RAJAN BAJAJ (B16072)

INDIAN INSTITUTE OF TECHNOLOGY MANDI

# Project Report

## Stock-Market Analysis using Distributed Deep Neural Network

*Course Instructor: Dr. Arti Kashyap*

December 1, 2019

***Abstract*—This paper is supported by the project on prediction of stock-market prices of Nifty-50. The technologies used here are based on Big Data application on Deep Neural Networks (DNNs). The deep learning model used here is Long Short Term Memory (LSTM), which is trained on the historical data of Nifty-50 of past 7 years. Also to compare the results of this deep learning model, the Statistical model is used, which statistically manipulate the given data to find the future values of the stock. Both models are distributed using Apache Spark. In LSTM the data as well whole model is distributed using Apache Spark framework 'Elephas'. Elephas is the extension of Keras. The Statistical model is also distributed on Apache Spark. Keeping Spark environment in both models make it easier to compare the results of two. This paper also discusses a previously proposed methodology and why it is not used in this application. It is observed that the cosine similarity of predicted values on validation set is approximately 0.99, which shows that the predicted values are approximately 99 percent similar to the actual values.**

## I. Introduction

Prediction of rise and fall of stock-prices had become a very essential part in investing. Other than primary income there is a huge potential to earn money from investment activities. But there are risks involved in investment as one can lose a large amount by bidding on wrong price and by making wrong decisions on when to buy a stock and when to sell. Also it is very difficult to predict stocks rise and fall due to their volatility, market trends of buying and selling, population behaviour of investment, closing price of stock, volume of stocks traded, etc. Researchers are continuously working on increasing the accuracy of models for prediction of stock prices trend. These trends can be measured for few minutes, days, weeks or for years. Based on the stock data available for seven years this paper shows the methodology used in predicting the daily closing price and plotting the graph based on deep learning model and statistical model and comparing their results based on cosine similarity and time efficiency metric. The common environment used for distribution of Deep Learning model as well as statistical model is Apache Spark for distributed computing. The distributed approach is performed on single machine but division is done on computation using single core and multiple cores. The time efficiency and cosine metric are than compared using the graphs between predicted set and validation set. The framework used for distribution of Deep Learning model is 'Elephas'. Spark is chosen over Hadoop because of its speed which is 100 times faster than Hadoop map-reduce and its scalability and enrichment of frameworks for distribution of even the Deep Learning models. To get the temporal relationship between historical data and future data Recurrent neural networks (RNN) and Long Short-Term Memory (LSTM) approach which are part of Deep Learning field are used.

## II. LSTM Neural Network

Using a recurrent neural network is helpful in a way that unlike multi-perceptron networks (MLP), RNN uses a feedback loop from forward layers to a previous layers which helps in getting the relationship between the current data and the output value. Based on this feedback the next output is adjusted in such a way that it is more accurately predicted. In a way these networks give a time relation of data. But there are two major problems with RNNs, these are:

- **Vanishing Gradient :-** In some cases while training a deep neural network, the gradient in back-propagation step is vanishingly small due to which the value of a weight is not changed significantly and hence sometimes tends a neural network to halt the process of training since there are no learning because there are no updates in the parameters.
- **Exploding Gradient :-** The value of factor of gradient for parameter updating becomes very bigger than 1 and hence the value becomes high and hence other values comparatively become negligible to this parameter value and hence network is unable to learn many important patterns in data.

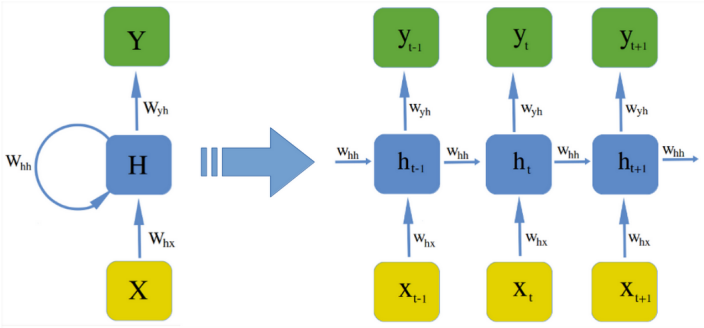
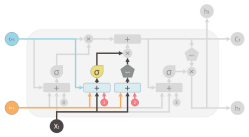


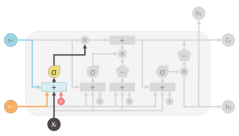

Figure 1: Recurrent Neural Network short and expanded form

$$\mathbf{h}_t = f_{\mathbf{W}}\left(\mathbf{h}_{t-1}, \mathbf{x}_t\right) \tag{1}$$
$$\mathbf{h}_t = f\left(\mathbf{W}_{hx} x_t + \mathbf{W}_{hh} h_{t-1} + \mathbf{b}_h\right) \tag{2a}$$
$$\mathbf{h}_t = \tanh\left(\mathbf{W}_{hx} x_t + \mathbf{W}_{hh} h_{t-1} + \mathbf{b}_h\right) \tag{2b}$$
$$\hat{\mathbf{y}}_t = \mathbf{softmax}\left(\mathbf{W}_{yh} h_t + \mathbf{b}_y\right) \tag{3}$$

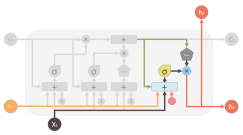
To solve these problems LSTM was designed. A common LSTM cell is made of different gates, which have specialized functionality. The solution to exploding gradients is very simple which has been used by researchers for many years: **clipping the gradient**.



(a) Input gate operation



(b) Forget gate operations.



(c) Output gate operations.

Figure 2: LSTM cell operations.

These gates include:

- **Input Gate :-** Input Gate is used to update the state of the cell.
- **Forget Gate :-** This gate decides which information to keep and thrown away. The data from the input gate

is passed through sigmoid function which ensures the values remains between 0 and 1. The value close to 0 means the forget gate has to forget this value and 1 means keep the value.

- **Output Gate :-** The last is the output gate. Output Gate decides the next cell state.

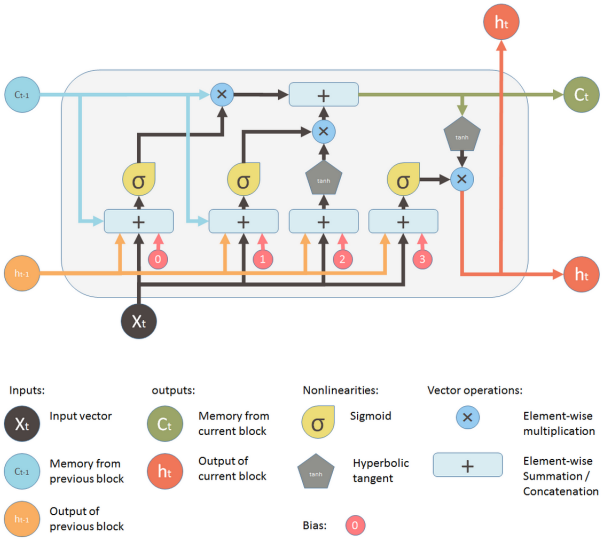



Figure 3: LSTM Building Block

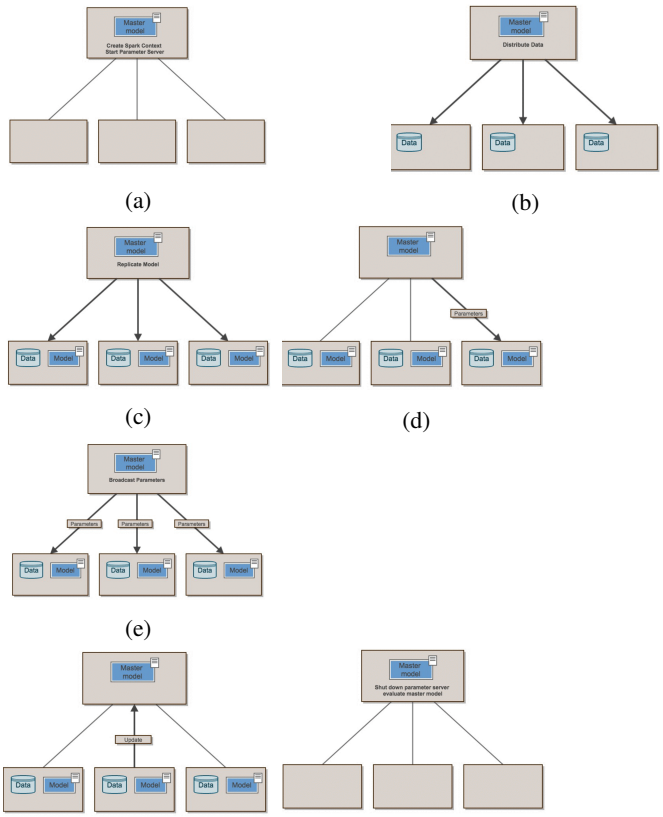


Figure 4: Elephas Work Flow

$$
\begin{aligned}
i_t &= \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \\
f_t &= \sigma(W_{xf}x_t + W_{hf}h_{t-1} + W_{cf}c_{t-1} + b_f) \\
c_t &= f_t c_{t-1} + i_t tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \\
o_t &= \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_t + b_o) \\
h_t &= o_t tanh(c_t)
\end{aligned}
$$

First the spark context is created and the data is distributed to worker nodes using RDD data structure. Then the model written using Keras is distributed to workers. Workers request for parameters from the master node. The master node broadcasts the parameters to all workers and use proper locking mechanism for updating the parameters. After the parameters are updated the parameter-node shuts down. This explains the work flow of Elephas framework.

## III. Elephas: Distributed LSTM using Keras and Spark

Elephas bring deep learning with spark[1]. It takes advantage of spark's Resilient Distributed Dataset (RDD), which helps in fast access to distributed data. It is very efficient data structure provided by spark platform. Elephas implements a class of data-parallel algorithms on top of Keras, using Spark's RDDs and data frames[1]. Keras models are initialised and then sent to the workers for further computation, keeping parameters in master node for broadcasting to workers and updates. Elephas provides distribution in three ways:

- Data-parallel training of deep learning models
- Distributed hyper-parameter optimization
- Distributed training of ensemble models

## IV. Advantages of Distributed training of LSTM

There are many advantages of distributing Deep Learning networks. Now a days data is increasing at a very high rate and the deep neural networks are getting deeper. Complex networks take a lot of time for training on a single machine. Hence to improve the efficiency of training and predicting values increases using this approach which is very important in real time stock market analysis and prediction of rise and fall of prices. These includes:

- Fast and scalable training and prediction which is very important in stock market analysis for optimization of profit.
- GPUs are not required for heavy computation as model is also distributed over many machines.
- Low cost approach.

## V. Dataset Used

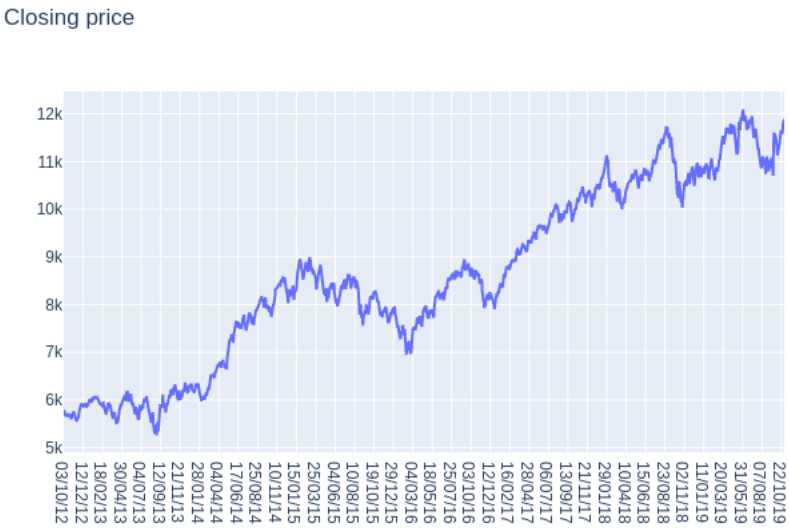



Figure 5: Nifty-50 Data-set.

Data set used in this project is past 7 years data of Nifty-50 stocks from investing.com. Sample data is shown in figure below.

| | Date | Price | Open | High | Low | Vol. | Change % |
|---|---|---|---|---|---|---|---|
| 0 | 3-Oct-12 | 5,731.25 | 5,727.70 | 5,743.25 | 5,715.80 | 173.22M | 0.22% |
| 1 | 4-Oct-12 | 5,787.60 | 5,751.55 | 5,807.25 | 5,751.35 | 179.19M | 0.98% |
| 2 | 5-Oct-12 | 5,746.95 | 5,815.00 | 5,815.35 | 4,888.20 | 255.57M | -0.70% |
| 3 | 8-Oct-12 | 5,676.00 | 5,751.85 | 5,751.85 | 5,666.20 | 142.32M | -1.23% |
| 4 | 9-Oct-12 | 5,704.60 | 5,708.15 | 5,728.65 | 5,677.90 | 119.30M | 0.50% |

Figure 6: Sample Data for Nifty-50.

## VI. Previous Works

Many previous works are done in the field of distribution of deep learning models from which this project is inspired. These include many tech giants such as Google, Amazon, Microsoft, etc

- DistBelief from Google
  - **Downpour SGD :-** Provides asynchronous and distributed implementation of stochastic gradient descent. The training instances are divided across different machines (Data Parallelism).
  - **Model Parallelism :-** For large deep learning neural network the model layers itself can distributed across as different machines. This will speed up the training process on CPUs and requirement of GPUs are reduced.

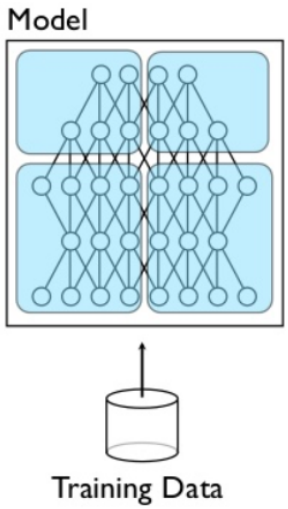



Figure 7: DistBelief From Google.

- Elephas: Distributed Deep Learning with Keras  Spark*
- Project Adams from Microsoft

## VII. Results and Observations

The predictions were made using statistical model, LSTM without distribution and LSTM with Distribution. It was observed the the statistical model used in this project to compare with LSTM performed poorly. It is possible that there are much better statistical models available on the internet from which comparison can be made. The Statistical model predicted comparatively at faster rate as there are very less parameters to compute using spark distributed system but under-performed Deep Learning approach. The LSTM model without distribution took more time to train for same number of iterations than the distributed LSTM. This can also be seen as the distributed LSTM model trained for same time interval as non-distributed model give more accurate predictions and better cosine similarity between validation set and predicted set. These inferences can be made from the graphs shown below.

- Blue Line: Training set.
- Orange/Red Line: Validation set.
- Green Line: Prediction over validation set.

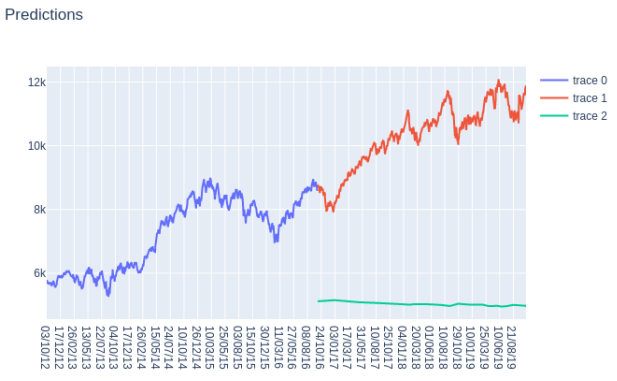



Figure 8: Statistical Model

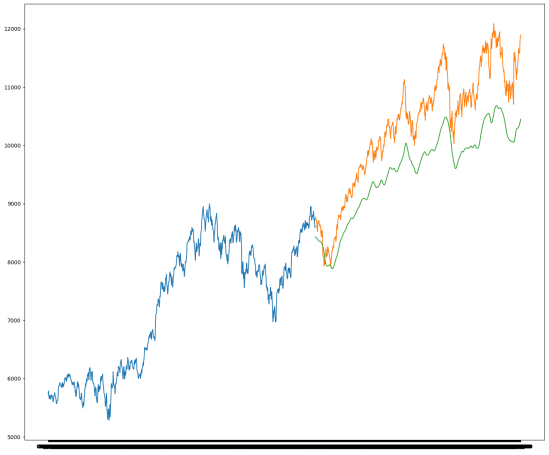


Figure 9: Non-Distributed LSTM Model

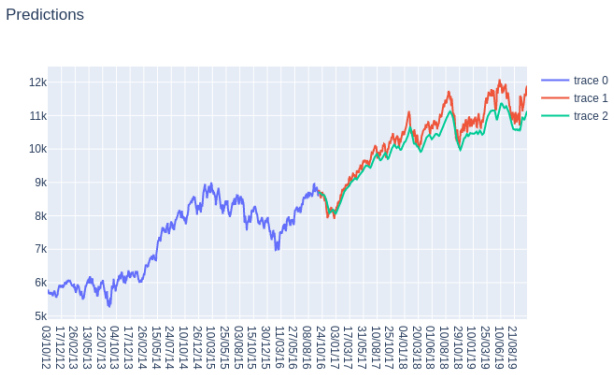



Figure 10: Distributed LSTM Model

## VIII. Challenges

- Main Challenge here is the distribution of model. There are some frameworks in spark for distribution of model. Two of them are BigDL and Elephas.
- Another challenge is how the data can be handled asynchronously. This can be handled using Downpour SGD.
- Identifying the right parameters for the prediction of stock prices.
- Initially working environment was Windows 10 for creating the model but found it difficult to combine the discrete components such as LSTM, pyspark and jupyter notebook. So the switch to Linux platform has to be made in later stages.

## IX. Conclusions

From the observations and results it is concluded that the a better statistical model with good accuracy can out perform Deep Learning models. This is essential in Stock Market analysis because of the sudden rise and fall of stock price and a need for faster predictions. Deep Neural network models have high latency in predictions so their efficiency is less as compared to statistical models but it can be improved using distributed LSTM and a very large data-sets. The similar analysis can be done on any other stock data-set but there will be need for different models and they had to be trained from scratch. It is possible to make a single model for all kind of stocks but the accuracy will reduce significantly in that case.

## References

[1] https://github.com/maxpumperla/elephasbasic-spark-integration
[2] Dean, Jeffrey, et al. *"Large scale distributed deep networks."* Advances in neural information processing systems. 2012.
[3] Shrivastava, Disha, Santanu Chaudhury, and Dr Jayadeva. *"A data and model-parallel, distributed and scalable framework for training of deep networks in apache spark."* arXiv preprint arXiv:1708.05840 (2017).
[4] https://software.intel.com/en-us/articles/bigdl-distributed-deep-learning-on-apache-spark
[5] J. Dean, G.S. Corrado, R. Monga, K. Chen, M. Devin, QV. Le, MZ. Mao, M'A. Ranzato, A. Senior, P. Tucker, K. Yang, and AY. Ng. Large Scale Distributed Deep Networks.
[6] https://github.com/cerndb/dist-keras?source=post_page———6d397c16abd———————-
[7] https://github.com/maxpumperla/elephas?source=post_page———6d397c16abd———————-