Trading Saudi Stock Market Shares using Multivariate Recurrent Neural Network with a Long Short-term Memory Layer

Fahd A. Alturki[1], Abdullah M. Aldughaiyem[2]
Electrical Engineering Department
King Saud University, Riyadh
Saudi Arabia

***Abstract*—This study tests the Saudi stock market weak form using the weak form of an efficient market hypothesis and proposes a recurrent neural network (RNN) to produce a trading signal. To predict the next-day trading signal of several shares in the Saudi stock market, we designed the RNN with a long short-term memory architecture. The network input comprises several time series features that contribute to the classification process. The proposed RNN output is fed to a trading agent that buys or sells shares based on the share current value, current available balance, and the current number of shares owned. To evaluate the proposed neural network, we used the historical oil price data of Brent crude oil in combination with other stock features (e.g., previous day) opening and closing price of the evaluated share). The results indicate that oil price variations affect the Saudi stock market. Furthermore, with 55% accuracy, the proposed RNN model produces the next-day trading signal. For the same period, the proposed RNN trading method achieves an investment gain of 23%, whereas the buy-and-hold method obtained 1.2%.**

Keywords*—Time series; neural network; long short-term memory; stock price; Tadawul

I. Introduction

Of all the presented works for forecasting stock markets, only very few have targeted the Saudi stock market. In this study, we presented a recurrent neural network (RNN) that utilizes the long short-term memory (LTSM) architecture for a multivariate time series prediction to generate a trading signal (buy, sell, or do nothing) for several Saudi stock indices that will be used in combination of a trading algorithm to buy and sell shares based on three factors: share current value, current available balance, and a current number of shares owned.

Neural networks have gained much attention in recent years, especially in stock market prediction. The nature of the randomness accosted with the stock market makes it hard to achieve high confidence in predicting the index price using normal statistical methods. By using neural networks with several futures, we can achieve a high prediction value. To study the effect of past historical prices on future prices and to develop a trading agent using neural networks, we tested the Saudi stock market for the weak form efficiency. In producing a trading signal, the developed neural network is an RNN with an LSTM architecture.

The remainder of the paper is organized as follows. Section II gives a literature review on the works undertaken to predict and forecast the stock market price. Section III tests the weak form of the Saudi stock market efficiency. (The test is useful for understanding the effect of the historical data of a share on future values.) Discussion on the proposed method and a brief neural network introduction is presented in Section IV. Section V evaluates the proposed method and compares it to a known trading method. Finally, we give the conclusions of this study in Section VI.

II. Literature Review

Recently, the stock market prediction has been a hot topic in the research field. To predict stock prices, many researchers have developed methods, but only a few have developed a trading strategy. Some of the reviews of the developed methods are published [1, 2]. For example, Shah et al. classified stock prediction methods into four categories: statistical methods, pattern recognition, machine learning, and sentiment analysis.

The autoregressive integrated moving average (ARIMA) model, which is one of the well-known statistical methods, uses a class of models to model the time series based on historical values. The model is fitted to the historical values of a stock price in predicting (forecasting) the stock's future price. The model consists of three parts: (1) an autoregressive (AR) model, in which the forecasted value is a linear combination of past lagged values; (2) a moving average (MA) model that forecasts the future value using the past forecast errors; and (3) the difference operation of past and future values. The model is denoted by ARIMA(p, d, q), where p is the order (number of time lags) of the AR model, d is the degree of differencing, and q is the order of the MA model.

Pattern recognition is closely related to machine learning but with a different implementation. Here, we focus on the methods of finding patterns in the stock's historical values. Then, by using computer algorithms, we predict future values using these patterns. Previous studies show an example of a pattern: the stock uptrend [3] and the open high–low close price candlestick charts [4].

Machine learning prediction uses historical data and the desired output as the training sets to build a mathematical model through an iterative process until an objective function is optimized. Previous studies have shown the usage of classification and regression as examples of machine learning in trading methods and the closing price of stock [5, 6, 7].

In sentiment analysis, it uses text information, such as news articles or social media feeds on stock markets. In predicting stock trends based on the feed provided, the analysis employs machine learning algorithms [8].

Idress et al. [9] built an ARIMA model to predict the Indian stock market, in which they found a deviation on a 5% mean percentage error.

Meanwhile, to predict the Saudi stock prices, Olatunji et al. [10] proposed an artificial neural network (ANN) model, applying on three major stock indices: Alrajhi bank, Saudi Telecom Company, and Saudi Basic Industries Corporation SABIC stocks. They only used the previous-day closing price as the model input. Moreover, the proposed model was used as an investment adviser, and it achieved a low root mean squared error (RMSE) of 1.8174 and a mean absolute percentage error of 1.6476.

Also, Jarrah and Salim [11] proposed an RNN and a discrete wavelet transform (DWT) to predict the Saudi stock price trends. The model consisted of two stages. The first stage uses DWT to break the stock price into both frequency and time domains to filter the noise associated with the signals, and the second stage is an RNN that performs the prediction. The model was tested to predict the next-seven days closing price of the Saudi stock. The prediction result was then compared with that obtained by a prediction process performed using the ARIMA model. Consequently, the proposed model (DWT + RNN) achieved an RMSE of 0.0522 when the RNN model used four batches and four neurons.

Alotaibi et al. [12] also used an ANN model to predict the Saudi stock market. Their ANN model consisted of three layers: input, hidden, and output layers. The input layer contained the historical close and open prices of the Saudi stock market and the historical close and open prices of oil. Bayesian regularization backpropagation was used for network training from 2003 to 2012. The test set training spanned from 2013 to the end of 2015.

Hua et al. [13] gave an introduction to deep learning with LSTM for time series prediction and proposed random connectivity for LSTM to overcome the computation cost.

Tilakaratne et al. [5] developed a neural network for predicting the trading signals of the Australian All Ordinary Index. Then, they compared an ANN to a probabilistic neural network (PNN), in which they found that the ANN outperformed the PNN.

On the basis of the previous studies mentioned above, many developed methods use historical information form the share itself without the combination of other factors (e.g., oil prices). These methods targeted different markets other than the Saudi stock market.

III. Weak Form of Efficient Market Test

The weak form of an efficient market hypothesis states that the future prices of a stock market with a weak efficiency cannot be predicted using historical information, such as trading volume, closing price, and earnings. It means that one cannot predict future values using the available information. Fama [14] divided the efficient market hypothesis into three: weak, semi-strong, and strong hypotheses.

Previous studies tested the Saudi stock market efficiency in its weak form and concluded the same; however, the presented studies are not up to date [15, 16].

To prove that the stock price under test can be predicted using historical values, we will be testing the Saudi stock indices used to evaluate the proposed RNN for the weak-form efficiency hypothesis. The weak form of the market efficiency for individual stocks is tested for randomness. If the stock does not follow a random walk, the hypothesis fails. The stock index can be predicted using historical data.

Several statistics tests are known for use in testing data randomness. Here, we used the Kolmogorov–Smirnov test (K–S test). The null hypotheses in the K–S test are that the data (stock returns) under the test follow a random walk, and the future value cannot be predicted. The alternative hypotheses are that the data under test are not random and that the data can be predicted using historical values.

Here, we used Alrajhi, Alinma, and SABIC stocks. The historical values are dated from January 2010 to the end of March 2020. The stocks' closing price was converted to the stock returns, as shown in Eq. (1), where R is the logarithmic stock return; $l(i)$ is the day i closing price; and $l(i-1)$ is the previous closing price of the day i:

$$R(i) = \log\left(\frac{l(i)}{l(i-1)}\right) \quad (1)$$

A. Kolmogorov–Smirnov Test

The K–S test is a nonparametric test for data randomness. The null hypotheses of the test assume that the cumulative distribution function (CDF) of the data under test is equal to the hypothesized CDF. The CDF of the data was computed herein and compared with the hypothesized CDF using Eq. (2), where D_n is the maximum amount of the hypothesized CDF ($F_n(x)$) exceeding the calculated CDF ($G_n(x)$). When both CDFs are equal to some factors, the data are random, and the test fails to reject the null hypothesis that the test statistics converge to zero as n goes to infinity. Detailed mathematical background on the K–S test is provided in [17].

$$D_n = \max_x |F_n(x) - G_n(x)| \quad (2)$$

B. Market Weak form Test Results

We performed the test on the three stocks used to evaluate the proposed RNN. Table I shows the result of the K–S test performed with a significance level of 0.05. (The p-value is the probability value of the test.) Smaller values (typically <0.05) indicate a strong rejection of the null hypothesis. The test statistic is a random variable calculated from the data under the test used in determining the null hypothesis rejection, whereas the z-value is the critical value. The K–S

test rejected the null hypotheses by comparing the *p*-value with the significance level. The null hypothesis is rejected if the *p*-value is less than the significance level (i.e., the data under test are not random).

Based on the test performed, Alinma, Alrajhi, and SABIC stock returns did not follow a random walk and were not independent of past values. This proved that the proposed stock prediction method and the trading agent could facilitate historical values to predict trading signals.

TABLE I. KOLMOGOROV–SMIRNOV TEST RESULTS

Stock	Hypothesis test result	*p*-value	Test statistic	*z*-value
Alinma	The null hypothesis is rejected.	0.00	0.1191	0.0268
Alrajhi	The null hypothesis is rejected.	0.00	0.0904	0.0268
SABIC	The null hypothesis is rejected.	0.00	0.1143	0.0268

IV. METHODOLOGY

Neural networks are a set of algorithms used to recognize underlying relationships in data sets. The process of a neural network is similar to the operation of a human brain. Here, we used an RNN with an LSTM architecture to produce a trading signal.

The input to the neural network is called a feature, which is a measurable characteristic of the observed data or a characteristic with an indirect effect on it. Accordingly, this section provides a brief introduction to neural networks. The introduction aims to familiarize the reader with the basics of neural networks and provide them the ability to understand some concepts. A detailed background regarding this matter is reported in [18].

A. RNN

RNNs are a class of neural networks best used in sequenced data sets, such as time series. An RNN has a one-to-one connection between its internal layers and the exact position in the time series [18]. An RNN can simulate any algorithm given sufficient data. These networks are based on the works by Rumelhart et al. [19], who described a new method for teaching a network through backpropagation. Unlike feedforward neural networks, RNNs have an advantage in using their internal memory to process a sequence of data, such as stock markets. Moreover, the network input (e.g., oil prices and index price) in RNNs are interrelated. On the contrary, an RNN suffers from exploding problems and gradient vanishing. Gradient vanishing is a term associated with neural network training, and a gradient is a vector of the calculated error during the network training process. The gradient is used to update the network weights to achieve a small error, such as an error in predicted stock value when compared with the actual value. The gradients in an RNN accumulate during the update process, which causes it to explode (i.e., it becomes large and goes to infinity).

Fig. 1 shows the basic building block of an RNN. The input to the block is a vectored time series x_t. In our case, we used the stock price and associated features. h_t is the output from the block to be fed to the subsequent titration at time $t + 1$. h_{t-1} is the output from the previous block. Both h_t and h_{t-1} are called the hidden layer vectors. w_h and w_x are the weight vectors for the hidden connection and the input vector, respectively. The weight vectors are chosen by network training, which is achieved by comparing the output (predicted) with the actual value and adjusting the weight vectors to achieve the smallest error. F is an activation function within the block. Activation functions are mathematical equations that determine the block output based on preset conditions. The most important activation function is the *tanh* function. b_t is a bias added to the block input. Equation (3) shows the math behind RNNs.

$$o_t = h_t = F(w_h\, h_{t-1} + w_x\, x_t + b_t) \quad (3)$$

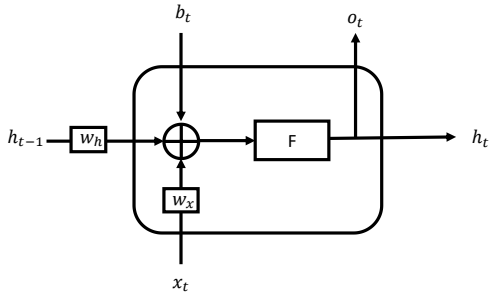

Fig. 1. The Basic Building Block of an RNN.

B. LSTM

Proposed by Hochreiter and Schmidhuber [20], LSTM is a type of RNN architecture used to solve the exploding and vanishing gradient problem that occurs in a normal RNN. The constant error carousel (CEC) LSTM was used to overcome the problems caused by the error back flow. The CEC controls the error flow by units, called gates, which are implemented in the memory block of the LMTS. The gates are categorized into the input gate, output gate, and forget gate, in which each gate has a function to achieve. The input gate controls the flow of the new sequence value. The output gate controls the usage of the value inside the cell using the activation function of the LSTM. The forget gate controls how long a value remains inside the memory cell.

Fig. 2 shows the building block of an LSTM unit, where C_t is the cell state, x_t is a vector input to the cell, f_t is the output from the sigmoid function that represents what cell state can be passed from adjacent cells, and i_t is the output from the sigmoid function that represents the output from the *tanh* function of the input gate to the cell. This updates the cell state with new values. O_t is multiplied by *tanh* of the cell state to choose what part to output to the adjacent cell.

Fig. 3 shows three hidden units for a vector input in an LSTM network. This number can be more than three, depending on the design. Equations (4)–(8) are the compact forms of the forward pass of an LSTM unit that contains a forget gate developed in [21]. In the equations, W, U, and b denote the weights and biases determined by network training. Each layer produces a single output, called ht, which is connected to a neuron at the final layer. The function of the neuron is to multiply each input by weight and sum them up to

produce an output $\hat{y}_t$ with length n, where n is the number of classifications produced (Fig. 4).

$$f_t = \sigma_g (W_f x_t + U_f h_{t-1} + b_f) \quad (4)$$

$$i_t = \sigma_g (W_i x_t + U_i h_{t-1} + b_i) \quad (5)$$

$$g_t = \sigma_c (W_g x_t + U_g h_{t-1} + b_g) \quad (6)$$

$$o_t = \sigma_g (W_o x_t + U_o h_{t-} + b_o) \quad (7)$$

$$f_t = \sigma_g (W_f x_t + U_f h_{t-1} + b_f) \quad (8)$$

$$h_t = o_t \circ o_h \quad (9)$$

The output $\hat{y}_t$ is connected to a softmax layer, which functions to convert the input vector $\hat{y}_t$ of n elements to a normalized probability distribution with n probabilities. The element with the highest probability is the network output. The produced classifications are two training signals: buy and sell. An in-depth discussion on pattern recognition and classification is shown in [22].

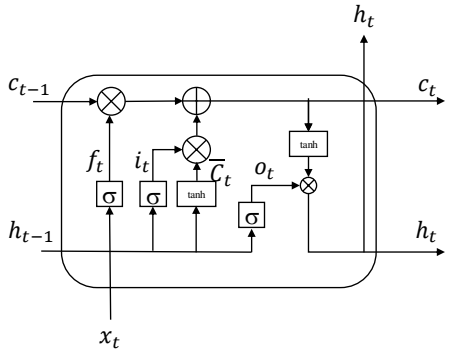

Fig. 2. LSTM basic Building Cell Called a Neuron or a Hidden Unit.

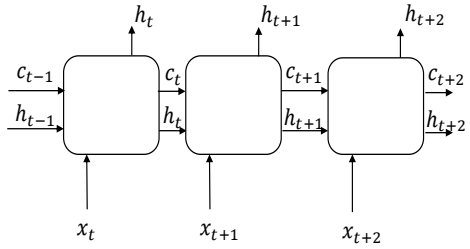

Fig. 3. The Network of the LSTM Units Known as Hidden Layers.

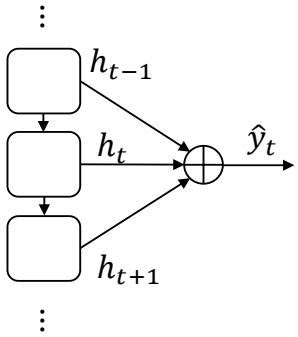

Fig. 4. Final Network Stage. The Output from each Cell is Added to Produce the Prediction.

C. Trading

The proposed design was constructed using LSTM layers connected in series. The RNN input comprised a set of time series data representing the features associated with the stock and oil closing price. The network setup consisted of the training method, the number of hidden elements (LSTM units), and the number of training titrations. Fig. 5 shows a history of three stock prices in Saudi Riyals that was used in this study. The data will be divided into two sets. The first set will be used to train the classifier, and the other data will be used to evaluate the proposed classifier. Fig. 6 shows the history of the oil prices that will be used as an input to the proposed network. Table II lists the options used in constricting the network.

1) Input features: Table III lists the features used for the buy and sell classification network. Several methods can be used for feature selection. However, in this study, we used a trial-and-error method to find the best feature combination because some feature selection methods fail when chart technical indicators are used in the stock price.

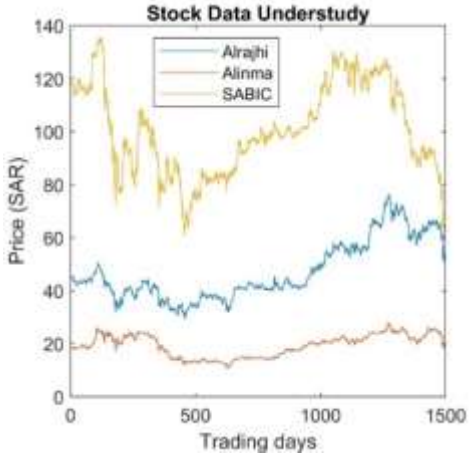

Fig. 5. Historical Data of Three Stocks from March 21, 2012, to April 24, 2020.

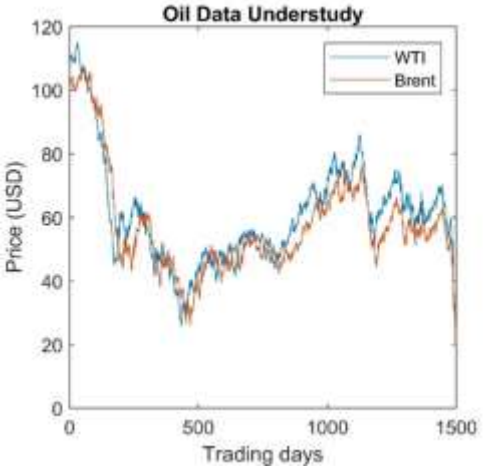

Fig. 6. Historical Data on Oil Prices in USD that are used in the Study. The Data are from March 21, 2012, to April 24, 2020.

TABLE II. LSTM NETWORK SETTINGS

Option	Description	Value
Solver	Training algorithm	ADAM
Epoch	Number of full training data passes	750
Hidden layers	Number of LTMS cell per time series	200
Gradient threshold	The gradient is clipped to the threshold if the gradient of the error passes the value	1
Initial learn rate	Specifying the rate of learning higher values will cause the learning to be faster, but could diverge the network	125

TABLE III. FEATURE DESCRIPTION

Features	Description
Stock closing price	The previous-day closing price of the stock
WTI daily price	West Texas Intermediate oil price
Brent daily price	Brent crude oil price
No. of trades	Number of trades placed on a stock for the previous day
Open price	The opening price of the same day
Highest price	The highest price of the previous day
Lowest price	The lowest price of the previous day
Month number	Current month in numerical form
Number of days	Since the last trading session Until the upcoming trading session
Relative strength index	Relative strength index for 7 days Relative strength index for 21 days
Accumulation/distribution (A/D) oscillator	Momentum indicator for detecting the changes in the A/D line by measuring the momentum of the first signal of change of trend
Moving average convergence/divergence	A trend-following momentum indicator that shows the relationship between two moving averages of a security's price
Stochastic oscillator	An indicator comparing a closing price of a stock to a range of its prices over a certain period
Logarithmic return	The logarithm of the closing price divided by the previous closing price shown in Eq. (3)

2) Network training: To obtain the required gains and biases in the hidden network layers, we must train the neural network. A data set must be prepared to perform the training and evaluation processes of the RNN. The required data were divided into two sets: a training set and an evaluation set. The data set comprised the historical values of the proposed futures from March 21, 2012, to April 24, 2020, and the required response (trading signal) of that interval. The trading agent responses were obtained from the stock returns, in which a buy signal was generated from a positive return, and a sell signal was entreated from a zero or negative return. The data were normalized using Eq. (9).

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)} \quad (9)$$

Each training run computes the generated responses with the required ones. An error is produced if the response is different, and the weights are updated in each training iteration. Adaptive moment estimation (ADAM), developed by Diederik Kingma and Jimmy Ba [23], was used as a solver to optimize the weights and biases of the neural network. The following lists the process undertaken to train the LSTM network.

- Initialize the LSTM network weights and biases randomly.
- Input the historical data to the network as a normalized time series.
- Compare the trading signal output with the required signal (buy and sell signal).
- Update the weights and biases using the ADAM solver and the computed error.
- Repeat the training process until the classification accuracy is higher than that in the previous run or stop when the required number of iterations has been satisfied.
- The evaluation data set was used to test the network after network training. This process is called the classification process.

3) Trading agent: The output of the neural network classification is connected to a trading agent. The presented trading agent strategy involves buying or selling a pre-defined number of shares in a trading session based on the number of shares and money currently owned. Fig. 7 depicts the trading process. The agent relies on the initial investment budget and the required shares to be bought and sold per trading session. These values are fixed in the current version of the trading agent.

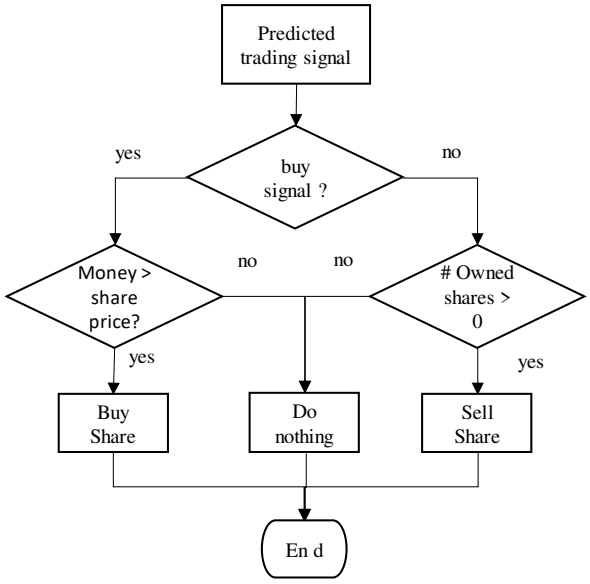

Fig. 7. Trading Agent Flow Chart.

V. Results

The proposed neural network and trading agent were evaluated using three stock shares from the Saudi stock market (i.e., Alinma Bank, Alrajhi Bank, and SABIC). The evaluation data set comprised of historical values from June 2018 to August 2019. The performance of the proposed agent was compared with that of the buy-and-hold trading strategy. Table IV shows the accuracy of the trading signal, trading agent initial values, and investment gain. The trading gain was affected by the initial values used, which were optimized to achieve the highest gain.

Fig. 8 and 9 denote the output of the trading agent for the Alinma and Alrajhi stocks, respectively. The trading agent was effective for both the Alinma and Alrajhi shares, as shown by the output. The agent bought shares in an upward trend and sold them at the local maximum in several instances.

TABLE IV. Trading Output Results

Stock	*Trading accuracy*			*Investment gain*	*Buy-and-hold gain*
	Buy	*Sell*	*Overall*		
Alinma	53.3%	50.3%	57.3	28.24%	6.9%
Alrajhi	51.9%	60.4%	57.3%	18.087 %	10.1 %
SABIC	47.8%	61.4%	57.3%	0.01%	−23%

Fig. 8. Trading Agent Signal when used in the Alinma Stock Trading.

Fig. 9. Trading Agent Signal when used in the Alrajhi Stock Trading.

Fig. 10. Trading Agent Signal when used in the SABIC Stock Trading.

Fig. 10 shows the trading signal of the SABIC shares. The agent predicted the correct trading signals when trading the SABIC shares, but the gain was not high compared with that of the other two shares because of the fixed amounts of shares that can be bought per trading session. This low gain can be fixed if the number of shares is dynamic and linked to the classification layer output score.

VI. Discussion and Conclusion

To predict the Saudi stock trading signals, we proposed the usage of a multivariate RNN with an LSTM architecture. The model used historical stock information, such as closing prices, the volume of trades, number of trades, current-day opening prices, and oil price. The model result was satisfying compared with that obtained using the buy-and-hold trading method.

In future studies, we must consider more factors, such as the Fibonacci retracement, and develop a feature selection method to select the best feature among the presented features. Other financial trading methods may also be considered to train a neural network and develop a trading agent instead of relying on the prediction of future returns.

Acknowledgment

The authors would like to thank Deanship of scientific research in King Saud University for funding and supporting this research through the initiative of DSR Graduate Students Research Support (GSR).

References

[1] Shah, H. Isah and F. Zulkernine, "Stock Market Analysis: A Review and Taxonomy of Prediction Techniques," International Journal of Financial Studies, vol. 7, p. 26, 5 2019.

[2] N. Singh, N. Khalfay, V. Soni and D. Vora, "Stock Prediction using Machine Learning a Review Paper," International Journal of Computer Applications, vol. 163, p. 36–43, 4 2017.

[3] P. Parracho, R. Neves and N. Horta, "Trading in financial markets using pattern recognition optimized by genetic algorithms," in Proceedings of the 12th annual conference comp on Genetic and evolutionary computation - GECCO, Portland, 2010.

[4] M. Velay and F. Daniel, "Stock Chart Pattern recognition with Deep Learning," 1 8 2018.

[5] C. D. Tilakaratne, M. A. Mammadov and S. A. Morris, "Predicting Trading Signals of Stock Market Indices Using Neural Networks," in AI 2008: Advances in Artificial Intelligence, Springer Berlin Heidelberg, 2008, p. 522–531.

[6] R. Dash and P. K. Dash, "A hybrid stock trading framework integrating technical analysis with machine learning techniques," The Journal of Finance and Data Science, vol. 2, p. 42–57, 3 2016.

[7] A. H. Moghaddam, M. H. Moghaddam and M. Esfandyari, "Stock market index prediction using artificial neural network," Journal of Economics, Finance and Administrative Science, vol. 21, p. 89–93, 12 2016.

[8] J.-L. Seng and H.-F. Yang, "The association between stock price volatility and financial news – a sentiment analysis approach," Kybernetes, vol. 46, p. 1341–1365, 9 2017.

[9] S. M. Idrees, M. A. Alam and P. Agarwal, "A Prediction Approach for Stock Market Volatility Based on Time Series Data," IEEE Access, vol. 7, p. 17287–17298, 2019.

[10] S. O. Olatunji, M. S. Al-Ahmadi, M. Elshafe and Y. A. Fallatah, "Forecasting the Saudi Arabia Stock Prices Based on Artificial Neural Networks Model," International Journal of Intelligent Information Systems, vol. 2, p. 77, 2013.

[11] M. Jarrah and N. Salim, "A Recurrent Neural Network and a Discrete Wavelet Transform to Predict the Saudi Stock Price Trends," International Journal of Advanced Computer Science and Applications, vol. 10, 2019.

[12] T. Alotaibi, A. Nazir, R. Alroobaea, M. Alotibi, F. Alsubeai, A. Alghamdi and T. Alsulimani, "Saudi Arabia Stock Market Prediction Using Neural Network," International Journal on Computer Science and Engineering, vol. 9, p. 62–70, 2 2018.

[13] Y. Hua, Z. Zhao, R. Li, X. Chen, Z. Liu and H. Zhang, "Deep Learning with Long Short-Term Memory for Time Series Prediction," IEEE Communications Magazine, vol. 57, p. 114–119, 6 2019.

[14] E. F. Fama, "Efficient Capital Markets: A Review of Theory and Empirical Work," The Journal of Finance, vol. 25, p. 383, 5 1970.

[15] U. Awan and M. Subayyal, "Weak Form Efficient Market Hypothesis Study: Evidence from Gulf Stock Markets," SSRN Electronic Journal, 2016.

[16] B. Asiri and H. Alzeera, "Is the Saudi stock market efficient? A case of weak-form efficiency," Research Journal of Finance and Accounting, vol. 4, no. 6, pp. 35-48, 2013.

[17] F. J. Massey Jr, "The Kolmogorov-Smirnov test for goodness of fit," Journal of the American statistical Association, vol. 46, p. 68–78, 1951.

[18] C. C. Aggarwal, Neural Networks and Deep Learning, Springer-Verlag GmbH, 2018.

[19] D. E. Rumelhart, G. E. Hinton and R. J. Williams, "Learning representations by back-propagating errors," Nature, vol. 323, p. 533–536, 10 1986.

[20] S. Hochreiter and J. Schmidhuber, "Long Short-Term Memory," Neural Computation, vol. 9, p. 1735–1780, 11 1997.

[21] F. A. Gers, J. Schmidhuber and F. Cummins, "Learning to Forget: Continual Prediction with LSTM," Neural Computation, vol. 12, p. 2451–2471, 10 2000.

[22] C. Bishop, Pattern recognition and machine learning, New York: Springer, 2006.

[23] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," 22 12 2014.