Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

RESEARCH ARTICLE

Multivariate and Univariate Prediction of Stock Prices using an Optimized Gated Recurrent Unit with a Time Lag Proportional to the Wavelet Approximation Coefficient

Luyandza Sindi Mamba*[1] | Antony Ngunyi[2] | Lawrence Nderu[3]

[1]Department of Mathematics, Institute for Basic Sciences, Technology and Innovation, Nairobi, Kenya, Email: sindi.luyandza@students.jkuat.ac.ke

[2]Department of Statistics and Actuarial Science, Dedan Kimathi University of Technology, Nyeri, Kenya, Email: antonyngunyi@gmail.com

[3]School of Computing and Information Technology, Department of Computing, Jomo Kenyatta University of Agriculture and Technology, Nairobi, Kenya, Email: lnderu@jkuat.ac.ke

Correspondence
*Luyandza Sindi Mamba. Email: sindi.luyandza@students.jkuat.ac.ke

Abstract

The advancement of precise prediction models is still very helpful across a wide range of fields. Deep learning models have demonstrated strong performance and great accuracy in stock price prediction. However, the vanishing gradient problem, which some activation functions have exacerbated, has a significant impact on these models. In order to combat poor convergence, disappearing gradients, and significant error metrics, this study suggests using the Optimized Gated Recurrent Unit (OGRU) model with a scaled mean Approximation Coefficient (AC) time lag. This study employed the Rectified Linear Unit (ReLU), Hyperbolic Tangent (Tanh), Sigmoid and Exponential Linear Unit (ELU) activation functions. Real-life datasets were used including the daily Apple and 5-minute Netflix closing stock prices, decomposed using the Stationary Wavelet Transform (SWT). The decomposed series formed a multivariate model which was compared to a univariate model with similar hyper-parameters and different default lags. The Apple daily dataset performed well with a Default_1 lag, using a univariate model and the ReLU, attaining 0.01312, 0.00854 and 3.67 minutes for RMSE, MAE and runtime. The Netflix data performed best with the MeanAC_42 lag, using a multivariate model and the ELU achieving 0.00620, 0.00487 and 3.01 minutes for the same metrics. The study concluded that the OGRU is made resilient to the vanishing gradient problem by avoiding the Sigmoid activation function and applying the proposed lag on high frequency data with the ELU activation function using decomposed data.

KEYWORDS:
Optimized Gated Recurrent Unit; Approximation Coefficient; Stationary Wavelet Transform; Activation Function; Time Lag

1 | INTRODUCTION

According to the Efficient Market Hypothesis (EMH), financial time series are virtually always unpredictable since every significant piece of information, including past values and volumes, that can affect the price, is already taken into consideration. This indicates that the price is independent of any trend or pattern and responds fast to new information. The stock price will always

[0]**Abbreviations:** AC, Approximation Coefficient; DC, Detail Coefficient; GRU, Gated Recurrent Unit; LSTM,Long Short-Term Memory; OGRU, Optimized Gated Recurrent Unit; ReLU, Rectified Linear Unit; RMSE, Root Mean Squared Error; SWT, Stationary Wavelet Transform

be the fair one, making it unpredictable, and any kind of forecasting or prediction will perform no better than random guessing, according to the Random Walk Theory (Ding et al., 2014)[1]. Financial time series are becoming even more non-stationary nowadays, in part due to the great pace at which big data is being produced, often even faster than real time. Because of large anomalies, traditional statistical methods of forecasting and prediction, such as filter and autoregressive models, are becoming less effective at predicting financial sequences (Benrhmach et al., 2020)[2].But with the development of artificial intelligence (AI), actual evidence has demonstrated that stock price movement is predictable (Jegadeesh and Titman, 1993)[3]. The vanishing gradient problem, which makes convergence difficult, raises prediction errors, and lengthens model computation time, has been a challenge for deep learning models over time. As a sequence gets longer, the gradient tends to get smaller and disappear, which is known as the "vanishing gradient problem." Finding a lagging mechanism for deep learning models is extremely desirable because it inhibits the model from taking into account all previous data, which lowers the likelihood that it would encounter the vanishing gradient problem (Nguyen et al., 2021)[4]. Additionally, studies have revealed that several academics have combined the wavelet transform with pure and hybrid deep learning models to predict stock prices. In order to predict stock price, this research took into account a model that combines the Wavelet Transform with an Optimized Gated Recurrent Unit (OGRU) neural network that is immune to vanishing gradients. Myriad stock prediction websites still have significant prediction errors, and the bigger they are, the more expensive they are for investors. In order to decompose the time series data and lessen its inherent noise, the study suggested using the Wavelet Transform. To establish a temporal lag and lessen long-term reliance, the wavelet transform's scaled mean of the approximation coefficient (AC) was applied. A vanishing gradient resilient OGRU model was trained using the deconstructed data in addition to a robust prediction model. Together, the time lag, wavelet-decomposed data, and OGRU model will solve the vanishing gradient problem and enable quicker and more precise predictions of the daily closing prices of Apple Inc. and Netflix.

2 | LITERATURE REVIEW

This section reviews relevant literature on methods for calculating the temporal lag. Additionally, a summary of pertinent research on forecasting models is provided, including everything from neural networks to conventional statistical techniques.

2.1 | Related Work on Time Lags

Numerous methods have been used to determine the time lag to be used in statistical forecasting models as well as neural networks. Xiao et al. (2017)[5] implemented an accurate multi-step-ahead time series forecasting using the Kalman Filtering Model (KFM) in conjunction with Echo Neural Networks (ESN), dubbed the E-KFM model using arbitrary lags of 1, 6, 12 and 18. The Recurrent Neural Network-based Granger Causality estimator (RNN-GC) model proposed by Wang et al. (2018)[6] was efficient in modelling directional linked analysis in multivariate series and it allowed varying-length time lags in the brain connectivity detection problem. Moreover, Petneházi (2019)[7] suggested using the Gated Recurrent Unit (GRU) and Long Short-Term Memory (LSTM) for forecasting hourly bike rentals. Two types of lags were used; recent values (1, 2 and 3 lags) and distant values (24, 48 and 168 lags). Also, Munkhdalai et al. (2020)[8] proposed a hybrid Vector Autoregressive and Gated Recurrent Unit (VAR-GRU) to establish the most important variables using Granger Causality and an appropriate lag length for multivariate stock-price prediction. The models used with the VAR proved to have the lowest error metrics in all experiments. Lastly, Surakhi et al. (2021)[9] implemented a comparative study of the autocorrelation function, an LSTM used with a Genetic Algorithm (GA) to enhance the choice of a time-lag value and another LSTM that chose the most accurate prediction given the optimal lag ranging between 24 and 168.

2.2 | Related Work on Forecasting Models

Al Wadi et al. (2010)[10] and Mousazadeh et al. (2015)[11] used Autoregressive Integrated Moving Average (ARIMA) and the Wavelet Transform in the prediction of stock prices. The wavelet transform left the financial data with no outlier, seasonal effects and non-constant mean and variance and improved accuracy of the ARIMA (Al Wadi et al., 2010)[10]. On the other hand, the ARIMA was compared to the LSTM in forecasting the stock price of four companies after the data was denoised using the wavelet transform (Skehin et al., 2018)[12]. The study used the pure ARIMA and LSTM models as well as the WAV-ARIMA and WAV-LSTM and these were compared using RMSE.

Some researchers used the wavelet transform in conjunction with Artificial Neural Networks (ANNs) for the prediction of stock prices. Lamhiri et al. (2014)[13] and Chandar et al. (2016)[14] used the Discrete Wavelet Transform (DWT) with a simple ANN for price index and stock price prediction. These models' performance in terms of error metrics as well as computational time was enhanced. Using the Haar wavelet transform with Multiple Time Windows for Apple Inc. stock price prediction also reduces the RMSE significantly (Kulaglic and Üstündağ, 2018)[15]. Meanwhile, Jarrah and Salim (2019)[16] predicted the Saudi stock price trends based on previous price history using the DWT and RNN using Back Propagation Through Time (BPTT). The method (DWT+RNN) predicted the period's price more accurately than the ARIMA model using MSE, RMSE and MAE criteria.

Some researchers have used deep neural networks to predict the financial variables. Štifanić et al. (2020)[17] proposed a model for forecasting stock and commodity prices by integrating a five-level Stationary Wavelet Transform (SWT) and the Bidirectional LSTM (BDLSTM) using a 128-day lookback period for the five-day West Texas Intermediate (WTI) crude oil forecast. Also, Qiu et al. (2018)[18] proposed a prediction model that used LSTM and an attention technique, in which the Wavelet Transform was used to denoise the long-term financial data as well as extract and train its features. Althelaya et al. (2021)[19] proposed multiresolution analysis and a stacked LSTM to predict financial time series with a comparison of multiresolution methods with SWT and the Empirical Wavelet Transform (EWT). Deep learning, multiresolution analysis and decomposition of data had impeccable effects on the performance of a model.

The wavelet transform was also used with the Gated Recurrent Unit (GRU) neural network. Biazon and Bianchi (2020)[20] developed the DWT Gated Recurrent Unit Network model (DWT-GRU) for stock exchange data. The DWT-GRU consisted of combining the DWT's denoising and decomposition capacity with pre-processed data to be trained by an RNN based primarily on the Gated Recurrent Unit Neural Network (GRUNN). The wavelet preprocessing significantly improved the results of both LSTM and GRU networks (Arévalo et al., 2018)[21]. Lastly, the Optimized Gated Recurrent Unit (OGRU) is the latest modification of the GRU was done by Wang et al. (2018)[22] to augment the learning and structure of the GRU and preventing present forgetting information hindering the update gate. The GRU significantly performed better that the GRU in both univariate and multivariate time series. In this study, the same model will be used in conjunction with the wavelet transform and a scaled mean AC time lag, while altering the activation functions.

3 | MATERIALS AND METHODS

In order to forecast time series data for Apple Inc. and Netflix Inc., this article suggests using the OGRU model with a time lag that is impervious to vanishing gradients. The direction the study took is outlined in the following figure:

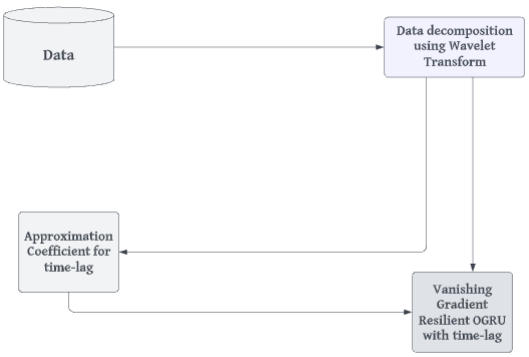

FIGURE 1 Work Flow

This section describes the vanishing gradient resilient OGRU model, the performance measures that were taken into consideration, the SWT that was used to decompose the data, and the determination of the time lag using the AC from the SWT.

3.1 | Stationary Wavelet Transform (SWT)

The Shifted Invariant (SI) wavelet transform or Translation Invariant (TI) wavelet transform is another name for the SWT. The SWT uses the identical formulas as the Discrete Wavelet Transform (DWT), with the exception that the signal is never sub-sampled. Instead, the signal is up-sampled with each level of decomposition by a factor of two, which makes the wavelet shift-invariant. The SWT is better for signal denoising since it is more redundant than DWT. The AC and DC in this case are the same length as the original signal at each level. The Daubenchies2 (db2) mother wavelet was employed in this study.

To determine the level to which we decompose the data, we use the rule:

$$j = log(n) \tag{1}$$

where n is the length of the series.

Thus, the stock price time series can be reconstructed by a series of projections on the mother and father wavelets with multilevel analysis indexed by $k \in 0, 1, 2, \ldots$ and by $j \in 0, 1, \ldots, J$, where J denotes the number of multi-resolution scales. The orthogonal wavelet series approximation to a signal $s(t)$ is formulated by:

$$s(t) = A_J(t) + D_J(t) + D_{J-1}(t) + \ldots + D_1(t) \tag{2}$$

where $A_J(t)$ is the coarsest approximation of the signal. The multi-resolution decomposition of $s(t)$ is the sequence of $\{A_J(t), D_J(t), D_{J-1}(t), \ldots, D_1(t)\}$

where,

$$A_J(t) = \sum_k a_{j,k}\varphi_{j,k}(t) \tag{3}$$

$$D_J(t) = \sum_k d_{j,k}\psi_{j,k}(t) \tag{4}$$

The expansion coefficients $a_{j,k}$ (known as the approximation coefficients) and $d_{j,k}$ (known as the detail coefficients).

3.2 | Determining the time lag

The lag for time series to be input into a neural network using the wavelet transform has never before been determined. This study used the AC of the Wavelet Transform, which is given by $A_J(t)$, which is defined in Equation (3), to compute the lag. The lag employed in the neural network was therefore the mean of the normalized AC increased by a factor of 100 because this portion of the Wavelet Transform indicates the trend element of the series or signal. Three procedures will be taken to determine the time lag: normalizing the AC, determining the average AC, and applying a factor of 100.

Normalisation of Approximation Coefficient

The AC will be a sequence of values that define the series' trend once the time series has been divided into AC and DC. The observation is more random and has a low correlation with the trend when the AC is less. Normalisation will take the form:

$$A'_J(t) = \frac{A - min(A)}{max(A) - min(A)} \tag{5}$$

where $A'_J(t)$ is the normalised value of the AC, the values are between the range of 0 and 1.

Average of the Approximation Coefficient

The normalised values of AC will then be averaged in order to find the average trend in the following manner:

$$\bar{A}'_J(t) = \frac{\sum_{i=1}^{n} A'_J(t)}{n} \tag{6}$$

where $\bar{A}'_J(t)$ is the average AC, $A'_J(t)$ is the normalised values of the ACs and n is the length of the series.

Applying a Multiplier

A mutiplier of 100 was used on the average lag, such that the lag will be given by:

$$MeanAC = 100 * \bar{A}'_J(t) \tag{7}$$

The $MeanAC$ will be used to determine h_{t-1} which is the hidden state for the previous time period. That is, how far back the model will look.

3.2.1 | Justifying the time lag

TABLE 1 Descriptive Statistics of the Approximation Coefficient

Statistic	Apple	Netflix
Minimum	0.95	824.37
Maximum	711.36	3918.08
Average	115.04	2125.75

- **Normalisation**: The range in AC is too large and in order to maintain the relationship between among the original data values.
- **Average of AC**: To find the average trend to determine the lag of the model.
- **Multiplier**: The lag must be in a scale that is relevant to the data.

This study adopted an approach that is similar to Surakhi et al. (2021)[9] where the autocorrelation coefficient was used to determine a lag for hourly data.

3.3 | Default lags

The study sought to compare the proposed Mean AC lag to other lags for both datasets, while the Default_1, the Default_21 were used for the Apple dataset. The Default_1 and the Default_24 were used for the Netflix dataset. The Default_21 time lag (for the Apple dataset) is the equivalent of a one month lookback period, taking into account that the stock market does not operate on weekends. Alternatively, the Default_24 (for the Netflix dataset) is the equivalent of a 2-hour lookback period.

TABLE 2 Justification of Default Lags

Dataset	Lag	Related Work	Citation	Application to study
Apple	Default_1	1, 6, 12 and 18 previous steps	Xiao et al. (2017)[5]	1 day to predict daily data
	Default_21	1, 2, 3, 24, 48 and 168 hours to predict daily data	Petneházi(2019)[7]	Extended 168 hours to 21 days (a month) to predict daily data
Netflix	Default_1	1 previous step to predict the next	Xiao et al. (2017)[5] and Petneházi(2019)[7]	1 previous period used to predict the next
	Default_24	24 hours to predict hourly data	Surakhi et al. (2021)[9]	24*5 = 120 minutes = 2 hours to predict 5-minute data

Table 2 shows a brief summary of the motivation behind the default lags whose performance was compared to the performance of the proposed Mean AC lag.

3.4 | Vanishing Gradient Resilient Optimized Gated Recurrent Unit with time lag

The decomposition coefficients that are resultant from the decomposition using the SWT were subjected to the Granger Causality test to determine if they each Granger-caused the closing stock price before being fed into the neural network. The proposed model differs slightly from the OGRU proposed by Wang et al. (2019)[22] in that it employs different activation functions for the OGRU layers. The paper used the Tanh, but this proposed methodology will use a ReLU, Sigmoid and the ELU. The ReLU is newer than all other activation functions, including the Sigmoid and Tanh. It is also very easy to use and effective at circumventing the limitations of other previously popular activation functions and it is not largely affected by the vanishing gradient problem. Likewise, the ELU activation function smooths slower than the ReLU and it produces negative inputs, thus making it a great substitute for the ReLU. This model was first fit on the training data and then tuning of parameters made use of the validation set. Lastly, the test set was used as the actual price, so that residuals are calculated using the predicted price.

3.4.1 | Model Assumptions

Before undertaking this study, the following assumptions were considered in determining the methodology:

1. Different activation functions perform differently for different models and datasets.
2. The OGRU model can be applied to both univariate and multivariate datasets.
3. The OGRU model works on time series data of different frequencies and sample sizes.

3.4.2 | Model Configuration

The model was configured using the following preprocessing, split, layers and trainable parameters.

Normalising Data

Before training the model, the data (decomposition coefficients used as regressors and the regressand) was first normalised using the MinMaxScaler as shown below:

$$x' = \frac{x_i - min(x_i)}{max(x_i) - min(x_i)} \tag{8}$$

Splitting Data

Thereafter, the training and test sets were determined. In this study, a ratio of 70:30 was used to determine the training:test split, and the test set was further split into the validation set. Of note here is that the splitting criteria was set "no shuffling" because this is time series data.

Dropout and Dense Layer

In order to avoid overfitting the model applied drop-outs of 0.1 or 0.2 (10 or 20 percent) and to make the model more robust, the model added a dense layer. This study also used 16, 32, 48 and 64 neurones depending on the most optimal. On the other hand, the study used 40 epochs for the hyperparameter tuning stage and 100 epochs to fit the optimal model.

Loss Function and Optimizer

The loss function that was used for the gradient descent stage of training the model was the Mean Square Error (MSE) and this study opted to use the 'ADAM' optimizer. This optimizer was chosen because it is the best compared to other optimizers in terms of computational time and limiting the parameters to be tuned.

Trainable Parameters

Trainable parameters are also known as weights of the neural network model. Too many weights in a model cause overfitting for simple task and small datasets. This was not an issue in this study because the data used was highly unstationary, which made more weights more suitable in the learning process. In this study, weight distribution relied on random numbers and these were not tuned. Parameters in a GRU and subsequently, an OGRU are calculated as:

$$p = 3(n^2 + nm + 2n) \tag{9}$$

where p is the number of parameters, m is the input dimension and n is the output dimension. The three comes from the 3 sets of operations requiring weight matrices; that is the new candidate vector, update gate and reset gate as specified below.

Early Stopping

In order to avoid overfitting and having unnecessarily long training periods, the study employed the *Early Stopping* callback. This function was specified as:

$$es = EarlyStopping(monitor =' root\_mean\_square\_error', mode =' min') \tag{10}$$

This callback functionality is employed at the training stage of the model, and it monitors the RMSE in this study which terminated the training process as soon as the RMSE stopped improving significantly. The callback also makes projections for the RMSE in future iteration and terminates the training process if the RMSE will not improve. In essence, Early Stopping terminates under the rule:

$$RMSE^* = min(RMSE_1, RMSE_2, \dots, RMSE_{n-1}, RMSE_n) \tag{11}$$

where $RMSE^*$ is the RMSE where the iterations terminate the training process, n is the total number of epochs or iterations. This study did not apply an improvement threshold because the training process could be easily tampered with, especially because the threshold could be any number.

3.4.3 | Model Architecture

Taking the input time series to be (x_1, x_2, …, x_t), the model architecture will take the form shown in Figure 2.

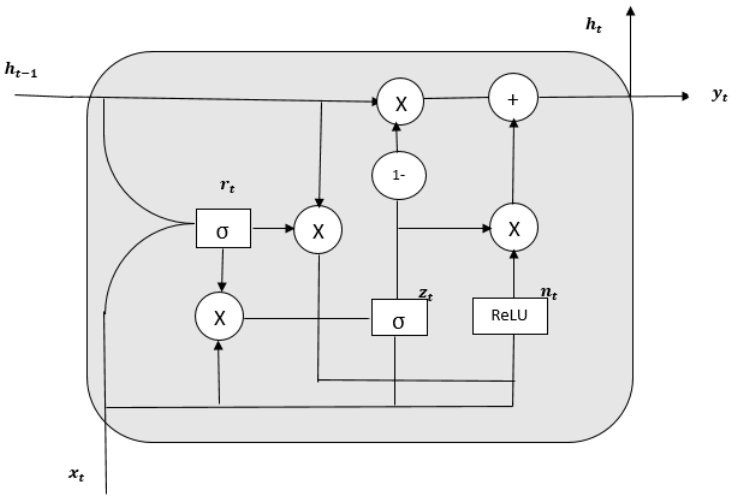

FIGURE 2 The Neural Structure of the vanishing gradient resilient OGRU

It is important to note that wherever there is h_{t-1}, $t-1$ signifies the time steps the model will look back at and this will be determined by the lag as shown in the previous subsection.

The reset gate will be given by:

$$r_t = \sigma(W_r[h_{t-1}, x_t]) \tag{12}$$

where σ is the sigmoid activation function, W_r is the weight between the input and h_{t-1} represents the standard GRU unit output at time $(t-1)$ and x_t represents the input at time t.

The update gate will be given by:

$$z_t = \sigma(W_z[h_{t-1}, x_t * r_t]) \tag{13}$$

where W_z represents the weight between the input and h_{t-1} in the update gate and r_t is the reset gate at time t.

The new candidate value vector created with the ReLU activation will be given by:

$$n_t = relu(W * [r_t * h_{t-1}, x_t]) \tag{14}$$

where W represents the update gate's output z_t and the weight between the inputs. The ReLU above is a placeholder that represents the other activation functions to be used. While the hidden layers will be given by:

$$h_t = (1 - z_t) * h_{t-1} + (z_t * n_t) \tag{15}$$

Finally, the output will be given by:

$$y_t = \sigma(W_o * h_t) \tag{16}$$

where W_o represents the weight of h_t.

The fine-tuned vanishing gradient resilient OGRU model which has been trained and validated will be used to make predictions and these will be compared to the actual test data. This will determine whether or not the model will be fit enough to be used by traders, portfolio managers and investors to hedge against risk and decision-making.

3.5 | Validating the model using Grid Search

Tuning for hyperparameters is very pivotal in deep learning because it massively improves the performance of models. Distinct combinations of the hyperparameters from Table 3 were assessed for the lowest error. Grid search performed loops of the different combinations and fit the model on the training data. The evaluation metrics for determining the best combination of hyperparameters was the RMSE.

TABLE 3 Dictionary of Hyperparameters

Hyperparameter	Vector
Neurones	[16, 32, 48, 64]
Batch Size	[8, 16, 32, 64]
Drop-out	[0.1, 0.2]

The hyperparameter table was formulated as an extension of work done by Miao (2019)[23], Adhinata and Rakhmadani (2021)[24] and Khalil et al. (2021)[25]. Advantages of grid search is that the search space is predetermined in the form of tuples, which makes it easy to control how long the process takes. When compared to manual search, it is computationally less intensive. Finally, grid search is advantageous because it allows the specification of the metric to be minimized or maximised, in this study, over and above the validation loss, RMSE was a chosen stopping metric. Even though it suffers from high dimensional spaces, it can easily to parallelized since the hyperparameter spaces are usually independent of each other.

3.6 | Evaluation Metrics

To determine the accuracy of the vanishing gradient resilient OGRU model and to be able to compare it to the OGRU model with Tanh activation function, the study will use the RMSE and MAE as stated above. These are the most commonly used measures of prediction accuracy according to literature. This is because both measures are easy to calculate and interprete, and they are scale-dependent. Using both measurements will be advantageous because models that minimise the MAE forecast the median, and those that minimise the RMSE forecast the mean. Specifically, these evaluation metrics will be calculated as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{17}$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right| \tag{18}$$

where n is the number of observations, y_i is the actual stock price and $\hat{y}_i$ is the estimated stock price. The study will also use computational time or runtime as another evaluation metric.

$$runtime = t_n - t_0 \tag{19}$$

where t_n is the time when the model converges and t_0 is the time when the model begin running.

3.7 | Scientific Contributions

In conclusion, this study has the following scientific contributions:

1. The OGRU model has never been stacked with drop-out layers in between and one dense layer.
2. The time lag for the OGRU neural network is determined using the scaled Mean AC. This method used has never been used to determine a lag before.
3. The OGRU model has not yet been used with the SWT.
4. The OGRU model has not been used with stock price datasets before.
5. The OGRU model has not been implemented with varying activation functions before.

3.8 | The Datasets

The study employed 5040 observations of the Apple Daily Closing Stock Price from April 1, 2002 to April 4, 2022. The study also employed 100,000 observations of the Netflix 5-Minute Closing Stock Price from May 22, 2017 at 12:15 PM through July 1, 2022 at 16:55 PM. While the Netflix data came from Forex Robot Factory, the Apple data came from Yahoo Finance. To more clearly show the structure and trend of the data, both time series were resampled using the weekly mean.

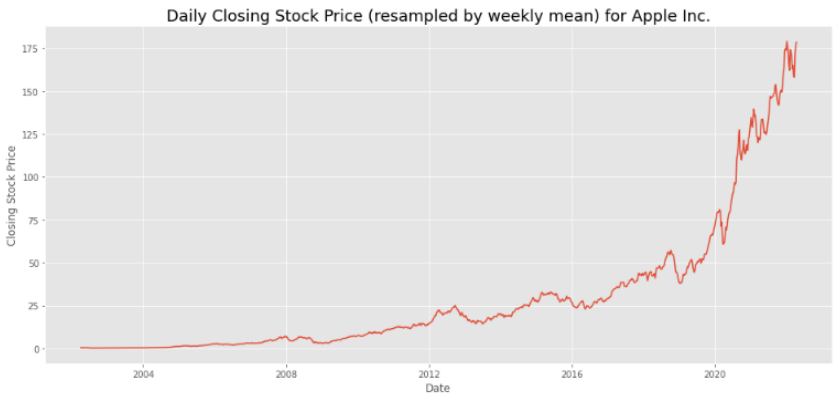

FIGURE 3 Daily (resampled by weekly mean) Closing Stock Price for Apple Inc. for the period between April 1, 2002 to April 4, 2022

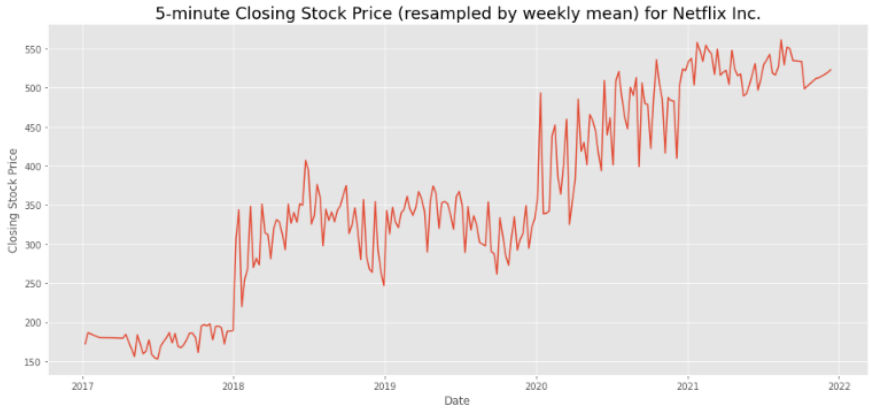

FIGURE 4 5-Minute (resampled by weekly mean) Closing Stock Price for Netflix Inc. for the period between May 22, 2017 at 12:15PM to July 1, 2022 at 16:55PM

4 | RESULTS AND DISCUSSION

All data-cleaning and preprocessing as well as experiments were performed in Python3 using a Tensorflow backend. The computer operating system used was the Windows 10, the basic configuration is: CPU is Intel Core i5 with 16GB RAM and 2.40GHz processing speed.

4.1 | Stationary Wavelet Transform (SWT) Decomposition

The study used the SWT to decompose the data into Approximation Coefficients (AC) and Detail Coeffients (DC). The Daubenchies2 (db2) mother wavelet was used for the decomposition on the Wavelet Toolbox.

TABLE 4 Results of SWT Decomposition

Dataset	Level of Decomposition	Approximation Coefficient	Detail Coefficients
Apple	$log(5040) = 3.7 \approx 4$	AC Level 4	DC Level 1, Level 2, Level 3 and Level 4
Netflix	$log(100000) = 5$	AC Level 5	DC Level 1, Level 2, Level 3, Level 4 and Level 5

The coefficients specified above were used as inputs for the next stage for the multivariate vanishing gradient-resilient OGRU model.

4.2 | Model Configuration

The Mean AC lags were established to be 16 and 42 for the Apple and Netflix datasets, respectively. The decomposition coefficients were used as inputs for the model. These inputs were normalised to improve training accuracy and reduce computational time. The data was split into training, validation and test sets and it was not shuffled because it is sequential data. The model was built using two OGRU layers with two drop-out layers after each OGRU and finally a dense layer at the end. The best model for the two datasets was searched out by using grid search subject to the hyperparameters specified in Table 3.

4.3 | Experiments and best model configuration

The study performed some graphical summaries for the errors according to the models, activation functions and lags for the different datasets in boxplots and scatter diagrams, before exploring the performance of the best models. A total of 48 models were tuned for hyperparameters for both datasets. The lag proposed in this study is depicted as MeanAC_16 (for Apple) and

MeanAC_42 (for Netflix), the Default_1 is used for both datasets and the Default_21 (for Apple) and the Default_24 (for Netflix). Figure 5 shows the summary of RMSE and Runtimes as depicted by the lag use and the activation function. On the left, it is noted that the Default_1 lag has very low runtimes and errors are evenly distributed.

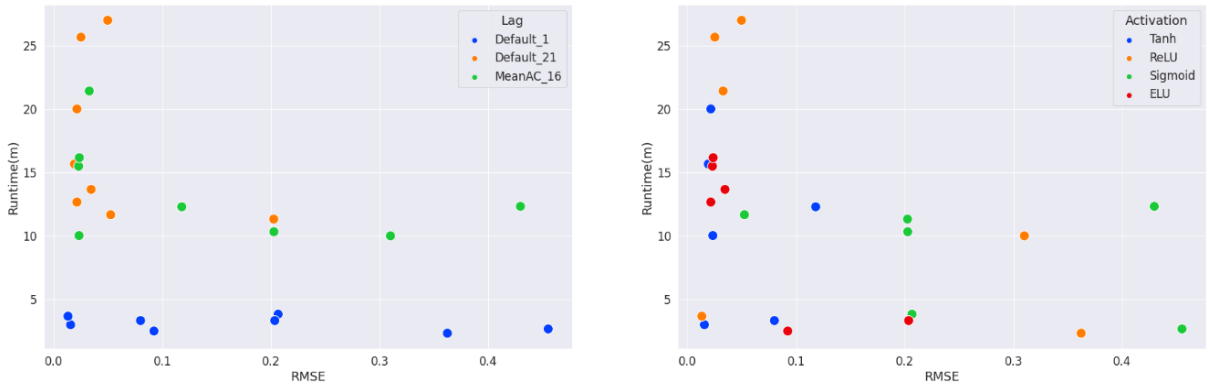

FIGURE 5 Graphical Summary of RMSE for the Apple Model Performance

The Default_21 lag has the lowest errors, but high runtimes while the MeanAC_16 lag has overall low errors and moderate runtimes. The proposed lag provides the perfect trade-off between errors and runtime. On the other hand, the ELU activation function is the best in terms of keeping RMSE below 0.2 and runtimes under 17 minutes. The Tanh activation function also performed well and the Sigmoid proved to be inefficient in terms of keeping errors low.

Moreover, Figure 6 shows that the model performance for Netflix is relatively better in terms of RMSE, but not runtime (which is expected since this dataset had 100000 observations). The MeanAC_42 lag contributed to high runtimes, especially because there was an outlier model with a runtime of 120 minutes.

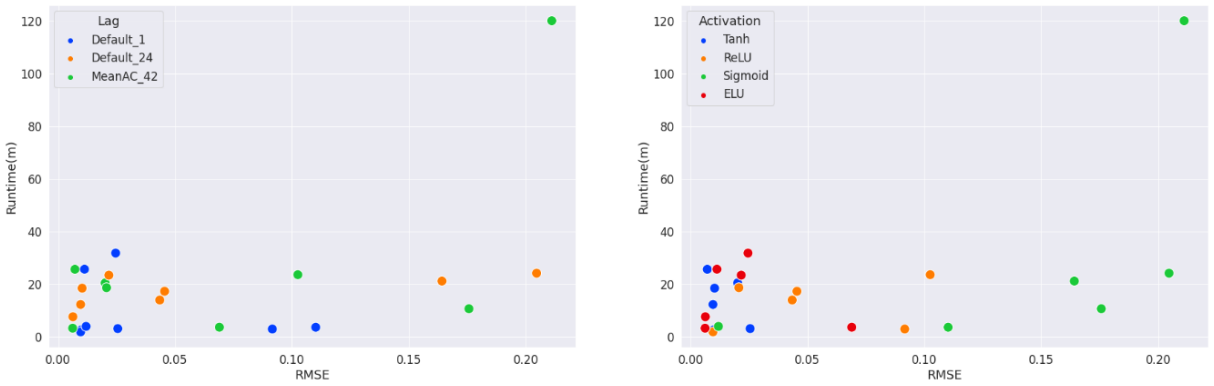

FIGURE 6 Graphical Summary of RMSE for the Netflix Model Performance

However, the proposed lag still managed to keep errors low and most runtimes were under 30 minutes. The Default_1 lag performed very well for this dataset. On the activation function part on the right, the ELU still performed best and kept errors under 0.1 and runtimes below 40 minutes. The ReLU also performed well, but the Sigmoid was the least accurate in terms of both errors and runtimes (just like in the Apple dataset).

Figure 7 shows summaries for both RMSE and MAE according to dataset and activation function. The Tanh contributed to very low error metrics, followed by the ELU. The highest contributor to large errors was the Sigmoid activation function and the ReLU was intermediate.

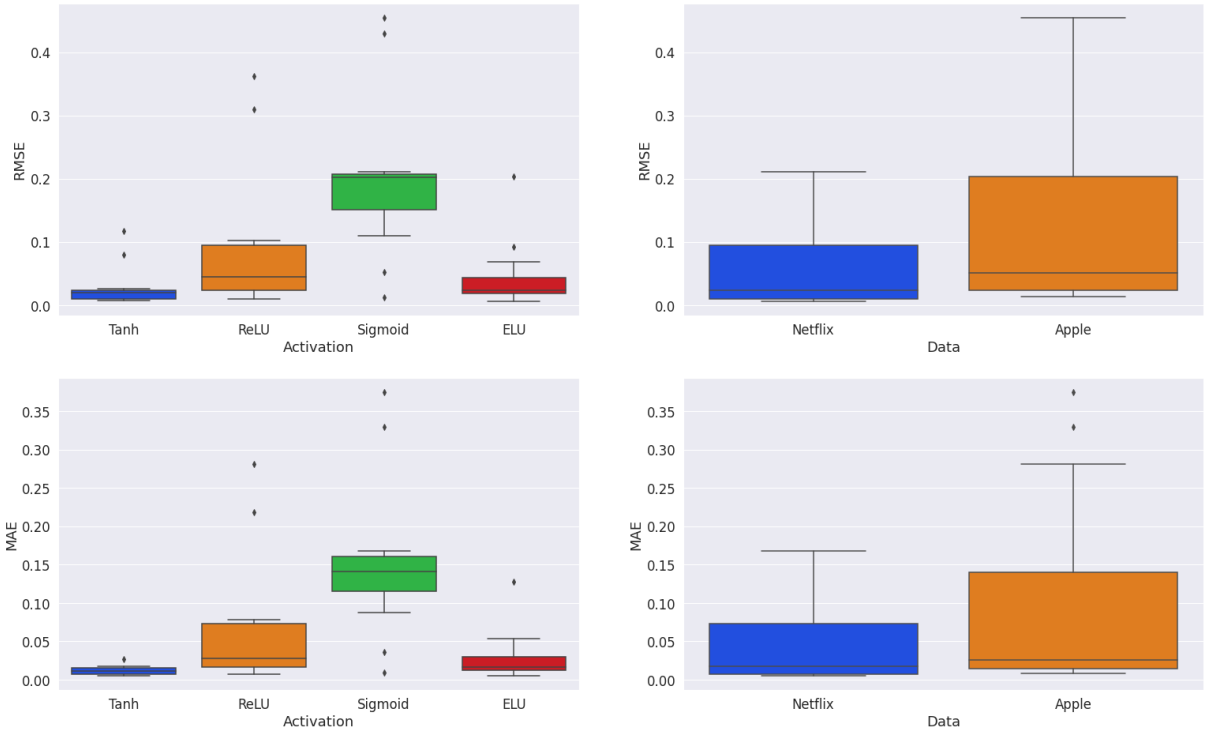

FIGURE 7 Overall Error Metrics by Activation Function and Dataset

The Netflix dataset has lower errors compared to the Apple dataset. The vanishing gradient resilient OGRU performs better for big data, which has a high frequency.

4.4 | Best Model Performance

The best four models were chosen from each activation function for each of the datasets.

4.4.1 | Apple Daily Closing Stock Price Model Performance

The Apple daily dataset performed best with the Default_1 and Default_21 lags as shown in Table 5. The indication here was that it was best to use the univariate model to predict the daily stock price. However, the second best model suggests otherwise because it showed that decomposition using the SWT keeps both errors and runtimes even lower than the best model.

TABLE 5 The best 4 models for the Apple daily dataset

S/N	Lag	Model	Activation	Neurons	Batch	Drop-out	RMSE	MAE	Runtime(m)
1.	Default_1	Multivariate	Tanh	64	64	0.2	0.01551	0.01040	3.00
2.	**Default_1**	**Univariate**	**ReLU**	**64**	**64**	**0.2**	**0.01312**	**0.00854**	**3.67**
3.	Default_21	Univariate	Sigmoid	48	16	0.2	0.05236	0.03567	11.67
4.	Default_21	Univariate	ELU	64	64	0.2	0.02136	0.01421	12.67

The best model had the highest hyperparameters with 64 neurons, batch size of 64 and the drop-out rate of 0.2. Since the multivariate model had the lowest runtime, it might be pivotal to decompose the series if time is of the essence.

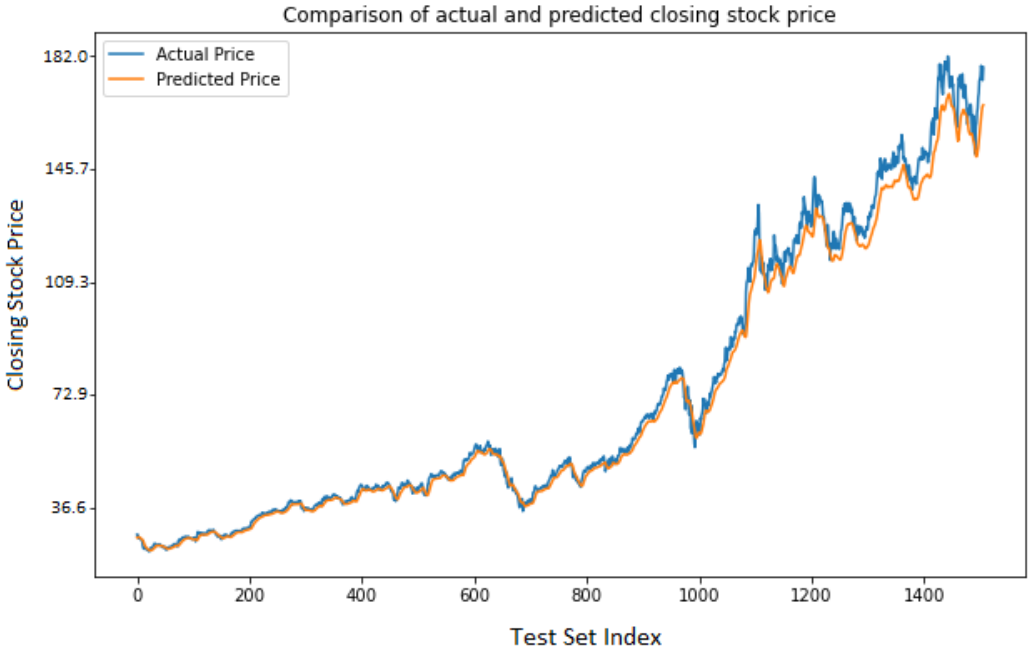

FIGURE 8 Out-of-sample predictions versus actual Apple daily data with ReLU activation and Default lag

The Default_1 lag is pivotal in accurately predicting the Apple daily stock price as shown in Figure 8. However, the performance of this model can still be improved especially considering the latter parts of the series where the data is highly volatile.

4.4.2 | Netflix 5-minute Closing Stock Price Model Performance

For the Netflix dataset however, the best performer was the proposed MeanAC_42 and the Default_1 as shown on Table 6. The best model had very low errors of 0.00620 and 0.00487, with the ELU activation function. This big dataset shows that the decomposition is necessary as 75 percent of the best models are multivariate. The best performing model had 32 neurons, a batch size of 64 and a 0.2 drop-out. The second best model was given by the ReLU activation function. This means the ELU, Tanh and the ReLU are recommended activation functions, but the Sigmoid is very costly in terms of errors.

TABLE 6 The best 4 models for the Netflix 5-minute dataset

S/N	Lag	Model	Activation	Neurons	Batch	Drop-out	RMSE	MAE	Runtime(m)
1.	MeanAC_42	Univariate	Tanh	64	64	0.2	0.00716	0.00533	25.67
2.	Default_1	Multivariate	ReLU	64	32	0.2	0.00960	0.00717	1.91
3.	Default_1	Mutlivariate	Sigmoid	64	32	0.2	0.01192	0.00927	4.00
4.	**MeanAC_42**	**Multivariate**	**ELU**	**32**	**64**	**0.2**	**0.00620**	**0.00487**	**3.01**

Figure 9 is the out-of-sample performance of the best Netflix model and it reflects the accuracy of the error metrics presented in Table 6. The Mean AC lag shows that it performed very well in making out-of-sample prediction for this dataset as there are very small deviations throughout the whole series.

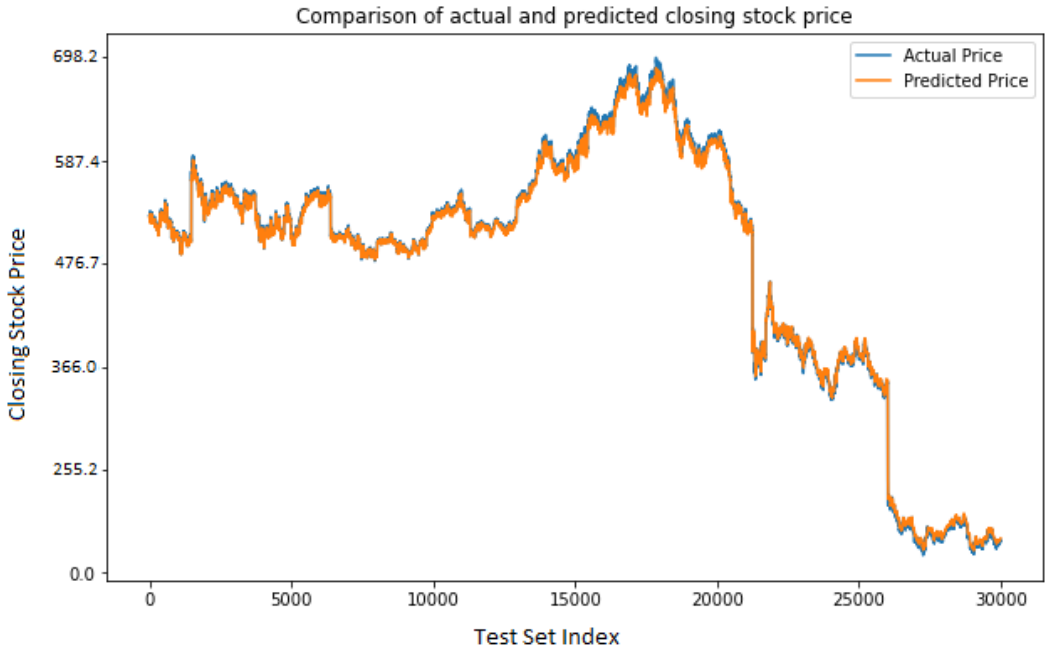

FIGURE 9 Out-of-sample predictions versus actual Netflix 5-minute data with ELU activation and Mean AC lag

5 | CONCLUSIONS

5.1 | Activation Function

After carefully studying the error patterns produced by the various activation functions, the model was observed to work best with the Exponential Linear Unit (ELU), Hyperbolic Tangent (Tanh) or the Rectified Linear Unit (ReLU) for the higher frequency data; the 5-minute Netflix data. The Sigmoid activation function must be avoided for the 5-minute data because it leads to explosive runtimes. On the other hand, the ReLU must be avoided for the daily dataset for the same reason. This study concludes that in order for the OGRU to be resilient to the vanishing gradient problem, it must be used with the Tanh or the ELU for the lower frequency data and strictly avoid the Sigmoid activation function for the higher frequency data.

5.2 | Decomposition

The best overall model performance came from the Netflix dataset as error metrics were kept very low, while they were on average larger for all other models. For the Apple daily dataset models performed better when using the univariate version instead of the multivariate with the Tanh activation function. For the Apple case, the only multivariate model that featured among the best performers kept errors extremely low and had the fastest runtime. Conversely, the 5-minute Netflix closing stock price performed best when the multivariate model was used. This was true for the case of the ELU and ReLU. This study thus concludes that the multivariate models are better applied for higher frequency data. Multivariate models also work well with the lower frequency data if there is a balance to be struck between runtime and errors.

5.3 | Lags

The proposed lagging mechanism has proven quite competitive for the higher frequency data as it performed better than all the other lags. However, it was not as efficient for the Apple daily dataset as the best models comprised of either the Default_1 or the Default_21. The best models for the Netflix dataset comprised of the Mean AC_42 and Default_1 lags. In terms of runtimes for the Apple dataset, the Default_21 were the worst while the Default_1 were the best. The Mean AC_16 lag had intermediate runtimes. In the Netflix case, the proposed lag performed very well by keeping runtimes low. However, when it was used with

the Sigmoid activation, it produced extremely high runtimes. This study thus concludes that the proposed lagging mechanism is recommended for higher frequency data as it wors well with the Tanh and ELU activations, as well as the multivariate and univariate models.

Author contributions

Conceptualization, L.S.M., L.N. and A.N.; methodology, L.S.M., L.N. and A.N.; software, L.S.M.; validation, L.S.M., L.N. and A.N.; formal analysis, L.S.M.; investigation, L.S.M.; resources, L.S.M.; data curation, L.S.M.; writing—original draft preparation, L.S.M.; writing—review and editing,L.S.M., L.N. and A.N.; visualization, L.S.M.; supervision, L.N. and A.N.; project administration, L.S.M., L.N. and A.N. All authors have read and agreed to the published version of the manuscript.

ACKNOWLEDGMENTS

This paper would not have been possible without the support of the Pan African University Institute of Basic Sciences, Technology and Innovation (PAUISTI) and the Jomo Kenyatta University of Agriculture and Technology (JKUAT).

Financial disclosure

None reported.

Conflict of interest

The authors declare no potential conflict of interests.

Data Availability

Data available in a publicly accessible repository that does not issue DOIs were analyzed in this study. The daily Apple stock price used in this study was downloaded from Yahoo Finance: [https://finance.yahoo.com/quote/AAPL/history?p=AAPL] and the 5-minute Netflix stock price data was downloaded from Forex Robot Factory: [https://www.forexrobotacademy.com/forex-historical-data].

ORCID

Luyandza Sindi Mamba: https://orcid.org/0000-0002-3126-9155.

References

1. Ding X, Zhang Y, Liu T, Duan J. Using structured events to predict stock price movement: An empirical investigation. In: ; 2014: 1415–1425.

2. Benrhmach G, Namir K, Namir A, Bouyaghroumni J. Nonlinear autoregressive neural network and extended Kalman filters for prediction of financial time series. *Journal of Applied Mathematics* 2020; 2020.

3. Jegadeesh N, Titman S. Returns to buying winners and selling losers: Implications for stock market efficiency. *The Journal of finance* 1993; 48(1): 65–91.

4. Nguyen HP, Baraldi P, Zio E. Ensemble empirical mode decomposition and long short-term memory neural network for multi-step predictions of time series signals in nuclear power plants. *Applied Energy* 2021; 283: 116346.

5. Xiao Q, Chaoqin C, Li Z. Time series prediction using dynamic Bayesian network. *Optik* 2017; 135: 98–103.

6. Wang Y, Lin K, Qi Y, et al. Estimating brain connectivity with varying-length time lags using a recurrent neural network. *IEEE Transactions on Biomedical Engineering* 2018; 65(9): 1953–1963.

7. Petneházi G. Recurrent neural networks for time series forecasting. *arXiv preprint arXiv:1901.00069* 2019.

8. Munkhdalai L, Li M, Theera-Umpon N, Auephanwiriyakul S, Ryu KH. VAR-GRU: A hybrid model for multivariate financial time series prediction. In: Springer. ; 2020: 322–332.

9. Surakhi O, Zaidan MA, Fung PL, et al. Time-Lag Selection for Time-Series Forecasting Using Neural Network and Heuristic Algorithm. *Electronics* 2021; 10(20): 2518.

10. Al Wadi S, Ismail MT, Altaher AM, Karim SAA. Forecasting volatility data based on Wavelet transforms and ARIMA model. In: IEEE. ; 2010: 86–90.

11. Mousazadeh A, Aghaei M, Moradzadeh F. Forecasting stock market using wavelet transforms and neural networks and arima (case study of price index of tehran stock exchange). *International Journal of Applied Operational Research* 2015; 5(3): 31–40.

12. Skehin T, Crane M, Bezbradica M. Day ahead forecasting of FAANG stocks using ARIMA, LSTM networks and wavelets. In: ; 2018: 334–340.

13. Lahmiri S. Wavelet low-and high-frequency components as features for predicting stock prices with backpropagation neural networks. *Journal of King Saud University-Computer and Information Sciences* 2014; 26(2): 218–227.

14. Chandar SK, Sumathi M, Sivanandam S. Prediction of stock market price using hybrid of wavelet transform and artificial neural network. *Indian journal of Science and Technology* 2016; 9(8): 1–5.

15. Kulaglic A, Üstündağ BB. Stock price forecast using wavelet transformations in multiple time windows and neural networks. In: IEEE. ; 2018: 518–521.

16. Jarrah M, Salim N. A recurrent neural network and a discrete wavelet transform to predict the Saudi stock price trends. *International Journal of Advanced Computer Science and Applications* 2019; 10(4): 155–162.

17. Štifanić D, Musulin J, Miočević A, Baressi Šegota S, Šubić R, Car Z. Impact of COVID-19 on forecasting stock prices: an integration of stationary wavelet transform and bidirectional long short-term memory. *Complexity* 2020; 2020.

18. Qiu J, Wang B, Zhou C. Forecasting stock prices with long-short term memory neural network based on attention mechanism. *PloS one* 2020; 15(1): e0227222.

19. Althelaya KA, Mohammed SA, El-Alfy ESM. Combining deep learning and multiresolution analysis for stock market forecasting. *IEEE Access* 2021; 9: 13099–13111.

20. Biazon V, Bianchi R. Gated Recurrent Unit Networks and Discrete Wavelet Transforms Applied to Forecasting and Trading in the Stock Market. In: SBC. ; 2020: 650–661.

21. Arévalo A, Nino J, León D, Hernandez G, Sandoval J. Deep learning and wavelets for high-frequency price forecasting. In: Springer. ; 2018: 385–399.

22. Wang X, Xu J, Shi W, Liu J. OGRU: An optimized gated recurrent unit neural network. In: . 1325. IOP Publishing. ; 2019: 012089.

23. Miao Y. A Deep Learning Approach for Stock Market Prediction. *Computer Science Department Stanford University* 2019.

24. Adhinata FD, Rakhmadani DP. Prediction of Covid-19 Daily Case in Indonesia Using Long Short Term Memory Method. *Teknika* 2021; 10(1): 62–67.

25. Khalil EAH, El Houby EM, Mohamed HK. Deep learning for emotion analysis in Arabic tweets. *Journal of Big Data* 2021; 8(1): 1–15.