Data Analysis and Optimization of Asset Liability Management Problem for Banks

CERTIFICATE

This is to certify that the dissertation entitled "**Overlapping Community Discovery in Social Networks**" submitted by **D Shiva Shankar**, bearing Reg. No. 12MCMI41, in partial fulfillment of the requirements for the award of Master of Technology in Artificial Intelligence is a bonafide work carried out by him under my supervision and guidance.

The dissertation has not been submitted previously in part or in full to this or any other University or Institution for the award of any degree or diploma.

Dr. S. Durga Bhavani
Project Supervisor
School of CIS
University of Hyderabad

Dean
School of CIS
University of Hyderabad

DECLARATION

I, D Shiva Shankar hereby declare that this dissertation entitled "**Overlapping Community Discovery in Social Networks**" submitted by me under the guidance and supervision of **Dr. S. Durga Bhavani** is a bonafide work. I also declare that it has not been submitted previously in part or in full to this University or other University or Institution for the award of any degree or diploma.

Date:	(D Shiva Shankar)
Place:	12MCMI41

To,

My Parents

Acknowledgments

I would like to express my sincere gratitude to **Dr. S. Durga Bhavani**, my project supervisor, for valuable suggestions and keen interest through out the progress of my course of research.

I am grateful to Dean, SCIS for providing excellent computing facilities and a congenial atmosphere for progressing with my project.

I would like to thank **The University of Hyderabad** for providing all the necessary resources for the successful completion of my course work. At last, but not the least I thank my classmates and other students of SCIS for their physical and moral support.

I would also like to thank the **Open Source Community** who provided the free Software and documentation to work with.

With Sincere Regards,
D Shiva Shankar

Abstract

Social networks are modeled as graphs with nodes representing individuals and edges denoting interactions between the individuals. A community is defined as a subgraph with dense internal connections and sparse external connections. Many heuristic algorithms have been proposed in the literature for disjoint community detection. Nodes that belong to more than one community form overlapping regions between communities. In this thesis, we proposed two novel overlapping community discovery algorithms based on the idea of consensus clustering. The first algorithm defines core members of a community as nodes that will be co-clustered by all the algorithms as belonging to one community. The rest of the nodes, unless on the periphery are potentially in the overlapping regions of communities. The second algorithm tries to retrieve overlapping nodes directly based on connectivity consensus. The intuition behind this algorithm is that the neighbours of an overlapping node most probably belong to different communities. Here again the 'overlappingness' of a node is decided based on the consensus arrived at by majority of the community discovery algorithms. The two algorithms are implemented and tested on many benchmark data sets for community discovery. The algorithms successfully detected overlapping nodes that can be verified visually for small networks. Since overlapping node information is not available for benchmark networks, we designed two additional data sets joining friendship networks of two friends using their individual facebook data. It is shown that both the algorithms retrieve friends who have high degree of interactions with many other friends as overlapping nodes. These nodes are not merely those that belong to intersection of two communities but belong to common regions of many hidden communities that are discovered by these algorithms which could not have been inferred easily.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter, first, we learn about the risks associated with banking. Then we will discuss Asset Liability Management (ALM) a solution over risk associated with banking and ALM for banks. Then we will discuss project objectives and problem statement. Then a glance over banking standards and the literature survey.[]

1.1 Risks Associated with Banking

The term risk can be defined in association with banking as the exposure of to loss. The several risks associated with banking are viz., Operational risk, Credit risk, Liquidity risk, Market risk, Foreign exchange risk, Interest rate risk and Information risk.

1.1.1 Operational Risk

It is defined as the loss occurred due to failure of peoples or system. It is faced in the various departments as of Information Technology department, Credit Department, Investment department.

1.1.2 Credit Risk

It is defined as the loss occurred due to failure of the borrower to repay to bank on the acknowledged terms. There is uncertainty about the repayment of dues and in the repayment in the agreed time frame. It happens due to borrowers lack of income, failure of business or reluctance to repay and lack of underwriting frameworks.

1.1.3 Market risk

It is defined as the loss occurred due to fluctuation in the market prices. Its components are as follows:

- **Equity risk:** Probable failure to generate profit due to fluctuation in stock prices.
- **Foreign exchange risk:** Probable failure to generate profit due to the fluctuation of exchange rates as bank does transaction in multiple currencies.
- **Interest rate risk:** Probable failure to generate profit due to the fluctuation of interest rates.
- **Commodity risk:** Probable failure to generate profit due to change in commodity prices as metals (as gold, silver, platinum), Energy (oil and gas) and Agriculture (as wheat, cotton, coffee, tea,etc.). The change in these prices occurs due to variations in demand and supply.

1.1.4 Liquidity risk

It is characterized as the lack of liquid cash available in hands of the bank to finance its day to day transactions or the situation where bank runs out of cash. Failure in managing the liquidity risk can lead to the destruction of the banks' reputation and losing its trust among customers.
The exposure of banks to these various risks and as the risks is also increasing with the liberalisation and increasing merger of local markets and global markets. Due to the deregulated operational space and the various products that requires the interest rates to be determined with the need to maintain the proper ratios between the profitability, spread and long term growth. Due to all these reasons, we need to perform Asset Liability management.

1.2 Asset Liability Management (ALM)

An asset is a valuable entity/resource that a person/organization owns for generating income. And the liability is a valuable entity/resource that a person/organization

needs to pay for. At any given point of time, total assets must be greater than the total liability. To balance the equation the concept of asset liability management had emerged.
In ALM we perform periodic monitoring of risk exposures involving collecting and analyzing the information in order to have the ability to anticipate, forecast and act so as to structure banks business to profit. ALM also involves transforming the asset and liability portfolio in a dynamic way to manage the risks which involve judgment and decision making.

1.3 ALM for Banks

As the business of banking involves lot of risks, the main problem of banking becomes risk management and the procedure to do so is ALM. The three pillars on which ALM resides are follows:

1.3.1 ALM Information System

The risk policies and tolerance limits need to be specified by ALM information system. Information is the key to ALM process which is now available due to the computerization of all banks and its respective branches. It also includes performing experimentation in a branch and studying its effects. If the results are positive then replicating the changes to other branches.

1.3.2 ALM Organization

For risk management to be successful the need is of the strong commitment of the senior management to take strategic decisions and integrate the basic operations. The Asset Liability Committee (ALCO) is formed for performing the above-mentioned tasks. It includes the CEO and the senior management of the bank, and their task is to decide business strategies with respect to banks budget and decide risk management objectives according to assets and liabilities.

1.3.3 ALM Processes

The ALM Processes are as shown in the following figure:

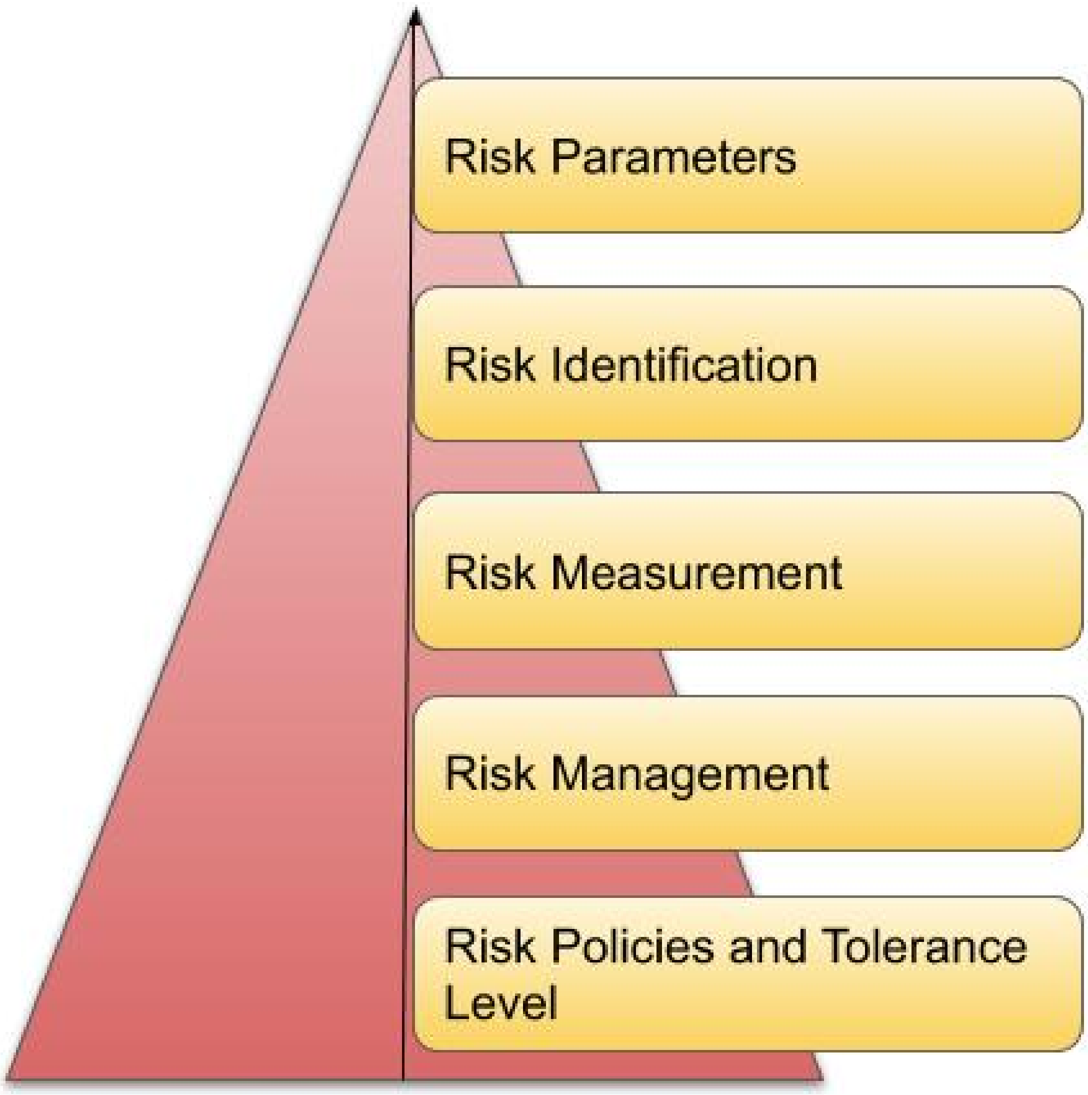
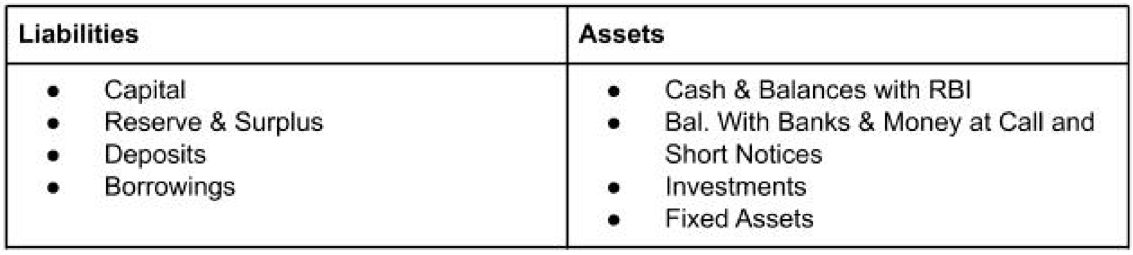


Figure 1.1: ALM Processes

All the risks discussed in section 1.1 are the problems to be handeled by the ALM Processes. The assets and liabilities of a bank involve the following:



Liabilities	**Assets**
• Capital • Reserve & Surplus • Deposits • Borrowings	• Cash & Balances with RBI • Bal. With Banks & Money at Call and Short Notices • Investments • Fixed Assets

Figure 1.2: Components of Banks Balance Sheet

The ALM cycle of a bank is as follows:

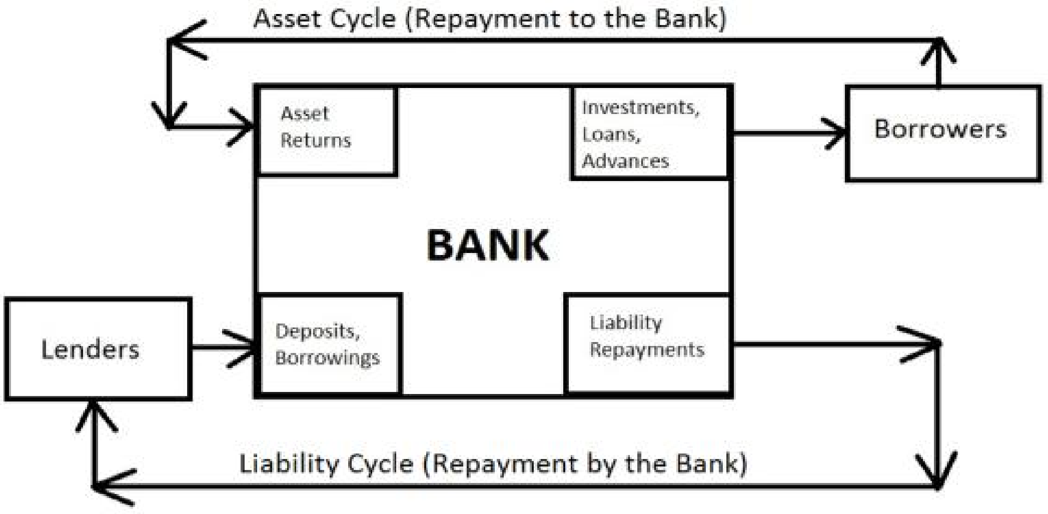

Asset Cycle (Repayment to the Bank)
Asset Returns
Investments, Loans, Advances
Borrowers
BANK
Lenders
Deposits, Borrowings
Liability Repayments
Liability Cycle (Repayment by the Bank)

Figure 1.3: ALM Cycle of a Bank

Throughout the cycle of the ALM the mismatch in asset and liabilities (negative gap) should not exceed 20% of the cash outflow during 2-14 days and 15-28 days time bucket.

1.4 Project Objective

Creation of strategies to stabilize financial networks i.e. banks and to improve the profitability of them from various risks.

1.5 Problem Statement

The two problems that have been tried to solve are as follows:

1.5.1 Liquidity Risk Management

In the single objective cash flow optimization for ALM, we have proposed a model of Linear Programming Problem (LPP) which ensures Liquidity and Profitability. Liquidity Risk is managed by considering the flow approach and profitability is ensured by taking the objectives function to maximize the profit of a bank. We have also studied and analyzed a 1999 Czech Financial dataset in search of the assets for the bank.

1.5.2 Prediction of Stock Price

The Prediction of Stock Price is a typical problem of time series and we propose a deep learning based solution for future price prediction.

1.6 Literature Survey

For first problem the literature survey done is: Hongxi Li et al. (2017) assess for both positive and negative duration gap to increase the net value of bank when interest rates fluctuate favorably[]. Nalan Gülpinar et al. (2013) uses the Vector Autoregressive process to model time-varying investment opportunities[]. Teng Fan et al. (2011) studied the interest rate risk of Chinese life insurers' liability[]. Mounika, P et al. (2011) have addressed the problem of single objective optimization for the maximization of wealth[]. Chaudhury, Rahul et al. (2014) created a fuzzy rule-based asset liability optimization model[].

For the second problem, the literature survey done is: Ashish Vaswani et al. (2017) proposed an attention mechanism for machine translation[]. Yao Qin et al. (2017) proposed a DA-RNN. It serves the purpose of attention to time series and encoding information of long sequences[]. Jian Liu et al. (2017) assesses the correlation between stock price movement with relevance to events happening in the world[]. Hao Li et al. (2018) proposed the MI-LSTM using attention which filters noise and extracts information[]. Huicheng Liu (2018) used attention based RNN for leveraging the news to predict stock price[].

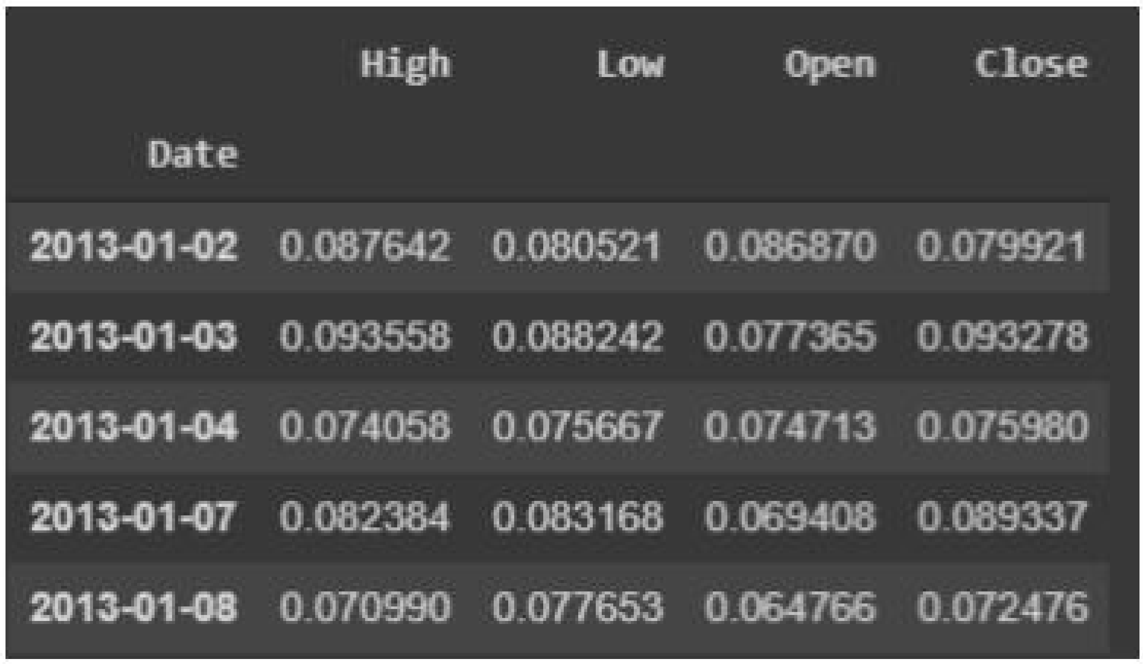
In this chapter we describe a few of the algorithms for our problems whose knowledge we assume in the dissertation. The implementation of some of these algorithms is available in a software package called *statsmodels, pandas, sklearn* [] which are also necessary statistical analysis.

1.6.1 Time Series Analysis for Stock Price Prediction

1.6.1.1 Introduction

The Time Series (TS) is defined as the collection of information or data at a regular interval of time. This TS contains the time-dependent data which may contain the

seasonality (i.e the deviation in data at specific time frame) and trend (i.e. the changing mean with respect to time) in it; i.e. the change in data with respect to specific time frame.
Ex. The Stock Data is collected per day at the start of the day, at the end of the day, the highest value of the day and the lowest value of the day.



	High	Low	Open	Close
Date				
2013-01-02	0.087642	0.080521	0.086870	0.079921
2013-01-03	0.093558	0.088242	0.077365	0.093278
2013-01-04	0.074058	0.075667	0.074713	0.075980
2013-01-07	0.082384	0.083168	0.069408	0.089337
2013-01-08	0.070990	0.077653	0.064766	0.072476

Figure 1.4: Ex of Time Series data of a stock

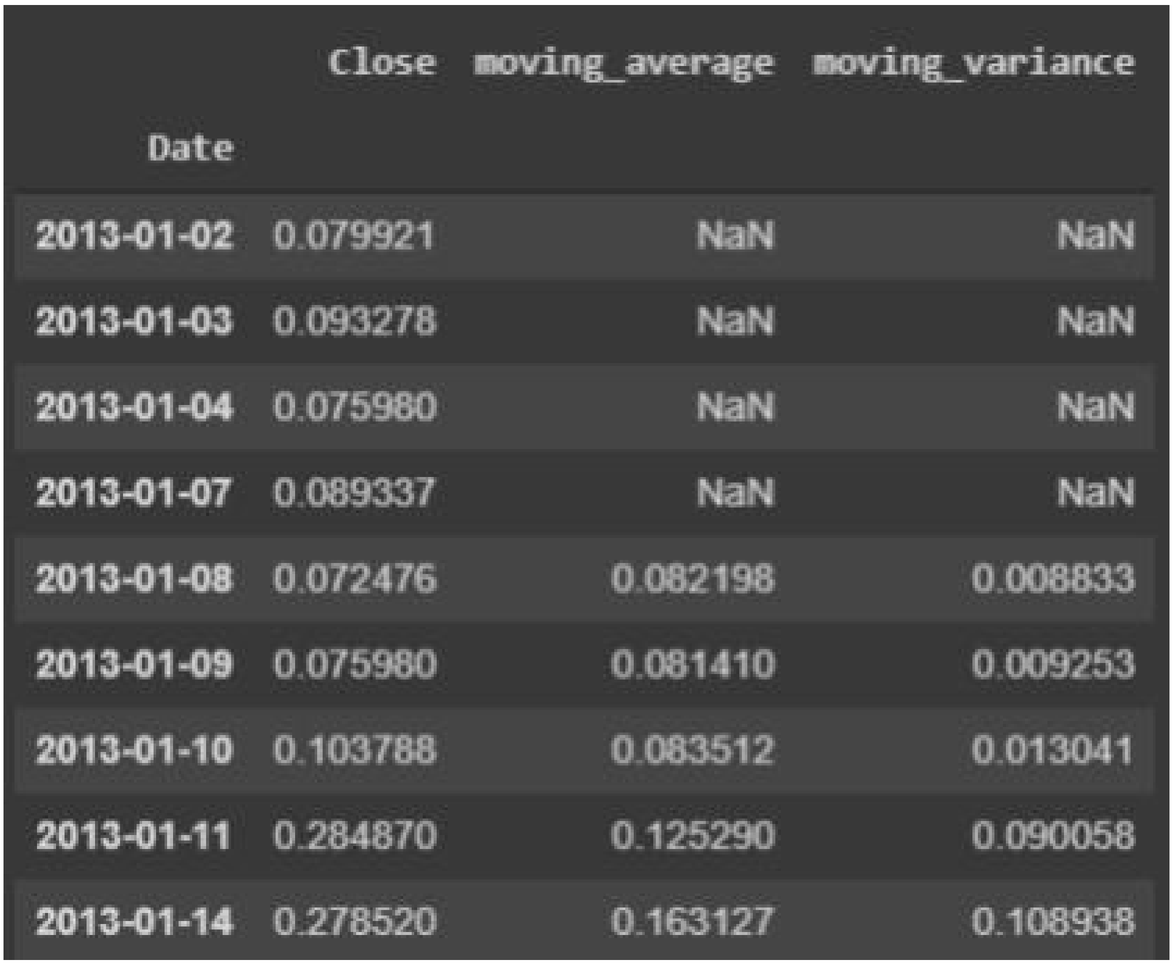
1.6.1.2 Issues with non-Stationarity of Data

The data is said to be stationarity if the mean and variance of the data is stable over time and the time-independent autocovariance. The statistical models which deal with the TS data have a premise that the data should be stationary. Because of this premise, we need to convert non-stationary data to stationary data. But how to check the data is stationary or not? There are two ways to do it:

- Plotting the Rolling Statistics
- Dickey-Fuller Test

1.6.1.2.1 Plotting the Rolling Statistics

In this procedure, we visualize the moving average and/or moving variance by plotting it against time.



Date	Close	moving_average	moving_variance
2013-01-02	0.079921	NaN	NaN
2013-01-03	0.093278	NaN	NaN
2013-01-04	0.075980	NaN	NaN
2013-01-07	0.089337	NaN	NaN
2013-01-08	0.072476	0.082198	0.008833
2013-01-09	0.075980	0.081410	0.009253
2013-01-10	0.103788	0.083512	0.013041
2013-01-11	0.284870	0.125290	0.090058
2013-01-14	0.278520	0.163127	0.108938

Figure 1.5: Moving Average and Moving Variance of the Closing Price with window size = 5

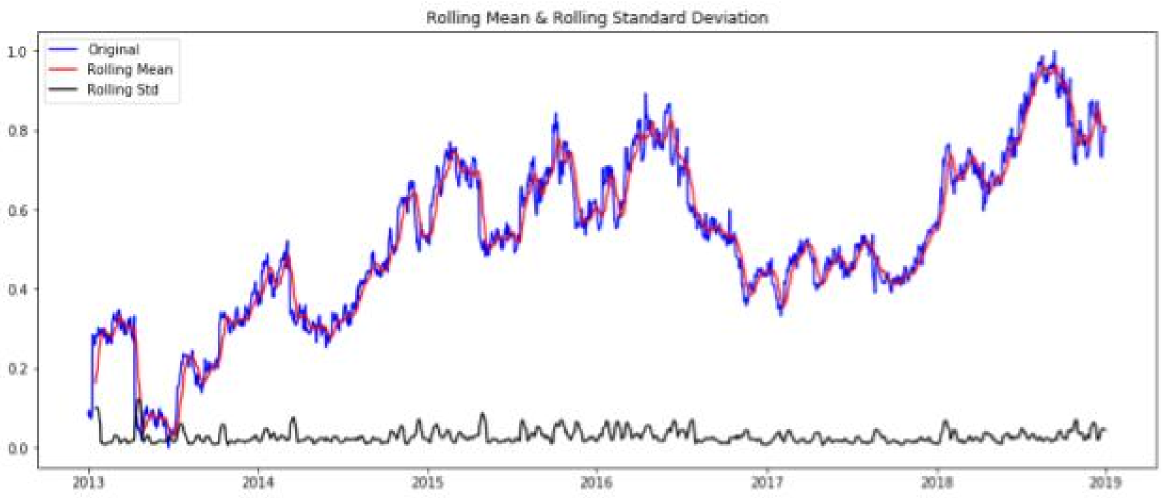
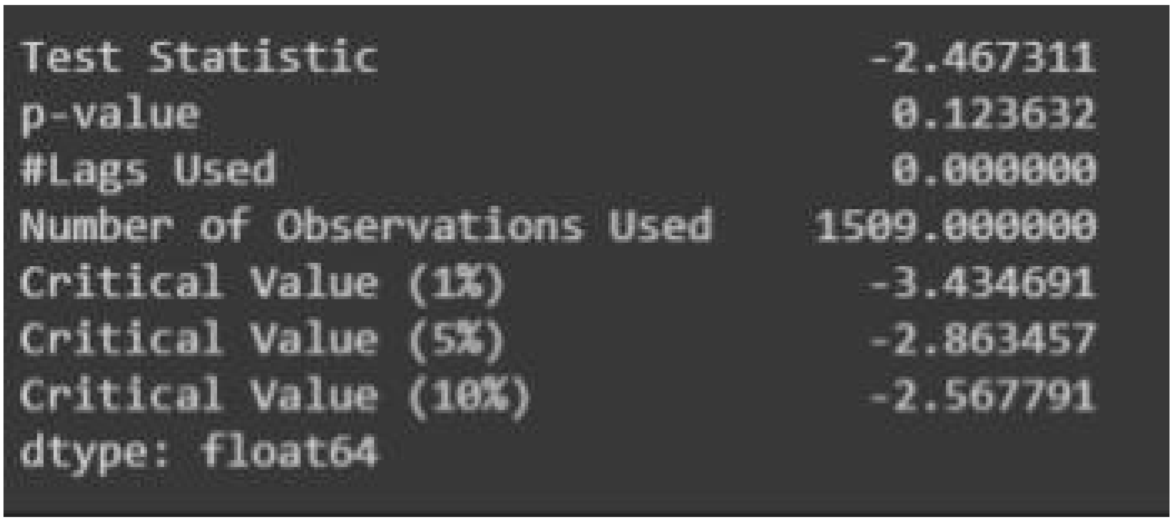


Figure 1.6: Plot of the Rolling Mean and the Rolling Standard Deviation of Moving Average of Closing Price

As we can infer from the plot, the rolling mean is changing with respect to time and there is a clear fluctuation in the standard deviation resulting in the data to be non-Stationary.

1.6.1.2.2 Dickey-Fuller Test (DF Test)

In statistics, there is a unit root test that is used to check the given TS is non-Stationary or not. DF Test is the most widely used unit root test. In statistical testing we have to make the hypothesis, the same in this test are:
H0: TS is non-Stationary. H1: TS is Stationary.
H0 is the Null hypothesis and the H1 is the Alternate Hypothesis. The hypothesis H0 is accepted if the p-value of the test is greater than 5% else the H0 is rejected and H1 is accepted.
For the data we are dealing with the results of the DF Test are as follows:



```
Test Statistic                  -2.467311
p-value                          0.123632
#Lags Used                       0.000000
Number of Observations Used   1509.000000
Critical Value (1%)             -3.434691
Critical Value (5%)             -2.863457
Critical Value (10%)            -2.567791
dtype: float64
```

Figure 1.7: DF Test Results for moving average of Closing price

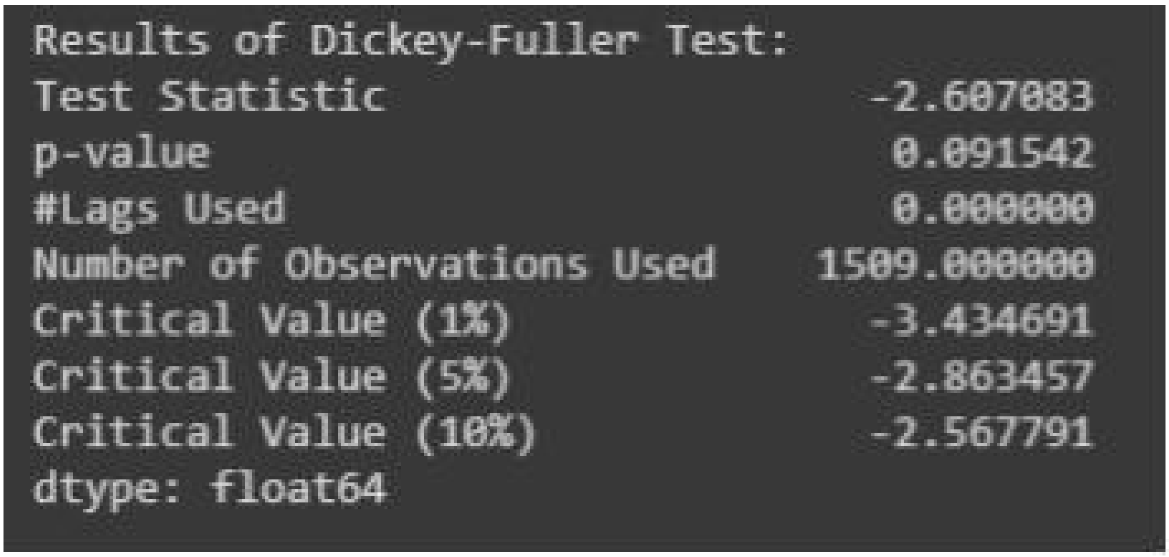
As the p-value is 0.1236 i.e. 12.36% which is greater than the threshold 5% so the null hypothesis H0 is accepted; meaning the TS is non-Stationary.

1.6.1.3 Converting non-Stationary TS to Stationary TS

To make the TS stationary we need to eliminate the trend and seasonality from the non-stationary TS. The first and the widely used approach to do it is Transforming the data using a log transform. Then use either Differencing or Decomposition. Let's see them one by one:

1.6.1.3.1 Log Transform

Taking the log of the data changes the recurrent pattern to a linear pattern and also stabilizes the variance of the data.



```
Results of Dickey-Fuller Test:
Test Statistic                  -2.607083
p-value                          0.091542
#Lags Used                       0.000000
Number of Observations Used   1509.000000
Critical Value (1%)             -3.434691
Critical Value (5%)             -2.863457
Critical Value (10%)            -2.567791
dtype: float64
```

Figure 1.8: DF Test Results for log-transformed Closing Price

Here we can see that the p-value is dropped from 12.36% to 9.15% which means that we can say we have 90% confidence that the TS is stationary.

1.6.1.3.2 Decomposing

A TS consists of three components: trend, seasonality and residual. If the decomposition be additive then:

$$Data_{\mathrm{t}} = Trend_{\mathrm{t}} + Seasonality_{\mathrm{t}} + Residual_{\mathrm{t}} \tag{1.1}$$

Where the subscript t denotes the time t. The multiplicative decomposition is:

$$Data_{\mathrm{t}} = Trend_{\mathrm{t}} * Seasonality_{\mathrm{t}} * Residual_{\mathrm{t}} \tag{1.2}$$

The addition/multiplication of these three components gives back the original data or TS. The multiplicative decomposition is used when the trend/seasonality of data is proportional to the time. The procedure of classical multiplicative decomposition is as follows:

Algorithm 1 Classical Multiplicative Decomposition

```
Assumption m - seasional period or frequency, MA - moving average, $DT_t$ - Detrended series.
 1: If m is even number then
        $$Trend_t = 2 * m - MA \quad (1.3)$$
    Else
        $$Trend_t = m - MA \quad (1.4)$$
 2: Compute Detrended series:
        $$DT_t = Data_t - Trend_t \quad (1.5)$$
 3: The seasonal component Seasonalitytof the respective season is the average of the $DT_t$ of the given season.
 4:
        $$Residual_t = Data_t/(Trend_t * Seasonality_t) \quad (1.6)$$
```

The plot looks as follows:

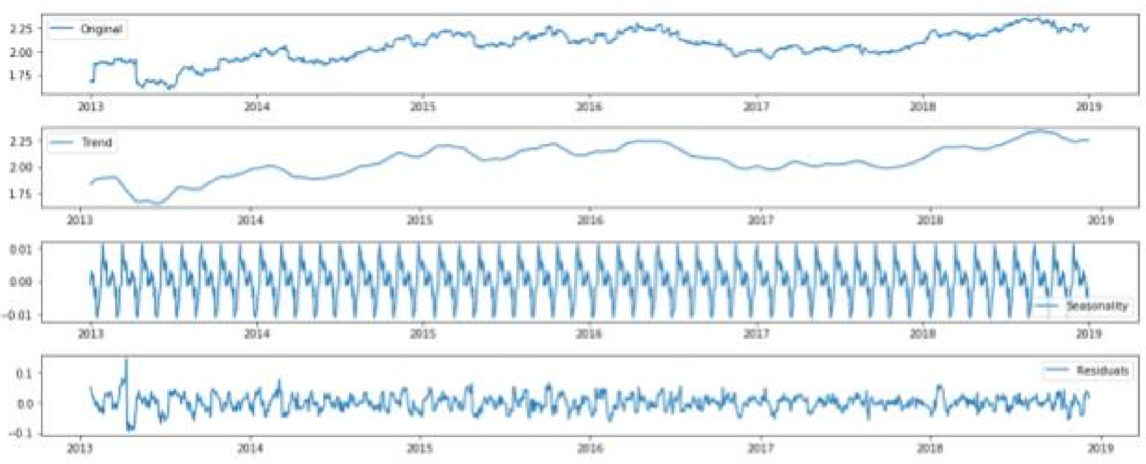
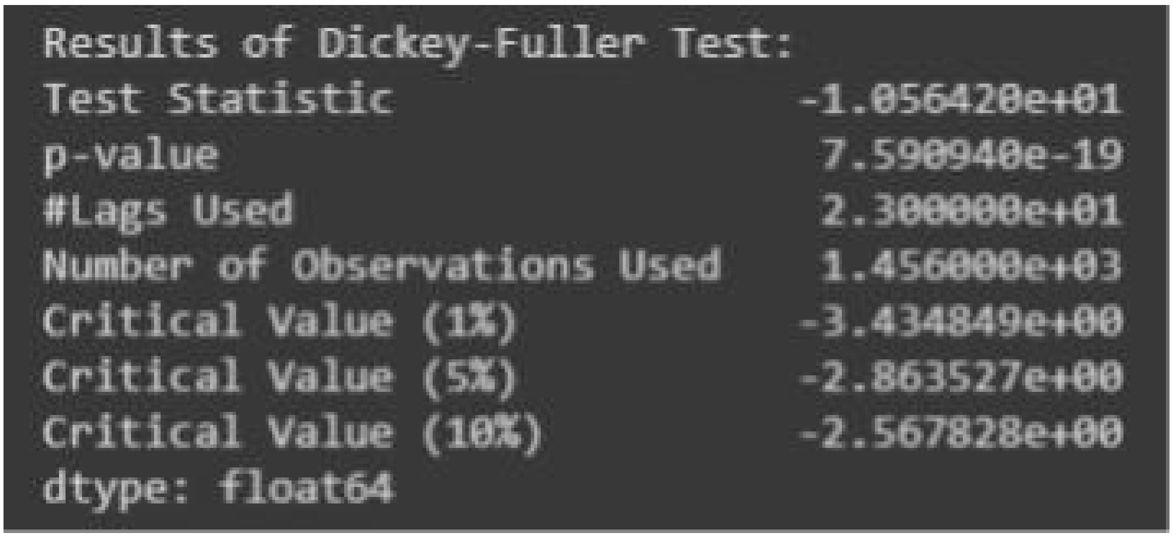


Figure 1.9: The trend, seasonality and residual of log-transformed data

The result of DF test results on the residual data is as follows:



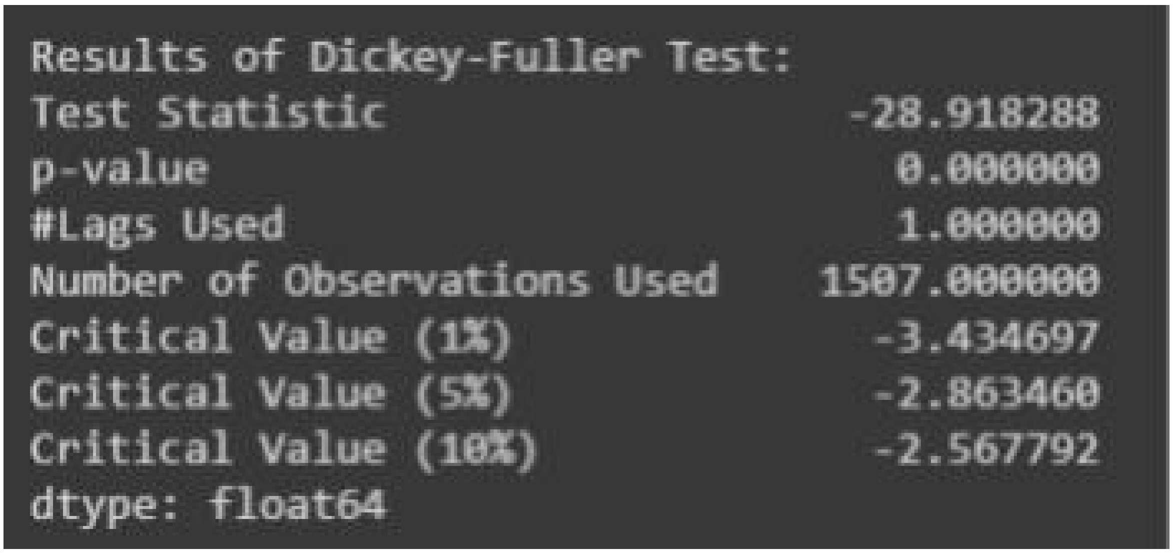
```
Results of Dickey-Fuller Test:
Test Statistic                 -1.056420e+01
p-value                         7.590940e-19
#Lags Used                      2.300000e+01
Number of Observations Used     1.456000e+03
Critical Value (1%)            -3.434849e+00
Critical Value (5%)            -2.863527e+00
Critical Value (10%)           -2.567828e+00
dtype: float64
```

Figure 1.10: DF Test Results on residual data

Here the p-value is 7.5e-17% which means that we can say the TS is stationary with 99.99% confidence.

1.6.1.3.3 Differencing

In this approach, we compute the difference between the data point at the current instance to the previous instance. Applying the DF test on the differencing of the log-transformed data gives the following results:



```
Results of Dickey-Fuller Test:
Test Statistic                  -28.918288
p-value                           0.000000
#Lags Used                        1.000000
Number of Observations Used    1507.000000
Critical Value (1%)              -3.434697
Critical Value (5%)              -2.863460
Critical Value (10%)             -2.567792
dtype: float64
```

Figure 1.11: DF Test Results for log-transformed Closing price after differencing

Here the p-value is 0.0% which means that we can say the TS is stationary with 100% confidence.

1.6.1.4 Disadvantages of Making TS stationary

There is huge information loss when we are converting the non-stationary TS to Stationary TS. The elimination of trend and seasonality leads to information loss which makes the model linear and it fails to predict the future trend and seasonality appropriately.

1.6.1.5 Inference

The statistical methods are good to work with the stationary data, but when the data is non-stationary the statistical methods fail due to the premise discussed in section 3.2. Although due to the advances in the research area of neural networks we are able to deal with the non-stationary data without converting it to stationary. The methods used to predict the TS data are discussed in following section.

1.6.2 Deep Learning Architectures for Prediction of Stock Price

In this chapter, we will study the different deep learning techniques used to predict the sequence of output after feeding them the time-based input sequence.
For all the tasks consider Data: $\mathrm{X_i}$: $\mathrm{Source_i}$; $\mathrm{Y_i}$: $\mathrm{Target_i}$ And the loss function: Mean Squared Error Loss as the problem is a regression problem.

1.6.2.1 Recurrent Neural Network (RNN)

Artificial neural networks have a special class of neural networks that works best with the input sequences, called the RNN. As stated in the name the word Recurrent stands for the repeating structure of the neurons with respect to time/sequence based input to them until the whole input sequence is over. Unlike other neural network architectures where we need to feed whole input at once to the input neurons; in RNN, we can feed the sequence one by one to the input neurons and the input gets processed in the same recurrent fashion throughout all the recurrent layers. The output of the neuron after processing the (t-1)th input sequence is fed again to the same neuron with the (t)th sequence of the input so that it can remember the whole input sequence. The idea basically is to remember the past, add it to the present and predict the future. The RNN looks as shown in the following Fig.

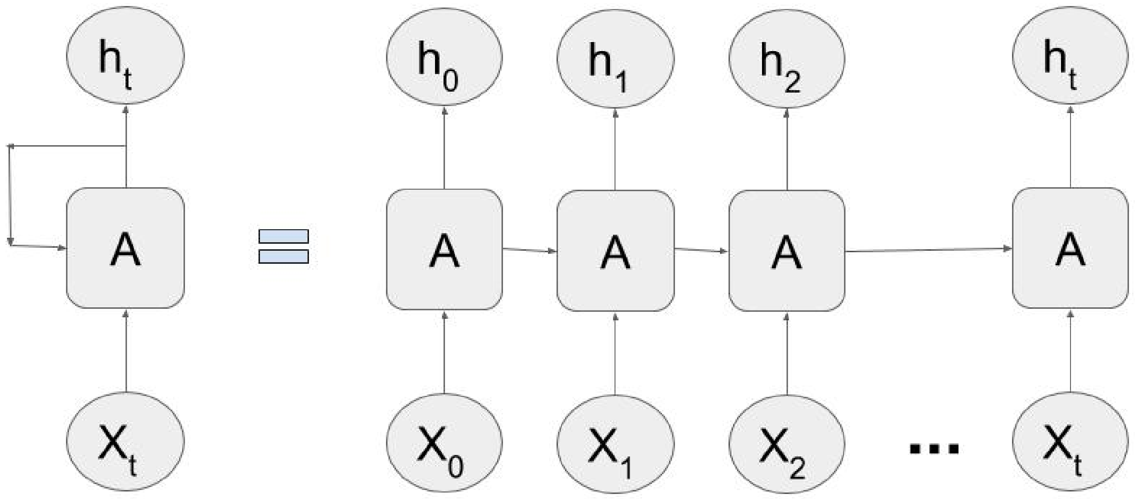


Figure 1.12: Unfolded Recurrent layer

A is the Recurrent layer of neurons, Xt is the input sequence and ht is the output of the recurrent layer

Now let's consider the Fig.4.3 which shows the activation function tanh in the neurons of recurrent layers. The equation of the recurrent layer is as follows:

$$h_t = tanh(w_i * X_t + w_o * h_{t-1} + b) \tag{1.7}$$

Where wi is the weight associated with the inputs and the wo is the weights associated with the output of the previous state. Using these weights we will learn about the sequenced input.

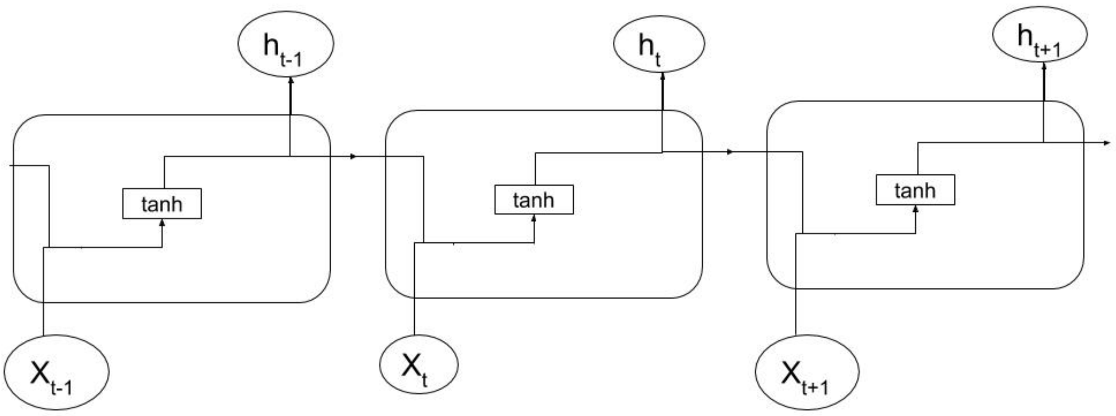


Figure 1.13: Unfolded Recurrent layer with activation function

1.6.2.1.1 Problems of Long Term Dependency

If the input sequences are small then the RNN is the best choice, but what if the input sequence is long, can RNN successfully learn the information from the past and using present information predict the future? The answer is No. In RNN we don't have an equation for how much to remember from the past or what to remember from the past. This is called the problem of long term dependency as shown in Fig.4.4. If the output ht+1 depends on the input X1 then RNN can't handle this dependency. The solution to this is the improved version of RNN discussed later.

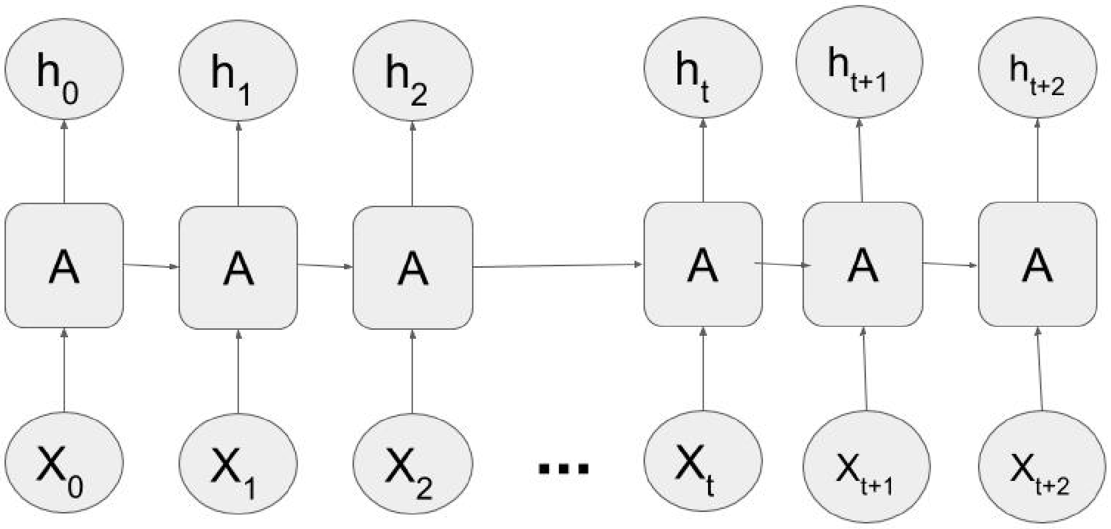


Figure 1.14: Long term dependency problem

1.6.2.1.2 Long Short Term Memory (LSTM)

The word LSTM can be interpreted as storing the Short information in the Memory for the Long Term. This happens due to the functional change in the cell of RNN. Instead of the single neuron function in the RNN, the LSTM has four neurons connected in a circuit creating three gates: input gate, forget gate and the output gate. Other than these gates there is a cell state which computes the update for the next state. The following are the notations that will be used to describe the architecture of the LSTM:

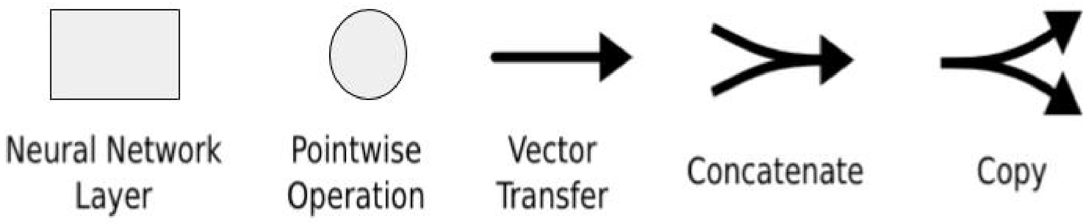


Figure 1.15: Notations

The comparison of RNN and LSTM is as follows:

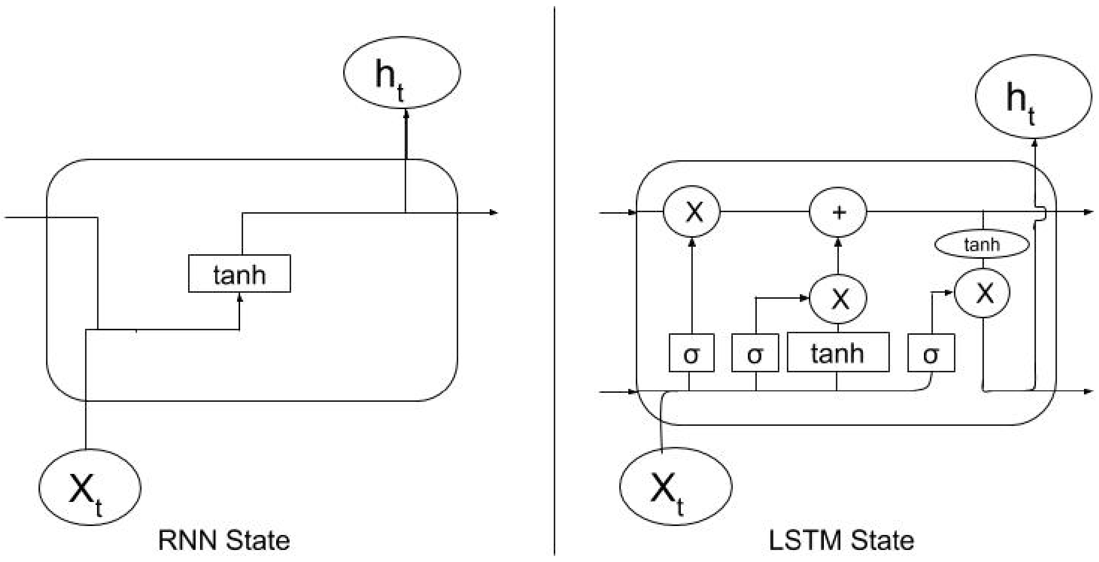


Figure 1.16: Comparison of RNN and LSTM State

Cell State The cell state decides how much of the original information from previous state is to be retained to the next state (Forget gate) and how much new information from the current state is to be added after forgetting the old information from the previous state (Input gate). It's like scaling and shifting operation. The information from the previous state is scaled between 0 to 1 and information from the current state is added (shifting).

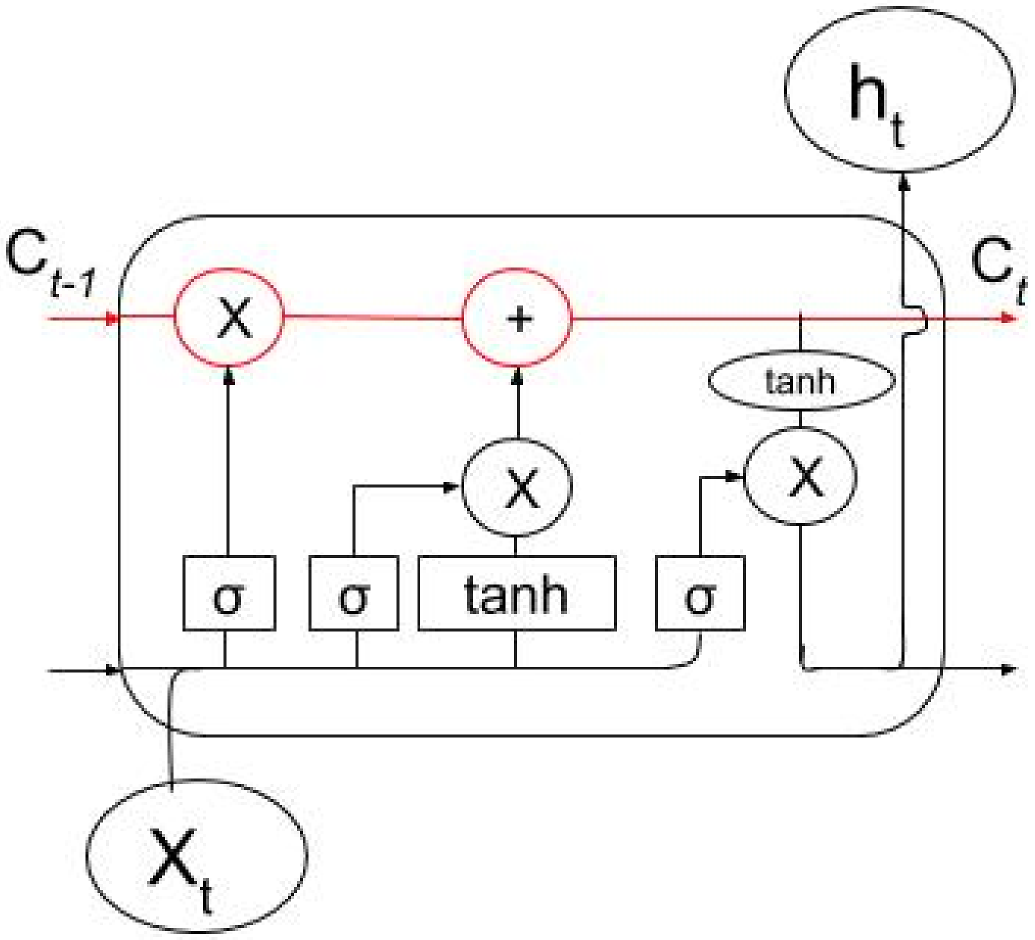


Figure 1.17: Cell State

Forget Gate As discussed in the cell state this gate uses the sigmoid function which gives the values between 0 to 1. These values determine the scaling of the forget gate. Equation for forget gate:

$$f_t = \sigma(W_f.[h_{t-1}, X_t] + b_f) \tag{1.8}$$

• Where f_t is the output of forget gate W_f is the weight associated with forget gate h_{t-1} is the previous state output X_t is the input to state t b_f is the bias term for forget get terms in [] reflects concatenation operation.

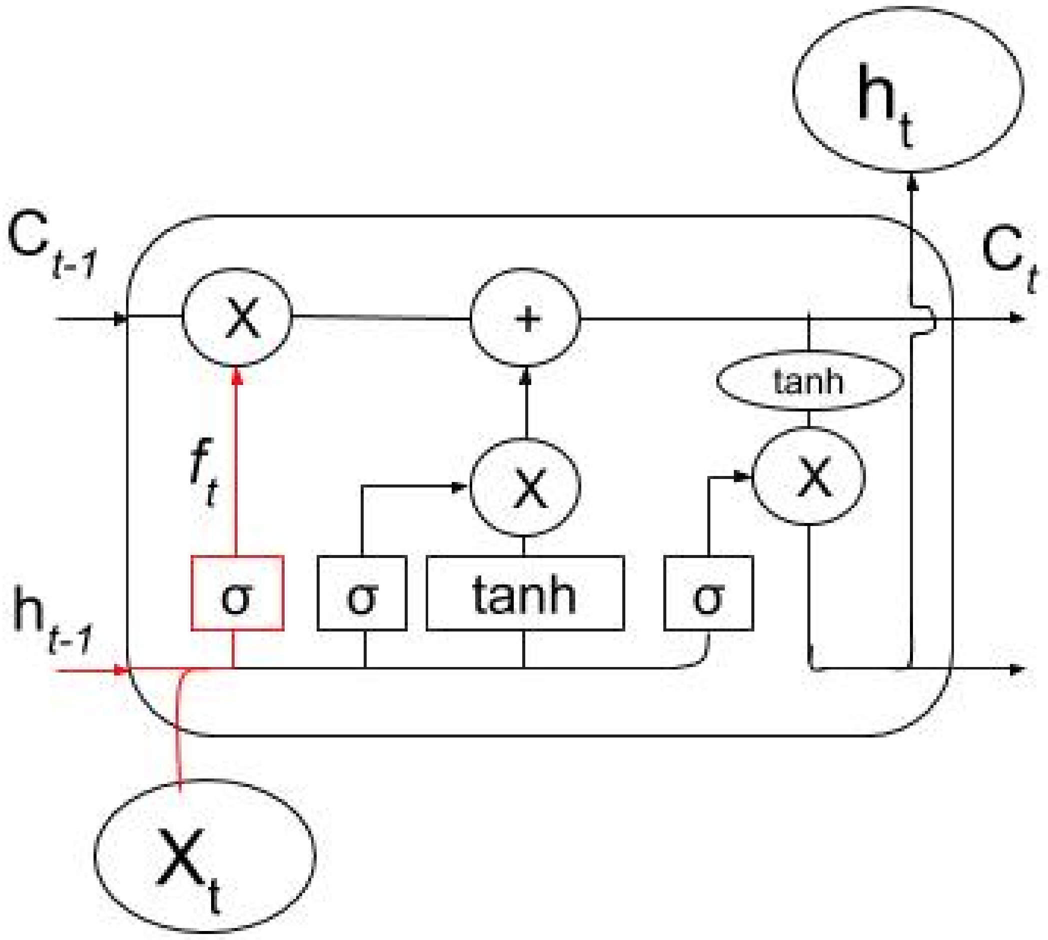


Figure 1.18: Forget Gate

Input Gate This gate decides how much information from the current state is to be added to the cell state. It uses a sigmoid and a tanh function for this purpose. The equation is as follows:

$$i_t = \sigma(W_i.[h_{t-1}, X_t] + b_i) \tag{1.9}$$

$$\tilde{I}_t = tanh(W_c.[h_{t-1}, X_t] + b_c) \tag{1.10}$$

The sigmoid function ranges between 0 to 1 and tanh function ranges between -1 to 1 which shifts the input accurately and creates the output of the input gate.

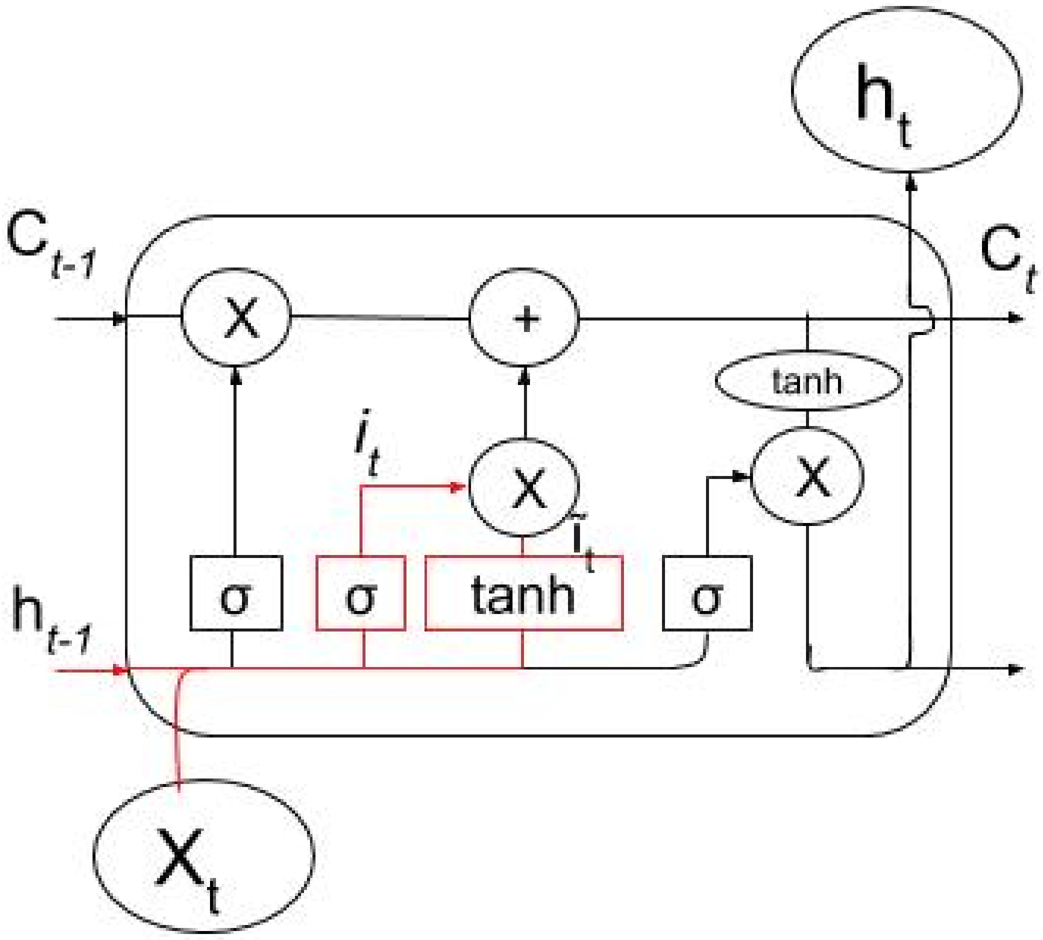


Figure 1.19: Input Gate

Updating Cell State The eq(2.8 to 2.10) computed the values for updating the cell state, now we just need to do pointwise multiplication and addition. The equation is as follows:

$$C_t = f_t * C_{t-1} + i_t * \tilde{I}_t \tag{1.11}$$

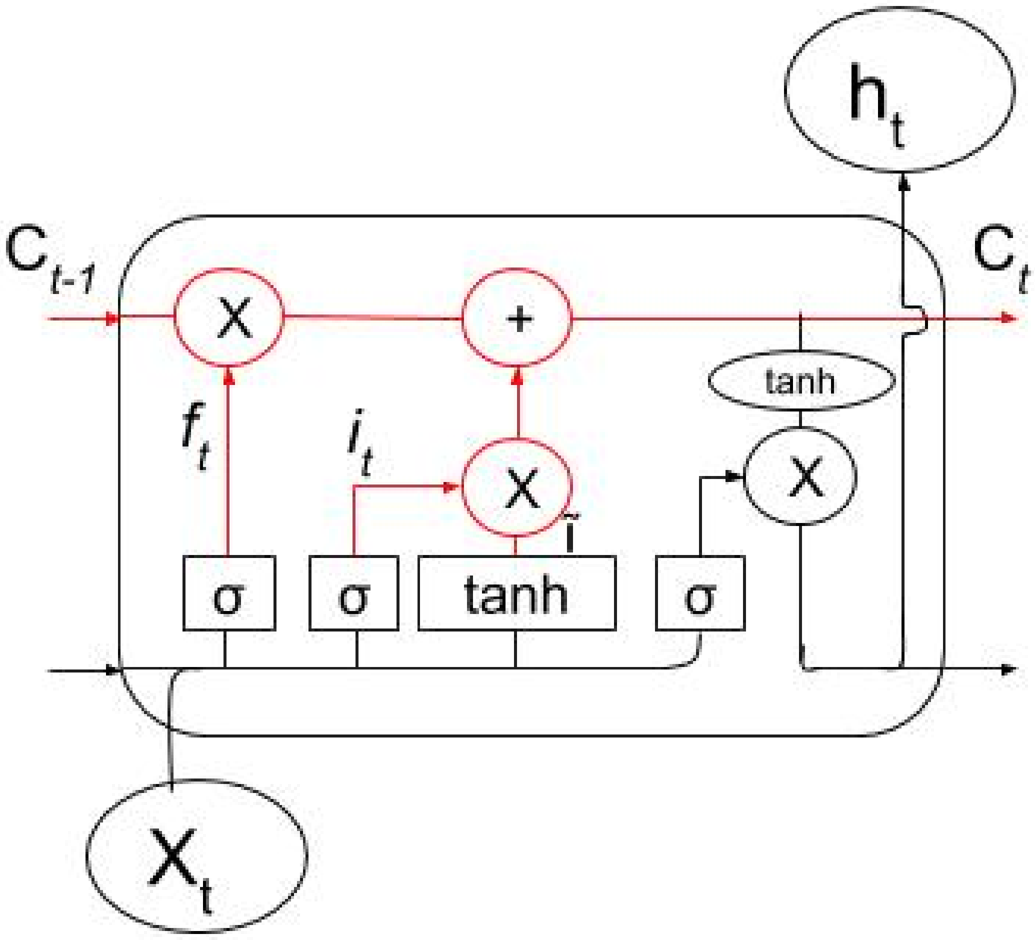


Figure 1.20: Update to new cell state

Output Gate Using sigmoid function on current states input and tanh function on the updated cell state we can output the parts we need. The equations are as follows:

$$o_t = \sigma(W_o.[h_{t-1}, X_t] + b_o) \tag{1.12}$$

$$h_t = o_t * tanh(C_t) \tag{1.13}$$

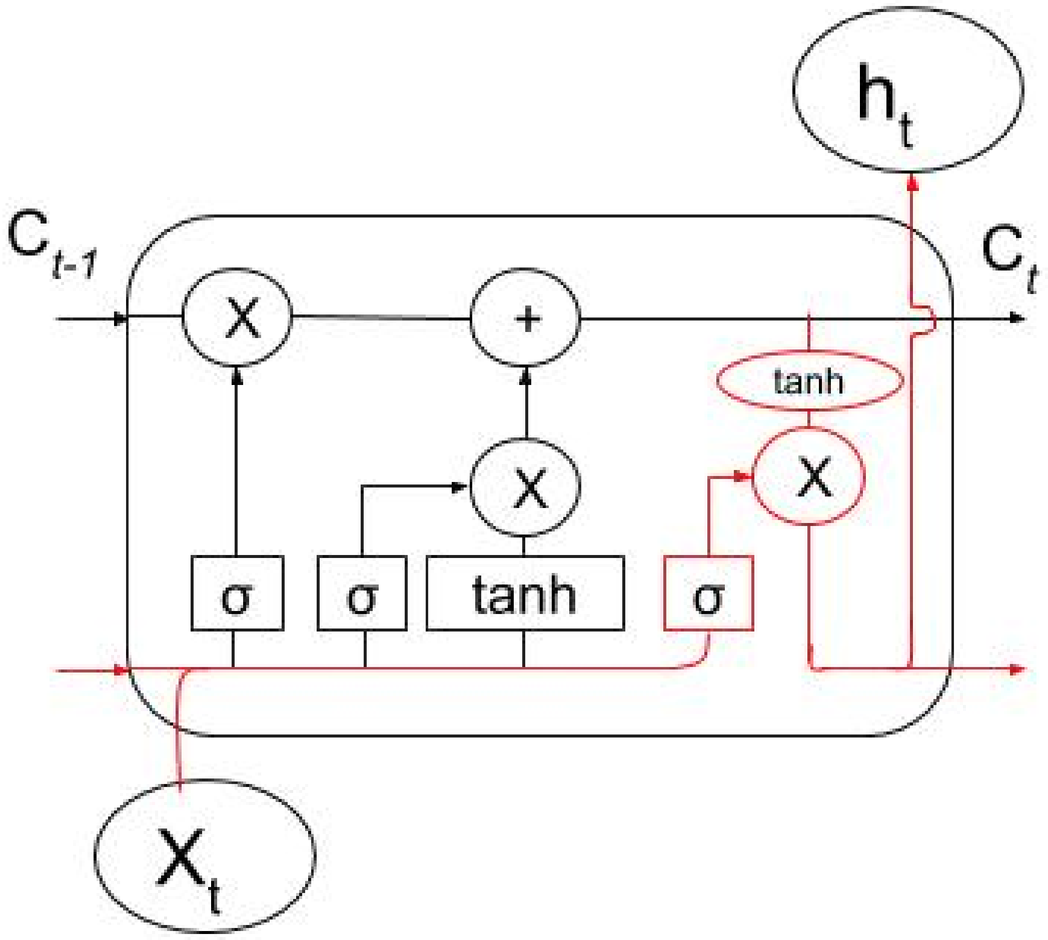


Figure 1.21: Output Gate

The Fig. below shows the unfolded LSTM layer.

For the understanding of the later equations let us consider the following equations:

For RNN:

$$h_t = RNN(h_{t-1}, X_t) \tag{1.14}$$

For LSTM:

$$h_t, C_t = LSTM(h_{t-1}, C_{t-1}, X_t) \tag{1.15}$$

1.6.2.1.3 Encoder - Decoder Models (Sequence to sequence models)

The Encoder - Decoder model is very much useful in a lot of applications. The Encoder which is a neural network works for computing the representation of the input while the decoder which is also a neural network uses this input representation from Encoder and the target output to learn about the relation between the input and output and makes the appropriate predictions. For our problem, both encoder and decoder use the RNN/LSTM as the encoding and decoding function. The Encoder - Decoder mechanism can be represented as in Fig. below:

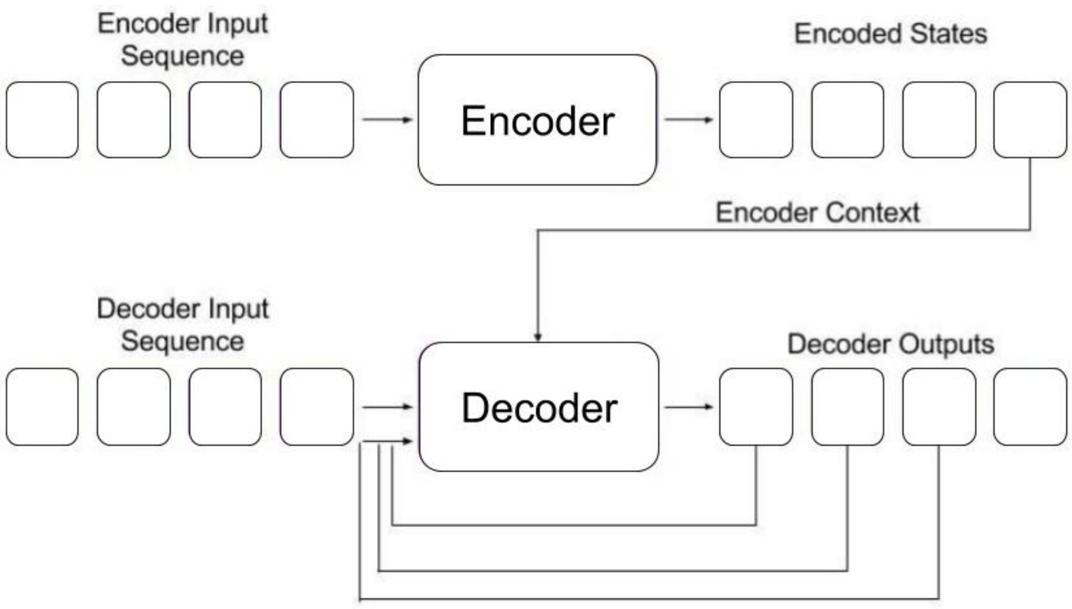


Figure 1.22: Encoder - Decoder Mechanism

The equations for the encoder-decoder model using RNN as the neural network architecture is as follows:
Encoder:

$$h_t = RNN(h_{t-1}, X_t) \tag{1.16}$$

Decoder:

$$C_0 = h_t \tag{1.17}$$

$$C_t = RNN(C_{t-1}, [h_{t-1}, e(\hat{y}_{t-1})]) \tag{1.18}$$

1.6.2.1.4 Encoder - Decoder Models with attention (Sequence to sequence models with attention)

The attention is a technique which tells the decoder how much to focus on the information at the given point of time. The attention of the overall input sequence adds to 1 due to the softmax function. The Encoder - Decoder mechanism with attention can be represented as in Fig.4.13.

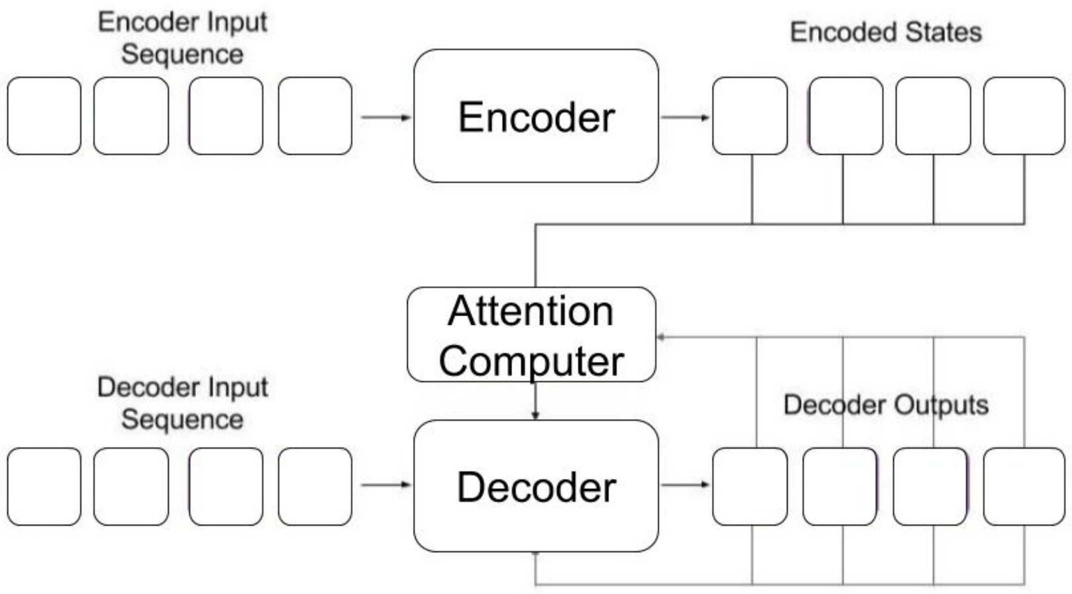


Figure 1.23: Encoder - Decoder Mechanism with Attention

The equations for the encoder-decoder model with attention using RNN as the neural network architecture is as follows:

Encoder:

$$h_t = RNN(h_{t-1}, X_t) \tag{1.19}$$

$$C_0 = h_t \tag{1.20}$$

Decoder:

$$e_{jt} = V_{attn} tanh(U * h_j + W * C_t) \tag{1.21}$$

$$\alpha_{jt} = softmax(e_{jt}) \tag{1.22}$$

$$S_t = \sum_{j=1}^{t} \alpha_{it} * h_j \tag{1.23}$$

$$C_t = RNN(C_{t-1}, [e(\hat{y}_{t-1}), S_t]) \tag{1.24}$$

Where e$_{jt}$ is the importance of jth input for decoding the tth output and α_{jt} is the focus probability on jth input with respect to tth output.

1.7 Overview of Project Report

Write it at the end.

Chapter 2

Linear optimization of Liquidity Risk Management

2.1 Analysis of Saving and Loan Accounts Data

2.2 LPP for Optimization of Assets

2.2.1 Prior Work

2.2.2 Algorithm

2.2.3 Illustration

2.2.4 Computational Complexity

2.2.5 Comparative Analysis

2.2.6 Conclusion

Chapter 3

LSTM implementation for Prediction of Stock Price

3.1 Prior Work

Prediction of stock prices is the famous problem for several years now. In the era of deep learning various researchers have done the contribution to this topic. Hao Li et al. (2018)[] proposed a Attention based Multi Input LSTM for prediction of opening price given the historical values. It uses attention mechanism for noise filtering and extracting the related information from related stocks. Yao Qin et al. (2017)[] proposed a dual attention based RNN for prediction of treading index using the historical prices of 81 different stocks. The dual attention consists of the input attention mechanism for encoder and a temporal attention mechanism for decoder. Rather et al. (2015)[] proposed the hybrid approach containing RNN and statistical models for prediction of stock price.

3.2 Algorithm

The data is collected prior from 2013 to 2018 for the historical stock price of Infosys stock named INFY. The algorithm is trained on the 90% of the data and tested on the rest of the 10% of the data. The algorithm is as follows:

- **Task:** Prediction of Stock Price

- **Data:** $\{\mathrm{X_i} : \mathrm{Open_i}, \mathrm{High_i}, \mathrm{Low_i}\ ;\ \mathrm{Y_i} : \mathrm{Close_i}\ \}_{i=1}^{N}$

- **Model:** LSTM:

$$h_t, C_t = LSTM(h_{t-1}, C_{t-1}, X_t) \tag{3.1}$$

- **Parameters:** W_f, b_f, W_i, b_i, W_c, b_c, W_o, b_o

- **Loss Function:**

$$MeanSquaredError(MSE) = \frac{1}{n}\Sigma_{i=1}^{n}(Predicted\_Y_i - Y_i)^2 \tag{3.2}$$

- **Algorithm:** Adam is the algorithm used for learning the parameters for the LSTM model.

Algorithm 2 Learning Parameters of LSTM for Prediction of Stock Price

```
1: Parameter Initialization: strategy = Uniform
2: $epoch \leftarrow Positive\_Integer$
3: while(eopch > 0):
4:         Forward Propagation of Data in Model
5:         Compute Loss using Loss Function
6:         Compute Gradients with respect to Loss
7:         Update the Parameters in Backward Propagation
8:         $epoch \leftarrow epoch - 1$
```

3.3 Illustration

The experimental setup was done in Google Colab, a facility provided by Google Inc. to students and researchers for training deep learning algorithms where they provide free GPU. The computational specification of Colab is as follows:

- CPU model name: Intel(R) Xeon(R) CPU @ 2.30GHz

- No. of Processors : 2

- CPU Cache Size : 46080 KB

- GPU Name : Tesla T4

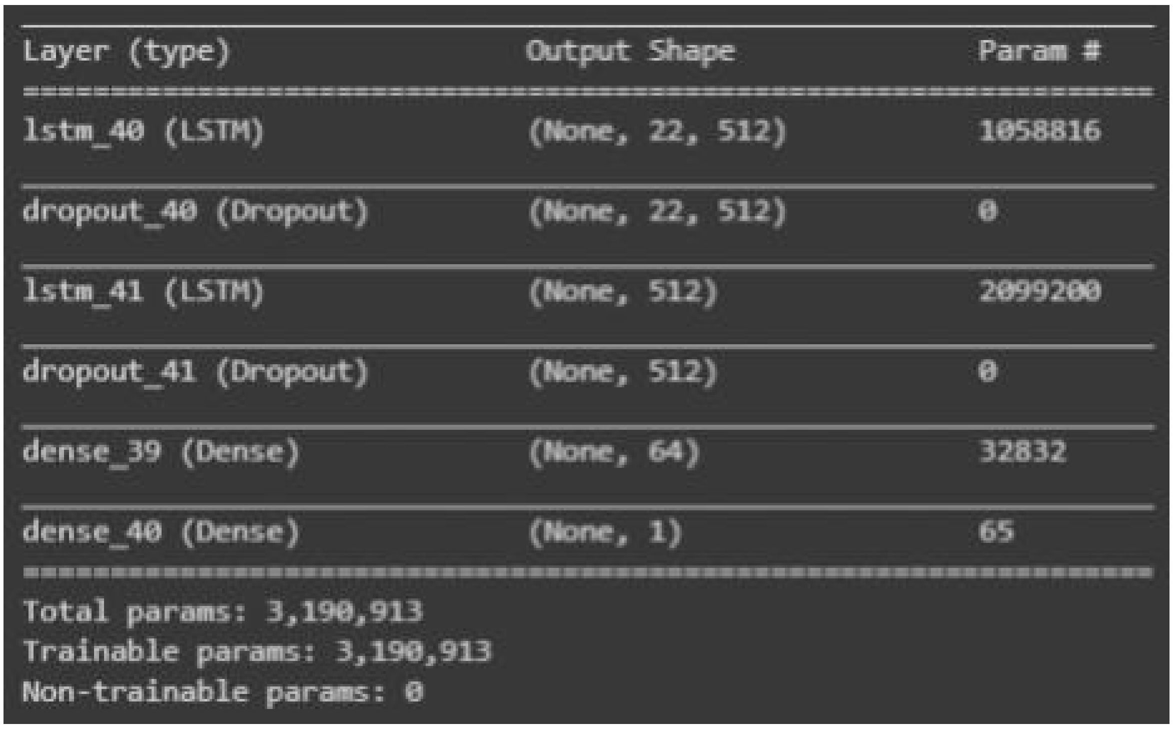
- GPU Memory : 16 GB (15079MiB usble)

- RAM : 12.9 GB

The programming language used for performing experiment is Python 3.6. The packages used for the experiment include sklearn, keras, math, time, pandas, numpy, pandas_datareader, matplotlib, h5py,statsmodels, etc.
The hyper-parameters that were fine tuned during the experiment for better performance of the model are as follows with their fine tuned values:

- seq_len = 22 # Input Window Size
- shape = [4, seq_len, 1] # No. of Features, Window, Output
- epochs = 90 # No. of times the the forward and backward propagation happens on whole training data through the model
- dropout rate = 0.3 # that percent of neurons from the model will be dropped for each epoch randomly
- decay = 0.2 # parameter decay rate for Adam optimizer
- neurons = [512, 512, 64, 1] # the different combinations of neurons is tried for optimization

The architecture of the model used for experimentation is as follows:



```
Layer (type)                 Output Shape              Param #
=================================================================
lstm_40 (LSTM)               (None, 22, 512)           1058816
_________________________________________________________________
dropout_40 (Dropout)         (None, 22, 512)           0
_________________________________________________________________
lstm_41 (LSTM)               (None, 512)               2099200
_________________________________________________________________
dropout_41 (Dropout)         (None, 512)               0
_________________________________________________________________
dense_39 (Dense)             (None, 64)                32832
_________________________________________________________________
dense_40 (Dense)             (None, 1)                 65
=================================================================
Total params: 3,190,913
Trainable params: 3,190,913
Non-trainable params: 0
```

Figure 3.1: The architecture of the LSTM model

After training the model we have tested the model on unseen test data and the MSE is as follows:

- MSE on Train Data:
- MSE on Test Data:

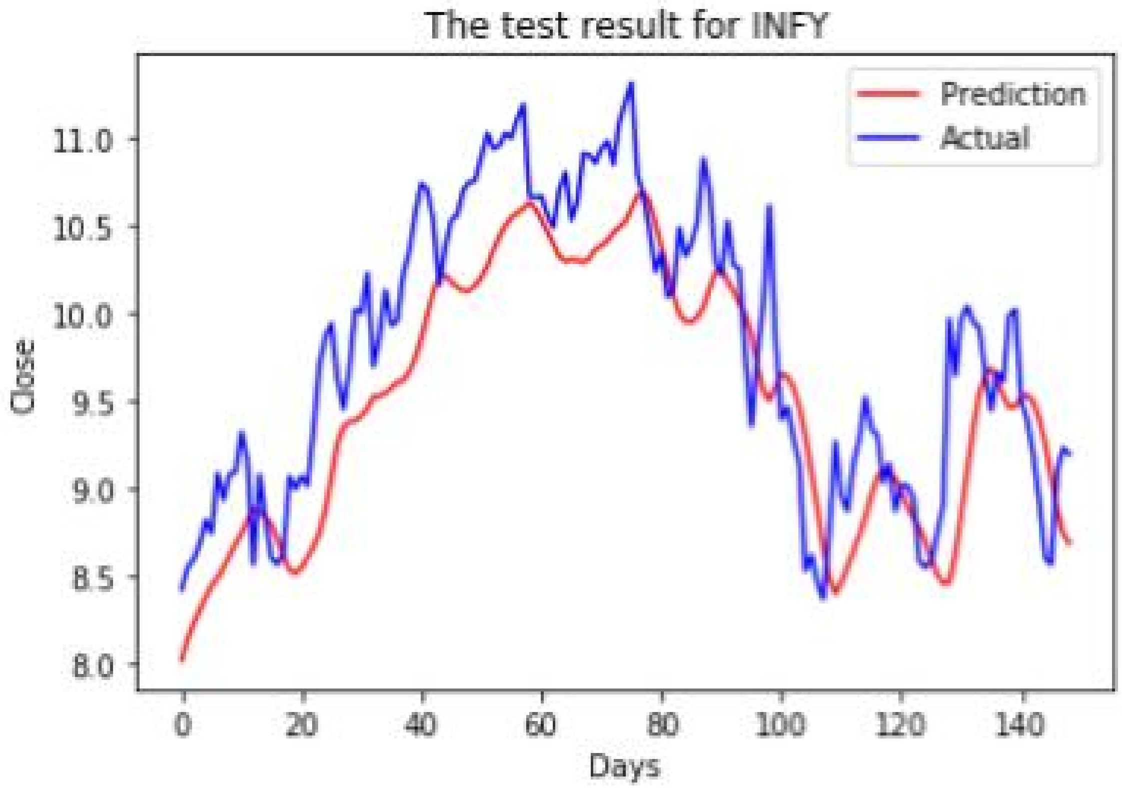


Figure 3.2: Plot showing of Actual and Predicted values

3.4 Computational Complexity

Computational complexity of LSTM is O(Z),

where Z = 4 * #IP * h + 4 * h^2 + 3 * h + h * #OP

#IP: Number of inputs

h: Number of cells in hidden layers

#OP: Number of outputs

3.5 Comparative Analysis

3.6 Conclusion

Chapter 4

Conclusion

Chapter 5

Future Work