# Comparative Study on Time Series Forecasting using Deep learning Models

**Preprint** · March 2020

DOI: 10.13140/RG.2.2.26561.71521/1



Abhishek Tiwari
Symbiosis International University

# UNIVARIATE AND MULTIVARIATE TIME SERIES FORECASTING USING DEEP LEARNING MODELS

Abhishek Tiwari [(1)] | Vipul Vaibhaw [(2)] | Adori Medhi [(3)]

Affiliation: Symbiosis Statistical Institute

## ABSTRACT
Time Series Forecasting methods help in predicting new variables by using the historical observations plotted against time. There are various traditional models for univariate and multivariate forecasting. They have a limitation of low number of predictions, inaccurate predictions for a dynamic dataset and inability to predict for discontinuous time. Deep Learning models overcome most of these limitations. In this paper, two datasets – one extremely dynamic and one comparatively stable are studied and univariate and multivariate forecasting are done. The Deep Learning models used are Long Short Term Memory (LSTM), Gated Recurrent Unit (GRU), Convolution Neural Networks (CNN) and Multilayer Perceptron (MLP). LSTM and GRU give excellent predictive results for univariate forecasting. For multivariate prediction, MLP is efficient for a dynamic dataset and LSTM is efficient in a comparatively stable dataset. All the Deep Learning models are compared by using Root Mean Square Error (RMSE). Deep Learning models yield a much accurate forecasting of most time series datasets.

## INTRODUCTION
Time series is sequential data points which is ordered in sequence and collected at regular time interval. [1] Time series analysis is applied on any variable which is changes over time. Time series forecasting has been used widely in various fields as a part of predictive analysis. Over the years, various techniques have been developed for forecasting data based on time. The traditional techniques like ARIMA which are widely used, have rigid assumptions which sometimes make it difficult for forecasting for a wide variety of datasets. Deep Learning models overcome a lot of challenges faced by the traditional time series methods.

Deep Learning is a subset of Machine Learning and Deep Learning algorithms are based on the workflow and functions of artificial neural network. Deep neural networks can automatically learn complex mappings between input and output function and support multiple inputs and outputs. [2]

In this paper, two time series datasets are used for both univariate and multivariate time series forecasting.

### Deep Learning Models

The Deep Learning models used for analysis are Long Short Term Memory (LSTM), Gated Recurrent Unit (GRU), Multilayer Perceptron (MLP) and Convolution Neural Network (CNN). A brief insight to the models is as follows:

Long Short Term Memory (LSTM)

To solve the problem of vanishing gradients in Recurrent Neural Networks (RNN), LSTM is used. LSTM are capable to learn order dependence in sequence prediction. LSTM remembers all the relevant past knowledge which the network has sequentially passed and forgets all the irrelevant data. This is done with the help of various activation function layers called gates. The LSTM has an Internal Cell state, which is a vector that keeps the information that was chosen to be retained by the previous LSTM unit.[3] A LSTM Network consists of four different gates which are: input gate, output gate, forget gate and input cell. These gates identify which data are required to be kept or omitted. After going through all the above gates, relevant information is passed down the sequence to make predictions.[4] The difference in the work flow of a LSTM Network and RNN is that the internal cell state is passed forward along with the hidden state.

The equations of an LSTM unit are:

$$f_t = \sigma_g(W_f x_t + U_f h_{t-1} + b_f)$$
$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i)$$
$$o_t = \sigma_g(W_o x_t + U_o h_{t-1} + b_o)$$
$$c_t = f_t \circ c_{t-1} + i_t \circ \sigma_c(W_c x_t + U_c h_{t-1} + b_c)$$
$$h_t = o_t \circ \sigma_h(C_t)$$

where the initial values are $c_0 = 0$ and $h_0 = 0$ (element-wise product). Here,

$x_t \epsilon\ \mathbb{R}^d$: input vector to the LSTM unit
$f_t \epsilon\ \mathbb{R}^h$: forget gate's activation vector
$i_t \epsilon\ \mathbb{R}^h$: input/update gate's activation vector
$o_t \epsilon\ \mathbb{R}^h$: output gate's activation vector
$h_t \epsilon\ \mathbb{R}^d$: hidden state or output vector
$c_t \epsilon\ \mathbb{R}^h$: cell state vector
$W \epsilon\ \mathbb{R}^{hXd}, U \epsilon\ \mathbb{R}^{hXd}$ *and* $b \epsilon\ \mathbb{R}^h$: weight matrices and bias vector parameters
The activation function is $\sigma_g$(sigmoid) and $\sigma_h$ (tanh). The superscripts d and h are the number of input features and number of hidden units.[5]

Gated Recurrent Unit (GRU)

GRU also tries to solve the vanishing gradient problem of RNN. GRU has lesser parameters which are more computationally efficient and can generalize result with lesser data. It has two gates – update gate and reset gate which decides on the information to be passed to the output.6 The update gate works like the input and forget gate in LSTM which acts as a deciding factor for the information. The reset gate decides the amount of past information to forget.

Initially, for t=0, the output vector $h_0$=0

$$z_t = \sigma_g(W_z x_t + U_z h_{t-1} + b_z)$$
$$r_t = \sigma_g(W_r x_t + U_r h_{t-1} + b_r)$$
$$h_t = z_t \circ h_{t-1} + (1 - z_t) \circ \phi_h(W_h x_t + U_h(r_t \circ h_{t-1}) + b_h)$$

Variables
$x_t$ : input vector

$h_t$ : output vector
$z_t$ : update gate vector
$r_t$ : reset gate vector
$W$, $U$ and $b$: parameter matrices and vector
The activation function are $\sigma_g$(sigmoid) and $\sigma_h$(tanh)[6]

Multilayer Perceptron (MLP)

MLP is an artificial neural network (ANN) model with at a single input, a single output and more than one hidden layer. MLP with a single hidden layer is called vanilla neural network. Perceptron is a single neuro model precursor to a larger neural network. Except for the input node, each node in neuron has an activation function. MLP uses a technique called back-propagation which helps in training its weights.[7]

Convolution Neural Network

Convolution Neural Network are the deep learning models for image processing. Convolution is a mathematical operation on two functions expressing how shape of one is modified by other. CNN consists of a Fully connected neural network at the end. CNN's hidden layers have a series of convolution layers which convolve with a dot product.

For image recognition, 2-D convolution is applied and for time series data 1D convolution is used. Consider a time series dataset of length *n* and width *m*. The length is the number of timesteps, and the width is the number of variables in a multivariate time series dataset. The convolution filters or kernels have the same number of variables as in the dataset, but their length can vary. In this way the kernel moves in the direction from the beginning of a time series data to its end, by performing convolution. The element of the kernel gets multiplied by the corresponding elements of the time series which they cover at a given point. The resulting value now becomes an element of a newly filtered univariate time series, then the kernel moves forward along the time series to formulate the next value. In the next step, max-pooling is applied to each filtered vector. The largest value is taken from each vector. A new vector is formed from these values. Then, in the end, new vector values are used as an input in the fully connected feed-forward network.[8]

## AIM

To perform a comparative study on the results of univariate and multivariate time series datasets by using Deep learning models. It also aims to identify the optimal deep learning model which can properly capture the dynamic fluctuations of a time series dataset.

## METHODOLOGY

As discussed, the following deep learning models are used: Long Short Term Memory (LSTM), Gated Recurrent Unit (GRU), Multilayer Perceptron (MLP) and Convolution Neural Network (CNN). All these models are used for testing for univariate as well as multivariate forecasting.

Two datasets are used for time series modelling. Details about the dataset are as follows:

Google Stock Price- This dataset is obtained from Kaggle which consists of the variables: opening price, closing price, high price, low price, volume of stocks traded. The dataset contains 1258 daily values from 03-01-2012 to 31-12-2016.
Occupancy – This dataset describes the occupancy of a room and the attributes of each room. There are 20560 observations. The source of the data is credited are from Luis M. Candanedo, VÃ©ronique Feldheim. Energy and Buildings. Volume 112, 15 January 2016, Pages 28-39. The attributes in the dataset are: Date time, Temperature, Relative Humidity, Light, CO2, Humidity Ratio and Occupancy

Each model was tested on both datasets for univariate and multivariate forecasting.

## RESULTS AND DISCUSSION

In the Google Stock Price Dataset, the variable Open Prices was used for univariate forecasting whose values range from 279 to 816 with a mean of 533. In the Occupancy Dataset, the variable Temperature was used for univariate forecasting whose values range from 19 to 24 with a mean of 20.
For univariate forecasting, each model was run for 1000 epochs on both the dataset. The dataset was split into a ratio of 90:10 for training and testing respectively. Root Mean Square Error is used as an indicator for checking the accuracy of the predictions made by the trained model with the test observations in the dataset. RMSE gives the standard deviation of the prediction errors.

| DATASETS | | | |
|---|---|---|---|
| **Google Stock Price** | | **Occupancy** | |
| **Model** | **RMSE** | **Model** | **RMSE** |
| LSTM | 7.710851 | LSTM | 0.04225 |
| GRU | 9.663648 | GRU | 0.03995 |
| MLP | 13.12884 | MLP | 0.06252 |
| CNN | 9.579383 | CNN | 0.06300 |

Table 1: Root Mean Square Error (RMSE) of each model on the two datasets

The above table shows the RMSE of the models. From the above, LSTM works the best for a highly dynamic dataset like opening prices of a stock market and GRU works well for a seasonally repeating dataset like temperature. MLP gives the highest RMSE for stock prices and CNN gives the highest RMSE for occupancy.

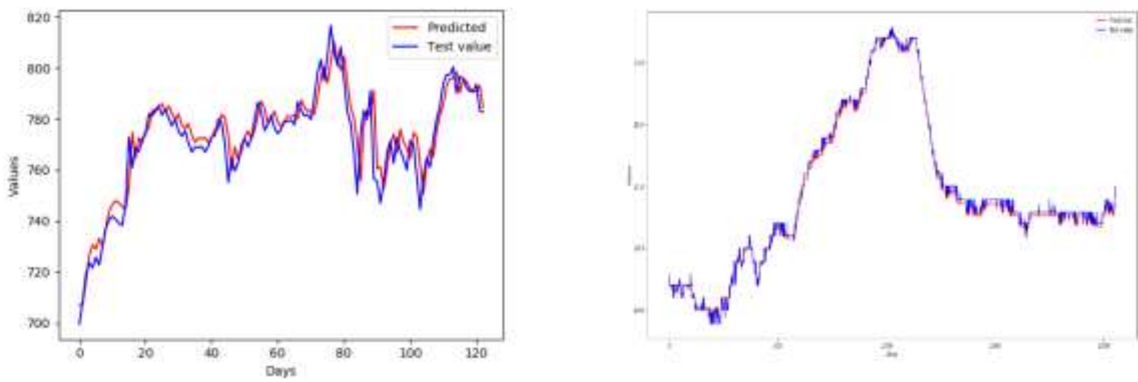


Fig 1: Predicted Vs Tested Values of LSTM and GRU for Stock Prices and Occupancy data respectively

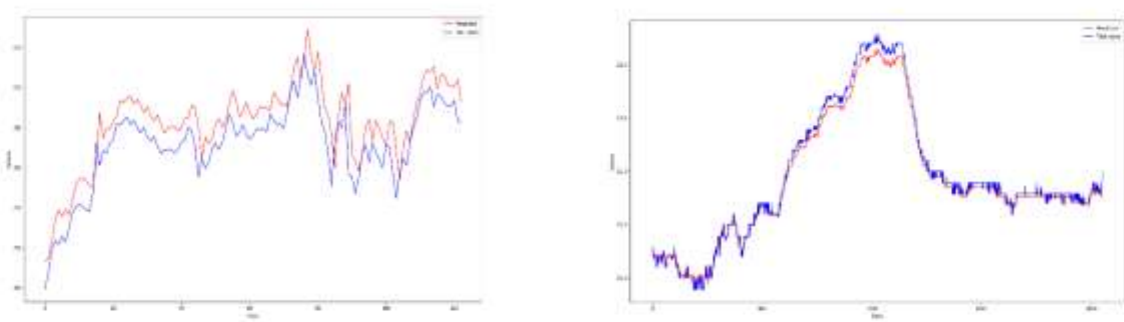

Fig 2: Predicted Vs Tested Values of MLP and CNN for Stock Prices and Occupancy data respectively

In the Google Stock Price Dataset, all the variables are used for multivariate forecasting which considers the dependencies amongst the variables along with its own historical values. Here, the values of all the variables are significantly different from each other with a very high difference in the value of the variable volume.  In the Occupancy dataset, all the variables except the occupancy variable are used for multivariate forecasting which considers the dependencies amongst the variables along with its own historical values. Here, the values of all the variables are significantly different from each other.

For multivariate forecasting, each model was run for 1000 epochs for Google Stock Price, a highly dynamic data. The dataset was split into a ratio of 90:10 for training and testing respectively. Root Mean Square Error is used as an indicator for checking the accuracy of the predictions made by the trained model with the test observations in the dataset. RMSE gives the standard deviation of the prediction errors.

| GOOGLE STOCK PRICES | | | | |
|---|---|---|---|---|
| Variables | LSTM | GRU | MLP | CNN |
| Open | 14.39879 | 26.05392 | 8.594959 | 15.88502 |
| High | 12.72302 | 25.24518 | 8.339229 | 15.45717 |
| Low | 16.89167 | 28.90806 | 11.59833 | 18.87729 |
| Close | 16.91848 | 28.59451 | 16.04845 | 11.25865 |
| Volume | 738112 | 625921.6 | 722253.9 | 798391.7 |

Table 2: Root Mean Square Error (RMSE) of each model for each variable without scaling

Due to the vast differences in values of the original variables, a high range of RMSE is obtained. Hence, all RMSE values are scaled in the values of 0 to 1.

| GOOGLE STOCK PRICES | | | | |
|---|---|---|---|---|
| Variables | LSTM | GRU | MLP | CNN |
| Open | 0.332427 | 1 | 0 | 0.417554 |
| High | 0.259304 | 1 | 0 | 0.421032 |
| Low | 0.305802 | 1 | 0 | 0.420512 |
| Close | 0.326481 | 1 | 0.276294 | 0 |
| Volume | 0.650492 | 0 | 0.558545 | 1 |
| AVERAGE | 0.374901 | 0.8 | 0.1669678 | 0.45181959 |

Table 3: Root Mean Square Error (RMSE) of each model for each variable with scaling

From the above table, it is observed that MLP gives the least RMSE and CNN has the highest RMSE for a dynamic dataset. Although MLP is comparatively a good model, its RMSE values in the original scale are still significantly high.

For multivariate forecasting, each model was run for 1000 epochs for Occupancy, a comparatively stable dataset. The dataset was split into a ratio of 90:10 for training and testing respectively. Root Mean Square Error is used as an indicator for checking the accuracy of the predictions made by the trained model with the test observations in the dataset. RMSE gives the standard deviation of the prediction errors.

| OCCUPANCY | | | | |
|---|---|---|---|---|
| Variables | LSTM | GRU | MLP | CNN |
| Temperature | 0.08605 | 0.121879 | 0.219675 | 0.096858 |
| Humidity | 0.432602 | 0.702202 | 0.683842 | 0.475885 |
| Light | 27.1558 | 31.81048 | 22.53739 | 82.99003 |
| CO2 | 159.0893 | 150.7947 | 143.2871 | 346.1368 |
| HumidityRatio | 0.000079 | 0.000116 | 0.00013 | 0.000072 |

Table 4: Root Mean Square Error (RMSE) of each model for each variable without scaling

Due to comparison purposes all RMSE values are scaled in the values of 0 to 1.

| OCCUPANCY | | | | |
|---|---|---|---|---|
| **Variables** | **LSTM** | **GRU** | **MLP** | **CNN** |
| Temperature | 0 | 0.268131 | 1 | 0.080883 |
| Humidity | 0 | 1 | 0.931899 | 0.160545 |
| Light | 0.076397 | 0.153394 | 0 | 1 |
| CO2 | 0.077901 | 0.037011 | 0 | 1 |
| HumidityRatio | 0.159091 | 1 | 1.318182 | 0 |
| AVERAGE | 0.062678 | 0.491707 | 0.650016 | 0.448286 |

Table 3: Root Mean Square Error (RMSE) of each model for each variable with scaling

From the above table, it is observed that LSTM gives the least RMSE and MLP has the highest RMSE for a comparatively stable dataset.

## CONCLUSION

Deep Learning models have a unique processing which allows them to be more efficient than other traditional models. The deep learning models used in this study focuses on training the model though various parameters and hence proves to be more efficient. Models like LSTM and GRU are efficient for time series univariate forecasting.

Deep Learning models yield a high accuracy at forecasting future values. Although this is the case, even deep learning models must be used at discretion for highly dynamic datasets. From the above, we have seen huge discrepancies in the RMSE of the stock price dataset.

For a relatively stable dataset or any dataset non extreme dynamic datasets, Deep Learning models work very efficiently. A very accurate prediction is given by models like LSTM and GRU. Deep Learning models work as an effective modeling technique for time series forecasting.

## ACKNOWLEDGEMENT

The authors acknowledge support from Dr. Sharvari Shukla, the Director of Symbiosis Statistical Institute. The authors also thank Mr Vipul Vaibhaw for mentoring for the paper.

## REFERENCES

1. Algorithmia Blog. 2020. *Introduction To Time Series Analysis*. [online] Available at: https://algorithmia.com/blog/introduction-to-time-series.
2. Mouti, S., Pelletier, C., Almalaq, A., Copur, O., Lachapelle, D., Bennatan, O. and Condran, C., 2020. *Deep Learning For Time Series Forecasting*. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/deep-learning-for-time-series-forecasting/
3. GeeksforGeeks. 2020. *Long Short Term Memory Networks Explanation - Geeksforgeeks*. [online] Available at: https://www.geeksforgeeks.org/long-short-term-memory-networks-explanation/
4. Medium. 2020. *Illustrated Guide To LSTM'S And GRU'S: A Step By Step Explanation*. [online] Available at: https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21
5. GeeksforGeeks. 2020. *Long Short Term Memory Networks Explanation - Geeksforgeeks*. [online] Available at: https://www.geeksforgeeks.org/long-short-term-memory-networks-explanation/

6. En.wikipedia.org. 2020. *Gated Recurrent Unit*. [online] Available at: <https://en.wikipedia.org/wiki/Gated_recurrent_unit>
7. Brownlee, J., 2020. *Crash Course On Multi-Layer Perceptron Neural Networks*. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/neural-networks-crash-course/
8. Medium. 2020. *How To Use Convolutional Neural Networks For Time Series Classification*. [online] Available at: https://towardsdatascience.com/how-to-use-convolutional-neural-networks-for-time-series-classification-56b1b0a07a57