Available online at https://www.gyanvihar.org/researchjournals/ctm_journals.php
SGVU International Journal of Convergence of Technology and Management
E-ISSN: 2455-7528
Vol.9 Issue 1 Page No 01-08

Deep Neural Networks for Financial Time Series Forecasting

[1]Jyoti Verma, [2]Sohit Agarwal
Department of Computer Science and Engineering
Suresh Gyan Vihar University
Jaipur, Rajasthan, India
Email: Jyoti.18223@mygyanvihar.com, Sohit.agarwal@mygyanvihar.com

Abstract—Deep Learning approaches have been used in forecasting financial time series data due to its capacity of achieving Higher accuracy. Multi-Layer Perceptron (MLP) and Recurrent neural networks (RNNs) are the most widely used neural network architectures for these model the complex nature of stock market datasets better than traditional models. In this paper, we present an analysis of the MLP as well as three variants of RNN namely, Long Short term Memory (LSTM), Bidirectional Long Short Term Memory and Gated Recurrent Units (GRU). The fundamentals of the architectures are explained along with experiments performed on Yahoo stock price datasets. The impact of deeper networks and the RNN variants on forecast performance is performed.

Index Terms—Recurrent Neural Networks, Long Short Term Memory, Gated Recurrent Units, Time Series Forecasting

I. INTRODUCTION

Time series forecasting is the prediction of the future values utilizing the past values. Time series data can be frequently observed in signal processing, econometrics such as stock prices, currency exchange rates as well as meteorology records of wind speeds, temperatures and rainfall. These data are prevalently used which is why time series forecasting is useful in many domains of day to day life. Traditional statistical methods as well as neural network models are used for the task of forecasting. Traditional models include Naive method, Holt's Winter, Exponential smoothing, ARIMA, ARIMAX, GAS [1] while artificial neural networks (ANN) consists of Multilayer Perceptron and Recurrent Neural Networks. Recurrent Neural Networks were developed mainly to handle sequential data problems [2]. Here, we perform stock price forecasting using multilayer perceptron along with three prevalent architectures of recurrent neural networks and analyze their performance. The main issue here is to perform analysis and forecasting of financial time series data and develop the qualitative forecasting models.

Correspondence to: Jyoti Verma, Department of Computer Science & Engineering, Suresh Gyan Vihar University, Jaipur
Corresponding author. E-mail addresses: Jyoti.18223@mygyanvihar.com

RNNs [9] are a special class of neural networks characterized by internal self connections in any nonlinear dynamical system. Stock market data consists of noise and volatile features which adds to its complexity. Now, prominent architectures of RNN include Deep RNNs with Multilayer Perceptron, Bidirectional RNN, Recurrent Convolutional Neural Networks, Multi-Dimensional Recurrent Neural Networks, Long-Short Term Memory, Gated Recurrent Unit, Memory Networks, Structurally Constrained Recurrent Neural Network, Unitary Recurrent Neural Networks, Gated Orthogonal Recurrent Unit and Hierarchical Subsampling Recurrent Neural Networks [3]. However, vanilla RNN is known to be having the underlying issue of vanishing as well as exploding gradients in order to tackle which various clipping strategies as well as other variants of RNN are proposed [4]. The LSTM variant of RNN have been analyzed for eight of its variants concluding that forget gate and the output activation function are the most critical component. Also, the learning rate is found to be the most crucial hyper parameters [5]. Now, RNN and its variants have been widely used for time series forecasting tasks in a wide range of domains. Long Short Term Memory has been used as a novel forecasting technique for solar energy forecasting proving LSTM as being robust and performing better than GBR and FFNN [6]. Petroleum time series data which are characterized by high dimensionality, non-stationary being highly non-linear in nature have also been used to test the performance of LSTM [7]. Furthermore, a deep architecture of RNN has been used to extract deep invariant daily features of financial time series outperforming other models in predictive accuracy and profitability performance [8]. The literature of ANN consisting of MLP and RNN indicate its capacity of handling the complex feature of financial time series data.

This paper aims to analyze the MLP, LSTM, Bidirectional LSTM and GRU architectures for the task of univariate time series forecasting. The paper is organized as follows. Section II discusses the architecture and mathematical formulation of the RNN models analyzed. Section III states the experimental details of the methodology adopted and the results while Section IV concludes the paper with future directions.

II. ARTIFICIAL NEURAL NETWORK ARCHITECTURES

This section summarizes the basics of the MLP and RNN architectures which are analyzed in this paper. RNN is the most basic of all the three and GRU and LSTM are the variants which were introduced at a later time.

A. Multi-Layer Perceptron Networks

Multilayer perceptron networks [10] are a class of artificial neural networks which consists of many layers of perceptrons. It consists of an input, hidden and an output layer. It is the most basic neural network architecture.

The input x_t along with its weight W^{xz} which is set for all connections is processed with a bias weight b which gives us the hidden layer values. Activation function φ is then applied to get the hidden layer output. This output is further processed in the output layer which gives us the final output. It uses the backpropagation algorithm for learning the appropriate weights. The number of hidden layers as well as hidden neurons is not fixed. Activation functions are not fixed as well.

$$z_t = W^{xz}x_t + b$$
$$a_t = \phi(z_t)$$
$$y_t = \sigma(W^{ay}a_t + b)$$
$$y_t = output \quad (1)$$

Correspondence to: Jyoti Verma, Department of Computer Science & Engineering, Suresh Gyan Vihar University, Jaipur
Corresponding author. E-mail addresses: Jyoti.18223@mygyanvihar.com

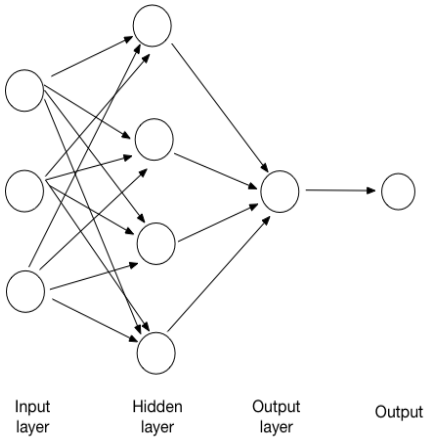

Fig. 1. MLP

B. Long Short Term Memory Networks

LSTM [10], as in Figure 2, introduces additional computation components to the RNN, the input gate, the forget gate and the output gate. The recurrence equation for the hidden vector is changed for LSTM with the use of long-term memory. The operations of the LSTM are designed to have fine-grained control over the data written into this long term memory. The equations for the forward pass are stated below:

$$
\begin{aligned}
a_t &= \tanh(W_c x_t + R_c h_{t-1}) \\
i_t &= \sigma(W_i x_t + R_i h_{t-1}) \\
f_t &= \sigma(W_f x_t + R_f h_{t-1}) \\
o_t &= \sigma(W_o x_t + R_o h_{t-1}) \\
c_t &= i_t \odot a_t + f_t \odot c_{t-1} \\
h_t &= o_t \odot \tanh(c_t) \qquad (2)
\end{aligned}
$$

The current input and the previous state are worked upon by a_t after which the input gate i_t decides upon which parts of a_t are required to be added to the long term state c_t. The forget gate f_t makes a decision as to which parts of c_{t-1} are to be erased and erases unnecessary parts, the output gate o_t decides on the parts of c_t to be read and shown as output. There exists a short term state h_t between the cells and a long term state c_t in which the memories are dropped and added by the respective gates.

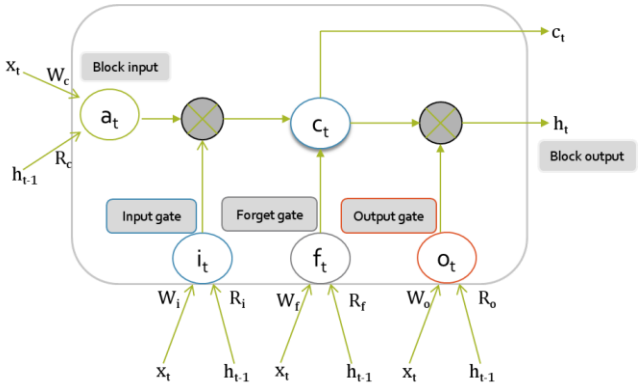

Fig. 2. LSTM

C. Gated Recurrent Units

The Gated Recurrent Unit [11] can be viewed as a simplification of the LSTM, which does not use explicit cell states. The main simplifications are that both state vectors are merged into a single vector. Figure 3 represents a GRU cell and (3) the equations followed during forward pass.

$$
\begin{aligned}
z_t &= \sigma(W_{xz}^T . x_{(t)} + W_{hz}^T . h_{(t-1)}) \\
r_t &= \sigma(W_{xr}^T . x_{(t)} + W_{hr}^T . h_{(t-1)}) \\
g_t &= tanh(W_{xg}^T . x_{(t)} + W_{hg}^T . (r_{(t)} \otimes h_{(t-1)})) \\
h_t &= (1 - z_t) \otimes tanh(W_{xg}^T . h_{(t-1)} + z_t \otimes g_t) \qquad (3)
\end{aligned}
$$

A single gate controller controls both the forget gate and the input gate. If the gate controller outputs a 1, the input gate is open and the forget gate is closed. If it outputs a 0, the opposite happens. In other words, whenever a memory must be stored, the location where it will be stored is erased first. This is actually a frequent variant to the LSTM cell in and of itself. There is no output gate; the full state vector is output at every time step. However, there is a new gate controller that controls which part of the previous state will be shown to the main layer.

Correspondence to: Jyoti Verma, Department of Computer Science & Engineering, Suresh Gyan Vihar University, Jaipur
Corresponding author. E-mail addresses: Jyoti.18223@mygyanvihar.com

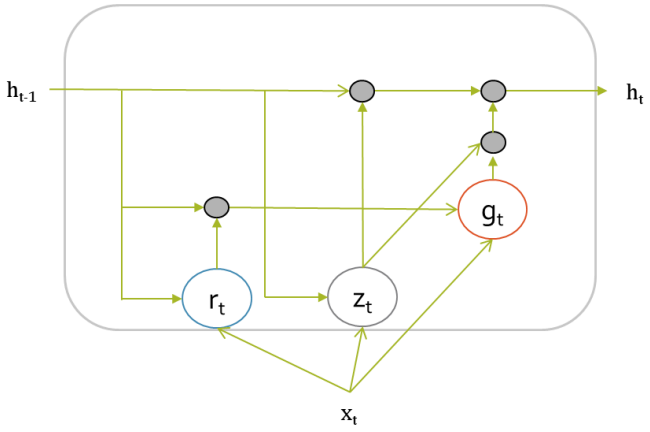

Fig.3.GRU

D. Bidirectional LSTM Network

Bidirectional LSTM [13] model can be treated as two independent LSTM models, one with input sequence in normal order and other having input sequence in reverse time step order. The output generated at each time step is the combined result of both the models. The specialty of this model is that it maintains both forward and backward information about the sequence at every time step which helps this model to predict output more accurately.

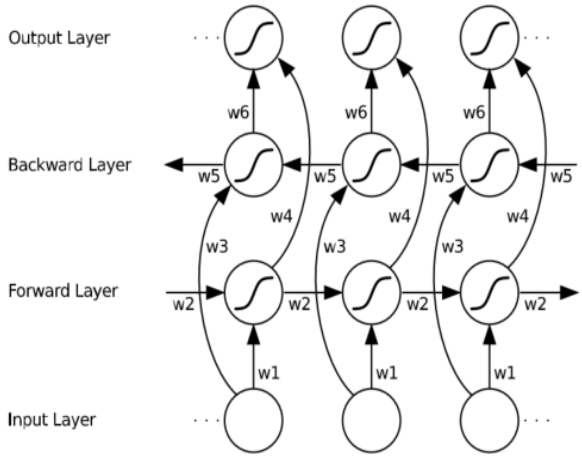

Fig.4.Bidirectional LSTM

III. EXPERIMENT AND RESULTS

The methodology followed for the proposed work and the results obtained is being discussed in this section. Figure 5 gives a more descriptive interpretation of the scheme followed. The time series data is first preprocessed for making it trainable using the neural network model. The models are further optimized, regularized and properly tuned for attaining generalized results avoiding underfitting as well as overfitting. The performance evaluation of the models is done using evaluation metrics which finally decides the best forecast model for the problem at hand. The experiments were carried out using the keras library with tensorflow backend and python programming language.

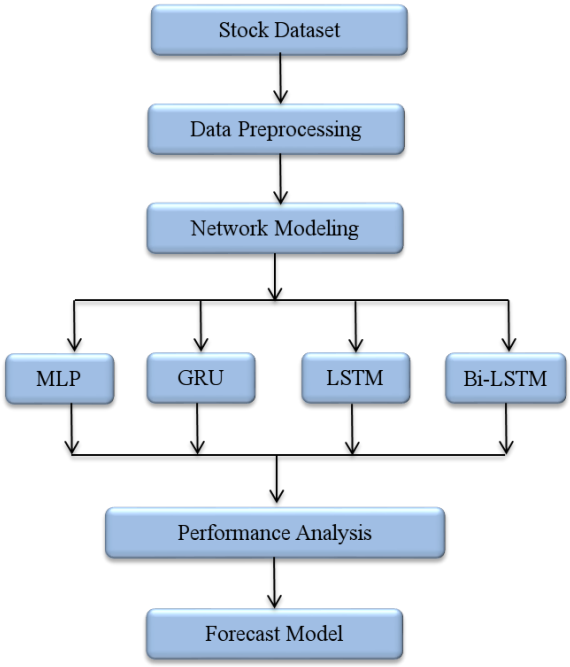

Fig.5.Methodology

A. Dataset Description

The analysis is carried out on financial time series of stock prices [14]. It consists of the Daily Yahoo stock price dataset ranging from 09-08-1996 to 22-02-2019. The closing price is aimed to be predicted accurately. From the figure of the dataset, the closing stock prices seem to be fluctuating in unequal intervals indicating more complexity. The aim is to analyze the performance of the models in such real world scenario of financial data analysis and prediction. Furthermore, preprocessing is required for making the data usable. Since, neural network architecture is being used, converting the series to

Correspondence to: Jyoti Verma, Department of Computer Science & Engineering, Suresh Gyan Vihar University, Jaipur
Corresponding author. E-mail addresses: Jyoti.18223@mygyanvihar.com

a stationary set is not a mandatory step. The values' range go high as well as low due to which data normalization is performed in order to standardize the inputs and approach faster towards the global minima. It also ensures that larger inputs do not overwhelm or become dominant. Min-max normalization, as described in (4), is performed in a range of 0 to 1. It does not change the data pattern or characteristics but only readjusts the scale of the data. Data pre-processing step also includes the conversion to a supervised data format since time series datasets are being dealt with in here. One-step ahead forecast is to be done where the next time step (t+1) is predicted. Originally the univariate time series dataset consists of only one feature column. We divide this time series into input (x) and output (y) using lag time method.

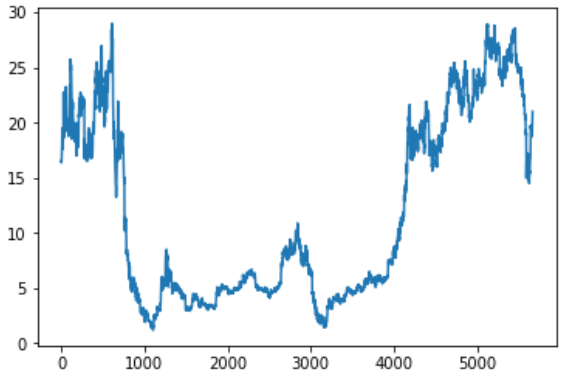

Fig.6.Stock Price Dataset

B. Network Modelling

Neural network architecture needs to be modelled for optimal perfomance which requires setting and tuning different configurations of the network. The supervised data is split into a train-validation-test split for proper estimation of error. Optimization is the minimization of loss function with respect to the parameters of our model. Here, ADAM optimizer is used as stated in [15]. ADAM optimizer is robust and is used frequently used for training ANN architectures. Now, optimizers mainly aim to decrease the training error. But, sometimes this results in overfitting, i.e., the model fits well on the training data but unable to fit on the test data. Regularization tries to overcome overfitting by choosing an approximately high value for the parameters governing the capacity of the model and then controlling it by adding a regularization term to the error function. In our work, we have used Dropout regularization as and when required [2]. A dropout layer blocks a random set of cell units in one iteration. Blocked units do not receive and do not transmit information. Removing connections in the network reduces the number of free parameters to be estimated during training and the complexity of the network. Consequently, dropout helps to prevent over-fitting. Dropout ratio of 0.2 is used in our work in the hidden layer. Hyperparameters are the settings that are not adapted by the learning algorithm since that would result in model overfitting. Hidden layers of size 2 and 3 were experimented upon. The number of hidden nodes was set to form a narrow architecture. Number of epochs is tuned for the problem at hand.

C. Performance Metrics

Four performance evaluation metrics are used to assess forecast accuracy. These error metrics are frequently used for assessing model accuracy. After evaluating all the forecast models according to the above stated metrics, the best configured forecast model is decided upon. These are shown in Table I below:

TABLE I
PERFORMANCE EVALUATION METRICS

Metric	**Description**	
MAE	Mean Absolute Error	$\frac{1}{N}\sum_{n=1}^{N}(y_n^{actual} - y_n^{predicted})$
MSE	Mean Squared Error	$\frac{1}{N}\sum_{n=1}^{N}(y_n^{actual} - y_n^{predicted})^2$

Correspondence to: Jyoti Verma, Department of Computer Science & Engineering, Suresh Gyan Vihar University, Jaipur
Corresponding author. E-mail addresses: Jyoti.18223@mygyanvihar.com

RMSE	Root Mean Squared Error	$\sqrt{\frac{1}{N}\sum_{n=1}^{N}(y_n^{actual} - y_n^{predicted})^2}$
R^2 score	R^2 score	$1 - \frac{\sum_{n=1}^{N}(y_n^{actual} - y_n^{predicted})^2}{\sum_{n=1}^{N}(y_n^{actual} - \bar{y}_n^{predicted})^2}$

Here, N is the total number of observations for which the error is evaluated. y_n^{actual} is the actual observation in nth position and $y_n^{predicted}$ the predicted value.

D. Results

As stated earlier, the experiments are carried out on stock price dataset. Three architectures of Recurrent Neural Networks, being LSTM, Bidirectional LSTM and GRU along with MLP are used for forecasting one step-ahead result. The results shown below in Table II show the performance evaluation parameter for optimally configured architectures.

TABLE II
PERFORMANCE EVALUATION

Architecture	MAE	MSE	RMSE	R^2 Score
MLP	0.0342	0.0015	0.0386	0.8721
LSTM	0.0312	0.0014	0.0376	0.8787
Bi-LSTM	0.0306	0.0015	0.0388	0.8710
GRU	0.0206	0.0007	0.0261	0.9414

The results are stated for well-tuned models achieved after experiments for generalizing the model for a better test performance on unknown samples. From the table of results, many observations can be made about the performance of the forecast model. In the case of network performance, it can be seen that GRU performs the best out of all four models. Other models also performs with good accuracy, however, GRU seems to outperform. It can be observed that the MLP also gives good results on proper tuning. It implies that deeper and complex networks are always not the solution to non-linear problems since these networks lead to more complex models and costly time consumption. Therefore, proper tuning of model parameters as well as hyperparameters is a must before adopting a deep architecture.

Figures 7 and 8 represent the training and test performance of the best performing model. The training performance is almost the same for all the models but the test performance creates all the difference. Better test performance implies better generalization, i.e., the model would perform well when new and unseen data is presented from the same problem set. The GRU is simpler and enjoys the advantage of greater ease of implementation and efficiency. It might generalize slightly better with less data because of a smaller parameter footprint. The bidirectional LSTM performs well too, however we tend to opt for the model with lesser complexity in order to generalize better for future instances. Therefore, the different architectures perform differently according to the problem at hand. The increase in data complexity for financial data require more complex models and the ANN models handle the issue efficiently, assuring high accuracy as can be observed from the performance evaluation metrics.

Correspondence to: Jyoti Verma, Department of Computer Science & Engineering, Suresh Gyan Vihar University, Jaipur
Corresponding author. E-mail addresses: Jyoti.18223@mygyanvihar.com

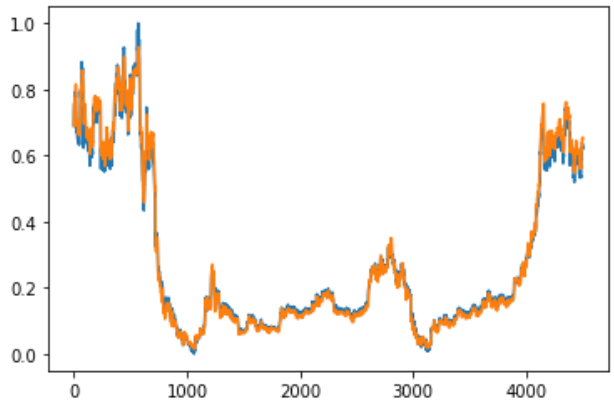

Fig.7.Training Performance

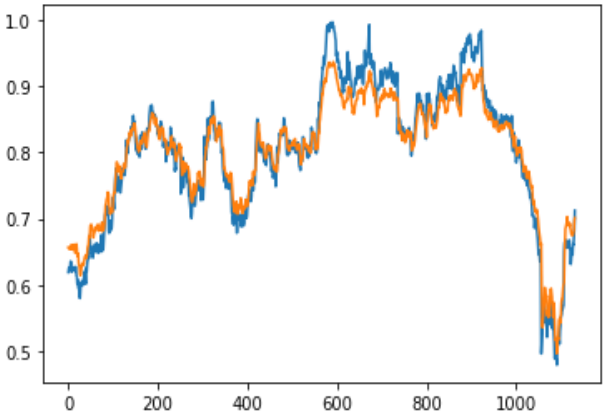

Fig.8.Testing Performance

IV. CONCLUSION

In the paper, an analysis is performed on the MLP architecture and the three variants of RNN, namely LSTM, Bidirectional LSTM and GRU on financial data. It was observed that GRU performs the best with model tuning being of utmost importance before increasing the complexity. In future, more architectures are sought to be analyzed for the multivariate case of forecasting data with more emphasis on the data characteristics.

REFRENCES

[1] P. J. Brockwell and R. A. Davis, Introduction to Time Series and Forecasting - Second Edition. 2002. n.d.

[2] I. Goodfellow, Y. Bengio, and A. Courville, Deep learning. 2017. n.d.

[3] H. Salehinejad, J. Baarbe, S. Sankar, J. Barfett, E. Colak, S. Valaee, "Recent advances in recurrent neural networks," 2017, [Online] Available: https://arxiv.org/abs/1801.01078 n.d.

[4] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in 30th International Conference on Machine Learning, ICML 2013, 2013, no. PART 3, pp. 2347–2355. n.d.

[5] K. Greff, R. K. Srivastava, J. Koutnik, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A Search Space Odyssey," IEEE Trans. Neural Networks Learn. Syst., vol. 28, no. 10, pp. 2222–2232, 2017. n.d.

[6] S. Srivastava and S. Lessmann, "A comparative study of LSTM neural networks in forecasting day-ahead global horizontal irradiance with satellite data," Sol. Energy, vol. 162, pp. 232–247, 2018. n.d.

[7] A. Sagheer and M. Kotb, "Time series forecasting of petroleum production using deep LSTM recurrent networks," Neurocomputing, vol. 323, pp. 203–213, 2019. n.d.

[8] W. Bao, J. Yue, and Y. Rao, "A deep learning framework for financial time series using stacked autoencoders and long-short term memory," PLoS One, vol. 12, no. 7, 2017. n.d.

[9] K. Unnikrishnan and K. P. Venugopal, "Alopex: A correlation-based learning algorithm for feedforward and recurrent neural networks," Neural Computation, vol. 6, no. 3, pp. 469–490, 1994. n.d.

[10] T. Hastie, R. Tibshirani, J. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer, New York, NY, 2009. n.d.

Correspondence to: Jyoti Verma, Department of Computer Science & Engineering, Suresh Gyan Vihar University, Jaipur
Corresponding author. E-mail addresses: Jyoti.18223@mygyanvihar.com

[11] F. A. Gers, J. Schmidhuber, F. Cummins, “Learning to forget: Continual prediction with LSTM”, Neural Comput., vol. 12, no. 6, pp. 2451-2471, 2000. n.d.

[12] K. Cho, D. Bahdanau, F. Bougare, H. Schwenk and Y. Bengio, “Learning Phrase Representations using RNN Encoder–Decoderfor Statistical Machine Translation,” arXiv, 2014. n.d.

[13] A. Graves, F. Santiago and S. Jürgen, “Bidirectional LSTM networks for improved phoneme classification and recognition,” in Artificial Neural Networks: Formal Models and Their Applications–ICANN 2005, Springer Berlin Heidelberg, 2005, pp. 799-804. n.d.

[14] [Online].Available:https://finance.yahoo.com/quote/CSV/history/ n.d.

[15] D. P. Kingma and J. L. Ba, “Adam: A method for stochastic optimization,” 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc., 2015. n.d.

Correspondence to: Jyoti Verma, Department of Computer Science & Engineering, Suresh Gyan Vihar University, Jaipur
Corresponding author. E-mail addresses: Jyoti.18223@mygyanvihar.com