Journal of Data Analysis and Information Processing, 2023, 11, 49-68
https://www.scirp.org/journal/jdaip
ISSN Online: 2327-7203
ISSN Print: 2327-7211

Predicting Wavelet-Transformed Stock Prices Using a Vanishing Gradient Resilient Optimized Gated Recurrent Unit with a Time Lag

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

[1]Department of Mathematics, Institute for Basic Sciences, Technology and Innovation, The Pan African University, Nairobi, Kenya
[2]Department of Statistics and Actuarial Sciences, Dedan Kimathi University of Technology, Nyeri, Kenya
[3]Department of Computing, School of Computing and Information Technology, Jomo Kenyatta University of Agriculture and Technology, Nairobi, Kenya
Email: sindi.luyandza@students.jkuat.ac.ke

How to cite this paper: Mamba, L.S., Ngunyi, A. and Nderu, L. (2023) Predicting Wavelet-Transformed Stock Prices Using a Vanishing Gradient Resilient Optimized Gated Recurrent Unit with a Time Lag. *Journal of Data Analysis and Information Processing*, **11**, 49-68.
https://doi.org/10.4236/jdaip.2023.111004

Received: December 8, 2022
Accepted: February 5, 2023
Published: February 8, 2023

Abstract

The development of accurate prediction models continues to be highly beneficial in myriad disciplines. Deep learning models have performed well in stock price prediction and give high accuracy. However, these models are largely affected by the vanishing gradient problem escalated by some activation functions. This study proposes the use of the Vanishing Gradient Resilient Optimized Gated Recurrent Unit (OGRU) model with a scaled mean Approximation Coefficient (AC) time lag which should counter slow convergence, vanishing gradient and large error metrics. This study employed the Rectified Linear Unit (ReLU), Hyperbolic Tangent (Tanh), Sigmoid and Exponential Linear Unit (ELU) activation functions. Real-life datasets including the daily Apple and 5-minute Netflix closing stock prices were used, and they were decomposed using the Stationary Wavelet Transform (SWT). The decomposed series formed a decomposed data model which was compared to an undecomposed data model with similar hyperparameters and different default lags. The Apple daily dataset performed well with a Default_1 lag, using an undecomposed data 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 decomposed data model and the ELU achieving 0.00620, 0.00487 and 3.01 minutes for the same metrics.

Keywords

Optimized Gated Recurrent Unit, Approximation Coefficient, Stationary Wavelet Transform, Activation Function, Time Lag

1. Introduction

According to [1], the Efficient Market Hypothesis (EMH) asserts that financial time series are almost always unpredictable because every piece of relevant information that can influence the price, including past values and volumes, is already taken into account. This means that the price reacts quickly to new information and is not tied to any trend or pattern. According to the Random Walk Theory, any type of prediction or forecasting will have no better performance than random guessing, and the stock price will always be the fair one, making it unpredictable [2]. Nowadays, financial time series are getting even more non-stationary especially because big data are generated at enormous speeds, sometimes as fast as real time. Classical statistical methods of prediction and forecasting such as filter and autoregressive models are getting less efficient in predicting financial sequences because of significant irregularities [3]. However, with advances in Artificial Intelligence (AI), it has been shown empirically that stock price movement is predictable [4].

Over the years, deep learning models have been faced with the vanishing gradient problem, which makes convergence impossible, increases prediction errors and model computational time. The vanishing gradient problem is when the gradient becomes particularly very small and disappears as the sequence becomes longer. The determination of a lagging mechanism for deep learning models is highly desirable because it prevents the model considering all past information, thus it reduces the chances of experiencing the vanishing gradient problem [5]. In addition, research has shown that numerous scholars have implemented pure and hybrid deep learning models for predicting stock prices in conjunction with the wavelet transform. Thus, this research considered a model consisting of a conjunction of the Stationary Wavelet Transform (SWT) and a vanishing gradient resilient Optimized Gated Recurrent Unit (OGRU) neural network for stock price prediction. The vanishing gradient problem will be addressed by using the lagging mechanism that will be informed by the approximation component of the wavelet transform and the use of a suitable activation function.

This problem is caused by long-term dependencies in neural networks as well as the activation functions that are used. The reality is that so many stock prediction sites still have large prediction errors, and the larger these are, the more financially costly they are. The study thus proposed to employ the SWT to decompose the time series data and reduce its inherent noisy property. The scaled mean of the Approximation Coefficient (AC) from the Wavelet Transform was used to determine a time lag to reduce long-term dependency. Furthermore, the decomposed data were used to train a robust prediction model in the form of a vanishing gradient resilient OGRU model. Collectively, the time lag, wavelet-decomposed data and the OGRU model will counter the vanishing gradient problem and allow for faster and more accurate predictions of the daily Apple Incorporated and 5-minute Netflix closing stock prices.

Predicting stock prices is of pivotal importance to investors, commercial banks, central banks, governments and stock forecasting sites. This research bridges the gap between models used in industry which still suffer large errors when used for High Frequency Data (HFD) and desirable fast and accurate models. Additionally, this research is significant in developing more accurate models for financial series such as exchange rates and interest rates and other instrumental variables. The proposed lag has the potential of leading time series modelling into a new and improve era of permanently reducing long-term dependence and using typical lags. From the innovation aspect, the research centers the limited exploration and manipulation of the OGRU model as it was developed about 3 years ago. Using it in a stacked fashion has never been implemented prior to this research. And most importantly, introducing a lagging mechanism that hinges on the trend-revealing Approximation Coefficient has never been done before. The scientific contributions of this paper are explained further in the methodology.

The organizational structure of this paper is as follows. Initially, we have a literature review of related work in Section 2. Furthermore, in Section 3, we have an explanation of the materials and methods employed to conduct this research. This section also contains the governing equations for the OGRU model, SWT and evaluation metrics. Moreover, the results of using the procedures mentioned in Section 3 are discussed and implemented in Section 4, and the study is concluded in Section 5.

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. [6] 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 [7] was efficient in modelling directional linked analysis in multivariate series and it allowed varying-length time lags in the brain connectivity detection problem.

Moreover, [8] 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, [9] proposed a hybrid Vector Autoregressive and Gated Recurrent Unit (VAR-GRU) to establish the most important variables using Granger Cau-

sality 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, [10] 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

On the other hand, [11] and [12] 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 [11]. 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 [13]. 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. [14] and [15] 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 [16]. Meanwhile, [17] 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. [18] 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, [19] 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. [20] 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. [21] 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 Net-

work (GRUNN). The wavelet preprocessing significantly improved the results of both LSTM and GRU networks [22]. Lastly, the Optimized Gated Recurrent Unit (OGRU) is the latest modification of the GRU was done by [23] to augment the learning and structure of the GRU and preventing present forgetting information hindering the update gate. The OGRU significantly performed better than 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 **Figure 1**.

This section describes the datasets used in this work, 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. 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 12:15PM through July 1, 2022 at 16:55PM. 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.

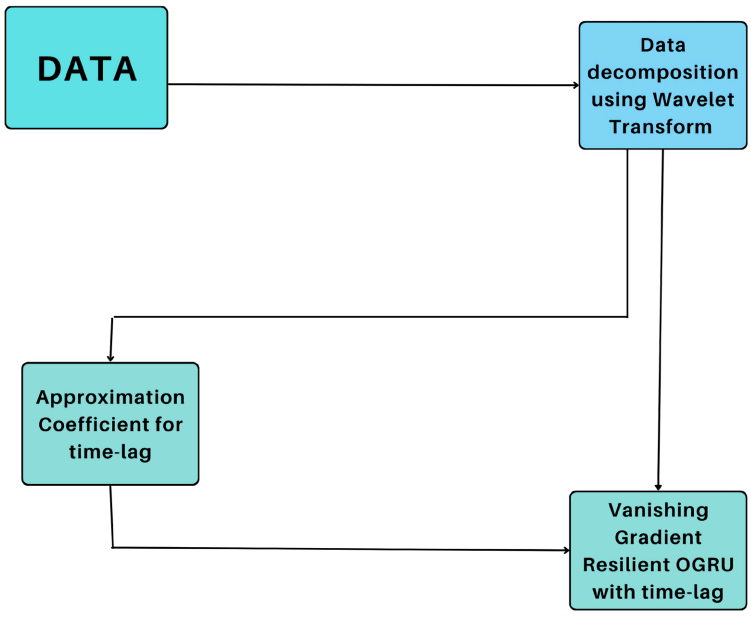

Figure 1. Work flow.

Figure 2 shows that the Apple dataset was not too volatile between the years 2002 and 2012. However, instability has been increasing each year since then. On the other hand, Figure 3 shows that the 5-minute Netflix data was more volatile compared to the daily Apple data because it has a higher frequency.

3.2. Stationary Wavelet Transform (SWT)

The Shift-Invariant (SI) 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.

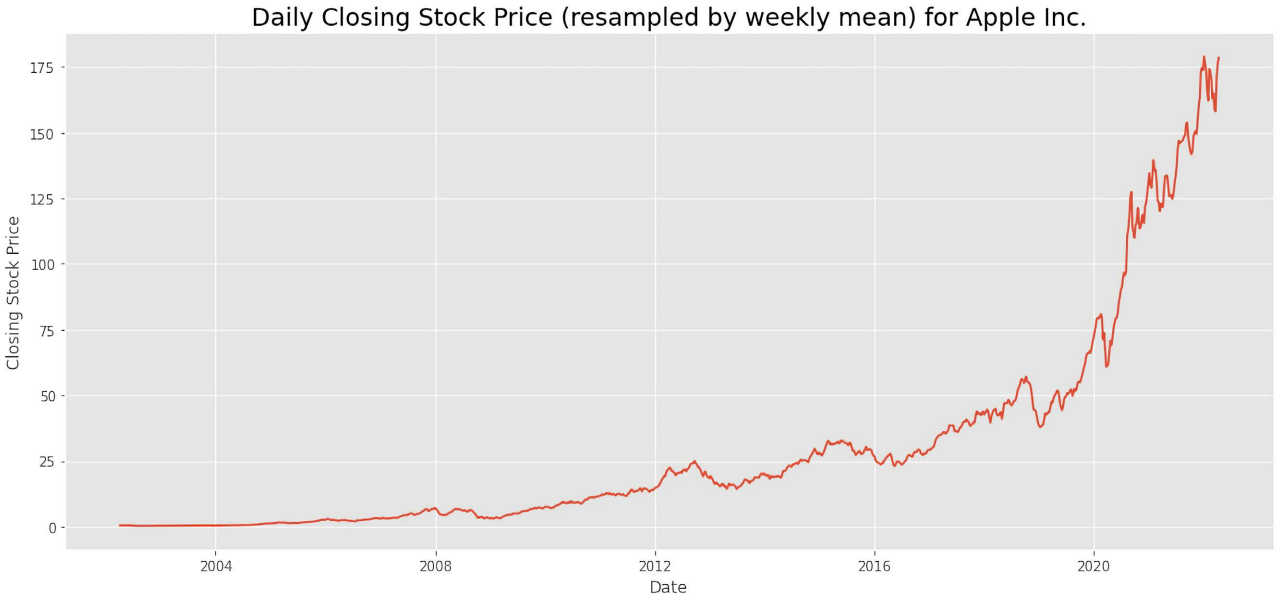

Figure 2. Daily closing stock price for Apple Inc.

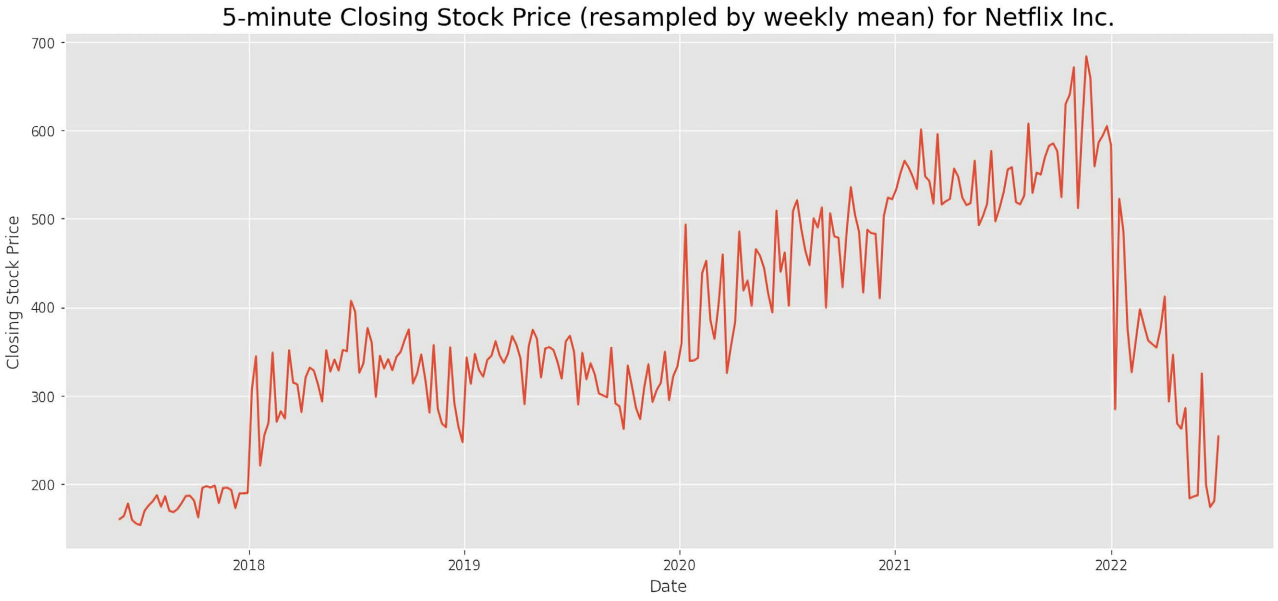

Figure 3. 5-minute closing stock price for Netflix Inc.

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. The Daubenchies wavelets are highly desirable because they are much smoother than the Haar wavelets and a very accurate generalization of the same. The Daubenchies are easily invertible, asymmetric, orthogonal and biorthogonal. Furthermore, the Daubenchies2 (db2) with two vanishing moments avoids the overly smoothing the signal, lengthening the wavelet feature and it has been proven efficient for financial series such as stock prices from the featured related work [14] and [24]. 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. The orthogonal wavelet series approximation to a signal $s(t)$ is formulated by:

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

the mother and father wavelets with multilevel analysis indexed by $k \in 0,1,2,\cdots$ and by $j \in 0,1,\cdots,J$, where J denotes the number of multi-resolution scales. $A_J(t)$ is the coarsest approximation of the signal. The multi-resolution decomposition of $s(t)$ is the sequence of $\{A_J(t), D_J(t), D_{J-1}(t), \cdots, D_1(t)\}$ 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). $\psi_{j,k}(t)$ is the mother wavelet and the $\varphi_{j,k}(t)$ is the father wavelet.

3.3. 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. As such, the proposed MeanAC lag is then given by:

$$\text{MeanAC} = 100 * \frac{\sum_{i=1}^{n}\left(\frac{A_i - \min(A_i)}{\max(A_i) - \min(A_i)}\right)}{n} \tag{5}$$

where 100 is the multiplier and A_i's are the Approximation coeffients.

3.3.1. Justifying the Time Lag

- **Normalisation**: The range in AC is too large and in order to maintain the relationship between among the original data values, especially because of the large ranges shown in Table 1.
- **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 [10] where the autocorrelation coefficient was used to determine a lag for hourly data.

3.3.2. 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 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

Table 1. Descriptive statistics of the AC.

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

Table 2. Justification of default lags.

Dataset	Lag	Related Work	Citation	Application to study
Apple	Default_1	1, 6, 12 and 18 previous steps	[6]	1 day to predict daily data
	Default_21	1, 2, 3, 24, 48 and 168 hours to predict daily data	[8]	Extended 168 hours to 21 days (a month) to predict daily data
Netflix	Default_1	1 previous step to predict the next	[6] and [8]	1 previous period used to predict the next
	Default_24	24 hours to predict hourly data	[10]	24 × 5 = 120 minutes = 2 hours to predict 5-minute data

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 [23] 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, loss functions, chosen optimizer and stopping criterion.

Normalising Data

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

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.

Early Stopping

In order to avoid overfitting and having unnecessarily long training periods, the study employed the *Early Stopping* callback. 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. 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 ($x_1, x_2, \cdots, x_t$), the model architecture will take the form shown in Figure 4.

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:

$$r_t = \sigma\left(W_r\left[h_{t-1}, x_t\right]\right) \tag{6}$$

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:

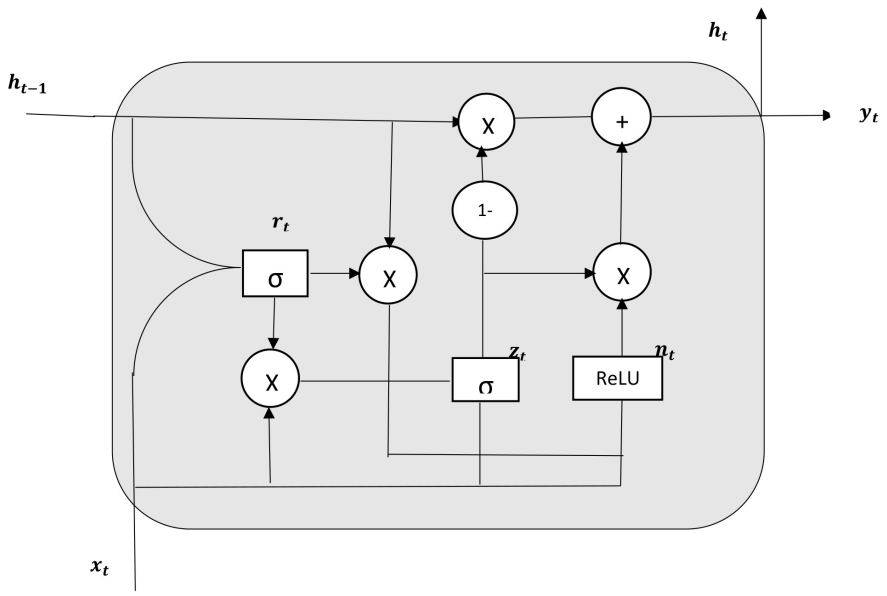

Figure 4. The neural structure of the model.

$$z_t = \sigma\left(W_z\left[h_{t-1}, x_t * r_t\right]\right) \tag{7}$$

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:

$$n_t = \text{relu}\left(W * \left[r_t * h_{t-1}, x_t\right]\right) \tag{8}$$

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:

$$h_t = \left(1 - z_t\right) * h_{t-1} + \left(z_t * n_t\right) \tag{9}$$

Finally, the output will be given by:

$$y_t = \sigma\left(W_o * h_t\right) \tag{10}$$

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.

3.4.4. Justification of Activation Functions

The selection of activation functions in this work starts from the OGRU which used the Hyperbolic Tangent (Tanh) activation function for the new candidate value vector [23]. Now, to reduce the possibility of the vanishing gradient problem affecting the models, this study varied the activation function by comparing the Tanh to the Rectified Linear Unit (ReLU), Sigmoid and an Exponential Linear Unit (ELU). The Sigmoid has the advantage of making prediction clear as the values are between 0 and 1, preventing the disappearance of the activation value. The ReLU is very easy to use and effective at circumventing the limitations of other previously popular activation functions. Likewise, the ELU activation function smooths slower than the ReLU and it produces negative inputs, thus making it a great substitute for the ReLU. In essence, all the activation functions where chosen on the basis that they had the potential to prevent the vanishing gradient problem.

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 [25] [26] and [27]. 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:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(y_i - \hat{y}_i\right)^2} \tag{11}$$

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}\left|y_i - \hat{y}_i\right| \tag{12}$$

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.

$$\text{runtime} = t_n - t_0 \tag{13}$$

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.

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 16 GB RAM and 2.40 GHz processing speed.

4.1. Stationary Wavelet Transform (SWT) Decomposition

The study used the SWT to decompose the data into Approximation Coefficients (AC) and Detail Coefficients (DC). The coefficients are specified in Table 4 and they were used as inputs for the decomposed data 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.

Table 4. Results of SWT decomposition.

Dataset	Level	AC	DCs
Apple	$\log(5040) = 3.7 \approx 4$	AC Level 4	DC Level 1, Level 2, Level 3, Level 4
Netflix	$\log(100000) = 5$	AC Level 5	DC Level 1, Level 2, Level 3, Level 4, Level 5

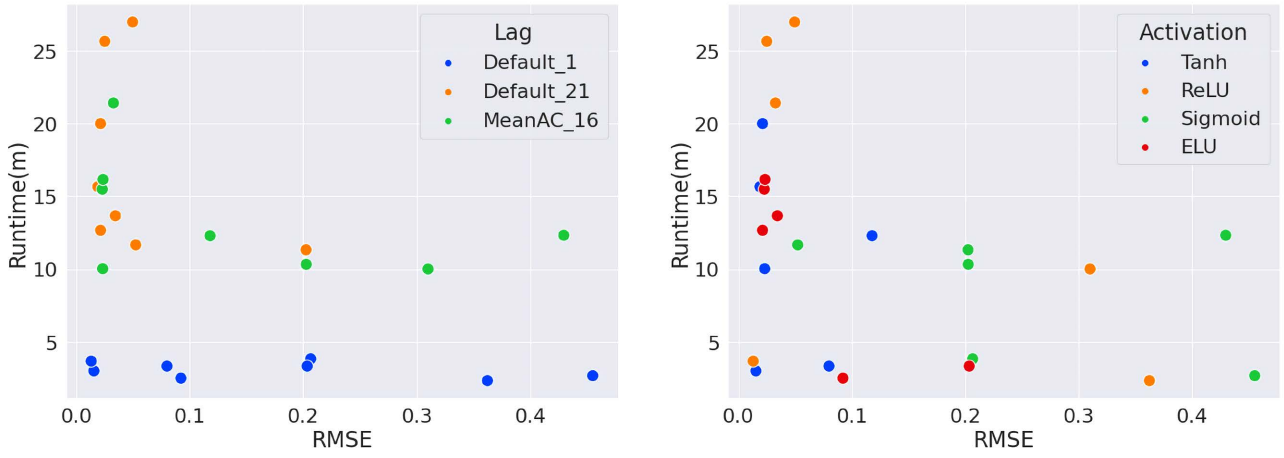

Figure 5. Graphical summary of RMSE for Apple.

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 100,000 observations). The MeanAC_42 lag contributed to high runtimes, especially because there was an outlier model with a runtime of 120 minutes.

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.

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

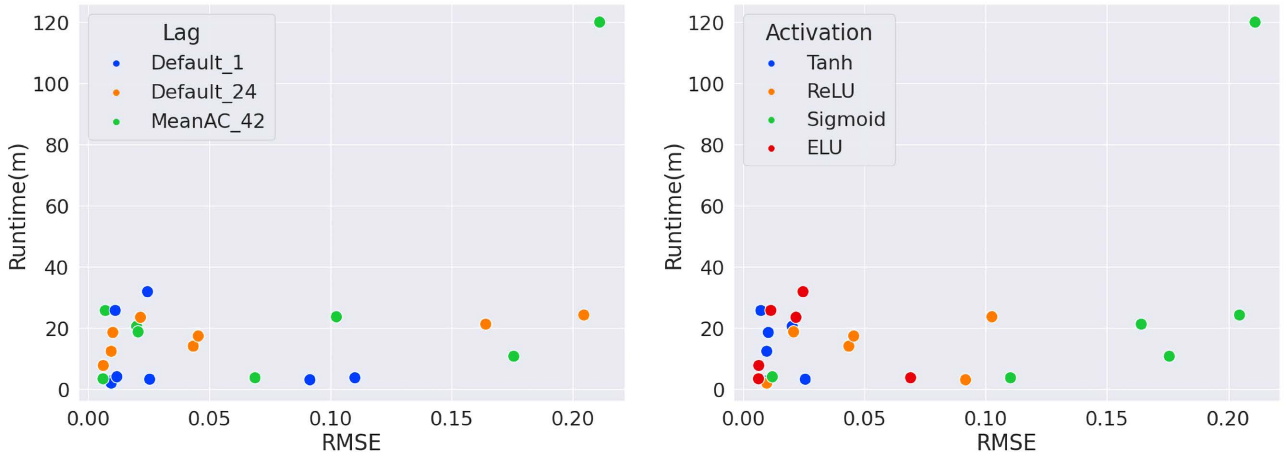

Figure 6. Graphical summary of RMSE for Netflix.

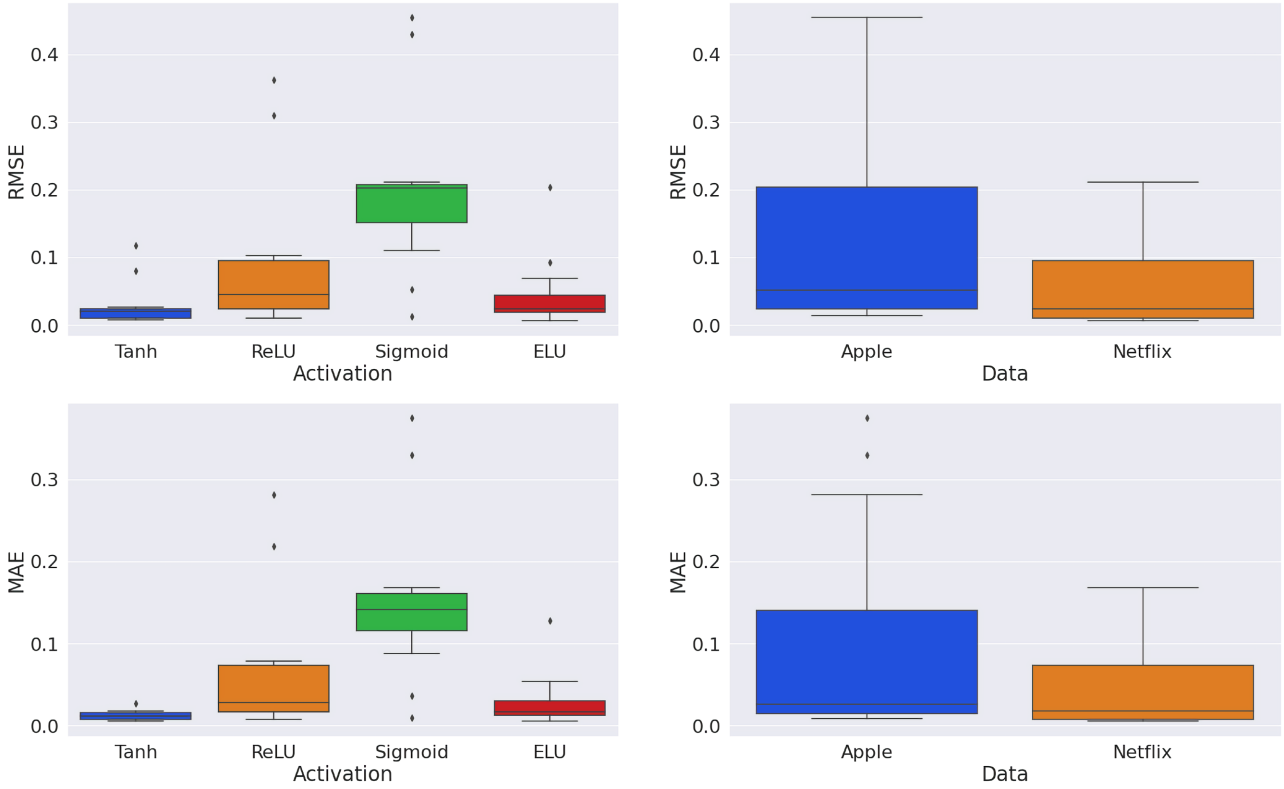

Figure 7. Overall error metrics by activation function and dataset.

Table 5. The best 4 models for the apple daily dataset.

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

undecomposed data 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.

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

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 decomposed data models. 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.

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.

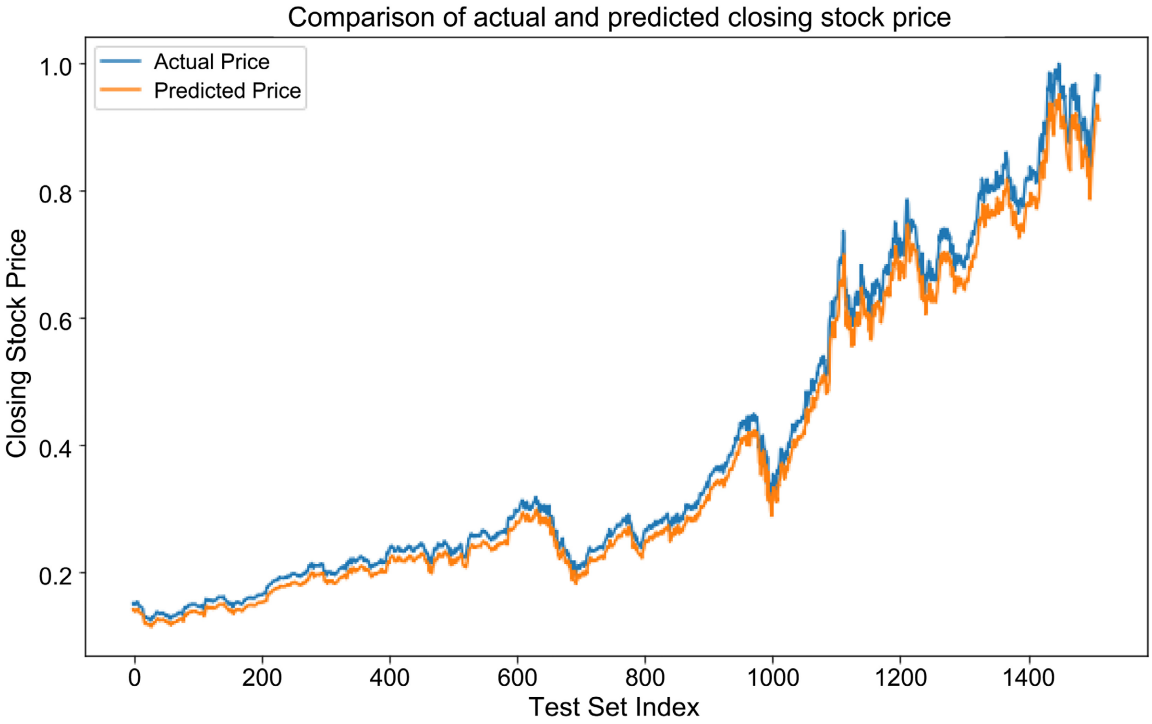

Figure 8. Best out-of-sample predictions for Apple.

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

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

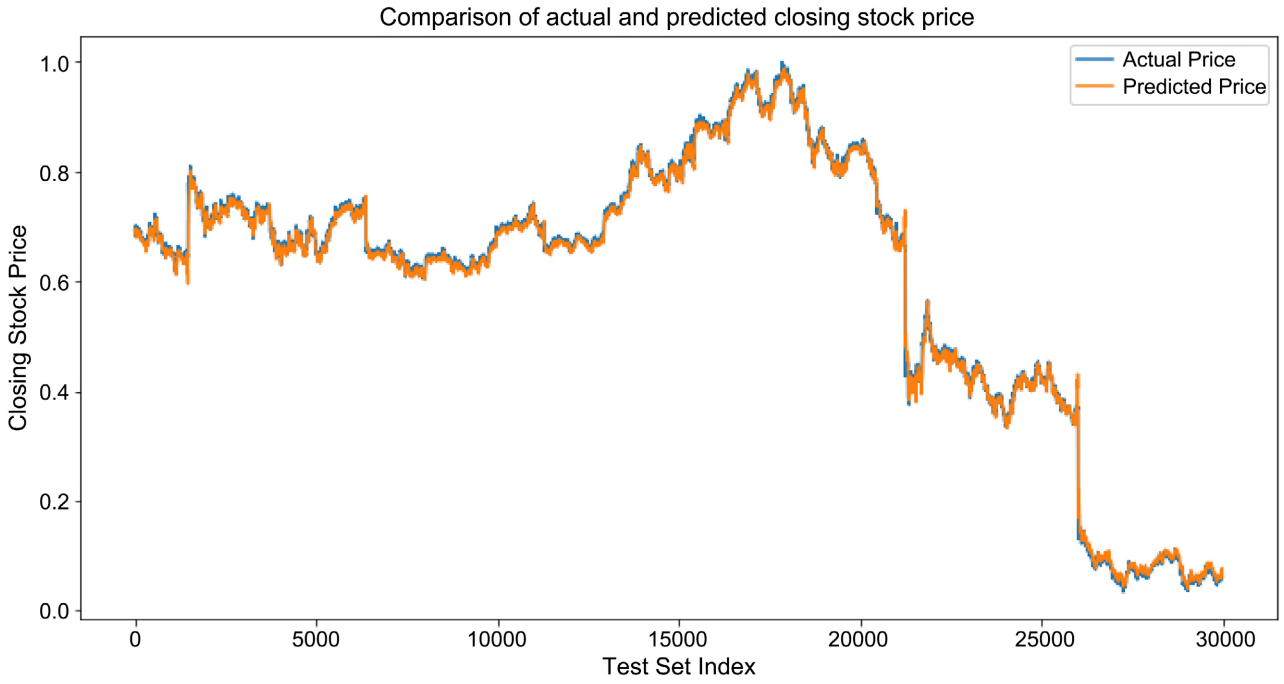

Figure 9. Best out-of-sample predictions for Netflix.

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.

The Apple daily dataset models performed better when using the undecomposed data version instead of the decomposed data model with the Tanh activation function. For the Apple case, the only decomposed data 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 decomposed data model was used. This was true for the case of the ELU and ReLU. This study thus concludes that the decomposed data models are better applied for higher frequency data. Decomposed data 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 works well with the Tanh and ELU activations, as well as the decomposed data and undecomposed data models.

Acknowledgements

I would like to thank God for this publication, the guidance of my supervisors, the support of my friends and the love of my family. I would love to dedicate this work to Pasha, Sabatini, Wilton and Sibongile.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Malkiel, B.G. (1989) Efficient Market Hypothesis. In: Eatwell, J., Milgate, M. and Newman, P., Eds., *Finance*, Springer, Berlin, 127-134. https://link.springer.com/chapter/10.1007/978-1-349-20213-3_13 https://doi.org/10.1007/978-1-349-20213-3_13

[2] Ding, X., Zhang, Y., Liu, T. and Duan, J. (2014) Using Structured Events to Predict Stock Price Movement: An Empirical Investigation. *Proceedings of the* 2014 *Conference on Empirical Methods in Natural Language Processing* (*EMNLP*), Doha, October 2014, 127-134. https://doi.org/10.3115/v1/D14-1148

[3] Benrhmach, G., Namir, K., Namir, A. and Bouyaghroumni, J. (2020) Nonlinear

Autoregressive Neural Network and Extended Kalman Filters for Prediction of Financial Time Series. *Journal of Applied Mathematics*, **2020**, Article ID: 5057801. https://doi.org/10.1155/2020/5057801

[4] Jegadeesh, N. and Titman, S. (1993) Returns to Buying Winners and Selling Losers: Implications for Stock Market Efficiency. *The Journal of Finance*, **48**, 65-91. https://doi.org/10.1111/j.1540-6261.1993.tb04702.x

[5] Nguyen, H., Baraldi, P. and Zio, E. (2021) Ensemble Empirical Mode Decomposition and Long Short-Term Memory Neural Network for Multi-Step Predictions of Time Series Signals in Nuclear Power Plants. *Electronics*, **283**, 116-346. https://doi.org/10.1016/j.apenergy.2020.116346

[6] Xiao, Q., Chaoqin, C. and Li, Z. (2017) Time Series Prediction Using Dynamic Bayesian Network. *Optik*, **135**, 98-103. https://doi.org/10.1016/j.ijleo.2017.01.073

[7] Wang, Y., Lin, K., Qi, Y., Lian, Q., Feng, S., Wu, Z. and Pan, G. (2018) Estimating Brain Connectivity with Varying-Length Time Lags Using a Recurrent Neural Network. *IEEE Transactions on Biomedical Engineering*, **65**, 1953-1963. https://doi.org/10.1109/TBME.2018.2842769

[8] Petneházi, G. (2019) Recurrent Neural Networks for Time Series Forecasting.

[9] Munkhdalai, L., Li, M., Theera-Umpon, N., Auephanwiriyakul, S. and Ryu, K.H. (2020) VAR-GRU: A Hybrid Model for Multivariate Financial Time Series Prediction. *Asian Conference on Intelligent Information and Database Systems*, Phuket, 23-26 March 2020, 322-332. https://doi.org/10.1007/978-3-030-42058-1_27

[10] Surakhi, O., Zaidan, M.A., Fung, P.L., Hossein, M.N., Serhan, S., AlKhanafseh, M., Ghoniem, R.M. and Hussein, T. (2021) Time-Lag Selection for Time-Series Forecasting Using Neural Network and Heuristic Algorithm. *Electronics*, **10**, 18-25. https://doi.org/10.3390/electronics10202518

[11] Al Wadi, S., Ismail, M.T., Altaher, A.M. and Karim, S.A.A. (2010) Forecasting Volatility Data Based on Wavelet Transforms and ARIMA Model. 2010 *International Conference on Science and Social Research*, Kuala Lumpur, 5-7 December 2010, 86-90. https://doi.org/10.1109/CSSR.2010.5773909

[12] Abbasi, N.M., Aghaei, M. and Moradzadeh, F. (2015) 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*, **5**, 31-40.

[13] Skehin, T., Crane, M. and Bezbradica, M. (2018) Day ahead Forecasting of FAANG Stocks Using ARIMA, LSTM Networks and Wavelets. *CEUR Workshop Proceedings*: *Day Ahead Forecasting of FAANG Stocks Using ARIMA*, *LSTM Networks and Wavelets*, Dublin, 6-7 December 2018, 334-340.

[14] Lahmiri, S. (2014) 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*, **26**, 218-227. https://doi.org/10.1016/j.jksuci.2013.12.001

[15] Chandar, S.K., Sumathi, M. and Sivanandam, S.N. (2016) Prediction of Stock Market Price Using Hybrid of Wavelet Transform and Artificial Neural Network. *Indian Journal of Science and Technology*, **9**, 1-5. https://doi.org/10.17485/ijst/2016/v9i8/87905

[16] Kulaglic, A. and Üstündağ, B.B. (2018) Stock Price Forecast Using Wavelet Transformations in Multiple Time Windows and Neural Networks. 2018 3*rd International Conference on Computer Science and Engineering* (*UBMK*), Sarajevo, 20-23 September 2018, 518-521. https://doi.org/10.1109/UBMK.2018.8566614

[17] Jarrah, M. and Salim, N. (2019) A Recurrent Neural Network and a Discrete Wavelet Transform to Predict the Saudi Stock Price Trends. *International Journal of Advanced Computer Science and Applications*, **10**, 155-162. https://doi.org/10.14569/IJACSA.2019.0100418

[18] Štifanić, D., Musulin, J., Miočević, A., Baressi Šegota, S., Šubić, R. and Car, Z. (2020) Impact of COVID-19 on Forecasting Stock Prices: An Integration of Stationary Wavelet Transform and Bidirectional Long Short-Term Memory. *Complexity*, **2020**, Article ID: 1846926. https://doi.org/10.1155/2020/1846926

[19] Qiu, J., Wang, B. and Zhou, C. (2020) Forecasting Stock Prices with Long-Short Term Memory Neural Network Based on Attention Mechanism. *PLOS ONE*, **15**, 222-227. https://doi.org/10.1371/journal.pone.0227222

[20] Althelaya, K.A., Mohammed, S.A. and El-Alfy, E.M. (2021) Combining Deep Learning and Multiresolution Analysis for Stock Market Forecasting. *IEEE Access*, **9**, 13099-13111. https://doi.org/10.1109/ACCESS.2021.3051872

[21] Biazon, V. and Bianchi, R. (2020) Gated Recurrent Unit Networks and Discrete Wavelet Transforms Applied to Forecasting and Trading in the Stock Market. *Anais do XVII Encontro Nacional de Inteligência Artificial e Computacional*, **68**, 650-661. https://doi.org/10.5753/eniac.2020.12167

[22] Arévalo, A., Nino, J., León, D., Hernandez, G. and Sandoval, J. (2018) Deep Learning and Wavelets for High-Frequency Price Forecasting. *International Conference on Computational Science*, Wuxi, 11-13 June 2018, 385-399. https://doi.org/10.1007/978-3-319-93701-4_29

[23] Wang, X., Xu, J., Shi, W. and Liu, J. (2019) OGRU: An Optimized Gated Recurrent Unit Neural Network. *Journal of Physics*: *Conference Series*, **1325**, 12-89. https://iopscience.iop.org/article/10.1088/1742-6596/1325/1/012089 https://doi.org/10.1088/1742-6596/1325/1/012089

[24] Ardila, D. and Sornette, D. (2016) Dating the Financial Cycle with Uncertainty Estimates: A Wavelet Proposition. *Finance Research Letters*, **19**, 298-304. https://doi.org/10.1016/j.frl.2016.09.004

[25] Miao, Y. (2019) A Deep Learning Approach for Stock Market Prediction. Computer Science Department, Stanford University, Stanford. http://cs230.stanford.edu/projects_fall_2020/reports/55614857.pdf

[26] Adhinata, F.D. and Rakhmadani, D.P. (2021) Prediction of Covid-19 Daily Case in Indonesia Using Long Short Term Memory Method. *Teknika*, **10**, 62-67. https://doi.org/10.34148/teknika.v10i1.328

[27] Khalil, E.A.H., El Houby, E.M.F. and Mohamed, H.K. (2021) Deep Learning for Emotion Analysis in Arabic Tweets. *Journal of Big Data*, **8**, 1-15. https://doi.org/10.1186/s40537-021-00523-w