Financial and Non-Stationary Time Series Forecasting using LSTM Recurrent Neural Network for Short and Long Horizon

1st Preeti
Department of Computer Science
Deen Dayal Upadhyaya College
University of Delhi
Delhi, INDIA
p06p05@gmail.com

2nd Rajni Bala
Department of Computer Science
Deen Dayal Upadhyaya College
University of Delhi
Delhi, INDIA
rbala@ddu.du.ac.in

3rd Ram Pal Singh
Department of Computer Science
Deen Dayal Upadhyaya College
University of Delhi
Delhi, INDIA
rprana@gmail.com

***Abstract*—Multi step ahead (long horizon) forecasting in a time series is a difficult task for various engineering applications in finance, geology, and information technology, etc. It is observed that different approaches mentioned in literature are at some disadvantage to give long term prediction for a non linear time series. Long Short-Term Memory (LSTM) network a type of Recurrent Neural Network (RNN) can be used to solve aforementioned problem and it may replace many practical implementations of the time series forecasting systems. This paper presents a novel LSTM model to give short and long horizon forecasting for a time series data. The LSTM method is preferable over other existing algorithms as LSTM network is able to learn non-linear and non-stationary nature of a time series which reduces error in forecasting. Also, long horizon forecasting is targeted in this paper which give wide range of its applicability. Comparative experiments with existing models demonstrate that our proposed LSTM ensemble method achieves state-of-the-art forecasting performance on four different real-life time series datasets publicly available.**

***Index Terms*—Deep Learning, Forecasting, LSTM, Recurrent Neural Network, Time Series**

I. Introduction

In recent years, Financial time series forecasting [1] has attracted substantial attention. A large number of applications have been developed to model and analyze the time series prediction [2].Specially, now a days forecasting for multiple step ahead in a time series has become very important and crucial. For instance, the stock market price prediction, exchange rates forecasting, energy, finance and several other benchmark time series problems. Most of the time series are inherently non-stationary and dynamic in nature as their statistical properties keep on changing over the time period. These series are significant from the fact because their analysis is beneficial for individual, companies and government.

II. Related Work

Since financial time series are so common in day to day life, numerous applications have been developed for this task [3]. A large number of statistical tools have already been applied and reported in literature for time series modelling and forecasting. These conventional statistical approaches include Fourier transform Autoregressive (AR), Autoregressive Moving Avergae (ARMA), Moving Average (MA) and Linear Regression. These traditional methods are unstable for such problems because of the data volatility, non-periodic and non-stationary nature of time series [4]. As compared to the statistical approaches, machine learning techniques such as fuzzy logic [5], support vector machines [6], neural network [7] give better results for time series forecasting in terms of different performance metrics. Apart from these techniques, a number of ensemble methods have also been studied and reported [8]. A hybrid approach using transfer learning and online sequential extreme learning machine is also proposed [9].

Recently, deep learning techniques have gained a lot of attention in financial time series forecasting applications. A deep learning network based framework on extreme learning machine with autoencoder (ELM-AE) has been reported in [10]. Recurrent neural network is one of them which has successfully been applied to time series prediction [8]. The major study in presented methods have only focused to find the short term prediction in a time series. However, a new challenge is to give prediction for long horizon that is to predict time series signal for a number of steps ahead. There are very few applications in time series for long-term prediction because of increasing uncertainties from various aspects such as accumulation of errors and the lack of information [2]. To name a few of them, Lai et al. [11] developed a model for long and short term temporal pattern in electricity, solar-energy and traffic data. Another author in his study [12] also presented LS-SVM to give long term prediction for electricity load.

It has been observed from existing study on non linear time series that the multi step ahead prediction is more valuable than just few step or one step ahead forecasting. Keeping in mind the importance of long horizon forecasting for real life problems, a type of recurrent neural network that is Long Short Term Memory (LSTM) has been studied in this paper. This paper provides an architecture which is applied to give

short and long horizon prediction for financial time series. The short term prediction is used to give comparison with previous study on time series data and further extended to long horizon forecasting. The experimental results shows the effectiveness and efficacy of discussed method for long term applicability. The generalization ability of proposed algorithm has been tested on four different benchmark datasets.

The rest of this paper is organized as follows: Section 2 describes the formulation of the problem and recurrent neural network architecture used for solving the formulated problem. In the next section, datasets used for study are described, including performance metrics and pre-processing steps used to clean the data. Subsequent subsection outlines the experimental details and obtained results. Finally, conclusion of study is presented in section 5.

III. Framework

In this section, we first formulate the time series forecasting problem for both short and long term horizon prediction. Problem formulation follows the details of the proposed LSTM recurrent neural network. Finally, an objective function is introduced for problem formulation.

A. Problem Formulation

In this phase, the original time series data is converted into an input-output matrix with a set of input features given a memory order m and a label as output variable. The transformed data set is used for further processing. We are interested in time series forecasting for both short and long horizons. More formally, original time series of n observed time signals is represented by $Y = y_1, y_2, \ldots, y_t$ where $y_t \in \mathbb{R}$ at time signals $t = 1, 2, 3, \ldots, n$, respectively. The given original data is just a single column depicting the closing value for time signal t. This series of closing values is transformed into a set of input features with memory order m and an output variable as label. The data set of records containing input vector and output label is used for further processing. The steps used for formulation of n^{th} instance of input-output phase reconstruction is following:

1) Consider the Memory order m as 4 which implies at current time t_{cur} the value of y_{cur} time series observation will be dependent upon 4 previous observations.The value of m is considered to be 4 through experimental study for memory order.
2) Take k as a skipping parameter to denote the gap among past four observations of Y time series.The value of k is varied in range $1, 2, \ldots, 10$ to find the best possible value of parameter k.
3) Consider v the number of sample steps ahead varying as $1, 5, 10, 15, 20$. When v value is 1 and k is also 1, it is treated as short horizon prediction at one step ahead. For remaining possible values of v, it is termed as long horizon forecasting at v step ahead using four past samples, with a gap of k parameter.
4) Hence, n^{th} input instance represented by x_n is

$$x_n = [y_{n-v-(k\times 0)}, y_{n-v-(k\times 1)}, y_{n-v-(k\times 2)}, y_{n-v-(k\times 3)}]^T$$

n^{th} output instance, s_n is y_n. For example, if k is 5 and v samples step ahead than corresponding nth instance of the reconstructed data is as follows:

$$x_n = [y_{n-v}, y_{n-v-5}, y_{n-v-10}, y_{n-v-15}]^T$$
$$s_n = y_n$$

5) Predicted value of time series signal will be stored in $\tilde{s_n}$.

B. Recurrent Neural Network Component

The architecture used to solve the stated problem is based on Recurrent Neural Network (RNN) [8]. The RNN is a type of neural network which uses current input and output obtained using previous input to compute current output. But the standard RNN is not able to handle the problem of exploding and vanishing gradient. This problem is addressed in one of the several types of RNN family that is Long Short Term Memory (LSTM) network [13] with the help of a memory cell referred as LSTM cell. This type of network posses the capability to learn long term dependencies in data which shows its applicability to a time series problem.

LSTM network architecture is a sequential model based upon two critical components states and gates. The states include hidden state which depicts the value of previous hidden layer and input state which is a linear combination of current input data and hidden state. The hidden state is also referred as output vector and denoted by s. The LSTM cell is consist of three gates that are input gate, forget gate and output gate denoted by i, f and o, respectively. Each unit of LSTM cell network is consisted of these three gates and uses an optimizer function to update the weights associated with units of network. At forget gate, f_t is computed using Eqn. (1) to find which information from previous state to be kept for further computation.

$$f_t = \sigma(W_{fx}x_t + W_{fs}s_{t-1} + b_f) \tag{1}$$

where, σ denotes the sigmoid activation function. After that, input gate is used to find an intermediate parameter i_t using Eqn. (2) and C_t using Eqn. (3) to decide whether internal state value serve as memory cell.

$$i_t = \sigma(W_{ix}x_t + W_{is}s_{t-1} + b_i) \tag{2}$$
$$c_t = \tanh(W_{cx}x_t + W_{cs}s_{t-1} + b_c) \tag{3}$$

Finally, the information to be kept is decided by merging input and forget gates output in Eqn. (4).

$$C_t = f_t \cdot C_{t-1} + i_t \cdot c_t \tag{4}$$

This sigmoid and tanh layers are used to compute new information being store in cell state. Then, output layer captures the output in Eqn. (5) which is used to give final output prediction s_t in Eqn. (6).

$$o_t = \sigma(W_{ox}x_t + W_{os}s_{t-1} + b_o) \tag{5}$$
$$\tilde{s_t} = o_t \cdot \tanh(C_t) \tag{6}$$

where, W_{fx}, W_{fs}, W_{ix}, W_{is}, W_{cx}, W_{cs}, W_{ox}, W_{os}, b_f, b_i, b_c and b_o are corresponding weights and biases used at different layers, respectively and $\tilde{s_t}$ denotes the output of LSTM network at time signal t.

C. Objective Function

The optimization objective for most of the regression problems such as time series forecasting is a square loss function. The formulation of corresponding objective function is

$$\min_{(W,b)} \sum_{t=1}^{t_{train}} ||s_t - o_t \cdot \tanh(f_t \cdot C_{t-1} + i_t \cdot c_t)||_F^2 \quad (7)$$

where, W denotes the set of parameters used in model, t_{train} depicts the set of time signals used for training the model and $||\cdot||_F$ denotes the Frobenius norm. The set of different metrics used to minimize the loss function including Root mean square error (RMSE). Since, square loss function is not robust towards outliers in the dataset. So, absolute error function is incorporated as an objective function of our model which can be formulated as

$$\min_{(W,b)} \sum_{t=1}^{t_{train}} |s_t - o_t \cdot \tanh(f_t \cdot C_{t-1} + i_t \cdot c_t)| \quad (8)$$

In the experiment section, both RMSE and MAE have been reported for both horizon forecasting for minimization of objective functions given in Eqns. 7 and 8.

IV. Evaluation

This section gives a brief description about the time series considered for experiments, performance metrics used to evaluate the proposed algorithm and detailed discussion about dataset-wise obtained results.

A. Dataset and Pre-processing

The proposed method is applied to four different financial and non stationary time series: Mackey Glass chaotic time series[1], Dow Jones Industrial Average (DJIA) (US stock market & Economy), three exchange rate data sets based on daily US Dollar (USD). Exchange rate data series include India, Japan and South Korea obtained from weblink & finance yahoo[2]. The descriptive statistics of these series used for long and short horizon forecasting is shown in Table I. Each of the data-sets is partitioned into 70% training set, 20% test set, and 10% validation set respectively.

The time series data sets used for experimental work are pre-processed before using them for training and testing of proposed method. After performing input-output phase space reconstruction III-A, the data set is normalized in the range of [0,1] under pre-processing. In order to get the obtained normalized data into the right shape for input of the LSTM sequential model, it is essential to reshape the input features of training and test data into 3 dimensional matrix. The dimensions of this $3D$ matrix are number of samples, time steps (1 in our case) and number of features (Memory order). The reshaped dataset is used further for training and testing the model.

[1]https://figshare.com/articles/Mackey-Glass_time_series/4233584/1

[2]https://finance.yahoo.com

B. Metrics

A number of performance metrics obtained through Eqns. 9-15 are adopted to establish the effectiveness and efficacy of proposed method on different time series.The different performance metrics used to find accuracy of method are based on error criteria. These include the correlation coefficient (r) , Root Mean Square Error (RMSE), Mean Absolute Error (MAE), relative error (%) based on RMSE value ($RMSE_{\overline{s}}$), Willmott's Index (WI), the Nash-Sutcliffe coefficient (E_{NS}), and Legates and McCabe Index (E_{LM}) are represented below:

$$r = \frac{\sum_{t=1}^{n}[(s_t - \overline{s})(s_t - \overline{\tilde{s}})]}{\sqrt{\sum_{t=1}^{n}((s_t - \overline{s}))^2} \cdot \sqrt{\sum_{t=1}^{n}(s_t - \overline{\tilde{s}})^2}} \quad (9)$$

$$RMSE = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(s_t - \tilde{s}_t)^2} \quad (10)$$

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|(s_t - \tilde{s}_t)| \quad (11)$$

$$RMSE_{\overline{s}} = 100 \times \frac{\sqrt{\frac{1}{n}\sum_{t=1}^{n}(s_t - \tilde{s}_t)^2}}{\overline{s}} \quad (12)$$

$$WI = 1 - [\frac{\sum_{t=1}^{n}(s_t - \tilde{s}_t)^2}{\sum_{t=1}^{n}(|(s_t - \tilde{s}_t)|) + (|(s_t - \tilde{s}_t)|)^2}], \text{ and } (0 \leq WI \leq 1) \quad (13)$$

$$E_{NS} = 1 - [\frac{\sum_{t=1}^{n}(s_t - \tilde{s}_t)^2}{\sum_{t=1}^{n}(s_t - \overline{s})^2}], \text{ and } (\infty \leq E_{NS} \leq 1) \quad (14)$$

$$E_{LM} = 1 - [\frac{\sum_{t=1}^{n}|(s_t - \tilde{s_t})|}{\sum_{t=1}^{n}|s_t - \overline{s}|}], \text{ and } (\infty \leq E_{LM} \leq 1) \quad (15)$$

where, n is the total number of records, s denotes the actual value of output variable, $\tilde{s}$ is a vector of forecasted value, and variables $\overline{s}$ and $\overline{\tilde{s}}$ denote the mean of set of actual and forecasted values, respectively.

The different error related statistics obtained through Eqns. 9-15, are adopted to establish the accuracy of forecasted $\tilde{s}$ with respect to actual s values. Both RMSE and MAE are used to capture the error distribution between actual and forecasted values whereas, RMSE is more representative only when error distribution is expected to be Gaussian [14]. However, [15] shows that MAE is an unambiguous and more natural measure of average error. According to Vapnik [16], $RMSE_{\overline{s}}$ depicts a model to be excellent if its value ¡ 10%, good if this metric value lie between 10% and 20%, fair in between 20% and 30%, and poor if it's value for some model is greater than 30%. The value of other metrics used here helps to better access the model performance. The efficiency coefficient E_{NS} and E_{LM} vary between $-\infty$ and 1.0, where value 1 is considered to be the optimum value. The $E_{NS} \leq 0.0$ indicates model is

TABLE I
DESCRIPTIVE STATISTICS OF DIFFERENT TIME SERIES DATASETS CONSIDERED FOR FORESCASTING.

Time Series	Count	Min	Max	Mean	Std. Dev	Skewness	25.00%	50.00%	75.00%
DJIA	1003	7286.27	11337.92	9683.42	876.56	-0.62	9054.26	9923.04	10383.92
Mackey Glass	1200	0.22	1.31	0.92	0.24	-0.45	0.73	1	1.12
India Exch. rate	11267	7.19	68.86	31.29	19.05	0.16	10.78	31.7	46.04
Japan Exch. rate	11769	75.72	358.44	161.39	73.35	0.94	108.18	124.1	226.55

not a good predictor, and $0 < E_{NS} < 1$ is considered as the acceptable range.

C. Experimental Details

The proposed LSTM method was implemented using Tensorflow and keras library in python and was trained with root mean square (rmsprop) optimizers. Two different models of proposed method were implemented. For both the models the count of hidden layers was two only. The number of hidden units for two models were 128×128 and 128×64, respectively. Further, addition of number of layers and number of hidden units did not affect the performance of proposed method. The grid search was performed on data formulated for short term forecasting to tune hyper-parameters that is number of epochs, k and choose the model between 128×128 and 128×64. Those optimized set of parameters were considered further for long horizon prediction.

So, experimental study was conducted for four different data sets discussed in section IV-A. For each data set, short term that is 1 step ahead prediction and long horizon that is 5, 10, 15 and 20 steps ahead prediction was conducted. For each horizon, grid search for a parameter k was performed between 1 to 10, where parameter k denotes the gap between the set of previous observations to be used in construction of input-output instances. After that the results on different metrics are calculated and presented in section IV-D. All the results reported are computed on test data-sets.

In order to demonstrate the effectiveness of given method, the obtained results are compared with other state-of-art algorithms: Extreme Learning Machine (ELMK) [17], Artificial Neural Network (ANN) [18], Online Support Vector Regression(OLSVR) [19], Online Sequential Extreme Learning Machine with Kernel (OS-ELMK) [20] as shown in Table VI. this table shows that proposed method is capable of learning the pattern of different time series for short as well as long term.

D. Results

The evaluation of proposed method to forecast the financial time series data-sets for short and long (5, 10, 15, 20 steps) horizon is presented in this section using the different performance metrics discussed from Eqns. 9 - 15. Table II - V present the one step ahead forecasting for four different time series such as DJIA, India exchange rate and Japan exchange rate, respectively. The set of results in these tables are obtained from two different models (1 & 2) that is 128 × 64 and 128 × 128, respectively and tries to find optimal number of epochs to be used while training the data for long horizon.

TABLE II
ONE STEP AHEAD PREDICTION FOR DJIA TIME SERIES USING TWO MODELS.

Model	128 × 64 model		128 × 128 model	
Epochs	Training RMSE	Testing RMSE	Training RMSE	Testing RMSE
40	0.00124	**0.01699**	0.00129	0.02642
35	0.00126	0.02664	0.00124	0.05864
30	0.0013	0.02011	0.00134	0.05768
25	0.00134	0.03541	0.00135	**0.01902**
20	0.00146	0.09199	0.00147	0.02418

TABLE III
ONE STEP AHEAD PREDICTION FOR MACKEY GLASS TIME SERIES USING TWO MODELS.

Model	128 × 64 model		128 × 128 model	
Epochs	Training RMSE	Testing RMSE	Training RMSE	Testing RMSE
40	0.00016	0.0233	0.0001	0.03067
35	0.00016	0.03018	0.0004	0.00676
30	0.00019	**0.00414**	0.00022	**0.00367**
25	0.00023	0.00872	0.00013	0.0224
20	0.00026	0.03067	0.00011	0.00782

Short horizon forecasting: Out of four time series datasets, model 2 is giving better RMSE on three of the time series. Table II shows model 1 gives better performance with RMSE value 0.01699 at 40 number of epochs. For Mackey glass series, least RMSE 0.00367 is obtained at 30 number of epochs using model 2. It is found that India and Japan exchange rate datasets are also obtaining best results on model 2 with RMSE 0.0165 and 0.00575 with number of epochs as 30 and 40, respectively. So, the obtained models and parameters can be used further for long term prediction.

Long Horizon prediction for DJIA, India and Japan exchange rate time series datasets is presented in Tables VII - IX. For every data set, the parameters obtained in short horizon forecasting are used for long term prediction. The results are

TABLE IV
ONE STEP AHEAD PREDICTION FOR INDIA EXCHANGE RATE TIME SERIES USING TWO MODELS.

Model	128 × 64 model		128 × 128 model	
Epochs	Training RMSE	Testing RMSE	Training RMSE	Testing RMSE
40	0.00012	0.03544	0.00014	0.0336
35	0.00014	**0.02245**	0.00015	0.03313
30	0.00017	0.03128	0.00015	**0.0165**
25	0.00018	0.02353	0.00019	0.01965
20	0.00021	0.04613	0.00025	0.01844
15	0.00027	0.03493	0.00029	0.07744

TABLE V
ONE STEP AHEAD PREDICTION FOR JAPAN EXCHANGE RATE TIME SERIES USING TWO MODELS.

Model	128 × 64 model		128 × 128 model	
Epochs	Training RMSE	Testing RMSE	Training RMSE	Testing RMSE
40	0.00023	0.00851	0.00025	**0.00575**
35	0.00028	0.00608	0.00026	0.00746
30	0.00029	0.00704	0.00029	0.00888
25	0.00034	0.00609	0.00035	0.0086
20	0.00041	0.01356	0.00042	0.00721
15	0.00049	**0.00607**	0.0005	0.01106

TABLE VI
COMPARISON OF DISCUSSED METHODS IN LITERATURE AND OUR CASE STUDIES.

Data-set	Model	RMSE
Mackey Glass	ELM [20]	0.0121
	OLSVR [20]	0.0192
	FLT-RNN [21]	0.0668
	OS-ELMK [20]	0.0242
	ANN+PSO [18]	0.0053
	TrOS-ELMK [9]	0.0082
	our method	**0.00367**
DJIA	ELM [20]	0.133
	OLSVR [20]	0.138
	OS-ELMK [20]	0.172
	our method	**0.0169**

computed using seven different performance metrics discussed in subsection IV-B for 5, 10, 15 and 20 step ahead prediction. It is also tried to find the best value of parameter k between 1 to 10. The best value obtained for different performance metrics at different steps ahead on different value of k is computed in Tables VII - IX.

In most of the results best possible value of a metric is obtained at $k \leq 6$. Since long term prediction is only possible after learning the pattern of the time series which is being learned with the help of skipping parameter k. The best value of k varies from one series to other because every series possess its own pattern. The finding about k is also depicted through line plots depicted in Figs. 1 - 3 as well for different steps ahead prediction and different data sets. The above result imply that proposed method is capable of giving good results on short term as well as long horizon prediction.

V. CONCLUSIONS

In this work, a recurrent neural network based on Long Short-Term Memory (LSTM) is used for short and long horizon forecasting for different time series. The study is conducted on four different time series data sets that are Mackey Glass, DJIA, India Exchange rate and Japan Exchange rate. It is tried to find the impact of gap among the past observations on short and long term horizons forecasting. It is observed that proposed method is capable to achieve significantly better performance with respect to other state-of-art algorithms in literature such as Neural Networks, ELM, Support vector Regression. It is also found that the skipping parameter used to represent the gap among past observations significantly affect the long horizon forecasting. The proposed method LSTM based on RNN outperforms the other methods as it is able to learn the long term dependencies in different types of time series datasets. In the future work, various types of recurrent neural network can be studied for different types of multivariate time series datasets such as energy, oil-price forecasting for long term horizon.

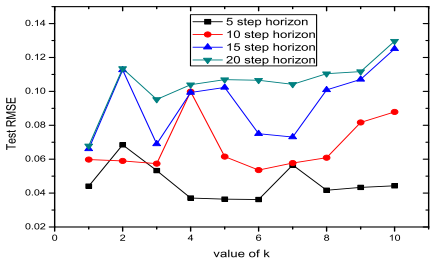

Fig. 1. DJIA long term prediction w.r.t. parameter k.

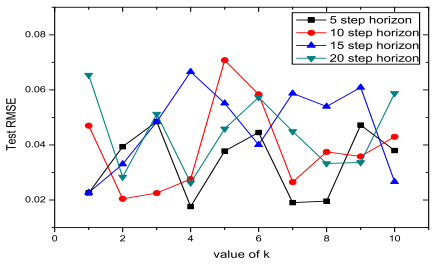

Fig. 2. India Exchange rate data long term prediction w.r.t. parameter k.

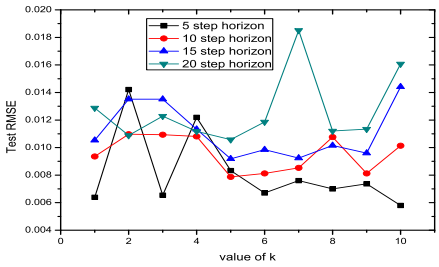

Fig. 3. Japan Exchange rate data long term prediction w.r.t. parameter k.

REFERENCES

[1] Y. S. Abu-Mostafa and A. F. Atiya, "Introduction to financial forecasting," *Applied Intelligence*, vol. 6, no. 3, pp. 205–213, 1996.
[2] A. S. Weigend, *Time series prediction: forecasting the future and understanding the past*. Routledge, 2018.
[3] R. Singh and S. Balasundaram, "Application of extreme learning machine method for time series analysis," *International Journal of Intelligent Technology*, vol. 2, no. 4, pp. 256–262, 2007.
[4] A. J. Hussain, R. Ghazali, D. Al-Jumeily, and M. Merabti, "Dynamic ridge polynomial neural network for financial time series prediction," in *Innovations in Information Technology, 2006*. IEEE, 2006, pp. 1–5.
[5] R. Khemchandani, S. Chandra *et al.*, "Regularized least squares fuzzy support vector regression for financial time series forecasting," *Expert Systems with Applications*, vol. 36, no. 1, pp. 132–138, 2009.
[6] K.-j. Kim, "Financial time series forecasting using support vector machines," *Neurocomputing*, vol. 55, no. 1-2, pp. 307–319, 2003.

[7] R. Adhikari, "A neural network based linear ensemble framework for time series forecasting," *Neurocomputing*, vol. 157, pp. 231–242, 2015.
[8] F. A. Gers, D. Eck, and J. Schmidhuber, "Applying lstm to time series predictable through time-window approaches," in *Neural Nets WIRN Vietri-01*. Springer, 2002, pp. 193–200.
[9] R. Ye and Q. Dai, "A novel transfer learning framework for time series forecasting," *Knowledge-Based Systems*, vol. 156, 05 2018.
[10] Preeti, A. Dagar, R. Bala, and R. Pal Singh, *Financial Time Series Forecasting Using Deep Learning Network*, 03 2018, pp. 23–33.
[11] G. Lai, W.-C. Chang, Y. Yang, and H. Liu, "Modeling long-and short-term temporal patterns with deep neural networks," in *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval*. ACM, 2018, pp. 95–104.
[12] A. Sorjamaa, J. Hao, N. Reyhani, Y. Ji, and A. Lendasse, "Methodology for long-term prediction of time series," *Neurocomputing*, vol. 70, no. 16-18, pp. 2861–2869, 2007.
[13] S. Hochreiter and J. Schmidhuber, "Lstm can solve hard long time lag problems," in *Advances in neural information processing systems*, 1997, pp. 473–479.
[14] T. Chai and R. R. Draxler, "Root mean square error (rmse) or mean absolute error (mae)? arguments against avoiding rmse in the literature," *Geoscientific Model Development*, vol. 7, no. 3, pp. 1247–1250, 2014. [Online]. Available: https://www.geosci-model-dev.net/7/1247/2014/
[15] C. J. Willmott and K. Matsuura, "Advantages of the mean absolute error (mae) over the root mean square error (rmse) in assessing average model performance," *Climate research*, vol. 30, no. 1, pp. 79–82, 2005.
[16] V. N. Vapnik, "An overview of statistical learning theory," *IEEE transactions on neural networks*, vol. 10, no. 5, pp. 988–999, 1999.
[17] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 2, pp. 513–529, 2012.
[18] C. López-Caraballo, I. Salfate, J. Lazzús, P. Rojas, M. Rivera, and L. Palma-Chilla, "Mackey-glass noisy chaotic time series prediction by a swarm-optimized neural network," in *Journal of Physics: Conference Series*, vol. 720, no. 1. IOP Publishing, 2016, p. 012002.
[19] J. Ma, J. Theiler, and S. Perkins, "Accurate on-line support vector regression," *Neural computation*, vol. 15, no. 11, pp. 2683–2703, 2003.
[20] X. Wang and M. Han, "Online sequential extreme learning machine with kernels for nonstationary time series prediction," *Neurocomputing*, vol. 145, pp. 90–97, 2014.
[21] S. L. Badjate and S. V. Dudul, "Novel ftlrnn with gamma memory for short-term and long-term predictions of chaotic time series," *Appl. Comp. Intell. Soft Comput.*, vol. 2009, pp. 2:1–2:9, Jan. 2009. [Online]. Available: http://dx.doi.org/10.1155/2009/364532

APPENDIX

TABLE VII

DJIA TIME SERIES LONG HORIZON PREDICTION USING SELECTED MODEL ON VARIOUS PERFORMANCE METRICS.

Step	Metrics	k: gap among observations									
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
5	**r**	0.78095	0.78742	**0.79947**	0.79463	0.79727	0.79357	0.79645	0.79729	0.77991	0.79343
	RMSE	0.04408	0.06845	0.05327	0.03708	0.03641	**0.03618**	0.05655	0.04167	0.04339	0.04432
	MAE	0.03633	0.05950	0.04489	0.03031	0.02972	**0.02957**	0.04756	0.03348	0.03603	0.03545
	$RMSE_{\overline{S}}$	6.00605	9.32508	7.25755	5.05027	4.95759	**4.92566**	7.69642	5.67133	5.90489	6.03102
	WI	0.83123	0.70065	0.78467	0.88114	0.88682	**0.88913**	0.77219	0.86228	0.85953	0.84359
	E_{NS}	0.43588	-0.35402	0.17984	0.60324	0.61802	**0.62292**	0.07962	0.50024	0.46083	0.44022
	E_{LM}	0.24845	-0.22681	0.07450	0.37525	0.38761	**0.39055**	0.01954	0.30985	0.26030	0.27498
10	**r**	0.47111	0.49008	0.51187	0.51463	0.50869	**0.52187**	0.51359	0.50773	0.49595	0.50524
	RMSE	0.05978	0.05901	0.05738	0.09986	0.06153	**0.05358**	0.05770	0.06089	0.08167	0.08789
	MAE	0.04911	0.04845	0.04709	0.08674	0.04937	**0.04349**	0.04647	0.05086	0.06449	0.07098
	$RMSE_{\overline{S}}$	8.14461	8.03637	7.81436	13.59531	8.37422	**7.29284**	7.85301	8.28603	11.11275	11.96086
	WI	0.66298	0.67109	0.67083	0.50637	0.65743	**0.71638**	0.70313	0.68995	0.60608	0.57339
	E_{NS}	-0.03291	-0.00465	0.05009	-1.87263	-0.08964	**0.17361**	0.04638	-0.06169	-0.90055	-1.19066
	E_{LM}	-0.01247	0.00139	0.02940	-0.78766	-0.01768	**0.10358**	0.04589	-0.04414	-0.31882	-0.44672
15	**r**	0.19567	0.25300	0.26965	0.27992	0.30213	0.29335	0.34206	0.28623	0.31953	**0.35329**
	RMSE	**0.06607**	0.11262	0.06897	0.09933	0.10233	0.07502	0.07310	0.10087	0.10711	0.12511
	MAE	0.05492	0.09670	**0.05380**	0.07945	0.08043	0.06030	0.05626	0.07968	0.08738	0.10837
	$RMSE_{\overline{S}}$	**8.99808**	15.33346	9.39053	13.51922	13.92652	10.20989	9.94710	13.72624	14.57682	17.03069
	WI	0.49344	0.43747	0.50884	0.46714	0.49017	**0.56760**	0.56073	0.48248	0.47388	0.44655
	E_{NS}	**-0.25949**	-2.65411	-0.37051	-1.83985	-1.99909	-0.61193	-0.52275	-1.89961	-2.25369	-3.42198
	E_{LM}	-0.13197	-0.99290	**-0.10873**	-0.63772	-0.65123	-0.23803	-0.15054	-0.62949	-0.78091	-1.20430
20	**r**	0.04384	0.06017	0.15983	0.16891	0.21921	0.18399	0.22489	0.26328	**0.37611**	0.29366
	RMSE	**0.06787**	0.11353	0.09526	0.10391	0.10693	0.10657	0.10418	0.11046	0.11167	0.12961
	MAE	**0.05661**	0.09396	0.07367	0.08061	0.08566	0.08357	0.08364	0.08988	0.09390	0.11015
	$RMSE_{\overline{S}}$	**9.24035**	15.45096	12.96478	14.14051	14.55108	14.50144	14.17820	15.03312	15.20115	17.64498
	WI	0.37620	0.40477	0.44764	0.44394	0.45773	0.45887	0.47121	0.45914	**0.48346**	0.42248
	E_{NS}	**-0.32702**	-2.70940	-1.61170	-2.09196	-2.25856	-2.23637	-2.07817	-2.46058	-2.52293	-3.72414
	E_{LM}	**0.16664**	-0.93692	-0.51849	-0.65495	-0.75189	-0.70908	-0.70460	-0.83183	-0.91005	-1.23454

TABLE VIII

INDIA EXCHANGE RATE TIME SERIES LONG HORIZON PREDICTION USING SELECTED MODEL.

Step	Performance metrics	k: gap among observations									
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
5	**r**	**0.99629**	0.99553	0.99478	0.99401	0.99339	0.99284	0.99223	0.99173	0.99130	0.99087
	RMSE	0.02257	0.03938	0.04849	0.01765	0.03783	0.04454	**0.01908**	0.01959	0.04724	0.03799
	MAE	0.01848	0.03477	0.04360	**0.01375**	0.03278	0.03956	0.01491	0.01535	0.04175	0.03216
	$RMSE_{\overline{S}}$	2.82155	4.92315	6.06200	**2.20677**	4.72849	5.56818	2.38451	2.44873	5.90462	4.74847
	WI	0.99206	0.97549	-4.29039	**0.99515**	0.97739	0.96867	0.99433	0.99401	0.96459	0.97701
	E_{NS}	0.97086	0.91128	0.86546	**0.98217**	0.91813	0.88647	0.97918	0.97803	0.87228	0.91738
	E_{LM}	0.84743	0.71298	0.64008	**0.88650**	0.72937	0.67340	0.87689	0.87322	0.65521	0.73446
10	**r**	**0.99312**	0.99229	0.99157	0.99091	0.99040	0.98504	0.98929	0.98890	0.98849	0.98815
	RMSE	0.04702	**0.02048**	0.02252	0.02761	0.07077	0.05839	0.02653	0.03750	0.03583	0.04297
	MAE	0.04168	**0.01567**	0.01738	0.02262	0.06582	0.05279	0.02127	0.03123	0.02961	0.03643
	$RMSE_{\overline{S}}$	5.87893	**2.56042**	2.81474	3.45100	8.84575	7.29795	3.31542	4.68687	4.47836	5.36927
	WI	0.96469	**0.99350**	0.99211	0.98830	0.92206	0.94598	0.98879	0.97742	0.97940	0.97025
	E_{NS}	0.87348	**0.97600**	0.97099	0.95639	0.71344	0.80495	0.95973	0.91951	0.92652	0.89434
	E_{LM}	0.65593	**0.87063**	0.85653	0.81324	0.45651	0.56413	0.82434	0.74214	0.75550	0.69916
15	**r**	**0.98992**	0.98921	0.98857	0.98801	0.98744	0.98695	0.98654	0.98613	0.98575	0.98526
	RMSE	**0.02273**	0.03306	0.04842	0.06655	0.05514	0.04007	0.05874	0.05398	0.06088	0.02660
	MAE	**0.01815**	0.02731	0.04242	0.06071	0.04906	0.03345	0.05243	0.04737	0.05429	0.02084
	$RMSE_{\overline{S}}$	**2.84168**	4.13250	6.05187	8.31901	6.89201	5.00781	7.34092	6.74573	7.60768	3.32381
	WI	**0.99178**	0.98257	0.96267	0.93002	0.95172	0.97415	0.94517	0.95336	0.94093	0.98882
	E_{NS}	**0.97044**	0.93747	0.86587	0.74655	0.82600	0.90813	0.80255	0.83323	0.78788	0.95950
	E_{LM}	**0.85016**	0.77455	0.64976	0.49877	0.59490	0.72381	0.56703	0.60875	0.55165	0.82785
20	**r**	**0.98695**	0.98630	0.98571	0.98511	0.98465	0.98426	0.98384	0.98342	0.98305	0.98278
	RMSE	0.06537	0.02836	0.05122	**0.02624**	0.04590	0.05729	0.04488	0.03326	0.03367	0.05874
	MAE	0.05948	0.02286	0.04457	**0.02036**	0.03884	0.05036	0.03754	0.02691	0.02743	0.05135
	$RMSE_{\overline{S}}$	8.17167	3.54548	6.40236	**3.27969**	5.73678	7.16038	5.60866	4.15555	4.20765	7.33915
	WI	0.93272	0.98712	0.95794	**0.98908**	0.96597	0.94734	0.96714	0.98189	0.98210	0.94440
	E_{NS}	0.75549	0.95396	0.84984	**0.96060**	0.87942	0.81214	0.88471	0.93669	0.93509	0.80245
	E_{LM}	0.50891	0.81128	0.63195	**0.83191**	0.67923	0.58415	0.68998	0.77769	0.77346	0.57577

TABLE IX

JAPAN EXCHANGE RATE TIME SERIES LONG HORIZON PREDICTION USING SELECTED MODEL.

Step	Performance metrics	k: gap among observations									
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
5	**r**	0.99313	0.99213	0.99162	0.99216	0.99373	0.99410	**0.99418**	0.99412	0.99410	0.99416
	RMSE	0.00639	0.01421	0.00654	0.01219	0.00833	0.00671	0.00760	0.00700	0.00737	**0.00579**
	MAE	0.00510	0.01300	0.00511	0.01078	0.00694	0.00538	0.00616	0.00561	0.00586	**0.00434**
	$RMSE_{\overline{S}}$	7.93723	17.65812	8.12761	15.15507	10.34772	8.33817	9.45014	8.70285	9.15997	**7.20306**
	WI	0.99556	0.97909	0.99548	0.98516	0.99258	0.99510	0.99393	0.99520	0.99425	**0.99663**
	E_{NS}	0.98325	0.91713	0.98244	0.93899	0.97156	0.98154	0.97631	0.97991	0.97775	**0.98624**
	E_{LM}	0.87863	0.69081	0.87835	0.74356	0.83486	0.87195	0.85349	0.86656	0.86065	**0.89676**
10	**r**	0.98768	0.98664	0.98633	0.98737	0.98864	**0.98876**	0.98876	0.98856	0.98856	0.98860
	RMSE	0.00935	0.01097	0.01094	0.01081	**0.00787**	0.00812	0.00853	0.01077	0.00812	0.01014
	MAE	0.00770	0.00917	0.00844	0.00855	0.00629	**0.00617**	0.00664	0.00878	0.00648	0.00839
	$RMSE_{\overline{S}}$	11.62149	13.63217	13.59431	13.43693	**9.78179**	10.09482	10.60613	13.38613	10.09974	12.60699
	WI	0.99063	0.98771	0.98697	0.98761	**0.99330**	0.99291	0.99279	0.98811	0.99306	0.98955
	E_{NS}	0.96410	0.95064	0.95091	0.95207	**0.97460**	0.97296	0.97017	0.95249	0.97297	0.95788
	E_{LM}	0.81675	0.78194	0.79928	0.79675	0.85043	**0.85320**	0.84213	0.79113	0.84603	0.80065
15	**r**	0.98210	0.98121	0.98096	0.98184	**0.98301**	0.98297	0.98293	0.98286	0.98261	0.98266
	RMSE	0.01053	0.01351	0.01351	0.01133	**0.00918**	0.00985	0.00924	0.01016	0.00960	0.01441
	MAE	0.00853	0.01144	0.01131	0.00863	**0.00703**	0.00777	0.00718	0.00768	0.00761	0.01232
	$RMSE_{\overline{S}}$	13.09038	16.79567	16.79598	14.08485	**11.41298**	12.24180	11.48639	12.63146	11.93538	17.92838
	WI	0.98823	0.98080	0.97955	0.98640	**0.99134**	0.98903	0.99095	0.98927	0.98976	0.97865
	E_{NS}	0.95448	0.92511	0.92511	0.94737	**0.96544**	0.96026	0.96504	0.95772	0.96227	0.91487
	E_{LM}	0.79716	0.72796	0.73108	0.79483	**0.83294**	0.81532	0.82924	0.81736	0.81916	0.70720
20	**r**	0.97672	0.97583	0.97535	0.97638	**0.97757**	0.97749	0.97749	0.97718	0.97746	0.97683
	RMSE	0.01288	0.01088	0.01228	0.01118	**0.01058**	0.01186	0.01851	0.01120	0.01134	0.01607
	MAE	0.01054	0.00827	0.00974	0.00858	0.00803	0.00965	0.01620	**0.00853**	0.00866	0.01346
	$RMSE_{\overline{S}}$	16.00376	13.53112	15.26729	13.90174	**13.15206**	14.75447	23.01824	13.93509	14.11035	19.99378
	WI	0.98227	0.98714	0.98474	0.98598	**0.98783**	0.98398	0.96541	0.98610	0.98715	0.97014
	E_{NS}	0.93201	0.95142	0.93816	0.94876	**0.95413**	0.94231	0.85967	0.94857	0.94730	0.89419
	E_{LM}	0.74938	0.80325	0.76840	0.79597	**0.80916**	0.77063	0.61502	0.79723	0.79419	0.67997