International Journal of Rough Sets and Data Analysis
Volume 7 • Issue 1

A RNN-LSTM-Based Predictive Modelling Framework for Stock Market Prediction Using Technical Indicators

Shruti Mittal, J. C. Bose University of Science and Technology, India
Anubhav Chauhan, J. C. Bose University of Science and Technology, India

ABSTRACT

The successful prediction of a stock's future price would produce substantial profit to the investor. In this paper, the authors propose a framework with the help of various technical indicators of the stock market to predict the future prices of the stock using recurrent neural network-based long short-term memory (LSTM) algorithm. The historical transactional data set is amalgamated with the technical indicators to create a more effective input dataset. The historical data is taken from 2010-2019, ten years in total. The dataset is divided into 80% training set and 20% test set. The experiment is carried out in two phases, first without the technical indicators and after adding technical indicators. In the experimental setup, it has been observed the LSTM with technical indicators have significantly reduced the error value by 2.42% and improved the overall performance of the system as compared to other machine learning frameworks that are not accounting the effect of technical indicators.

KEYWORDS

Artificial Neural Network, Deep Learning, LSTM, Machine Learning, Recurrent Neural Network, Stock Market Prediction

1.INTRODUCTION

To have a competitive edge, it is critical to have an insight into future and outcomes that challenge conventional assumptions. Thus, in almost all domains of life we are looking for the mechanisms which can help us in understanding what is likely to happen in near and far future and how we can take appropriate measures for gains and survival. Predicting about the future has neither been easy nor accurate particularly when it is based more on assumptions and intuitions. Science and mathematics have always been looking for the methods and techniques which can help us in taking conscious decisions based upon the facts and past history. For a credible prediction, the required ingredients can be accurate facts, precise history and a reliable evaluation mechanism. Thanks to the digitization of the data, precise historical data and accurate facts from the processed data are available on the desktops of the organizations and the only requirement that remains to be fulfilled is to design an appropriate predictive mechanism. The field of predictive analytics has grown richer with the time due to availability of accurate data sets and open source tools to carry out the computations making it feasible to predict in the areas which were otherwise considered unpredictable in general. One such area is stock market. Burton Malkiel, in his book "A Random Walk Down Wall Street", claimed that stock prices could not be accurately predicted (2007).

DOI: 10.4018/IJRSDA.288521

Online availability of accurate detailed data about the stock market transactions and performance of various indices has attracted the attentions of many data scientists to search for a reliable mechanism to predict the future trends in stock(s) in quite reasonable manner, if not accurate. Work proposed in this paper is an effort in this direction.

The paper has been divided into 6 sections. Section 2 takes up the literature survey wherein various efforts used in predicting the stock prices have been discussed. This section also takes up the shortcomings of these methods. Section 3 takes up the framework of the proposed methodology. Section 4 discusses the basic details of the proposed framework. Section 5 discusses the experimental setup and the results obtained. Section 6 concludes with the outcomes of the proposal.

2.LITERATURE SURVEY

Atsalakis et al. (Atsalakis & Valavanis, 2009; Zopounidis et al., 2013), have defined three major components for the accurate prediction of stock prices. These components include: Fundamental analysis, Technical analysis and the choice of appropriate technological methods. Fundamental Analysis includes the study of various parameters indicating the performance and credibility of the company over a period of time. Technical analysis is based on Dow Theory and uses price history for prediction. It is a form of a time series analysis, which determines the future price of a stock on the basis of the past price, exponential moving average (EMA), oscillators, support and resistance levels or momentum and volume indicators (Agrawal et al., 2019; Teixeira & De Oliveira, 2010).

The choice of appropriate technological methods for stock price forecasting has always been a challenging task. The Artificial Neural Networks (ANN), with their ability to learn, have always attracted the data scientists to use them in prediction problems with Stock Market being no exception (Shah, 2019; Wang, 2011). With the time, the machine learning techniques have improved and it has been possible to use multiple hidden layers or to cascade various layers, as in case of Convolutional Neural Network (CNN) making it feasible to extract the factual relationships which are otherwise hidden from the normal human analysis.

Singh et al. (Singh & Srivastava, 2016), have used Deep Neural Network algorithm to gain future price information. The performance was evaluated on Google stock price multimedia data (chart) from NASDAQ (National Association of Securities Dealers Automated Quotations exchange) and demonstrated that deep learning can improve stock market forecasting accuracy.

Guresen et al.(2011), analyzed the performance of the MLP (Multilayer perceptron), DAN2 (Dynamic Architecture for Artificial Neural Network, Hybrid Neural Network models for obtaining accurate prediction results. Studies have been mostly preoccupied with forecasting volatilities.

Pang et al. (2018), proposed a deep long short term memory neural network (LSTM) with embedded layer to predict the stock market.

Hegazy et al. (2013), proposed algorithm integrates Particle swarm optimization (PSO) and least square support vector machine (LS-SVM). The PSO algorithm is employed to optimize LS-SVM to predict the daily stock prices.

Nayak et al. (2016), proposed a model that predicts the future trend for the next day. Outcome of sentiment analysis is considered along with open price, close price of stock with extracted statistical parameters to build model.

The neural network models and frameworks that have been discussed above have taken into account different types of neural network techniques with dimensionality reduction. However, the domain of technical indicators is yet to be explored. In this paper, we analyze the accuracy of prediction using LSTM with technical indicators of the stock market. As many as 33 parameters are taken into account for making predictions. The next section discusses in detail recurrent neural network which is used to carry out predictions in the proposed framework.

3.RESEARCH METHODOLOGY

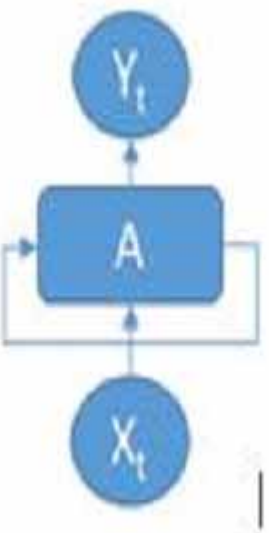
The proposed methodology uses various technical parameters apart from the data set for predicting the stock price. These parameters are computed with the help of the original data set. This research uses RNN as a prediction algorithm as it gives best performance in time series data as it takes the input based on the output of the previous timestamp.

Calculation are carried out for computing various technical parameters used in stock market prediction namely Simple Moving averages, MACD, Momentum, Moving Average Envelops (lower and upper), Money flow, McClellan Oscillator, On balance volume average directional index, Resistance, support, Bollinger bands, Average true Range, accumulation index, force index, pivot point, relative strength index, trending, McClellan summation index, moving average. The original data set along with these technical parameters data set are fed as input to the DNN (Deep Neural Network) that predicts the close price for the next day.

A Reccurent Neural Network

Unknowingly, predicting the future is we do all the time, whether we are trying to finish a friend's sentence or anticipating the smell of tea at breakfast. Recurrent neural networks (RNN) are the neural networks that can be used to predict the future accurately. They can analyze time series data such as stock prices, air quality index, bank fraud detection, autonomous driving systems, etc (Lien, 2017). The block diagram in Figure1 presents the architecture of Recurrent Neural Network.

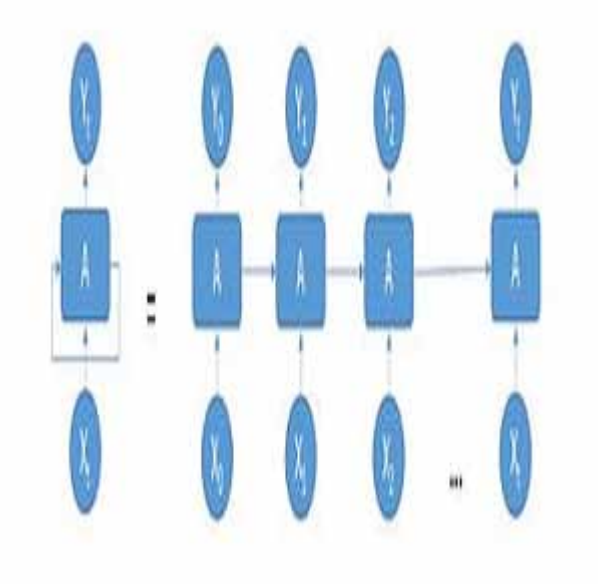
Figure.1. Recurrent Neural Network



RNNs are neural networks with loops in them which allow information to persist. A recurrent neural network, as shown in Figure 1, look like a feed forward neural network, except it also has connections pointing backward. At each time step t, this recurrent neuron receives the inputs X(t) as well as its own output from the previous time step, Y(t–1). We can represent this tiny network

against the time axis, as shown in Figure 2. This is called unrolling the network through time. The block diagram in Figure2 presents the architecture of Recurrent Neural Network.

Figure.2. An unrolled recurrent neural network



Each recurrent neuron has two sets of weights namely W_x and W_y. W_x is for the inputs $X_{(t)}$ and W_y is for the outputs of the previous time step, Y_{t-1}, which is taken as an input to this step. The output of a recurrent layer can be computed as shown in Equation 1, with the inputs X_t and Y_{t-1}, where b is the bias and Æ (•) is the activation function, (e.g. Rectified Linear Unit- ReLU, Hyperbolic Tangent function- Tanh).

$$\boldsymbol{H}_{(t)} = \varnothing(\boldsymbol{X}_{(t)}.\boldsymbol{W}_x + \boldsymbol{H}_{(t-1)}.\boldsymbol{W}_y + \boldsymbol{b}$$

$$= \varnothing\left[\boldsymbol{X}_{(t)}\boldsymbol{Y}_{(t-1)}\right].\boldsymbol{W} + \boldsymbol{b} \text{ with } \boldsymbol{W} = \begin{bmatrix}\boldsymbol{W}_x \\ \boldsymbol{W}_y\end{bmatrix}$$

$$\boldsymbol{Y}_{(t)} = \varnothing\left(\boldsymbol{H}_{(t)}.\boldsymbol{W}_h + \boldsymbol{b}\right)$$

Æ is the activation function (hyperbolic tangent)

$Y_{(t)}$ is the output layer matrix of size m ×n containing the layer's outputs at time step t for each instance in the mini batch

X_t is the input layer matrix of size m ×n inputs matrix containing the inputs for all instances (n is the number of input features).

W_x is the weight matrix of size n×n containing the connection weights for the inputs of the current time step.

W_y is the weight matrix of size n×n containing the connection weights for the outputs of the previous time step.

The weight matrices W_x and W_y are often concatenated into a single weight matrix W of shape (n inputs + n neurons) ×n neurons

b is a vector of size n neurons containing bias for each neuron.

Here $Y_{(t)}$ can be calculated from $X_{(t)}$ and $Y_{(t-1)}$, which in turn is a calculated from $X_{(t-1)}$ and $Y_{(t-2)}$, which in turn is dependent on $X_{(t-2)}$ and $Y_{(t-3)}$, and the recurrent cycle goes on . This makes $Y_{(t)}$ a function of all the inputs, $X_{(0)}$, $X_{(1)}$, …, $X_{(t)}$) in time t. In the first iteration, when t = 0, there value of $Y_{(t-1)}$ is assumed to be 0.

LSTM Network

LSTM is an advance version of RNN capable of learning long term dependencies. The problem of vanishing gradient does not exist in LSTM they are capable of handling long sequences of sentences easily. Suppose a language model trying to predict the next word in a sentence based on the previous ones. If we are trying to predict the last word in the sentence say "The clouds are in the sky**"**. The context here was pretty simple and the last word ends up being sky all the time. In such cases, the gap between the past information and th**e** current requirement are often bridged really easily by using recurrent neural networks. So, problems like Vanishing and Exploding Gradients which are present in don't exist and this makes LSTM networks handle long-term dependencies easily. LSTM have chain-like neural network layer. In a standard recurrent neural network, the repeating module consists of one single function.

There are various gates present in the LSTM to control the flow of information through the network called as forget gate, input gate and output gate and the key to these gates is called cell state as it acts as a conveyer belt which carries the information thorough these gates, finally giving output at the output gate. The first gate is known as forget gate the aim is to identify the information which is not required and thrown out of the cell state. The calculation is done by taking the new input and the input present at previous timestamp which then lead**s** to the output of a number between 0 and 1 for each number in that cell state. The value closer to the 1 means we have to keep that information and value closer to the 0 means we have to discard that information.

$$f_t = \sigma\left(W_{f.}\left[h_{t-1}, x_t\right] + b_f\right)$$

Where:

1) $f_{t=}$ forget gate's activation vector
2) w = weight matrices and bias vector parameters
3) h_{t-1} = hidden state vector present at previous timestamp
4) x_t = input vector to the LSTM unit
5) b_f = weight matrices and bias vector parameters
6) σ = activation function

The next step is to decide which new information we are going to store in the cell state which will be beneficial to us in making better predictions. Sigmoid layer gate is called Input Gate Layer which uses sigmoid activation function to decide which information is to be updated. Then the tanh layer creates the vector or product of new candidate $\hat{c}_t$ values, that will be added to the state.

$$i_t = \sigma\left(W_{i.}\left[h_{t-1}, x_t\right] + b_i\right)$$

$$\hat{c}_t = \tanh\left(W_c\left[h_{t-1}, x_t\right] + b_c\right)$$

Where:

1) i_t = input/update gate's activation vector
2) $\hat{c}_t$ = new candidate value

In this step we update the old cell state Ct-1 into new cell state that is Ct. Now we will multiply the old state by ft and forget the things that we decided earlier. Then we will add the product of $i_t * \hat{c}_{t-1}$. These are the new scaled values that tells us how much we have to update each state value. In the case of language model this is the step where we drop the information about the Old subject's gender and add the new things that we decided earlier.

$$c_t = f^* c_{t-1} + i_t^* \hat{c}_t$$

Now in the last step we finally decide what we are going to output. The final Output will be based on the cell state. First we will apply the sigmoid function to decide which part of the cell we are going to output, then we will pass it through the tanh function thus getting the values between (-1 to 1) and multiply it by the value of sigmoid gate output to so that we will only output the parts we decided to.

$$o_t = \sigma\left(W_{o.}\left[h_{t-1}, x_t\right] + b_o\right)$$

$$h_t = o_t^* tanhc_t$$

Where:

1) O_t = output gate's activation vector
2) h_t = hidden vector or output vector of LSTM unit

The model uses Root Mean Square Error (RMSE) to compute the loss function. It basically measures the difference between values predicted by a model and the actual values. RMSE is used in regression analysis to verify experimental results. The equations to compute the same are:

$$\boldsymbol{L_{RMSE}} = \left(\left(\sum_{\boldsymbol{i}-1}^{\boldsymbol{n}}\left(\boldsymbol{Y_{pi}} - \boldsymbol{Y_{ai}}\right)^2\right) / \boldsymbol{n}\right)^{1/2}$$

Where:

1) Y_{pi} = predicted value (expected values or unknown results),
2) Y_{ai} = actual value (known results).

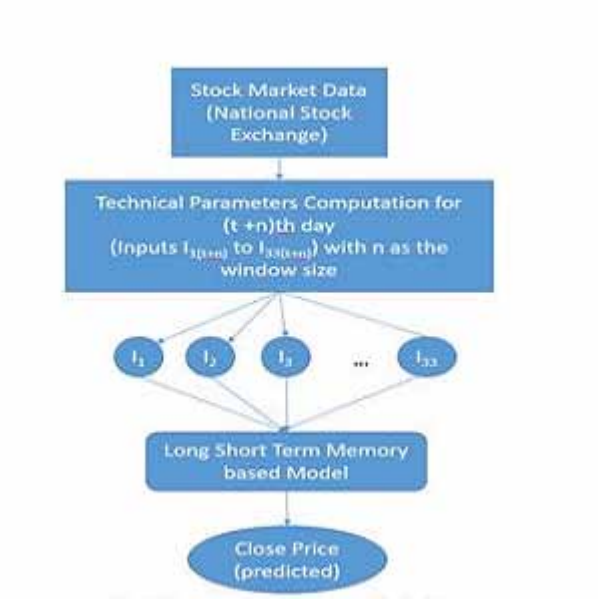
The mean absolute percentage error (MAPE), is used for measuring prediction accuracy of a model. The accuracy is expressed in percentage, and is defined as:

$$MAPE = \frac{100}{n}\left(\sum_{i=1}^{n}\left(\left|Y_{ai} - Y_{pi}\right|\right) / Y_{ai}\right)$$

4.PROPOSED METHOD

This paper introduces a predictive modeling framework based on LSTM for stock price prediction. It demonstrates the performance of this method on various factors. Fig 3 represents the proposed model for the same.

Figure 3. Framework of the Proposed Model



A Data Collection

The data is collected from National Stock Exchange website for ITC Company for the past 9 financial years. This paper focus on individual stock price prediction as it is more meaningful with respect to the investors (Pan et al., 2003). The data set used for experiment is from April 1, 2010 to December 31, 2019. Hence, the model is built for working 2418 days. The data set is divided into two categories namely 80% training set and 20% test set. Training set consists of data from April 1 2010 to Jan 10, 2018 and test set from Jan 11, 2018 to December 31, 2019. Each record in the dataset comprises of Open Price, High Price, Low Price, Previous Day Close Price, Last Traded Price, Close Price, Volume Weighted Average Price, 52-week High Price, 52-week Low Price, Volume, Value and Number of trades. Apart from these, various technical parameters as shown in table 1, are also computed and taken as inputs along with these to generate a composite result. The Block diagram in Figure3 represents the framework of proposed model.

This paper uses as many as 33 inputs (Teixeira & De Oliveira, 2010) to predict the composite output which is the close price for the next day. The forecasting is carried out in two stages. In stage1 the model is trained to compute the weights and the bias. In the second stage, the trained model is used to compute the output on test data set.

Table 1. List of all inputs (Original Dataset (I1 to I12) and Computed Technical Indicators (I13 to I33)) used in the Proposed Prediction Model

SNO	Name of the Input Variable	Symbol Used	Name of the Input Col
1	Open Price	OP	I1
2	High Price	HP	I2
3	Low Price	LP	I3
4	Previous Day Close	PDC	I4
5	Last Traded Price	LTP	I5
6	Close Price	CP	I6
7	Volume Weighted Average Price	VWAP	I7
8	52 Week Highest Price	52WH	I8
9	52 Week Lowest Price	52WL	I9
10	Volume	Vol	I10
11	Value	Val	I11
12	Number of Trades	NT	I12
13	Simple Moving Average	SMA	I13
14	Moving Average Convergence Divergence	MACD	I14
15	Momentum	Mom	I15
16	Moving Average Envelope Below	MAEB	I16
17	Moving Average Envelope Above	MAEA	I17
18	Money Flow Index	MFI	I18
19	McClellan Oscillator	MO	I19
20	On Balance Volume	OBV	I20
21	Average Directional Index	ADX	I21
22	Resistance Level	RL	I22
23	Support Level	SL	I23
24	Bollinger's Upper Band	BU	I24
25	Bollinger's Lower Band	BL	I25
26	Average True Range	ATR	I26
27	Accumulation Index	ACC	I27
28	Force Index	FI	I28
29	Pivot Point	PP	I29
30	Relative Strength Index	RSI	I30
31	Trending	TR	I31
32	McClellan Summation Index	McSI	I32
33	Exponential Moving Average	EMA	I33

5.EXPERIMENTAL SETUP

The experiments were conducted to predict the stock closing price using the Long Short-Term Memory based Neural Network algorithm with dataset and computed technical parameter as inputs. For the calculation of the technical parameter, Window size is taken into account. Window Size (WS) is the number of days taken into account for predicting the next day's price. For example, WS=40 implies, the data of 40 days is considered for predicting the value on the 41st day. Once the technical parameters are calculated, the output is computed for each day based on Deep learning equations of LSTM. Both the input data and output data for the training set are passed to the LSTM based model for training purpose. The loss function is set to the mean square error) and the number of epochs is set to 100. Early stopping criterion is used to decide the number of epochs, it uses a validation loss to check the overfitting condition when validation loss starts to rise that means we have to stop training the model. ADAM (adaptive moment estimation) (Kingma & Ba, 2014) algorithm for ðrst-order gradient-based optimization is used. The performance was measured using different error parameters such as Root Mean Square Error (RMSE) and Mean Average Percentage Error (MAPE). The experiments are conducted using a laptop with 4GB RAM, 2.71 GHz i3 processor on python of version 3.7.3. Different set of experiments are executed with varying the number of hidden layers and window sizes.

Table 2. MAPE and RMSE values for Original Dataset (inputs=12) and with technical indicators (inputs=33) for Different WS: (a) With Original Dataset(I1 to I12)

Window size	Mean Absolute Percentage Error	Root Mean Squared Error
20	**0.1305**	**0.282**
40	**0.1217**	**0.282**
60	**0.1019**	**0.246**
80	**0.1221**	**0.276**
100	**0.1188**	**0.0316**

Table 3. MAPE and RMSE values for Original Dataset (inputs=12) and with technical indicators (inputs=33) for Different WS: (b) with Computed Technical Indicators (I1 to I33)

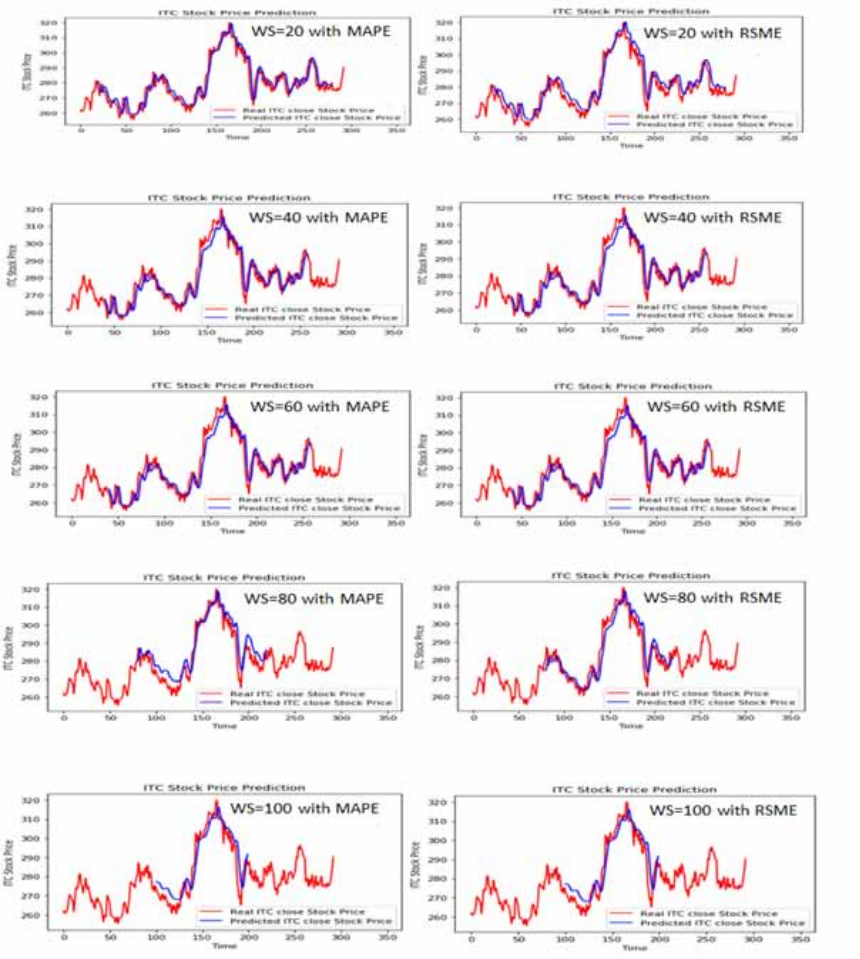
Window size	Mean Absolute Percentage Error	Root Mean Squared Error
20	0.1188	0.0278
40	0.1167	0.0278
60	0.1186	0.0268
80	0.1185	0.0257
100	0.1150	0.0242

6.RESULTS AND ANALYSIS

The experiments are executed in two different categories. Cat1 only with the original input data set and Cat 2 with the processed data set that contains the original input set along with the computed technical parameters. The errors (RMSE and MAPE) are observed for different window sizes from

the set WS={20,40,60,80,100} with epochs set to 100 for both the categories. The experimental results for different Window Sizes with different types of errors are shown in Fig.4& Fig.5 and Table 2(a) and (b). In the fig.4 and fig. 5, X-axis denotes the closing Price of the day and Y-axis represents the number of days. The results are different for different window sizes and different types of errors i.e. with initial set of inputs, and WS=20 with MAPE, the performance of the system was relatively better. Similarly, it has been observed that for WS=40, 60 and 80 the model had less MAPE values as compared to RMSE. However, for WS=100 the RMSE value was as less as 3.16%. The graph in Figure 4 represents the predicted vs actual without technical indicators.

Figure. 4. Graphs of Real and Predicted Values for different Window Sizes with Inputs (I_1 to I_{12})

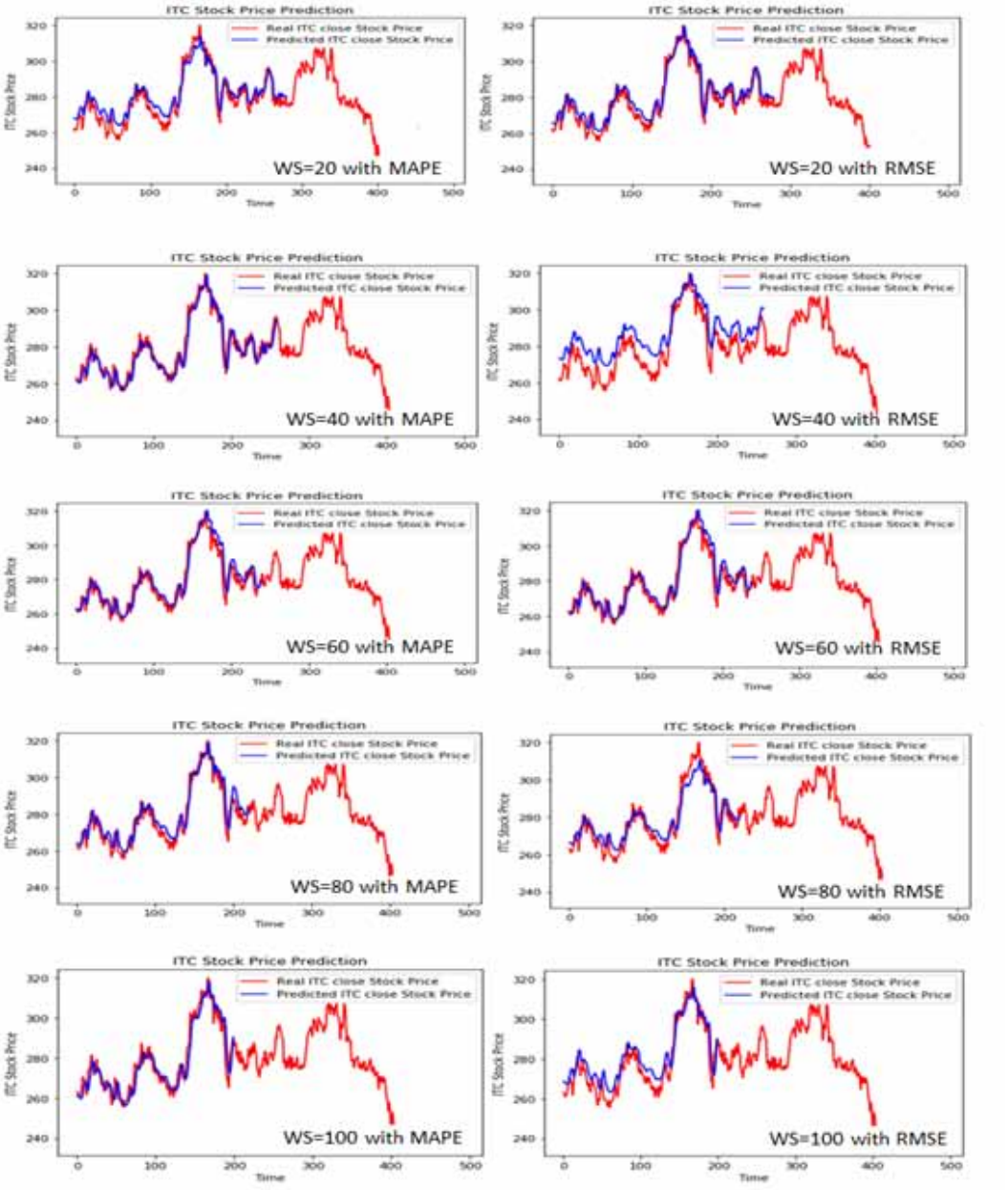


The same set of experiments were repeated for inputs I1 to I33, epochs set to 100 for different window sizes and different types of errors. It has been observed that increasing the number of inputs by computing the technical indicators has improved the performance of the model. This error percentage has reduced to 2.42%. It is also noted that with increased technical indicators in the input set, the WS

changes has very less effect on the error values. Moreover, the overall least error percentages were observed with WS=100.

The graph in Figure 5 represents the predicted vs actual with technical indicators.

Figure 5. Graphs of Real and Predicted Values for different Window Sizes with Inputs (I_1 to I_{33})



7.CONCLUSION AND FUTURE SCOPE

With experimentation, it has been observed that with DNN-LSTM, the stock Market Prices can be predicted with very least percentages of error. Moreover, it has been noted that increasing the input set by computing the various technical indicators has also improved the overall performance of the proposed model. It is found that with limited inputs, the WS=60 was performing better. However, as the number of inputs were increased with the technical indicators, the WS had very limited effect on the performance of the model. Experimental results clearly indicate that the proposed model provides a promising method for stock price forecasting. It is also found that the proposed model can perform more effectively if the number of inputs is increased further. Also, with the use of different Deep Learning Algorithm like Deep Belief Network, RBFNN etc. The overall performance of the system can be made better. Also, by adding different optimization techniques, the performance can tailored. The proposed work is focusing on the original data set and computed technical indicators for forecasting the close price of the stocks. External factors like trending stocks, portfolio and trading strategies shall also be 4onsidered for forecasting the close price.

REFERENCES

Agrawal, Khan, & Shukla. (2019). Stock Price Prediction using Technical Indicators: A Predictive Model using Optimal Deep Learning. *International Journal of Recent Technology and Engineering, 8*(2).

Atsalakis. & Valavanis. (2009). Surveying stock market forecasting techniques – Part II: Soft computing methods. *Expert Systems with Applications, 36*(3), 5932-5941.

Atsalakis, G., & Valavanis, K. (2013). Surveying stock market forecasting techniques - Part I: Conventional methods. In C. Zopounidis (Ed.), *Computation Optimization in Economics and Finance Research Compendium* (pp. 49–104). Nova Science Publishers, Inc.

Guresen, E., Kayakutlu, G., & Daim, T. U. (2011, August). Using artificial neural network models in stock market index prediction. *Expert Systems with Applications*, *38*(8), 10389–10397. doi:10.1016/j.eswa.2011.02.068

Hegazy, Soliman, & Salam. (2013). A Machine Learning Model for Stock Market Prediction. *International Journal of Computer Science and Telecommunications, 4*(12).

Kingma, D., & Ba, J. (2014). Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations*.

Lien. (2017). *Hands-On Machine Learning with Scikit-Learn and Tensor Flow*. Academic Press.

Malkiel. (2007). *A random walk down Wall Street: The Time Tested Strategy for Successful Investing*. WW Norton & Company.

Nayak, Pai, & Pai. (2016). Prediction Models for Indian Stock Market. *12th International Multi-Conference on Information Processing-2016 (IMCIP-2016)*.

Pan, Tilakaratne, & Yearwood. (2003). Predicting the Australian stock market index using neural networks exploiting dynamical swings and intermarket influences. Australas. Jt. Conf. Artif. Intell., 327–338.

Pang, X., Zhou, Y., Wang, P., Lin, W., & Chang, V. (2018). An innovative neural network approach for stock market prediction. *The Journal of Supercomputing*. Advance online publication. doi:10.1007/s11227-017-2228-y

Shah, D. (2019). *Stock Market Analysis: A Review and Taxonomy of Prediction Techniques*. Intl Journal of Financial Studies. doi:10.3390/ijfs7020026

Singh, R., & Srivastava, S. (2016). Stock prediction using deep learning. *Multimedia Tools and Applications*, 1–16. doi:10.1007/s11042-016-4159-7

Teixeira, L. A., & De Oliveira, A. L. I. (2010). A method for automatic stock trading combining technical analysis and nearest neighbor classification. *Expert Systems with Applications*, *37*(10), 6885–6890. doi:10.1016/j.eswa.2010.03.033

Wang. (2011, October). Forecasting stock indices with back propagation neural network. *Expert Systems with Applications*, *38*(11), 14346–14355.