a simple statistical analysis of the models' forecast residuals is performed to assess their distribution and extreme values. In addition, Diebold-Mariano (DM) tests are performed with the forecasts of the proposed LSTM model in this study (Model 2) and all other five models. This test is used to determine whether there exists a statistically significant difference in the forecasts of a pair of models or that this difference is simply due to the specific sample choice.

Furthermore, to account for the fact that ANNs can yield different forecast accuracy depending on where they start their gradient descent algorithm (Karsoliya & Azad, 2012; Vujičić et al., 2016), ten trials are performed for each LSTM model. The average and standard deviation of RMSE and MA are taken to compare the LSTM models' forecast power and create Confidence Intervals (CI) of 95% under the Central Limit Theorem (CLT) assumption. Further, Mann--Whitney U (MWU) tests, T-tests and F-tests with the RMSE and MA trials results are performed for the LSTM models. MWU tests and T-tests are used to determine whether a pair of LSTM models yield statistically significantly different forecast accuracy. F-tests are used to determine whether the variance in the forecast accuracy of a certain LSTM model is statistically significantly different to another LSTM model.

The results of this research indicate that the proposed model (Model 2) yields statistically significantly more accurate and robust forecasts than all other studied models when applied to stocks with middle-to-high volatility. Yet, when considering stocks with low volatility, it can only be confidently said that Model 2 yields statistically significantly more robust forecast results than all other considered models. It could be hypothesized that Model 2 performs relatively better than the other considered models when applied to stocks with middle-to-high volatility, yet this is not the case with stocks with low volatility. However, more research would be needed to properly test this hypothesis. Thus, the authors of this paper invite the scientific community to perform such a study with the proposed model.

Finally, it is important to notice that this study has its limitations. For instance, the assumption that ten trials were enough to fulfill CLT conditions when estimating the RMSE and MA 95% CI can easily be considered a weakness of this study. Not exploring other sample periods and ANNs types, and only exploring stock indexes are also other limitations of this study. Another limitation of this study is the fact that DM tests are only performed for the LSTM models forecasts of the first trial and not of all trials (or an average thereof). This limitation can be clearly seen in Section 5 where the DM test result for Model 2 in respect to Model 3 shows no statistically significant difference although all other error measures and statistical tests results indicate the opposite. Therefore, the scientific community is invited to further explore the proposed model whilst partially or fully addressing the limitations of this study.

Data availability and code

The Python code used for the proposed LSTM model plus all data, code, calculations and results of this research can be found on https://github.com/hugogobato/Forecasting-Realized-Volatility-through-Financial-Turbulence-and-Neural-Networks.git.

Appendix

Table A1. Variables description and sources

Symbol	Variable	Description	Source
RV	Yang and Zhang's Realized Volatility	a robust yet simple realized volatility proxy	estimated with prices retrieved from Compustat – Capital IQ
FT	S&P 500 Financial Turbulence	risk parameter proposed by Kritzman & Li (2010)	estimated with prices retrieved from Compustat – Capital IQ
DP	Dividend Yield Ratio S&P 500	dividends over the past year relative to current market prices	Nasdaq Data link
MKT	Market Excess Return	Fama-French's market factor: Return of U.S. stock market minus one--month T-Bill rate	Kenneth French's website
STR	Short-Term Reversal Factor	Fama-French's STR: Average return on stocks with low prior return minus average return on stock with high prior return	Kenneth French's website
DEF	Default Spread	measure of default risk of corporate bonds: difference of BAA and AAA bond yields	Federal Reserve Bank of St. Louis

Source: Adapted from (Bucci, 2020).

References

Aaltio, J. (2022). Volatility forecasting with artificial neural networks [unpublished PHD dissertation]. Hanken School of Economics. https://helda.helsinki.fi/dhanken/bitstream/handle/10227/509483/Aaltio_Juho.pdf?sequence=1

Andersen, T. M., & Bollerslev, T. (1998). Answering the skeptics: Yes, standard volatility models do provide accurate forecasts. *International Economic Review*, *39*(4), 885. https://doi.org/10.2307/2527343

Awais, M., Raza, M., Singh, Y., Bashir, K., Manzoor, U., Islam, S., & Rodrigues, J. J. P. C. (2021). LSTM-based emotion detection using physiological signals: IoT framework for healthcare and distance learning in COVID-19. *IEEE Internet of Things Journal*, *8*(23), 16863–16871. https://doi.org/10.1109/jiot.2020.3044031

Baffour, A. A., Feng, J., & Taylor, E. K. (2019). A hybrid artificial neural network-GJR modeling approach to forecasting currency exchange rate volatility. *Neurocomputing*, *365*, 285–301. https://doi.org/10.1016/j.neucom.2019.07.088

Bauwens, L., Laurent, S., & Rombouts, J. V. (2006). Multivariate GARCH models: A survey. *Journal of Applied Econometrics*, *21*(1), 79–109. https://doi.org/10.1002/jae.842

Black, F. (1986). Noise. *Journal of Finance*, *41*, 529–543.

Bollerslev, T. (1986). Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, *31*(3), 307–327. https://doi.org/10.1016/0304-4076(86)90063-1

Borup, D., & Jakobsen, J. S. (2019). Capturing volatility persistence: A dynamically complete realized EGARCH-MIDAS model. *Quantitative Finance*, *19*(11), 1839–1855. https://doi.org/10.1080/14697688.2019.1614653

Brandt, M. W., & Jones, C. W. (2006). Volatility forecasting with range-based EGARCH models. *Journal of Business & Economic Statistics*, *24*(4), 470–486. https://doi.org/10.1198/073500106000000206

Bucci, A. (2020). Realized volatility forecasting with neural networks. *Journal of Financial Econometrics*, *18*(3), 502–531. https://doi.org/10.1093/jjfinec/nbaa008

Chen, Q., & Robert, C. (2022). *Multivariate realized volatility forecasting with graph neural network*. Proceedings of the Third ACM International Conference on AI in Finance. https://doi.org/10.1145/3533271.3561663

Chen, W., Yao, J., & Shao, Y. (2022). Volatility forecasting using deep neural network with time-series feature embedding. *Ekonomska Istrazivanja–Economic Research*, 1377–1401. https://doi.org/10.1080/1331677x.2022.2089192

D'Ecclesia, R. L., & Clementi, D. (2021). Volatility in the stock market: ANN versus parametric models. *Annals of Operations Research*, *299*(1–2), 1101–1127. https://doi.org/10.1007/s10479-019-03374-0

Diebold, F. X., & Mariano, R. S. (1995). Comparing predictive accuracy. *Journal of Business & Economic Statistics*, *13*(3), 253–263. https://doi.org/10.2307/1392185

Donaldson, R. G., & Kamstra, M. J. (1996a). Forecast combining with neural networks. *Journal of Forecasting*, *15*(1), 49–61. https://doi.org/10.1002/(SICI)1099-131X(199601)15:1<49::AID-FOR604>3.0.CO;2-2

Donaldson, R. G., & Kamstra, M. J. (1996b). A new dividend forecasting procedure that rejects bubbles in asset prices: The case of 1929's stock crash. *Review of Financial Studies*, *9*(2), 333–383. https://doi.org/10.1093/rfs/9.2.333

Donaldson, R. G., & Kamstra, M. J. (1997). An artificial neural network-GARCH model for international stock return volatility. *Journal of Empirical Finance*, *4*(1), 17–46. https://doi.org/10.1016/s0927-5398(96)00011-4

Engle, R. F. (1982). Autoregressive conditional heteroscedasticity with estimates of the variance of United Kingdom inflation. *Econometrica*, *50*(4), 987–1007. https://doi.org/10.2307/1912773

Engle, R. F., Ghysels, E., & Sohn, B. (2013). Stock market volatility and macroeconomic fundamentals. *The Review of Economics and Statistics*, *95*(3), 776–797. https://doi.org/10.1162/rest_a_00300

Gajdka, J., & Pietraszewski, P. (2017). Stock price volatility and fundamental value: Evidence from Central and Eastern European countries. *Economics and Business Review*, *3*(4), 28–46. https://doi.org/10.18559/ebr.2017.4.2

Garman, M. B., & Klass, M. J. (1980). On the estimation of security price volatilities from historical data. *The Journal of Business*, *53*(1), 67–68. https://doi.org/10.1086/296072

Graves, A., Liwicki, M., Fernández, S., Bertolami, R., Bunke, H., & Schmidhuber, J. (2009). A novel connectionist system for unconstrained handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *31*(5), 855–868. https://doi.org/10.1109/tpami.2008.137

Hajizadeh, E., Seifi, A., Zarandi, M. H. F., & Turksen, I. (2012). A hybrid modeling approach for forecasting the volatility of S&P 500 Index return. *Expert Systems with Applications*, *39*(1), 431–436. https://doi.org/10.1016/j.eswa.2011.07.033

Hamid, A., & Iqbal, Z. (2004). Using neural networks for forecasting volatility of S&P 500 Index futures prices. *Journal of Business Research*, *57*(10), 1116–1125. https://doi.org/10.1016/s0148-2963(03)00043-2

Harvey, D., Leybourne, S. J., & Newbold, P. (1997). Testing the equality of prediction mean squared errors. *International Journal of Forecasting*, *13*(2), 281–291. https://doi.org/10.1016/s0169-2070(96)00719-4

Haugom, E., Westgaard, S., Solibakke, P. B., & Lien, G. (2010). *Modelling day ahead Nord Pool forward price volatility: Realized volatility versus GARCH models*. International Conference on the European Energy Market. https://doi.org/10.1109/eem.2010.5558687

Hu, Y., Ni, J., & Wen, L. (2020). A hybrid deep learning approach by integrating LSTM-ANN networks with GARCH model for copper price volatility prediction. *Physica D: Nonlinear Phenomena*, *557*, 124907. https://doi.org/10.1016/j.physa.2020.124907

Kambouroudis, D. S., McMillan, D. G., & Tsakou, K. (2016). Forecasting stock return volatility: A comparison of GARCH, implied volatility, and realized volatility models. *Journal of Futures Markets*, *36*(12), 1127–1163. https://doi.org/10.1002/fut.21783

Kamijo, K., & Tanigawa, T. (1990). *Stock price pattern recognition-a recurrent neural network approach*. 1990 IJCNN International Joint Conference on Neural Networks. https://doi.org/10.1109/ijcnn.1990.137572

Karsoliya, S., & Azad, M. (2012). Approximating number of hidden layer neurons in multiple hidden layer BPNN architecture. *International Journal of Engineering*

Trends and Technology, *3*(6). http://www.ijettjournal.org/volume-3/issue-6/IJETT-V3I6P206.pdf

Keras Team. (n.d.). *Keras documentation: LSTM layer*. Keras.io. https://keras.io/api/layers/recurrent_layers/lstm/

Khan, A. I. (2011). Financial Volatility forecasting by nonlinear support vector machine heterogeneous autoregressive model: Evidence from Nikkei 225 Stock Index. *International Journal of Economics and Finance*. https://doi.org/10.5539/ijef.v3n4p138

Kritzman, M., & Li, Y. (2010). Skulls, financial turbulence, and risk management. *Financial Analysts Journal*, *66*(5), 30–41. https://doi.org/10.2469/faj.v66.n5.3

Latoszek, M., & Ślepaczuk, R. (2020). Does the inclusion of exposure to volatility into diversified portfolio improve the investment results? Portfolio construction from the perspective of a Polish investor. *Economics and Business Review*, *6*(1), 46–81. https://doi.org/10.18559/ebr.2020.1.3

Li, J. (2022). *The comparison of LSTM, LGBM, and CNN in stock volatility prediction*. Proceedings of the 2002 7th International Conference on Financial Innovation and Economic Development (ICFIED 2022). https://doi.org/10.2991/aebmr.k.220307.147

Li, X., & Wu, X. (2015). *Constructing long short-term memory based deep recurrent neural networks for large vocabulary speech recognition*. International Conference on Acoustics, Speech, and Signal Processing (ICASSP). https://doi.org/10.1109/icassp.2015.7178826

Lin, Y., Lin, Z., Liao, Y., Li, Y., Xu, J., & Yan, Y. (2022). Forecasting the realized volatility of stock price index: A hybrid model integrating CEEMDAN and LSTM. *Expert Systems with Applications*, *206*, 117736. https://doi.org/10.1016/j.eswa.2022.117736

Liu, R., Demirer, R., Gupta, R., & Tiwari, A. K. (2020). Volatility forecasting with bivariate multifractal models. *Journal of Forecasting*, *39*(2), 155–167. https://doi.org/10.1002/for.2619

Liu, X., Yang, H., Gao, J., & Wang, C. (2021). FinRL: Deep reinforcement learning framework to automate trading in quantitative finance. *Social Science Research Network*. https://doi.org/10.2139/ssrn.3955949

Loang, O. K., & Ahmad, Z. (2021). Does volatility mediate the impact of analyst recommendations on herding in Malaysian stock market? *Economics and Business Review*, *7*(4), 54–71. https://doi.org/10.18559/ebr.2021.4.4

Maciel, L., Gomide, F., & Ballini, R. (2016). Evolving fuzzy-GARCH approach for financial volatility modeling and forecasting. *Computational Economics*, *48*(3), 379–398. https://doi.org/10.1007/s10614-015-9535-2

Mayer, H., Gomez, F., Wierstra, D., Nagy, I., Knoll, A., & Schmidhuber, J. (2006). A system for robotic heart surgery that learns to tie knots using Recurrent Neural Networks. *Advanced Robotics*, *22*(13–14), 1521–1537. https://doi.org/10.1163/156855308x360604

Naidu, G. P., & Govinda, K. (2018). *Bankruptcy prediction using neural networks*. 2018 2nd International Conference on Inventive Systems and Control (ICISC). https://doi.org/10.1109/icisc.2018.8399072

Nystrup, P., Boyd, S., Lindström, E., & Madsen, H. (2019). Multi-period portfolio selection with drawdown control. *Annals of Operations Research*, *282*(1–2), 245–271. https://doi.org/10.1007/s10479-018-2947-3

Nystrup, P., Madsen, H., & Lindström, E. (2018). Dynamic portfolio optimization across hidden market regimes. *Quantitative Finance*, *18*(1), 83–95. https://doi.org/10.1080/14697688.2017.1342857

Parkinson, M. H. (1980). The extreme value method for estimating the variance of the rate of return. *The Journal of Business*, *53*(1), 61–65. https://doi.org/10.1086/296071

Rodikov, G., & Antulov-Fantulin, N. (2022). *Can LSTM outperform volatility-econometric models?* ArXiv Preprint. https://doi.org/10.48550/arXiv.2202.11581

Rodriguez, J. (2018, July). The science behind OpenAI Five that just produced one of the greatest breakthrough in the history of AI. *Towards Data Science*. https://www.linkedin.com/pulse/science-behind-openai-five-just-produced-one-greatest-jesus-rodriguez/

Rogers, L. C. G., & Satchell, S. (1991). Estimating variance from high, low and closing prices. *Annals of Applied Probability*, *1*(4), 504–512. https://doi.org/10.1214/aoap/1177005835

Rogers, L. C. G., Satchell, S., & Yoon, Y. (1994). Estimating the volatility of stock prices: A comparison of methods that use high and low prices. *Applied Financial Economics*, *4*(3), 241–247. https://doi.org/10.1080/758526905

Rossi, E., & De Magistris, P. S. (2014). Estimation of long memory in integrated variance. *Econometric Reviews*, *33*(7), 785–814. https://doi.org/10.1080/07474938.2013.806131

Sahidullah, M., Patino, J., Cornell, S., Yin, R., Sivasankaran, S., Bredin, H., Korshunov, P., Brutti, A., Serizel, R., Vincent, E., Evans, N., Marcel, S., Squartini, S., & Barras, C. (2019). *The speed submission to DIHARD II: Contributions & lessons learned*. HAL (Le Centre Pour La Communication Scientifique Directe). https://hal.inria.fr/hal-02352840v2/file/Speed_DIHARDII_Manuscript.pdf

Salisu, A. A., Demirer, R., & Gupta, R. (2022). Financial turbulence, systemic risk and the predictability of stock market volatility. *Global Finance Journal*, *52*, 100699. https://doi.org/10.1016/j.gfj.2022.100699

Sheela, K. G., & Deepa, S. N. (2013). Review on methods to fix number of hidden neurons in neural networks. *Mathematical Problems in Engineering*, 425740. https://doi.org/10.1155/2013/425740

Souto, H.G. (2023a) Distribution analysis of S&P 500 financial turbulence. *Journal of Mathematical Finance*, *13*, 67–88. https://doi.org/10.4236/jmf.2023.131005

Souto, H.G. (2023b) Time series forecasting models for S&P 500 financial turbulence. *Journal of Mathematical Finance*, *13*, 112–129. https://doi.org/10.4236/jmf.2023.131007

Vidal, A., & Kristjanpoller, W. (2020). Gold volatility prediction using a CNN-LSTM approach. *Expert Systems with Applications*, *157*, 113481. https://doi.org/10.1016/j.eswa.2020.113481

Vujičić, T. M., Matijević, T., Ljucović, J., Balota, A., & Sevarac, Z. (2016). *Comparative analysis of methods for determining number of hidden neurons in artificial neural network*. Central European Conference on Information and Intelligent Systems.

White. (1988). *Economic prediction using neural networks: The case of IBM daily stock returns*. IEEE 1988 International Conference on Neural Networks. https://doi.org/10.1109/icnn.1988.23959

Wilson, R. K., & Sharda, R. (1994). Bankruptcy prediction using neural networks. *Decision Support Systems*, *11*(5), 545–557. https://doi.org/10.1016/0167-9236(94)90024-8

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A. S., Johnson, M., Liu, X., Kaiser, Ł., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., ..., Dean, J. (2016). *Google's neural machine translation system: Bridging the gap between human and machine translation*. ArXiv. https://arxiv.org/pdf/1609.08144.pdf

Yan, Y., & Yang, D. (2021). A stock trend forecast algorithm based on deep neural networks. *Scientific Programming*, 1–7. https://doi.org/10.1155/2021/7510641

Yang, D., & Zhang, Q. (2000). Drift independent volatility estimation based on high, low, open, and close prices. *The Journal of Business*, *73*(3), 477–492. https://doi.org/10.1086/209650

Zhu, X., Wang, H., Xu, L., & Li, H. (2008). Predicting stock index increments by neural networks: The role of trading volume under different horizons. *Expert Systems with Applications*, *34*(4), 3043–3054. https://doi.org/10.1016/j.eswa.2007.06.023