

Data Analysis and Optimization of Asset Liability Management Problem for Banks

A dissertation submitted to the University of Hyderabad
in partial fulfillment of the degree of

Master of Technology in Artificial Intelligence

By
Gopal Kumar Kalpande
17MCMI10



School of Computer and Information Sciences
University of Hyderabad

Prof. C. R. Rao Road, Gachibowli, Hyderabad - 500 046



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

Acknowledgments	iv
Abstract	v
1 Introduction	1
1.1 Risks Associated with Banking	1
1.1.1 Operational Risk	1
1.1.2 Credit Risk	1
1.1.3 Market risk	2
1.1.4 Liquidity risk	2
1.2 Asset Liability Management (ALM)	2
1.3 ALM for Banks	3
1.3.1 ALM Information System	3
1.3.2 ALM Organization	3
1.3.3 ALM Processes	3
1.4 Project Objective	5
1.5 Problem Statement	5
1.5.1 Liquidity Risk Management	5
1.5.2 Prediction of Stock Price	6
1.6 Literature Survey	6
1.7 Overview of Project Report	6
2 Background and Related Literature	7
2.1 Time Series Analysis for Stock Price Prediction	7
2.1.1 Introduction	7
2.1.2 Issues with non-Stationarity of Data	8
2.1.3 Converting non-Stationary TS to Stationary TS	10

2.1.4	Disadvantages of Making TS stationary	14
2.1.5	Inference	14
2.2	Deep Learning Architectures for Prediction of Stock Price	14
2.2.1	Recurrent Neural Network (RNN)	14
2.2.2	Problems of Long Term Dependency	16
2.2.3	Long Short Term Memory (LSTM)	16
2.2.4	Encoder - Decoder Models (Sequence to sequence models) . . .	22
2.2.5	Encoder - Decoder Models with attention (Sequence to sequence models with attention)	23
3	Proposed Algorithms for Overlapping Community Discovery	25
3.1	Motivation	25
3.2	Algorithm based on Consensus Clustering	26
3.2.1	Implementation on Zachary karate club	27
3.2.2	Zachary network figures for various community discovery algorithms	28
3.3	Algorithm based on Connectivity Consensus	33
3.3.1	Implementation on Zachary karate club	34
3.3.2	Conclusion	35
4	Implementation and Results	37
4.1	Datasets	37
4.1.1	Benchmark Datasets	38
4.1.2	Synthetic Datasets	38
4.2	Results on Benchmark Datasets	39
4.3	Results on Synthetic Datasets	42
4.4	Conclusion	46

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:

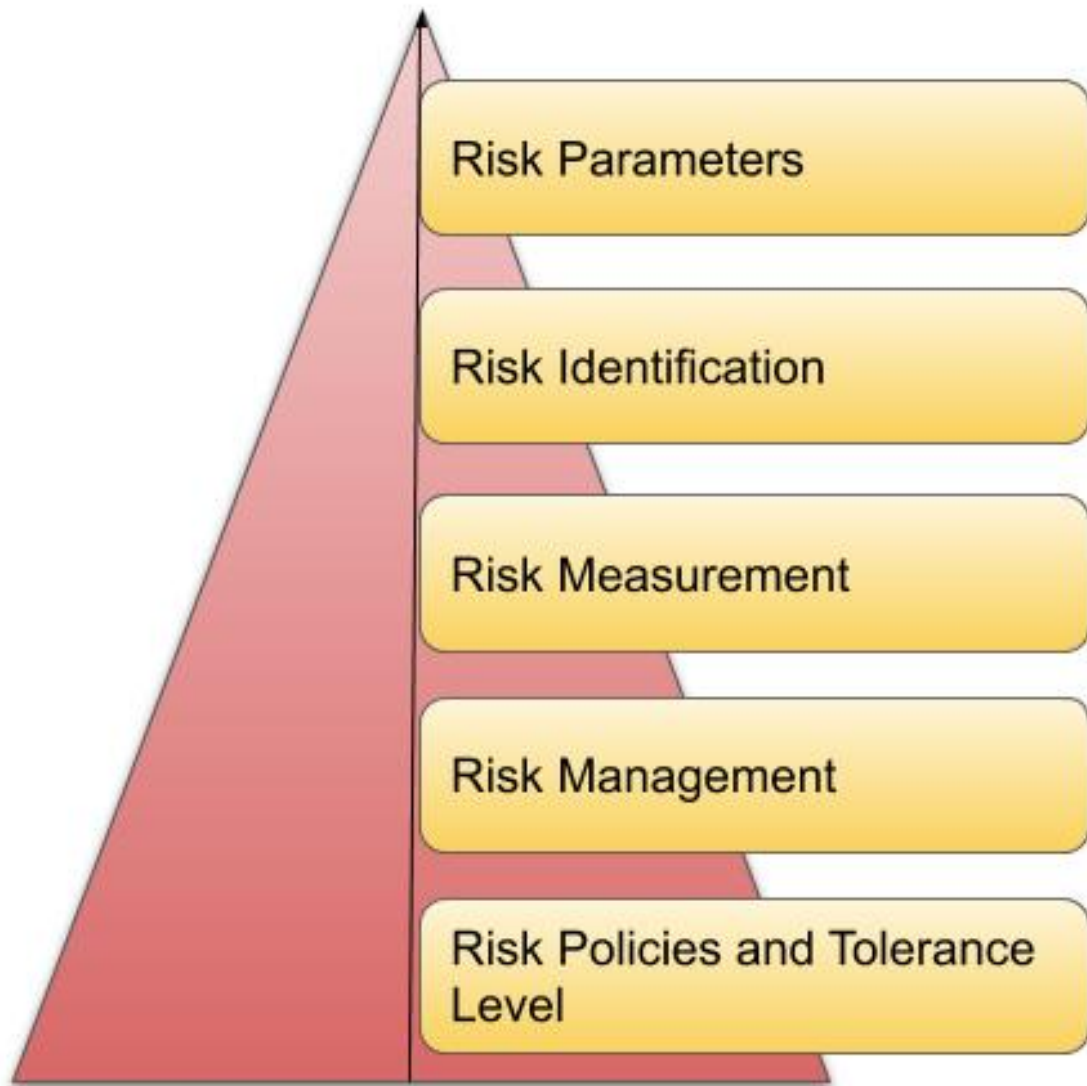


Figure 1.1: ALM Processes

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

Liabilities	Assets
<ul style="list-style-type: none">• Capital• Reserve & Surplus• Deposits• Borrowings	<ul style="list-style-type: none">• 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:

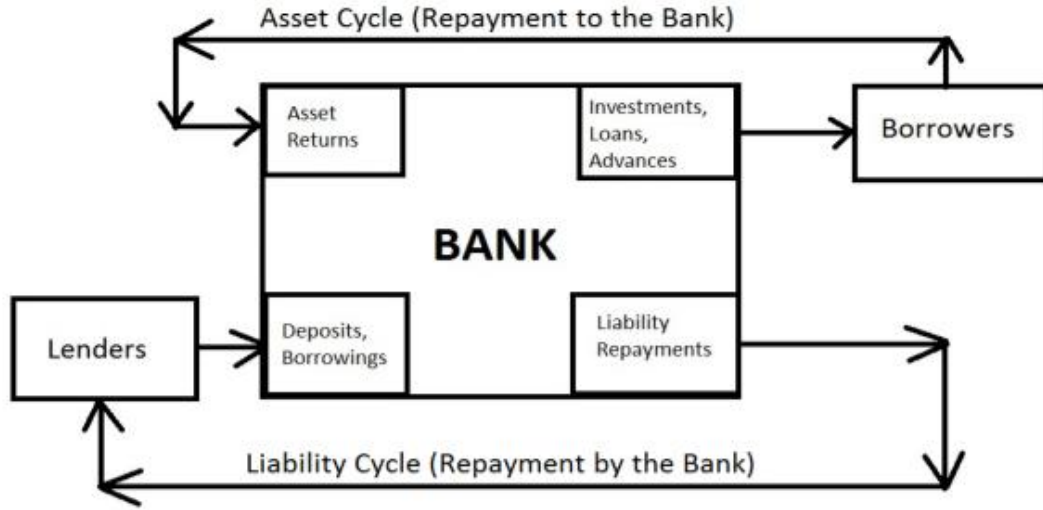


Figure 1.3: ALM Cycle of a Bank

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[].

1.7 Overview of Project Report

Write it at the end.

Chapter 2

Background and Related Literature

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.

2.1 Time Series Analysis for Stock Price Prediction

2.1.1 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 2.1: Ex of Time Series data of a stock

2.1.2 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

2.1.2.0.1 Plotting the Rolling Statistics

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

	Close	moving_average	moving_variance
Date			
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 2.2: Moving Average and Moving Variance of the Closing Price with window size = 5



Figure 2.3: 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.

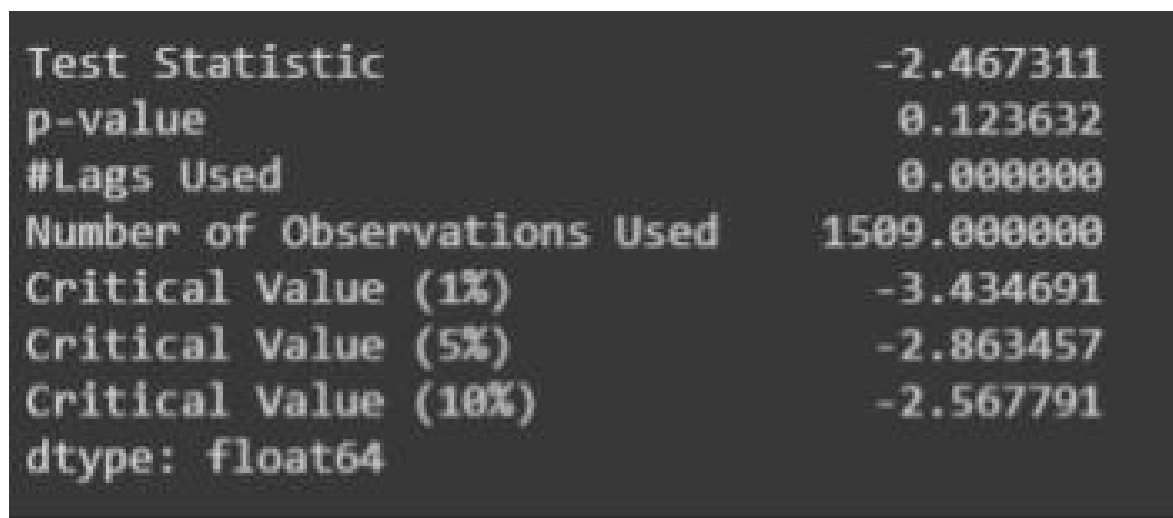
2.1.2.0.2 Dickey-Fuller Test (DF Test)

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 2.4: 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.

2.1.3 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:

2.1.3.0.1 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 2.5: 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.

2.1.3.0.2 Decomposing

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

$$Data_t = Trend_t + Seasonality_t + Residual_t \quad (2.1)$$

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

$$Data_t = Trend_t * Seasonality_t * Residual_t \quad (2.2)$$

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 - seasonal period or frequency MA - moving average DT_t - Detrended series

1: If m is even number then

$$Trend_t = 2 * m - MA \quad (2.3)$$

Else

$$Trend_t = m - MA \quad (2.4)$$

2: Compute Detrended series:

$$DT_t = Data_t - Trend_t \quad (2.5)$$

3: The seasonal component $Seasonality_t$ of the respective season is the average of the DT_t of the given season.

4:

$$Residual_t = Data_t / (Trend_t * Seasonality_t) \quad (2.6)$$

The plot looks as follows:

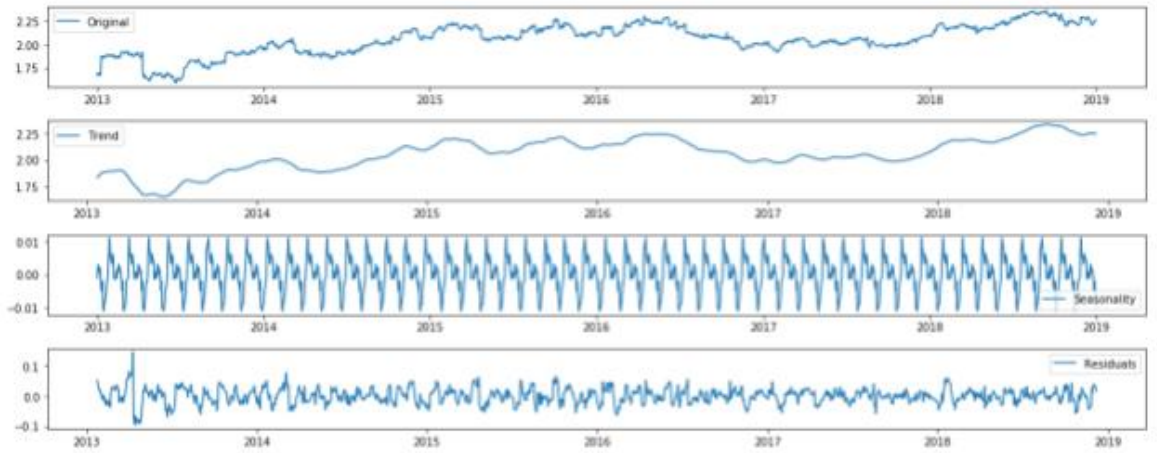


Figure 2.6: The trend, seasonality and residual of log-transformed data

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

```

Results of Dickey-Fuller Test:
Test Statistic          -1.056428e+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 2.7: 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.

2.1.3.0.3 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 2.8: 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.

2.1.4 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.

2.1.5 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.

2.2 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: X_i : Source_i ; Y_i : Target_i And the loss function: Mean Squared Error Loss as the problem is a regression problem.

2.2.1 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.

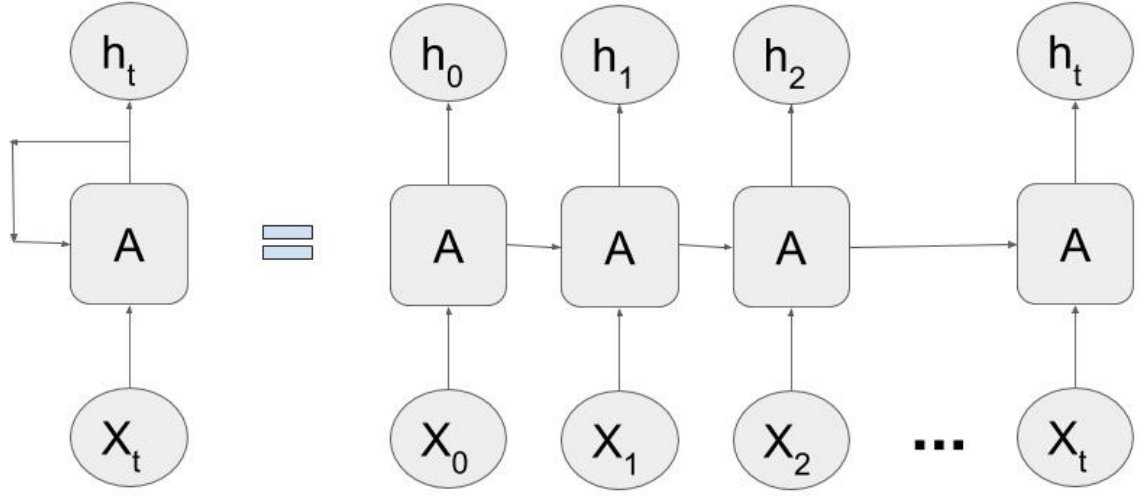


Figure 2.9: Unfolded Recurrent layer

A is the Recurrent layer of neurons, X_t is the input sequence and h_t 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:

$$h_t = \tanh(w_i * X_t + w_o * h_{t-1} + b) \quad (2.7)$$

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

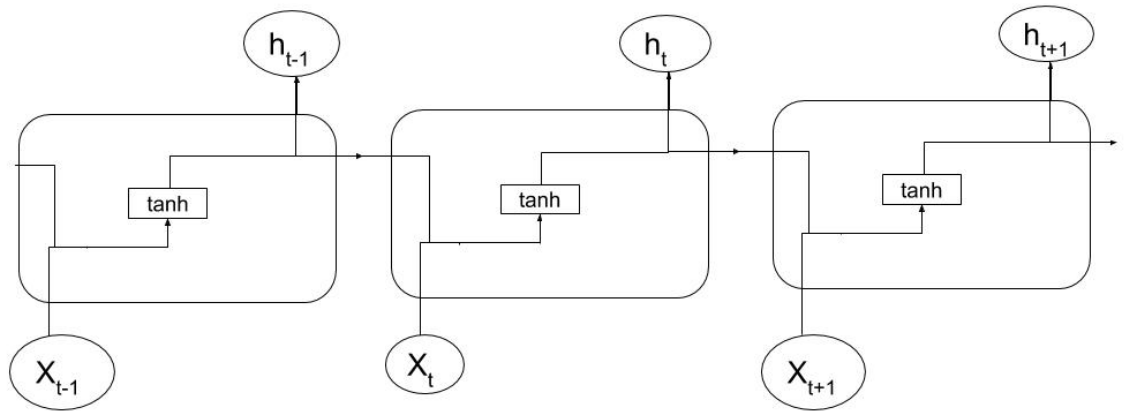


Figure 2.10: Unfolded Recurrent layer with activation function

2.2.2 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 h_{t+1} depends on the input X_1 then RNN can't handle this dependency. The solution to this is the improved version of RNN discussed later.

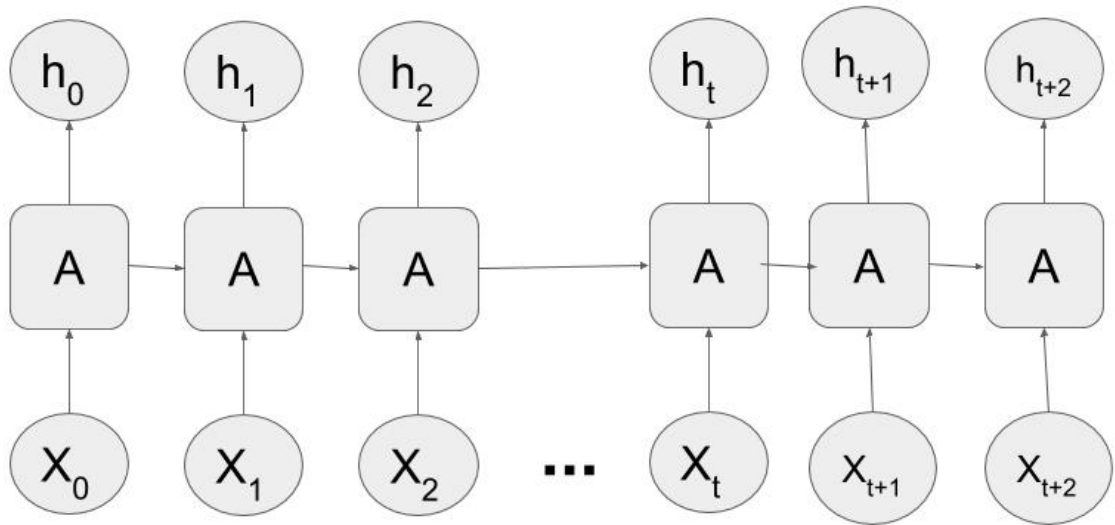


Figure 2.11: Long term dependency problem

2.2.3 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:

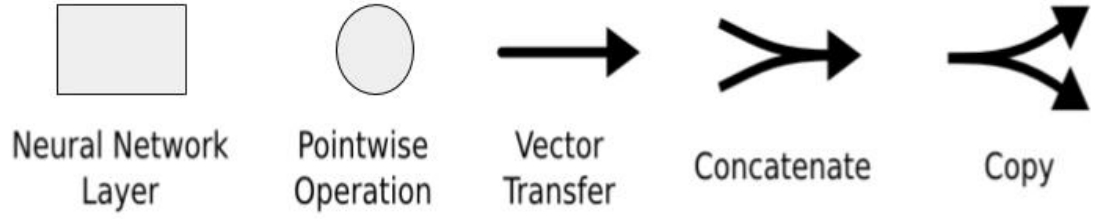


Figure 2.12: Notations

The comparison of RNN and LSTM is as follows:

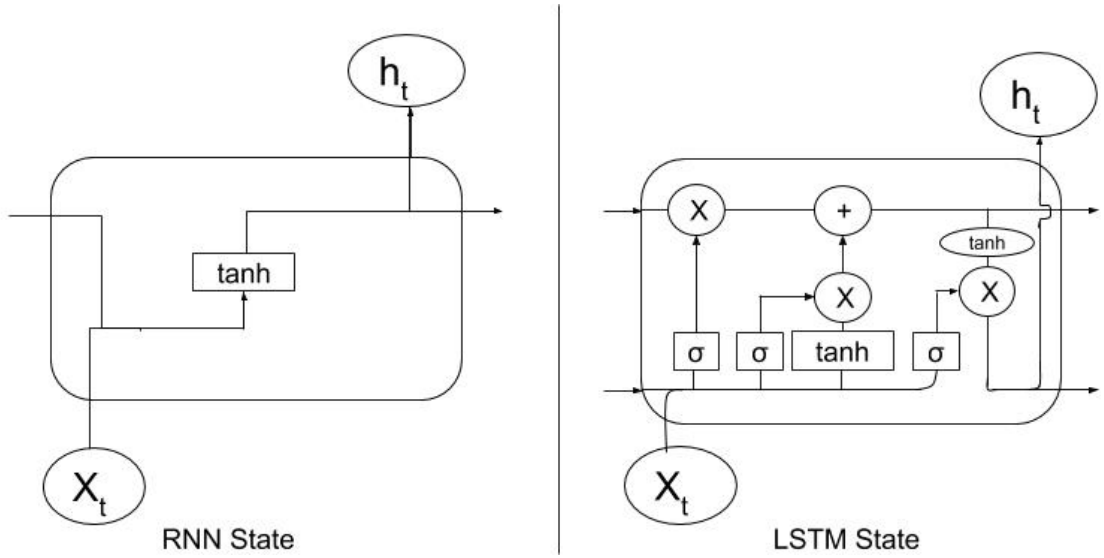


Figure 2.13: Comparison of RNN and LSTM State

2.2.3.0.1 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).

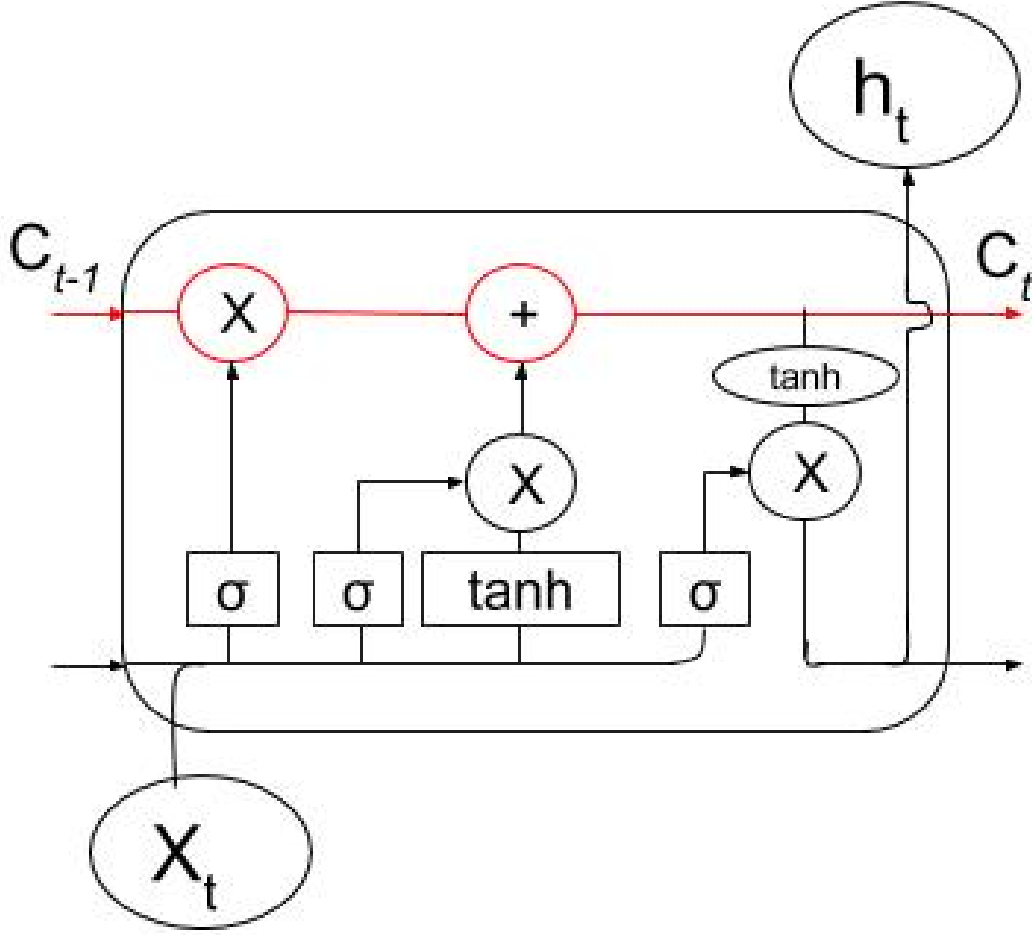


Figure 2.14: Cell State

2.2.3.0.2 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:

$$f_t = \sigma(W_f \cdot [h_{t-1}, X_t] + b_f) \quad (2.8)$$

- 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 gate terms in $[]$ reflects concatenation operation.

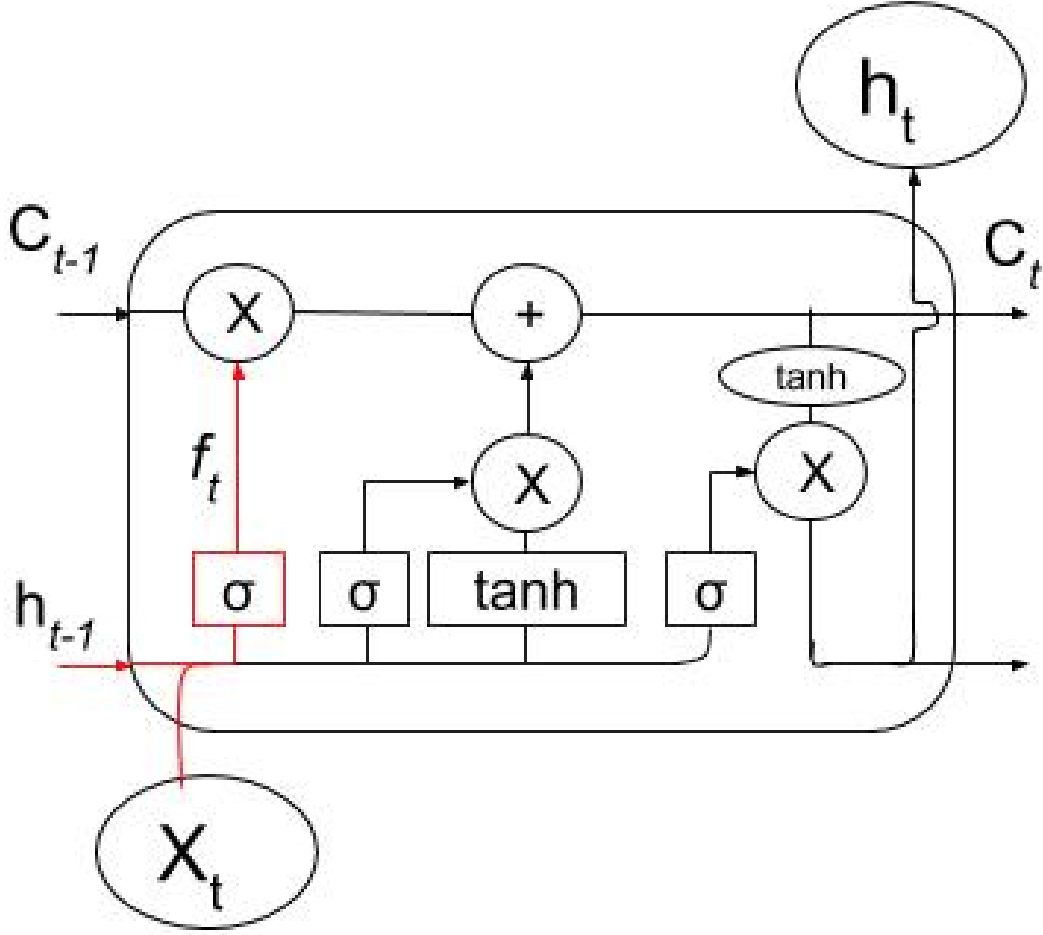


Figure 2.15: Forget Gate

2.2.3.0.3 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:

$$i_t = \sigma(W_i.[h_{t-1}, X_t] + b_i) \quad (2.9)$$

$$\tilde{I}_t = \tanh(W_c.[h_{t-1}, X_t] + b_c) \quad (2.10)$$

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.

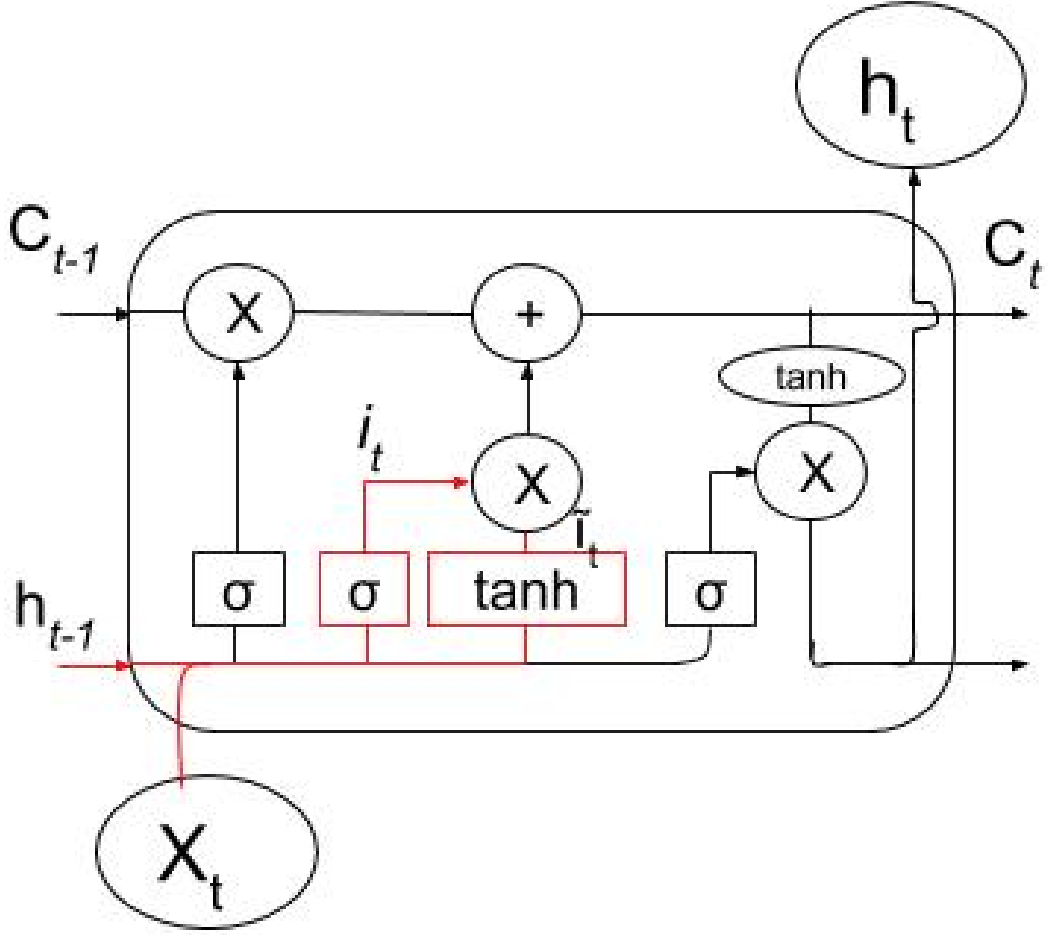


Figure 2.16: Input Gate

2.2.3.0.4 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:

$$C_t = f_t * C_{t-1} + i_t * \tilde{I}_t \quad (2.11)$$

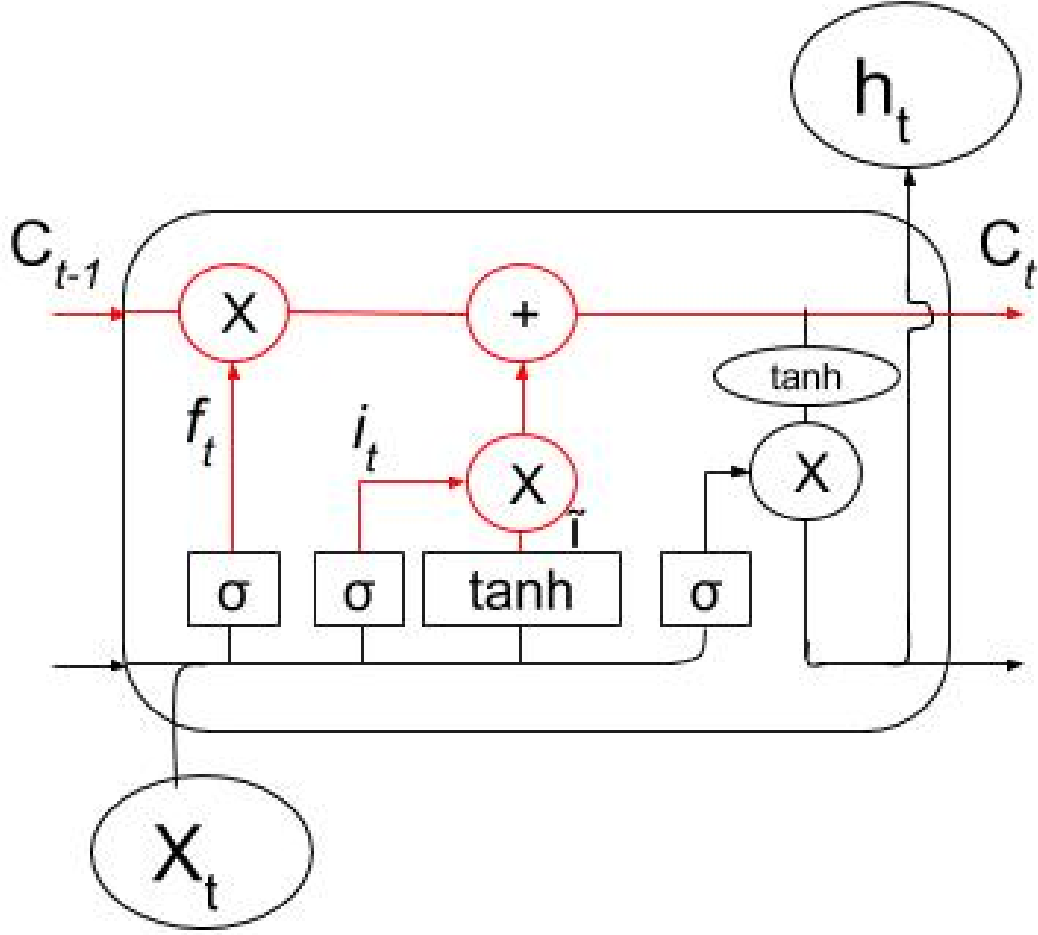


Figure 2.17: Update to new cell state

2.2.3.0.5 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:

$$o_t = \sigma(W_o \cdot [h_{t-1}, X_t] + b_o) \quad (2.12)$$

$$h_t = o_t * \tanh(C_t) \quad (2.13)$$

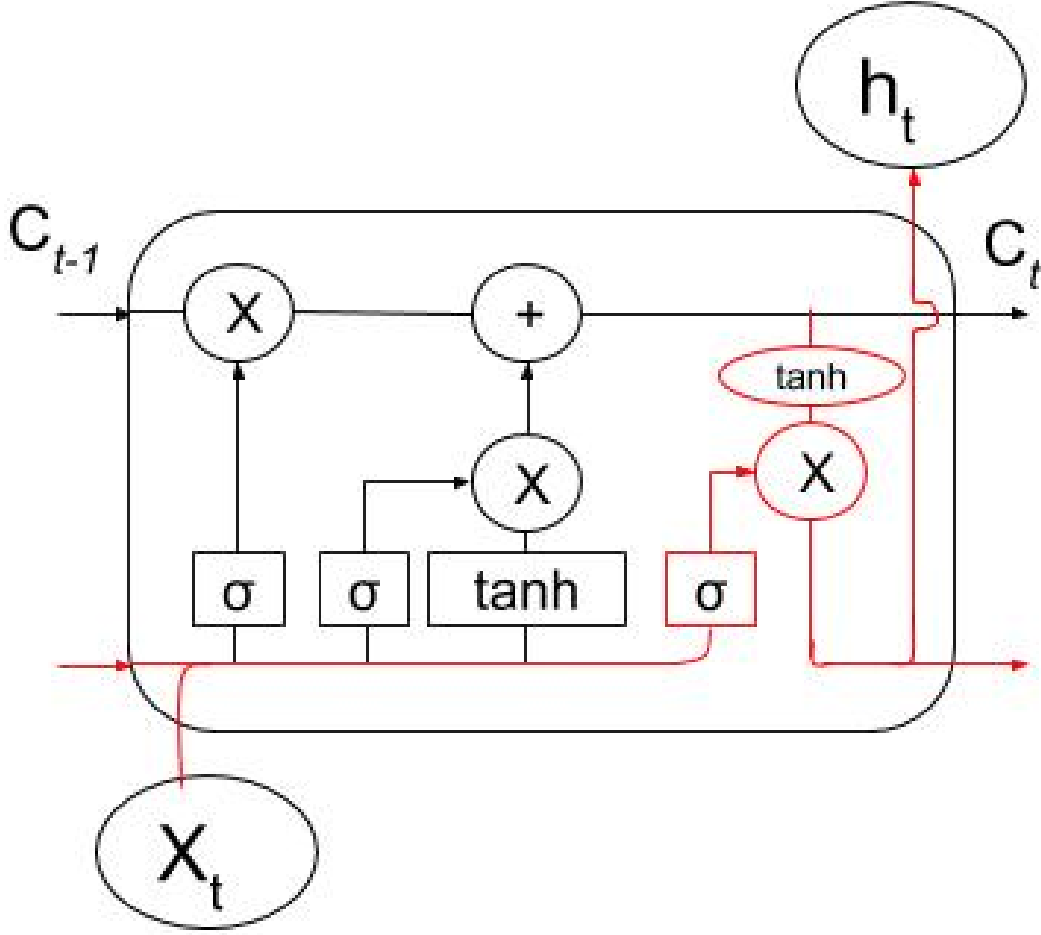


Figure 2.18: Output Gate

The Fig. below shows the unfolded LSTM layer.

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

For RNN:

$$h_t = RNN(h_{t-1}, X_t) \quad (2.14)$$

For LSTM:

$$h_t, C_t = LSTM(h_{t-1}, C_{t-1}, X_t) \quad (2.15)$$

2.2.4 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:

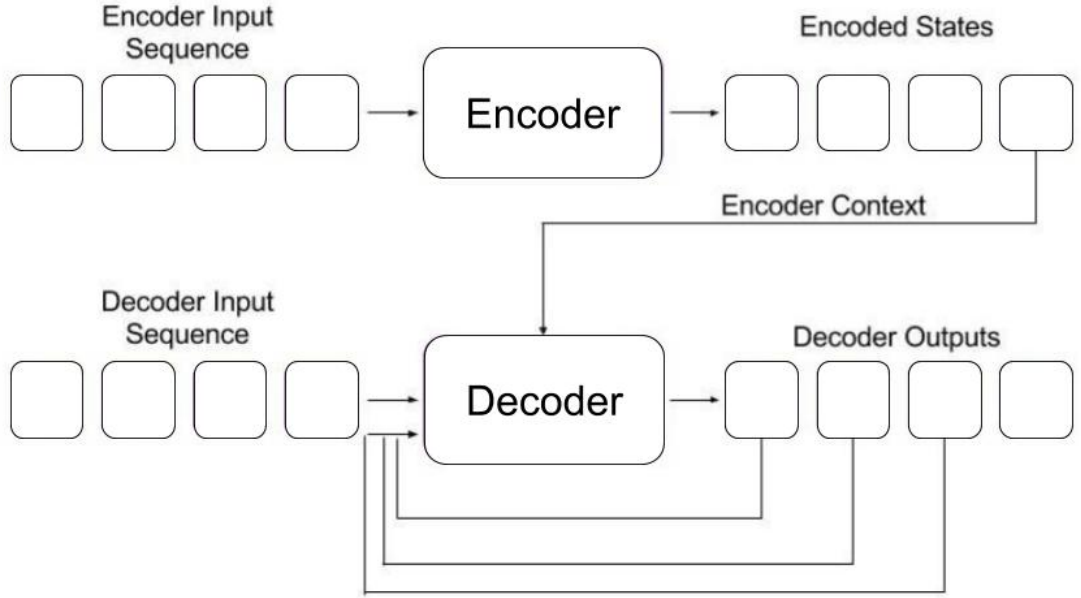


Figure 2.19: Encoder - Decoder Mechanism

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

Encoder:

$$h_t = RNN(h_{t-1}, X_t) \quad (2.16)$$

Decoder:

$$C_0 = h_t \quad (2.17)$$

$$C_t = RNN(C_{t-1}, [h_{t-1}, e(\hat{y}_{t-1})]) \quad (2.18)$$

2.2.5 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.

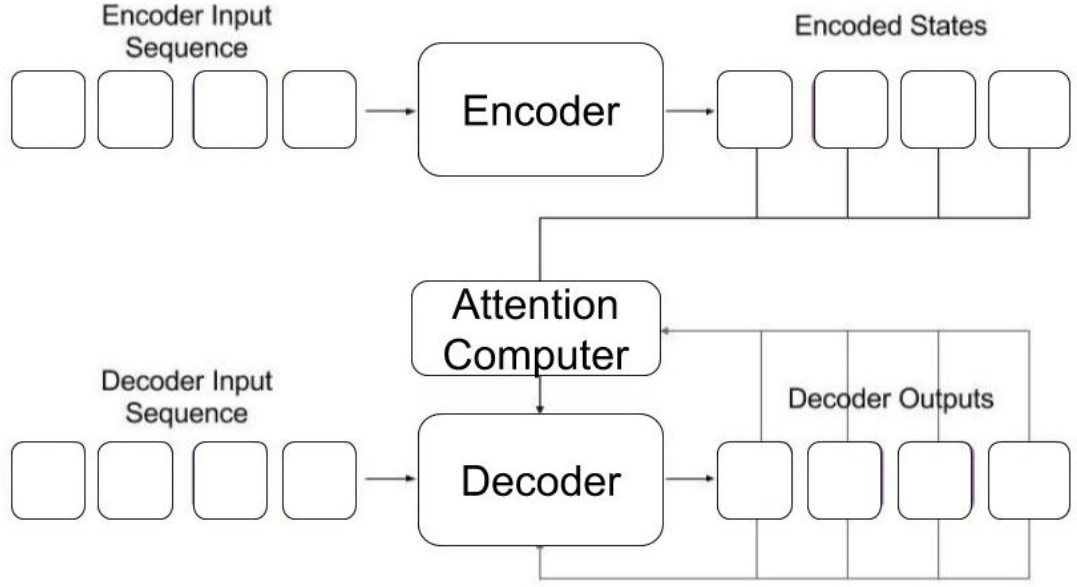


Figure 2.20: Encoder - Decoder Mechanism with Attention

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

Encoder:

$$h_t = RNN(h_{t-1}, X_t) \quad (2.19)$$

$$C_0 = h_t \quad (2.20)$$

Decoder:

$$e_{jt} = V_{attn} \tanh(U * h_j + W * C_t) \quad (2.21)$$

$$\alpha_{jt} = \text{softmax}(e_{jt}) \quad (2.22)$$

$$S_t = \sum_{j=1}^t \alpha_{jt} * h_j \quad (2.23)$$

$$C_t = RNN(C_{t-1}, [e(\hat{y}_{t-1}), S_t]) \quad (2.24)$$

Where e_{jt} is the importance of j^{th} input for decoding the t^{th} output and α_{jt} is the focus probability on j^{th} input with respect to t^{th} output.

Chapter 3

Proposed Algorithms for Overlapping Community Discovery

We adopt the approach of Consensus Clustering to detect overlapping communities. In the context of clustering of nodes, different ways of clustering of nodes may be obtained by different algorithms. Also, as an internal parameter like K in K -Means algorithm is changed, different clusterings are obtained. In order to choose a single clustering that agrees most with the other clusterings, the idea of consensus clustering was proposed by [?].

Consensus clustering is an NP-hard problem. There are many approximation algorithms proposed in the literature like Best-of- K , Majority rule, Best One Element Move, average linkage etc [?].

3.1 Motivation

We adopt these ideas behind consensus clustering to the context of overlapping community discovery. Since existing community discovery algorithms attach a community label to each node, can we check if majority of algorithms agree that a pair of nodes belongs to a specific community then those two nodes belong to that community. The nodes (of degree > 1) for which the algorithms do not form a consensus are considered to be overlapping nodes.

3.2 Algorithm based on Consensus Clustering

A node in a social network either may be embedded within the core of a community or the nodes may be either on the periphery of a community or may belong to more than one community. In order to distinguish the location of a node, we propose the Algorithm 2, in which we define the core of a community as nodes that are co-clustered by all the algorithms as belonging to one single community. If algorithms do not have consensus on cluster labels of two nodes, then there exists some ambiguity regarding the membership of these nodes. These nodes are potentially in the overlapping regions if they are not isolated or on the periphery (of degree 1).

Algorithm 2 Overlapping community discovery using consensus clustering.

Input: Community labels obtained from k community discovery algorithms on a graph $G = (V, E)$, $|V| = n$, $|E| = m$

Output: overlapping nodes.

```

1: read the input file and save as  $A(i, j)$  =community labels given to node  $i$  by the
   algorithm  $j$ 
2:  $M(i, j) = 0$  for each  $i, j = 1, 2, \dots, n$ 
3: for  $l = 1, 2, \dots, k$  do
4:   if  $A(i, l) = A(j, l)$  then
5:      $count(i, j)++$ 
6:   end if
7:   if  $count = k$  then
8:      $M(i, j) = 1$ 
9:   end if
10: end for
11: for  $i = 1, 2, \dots, n$  do
12:    $G_i = \{i\}$ 
13:   for  $j = 1, 2, \dots, n$  do
14:     if  $M(i, j) = 1$  then
15:        $G_i = G_i \cup \{j\}$ 
16:     end if
17:   end for
18: end for
19: for  $i = 1, 2, \dots, n$  do
20:   if  $length(G_i) = 1 \ \&\& \ degree(i) > 1$  then
21:     node  $i$  is overlapping node
22:   end if
23: end for

```

Time complexity:

In this algorithm it has to check labels of each node with all other nodes for all the given algorithms that takes nk time, and this checking should happen for all the remaining

nodes also with all other nodes. Total time will be taken by this algorithm is $\mathcal{O}(n^2k)$.

3.2.1 Implementation on Zachary karate club

Zachary Karate club dataset is a graph with members of a university karate club as nodes and interactions between the members as edges and has been collected by Wayne Zachary. It has 34 nodes and 78 edges. We choose seven community discovery algorithms for computing the consensus on clustering. The algorithms of Edge betweenness, Fastgreedy, Infomap, Multi level, Leading eigenvector, Optimal, Walktrap algorithm available in a software package called *RStudio* are executed. We have not considered Label propagation algorithm and Spinglass algorithm since they are not found to be stable and community labels are getting changed in different executions. A snapshot of the community labels assigned to the nodes by these algorithms is given in Table 3.1.

Node	EBC	FG	INF	LEV	ML	OPT	WT
1	1	1	1	2	1	2	1
2	1	3	3	2	1	2	1
3	2	3	3	2	1	2	2
4	1	3	3	2	1	2	1
5	3	1	1	1	2	3	5
6	3	1	1	1	2	3	5
7	3	1	1	1	2	3	5
.
.
.
34	4	2	2	4	3	1	3

Table 3.1: Zachary nodes & its community labels

In the above table first column represents node numbers of Zachary network and remaining seven columns represent community labels of that algorithm.

EBC: Edge betweenness centrality algorithm

FG : Fastgreedy algorithm

INF: Infomap algorithm

LEV: Leading eigenvector algorithm

ML : Multi level algorithm

OPT: Optimal algorithm

WT : Walktrap algorithm

Actually there exists only two disjoint communities in the Zachary network in the ground truth [?].

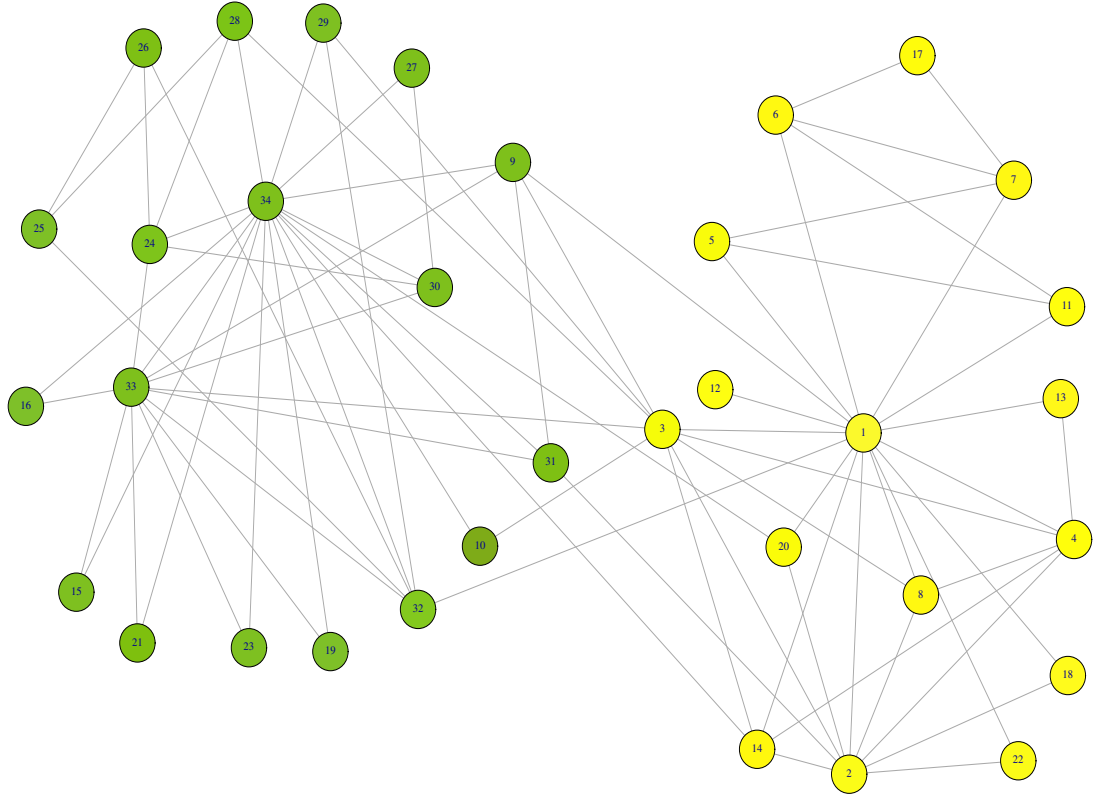


Figure 3.1: Zachary network with two communities represented in green colour and yellow colour [?]

In the above figure we can see two communities with green colour and yellow colour. There are no overlapping nodes as of now, but in some of the papers says that there are two nodes which are in overlapping they are, node 3 and node 10.

3.2.2 Zachary network figures for various community discovery algorithms

In this section we look at seven figures [?] of Zachary network with different communities coloured differently. Each figure is an output of different community discovery algorithm that is mention in the figure caption.

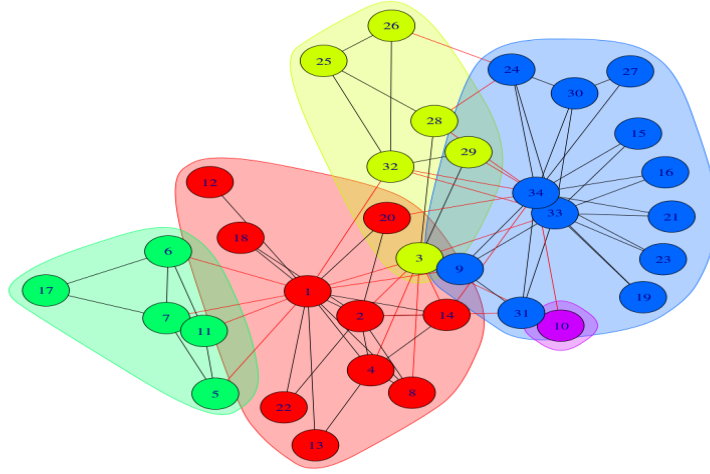


Figure 3.2: Zachary network with communities according to EBC

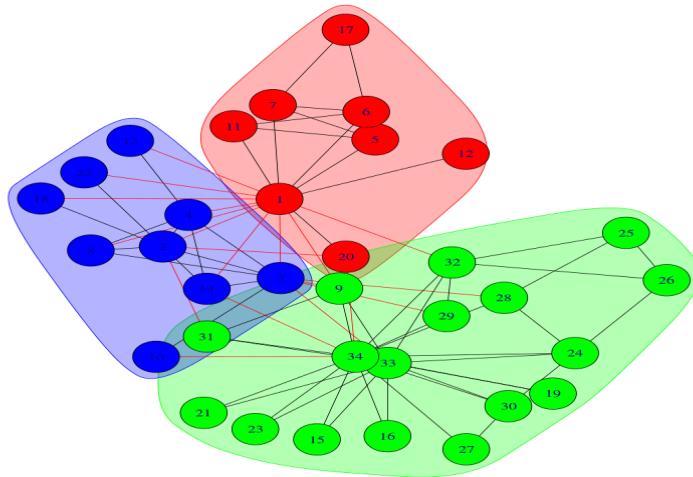


Figure 3.3: Zachary network with communities according to FG

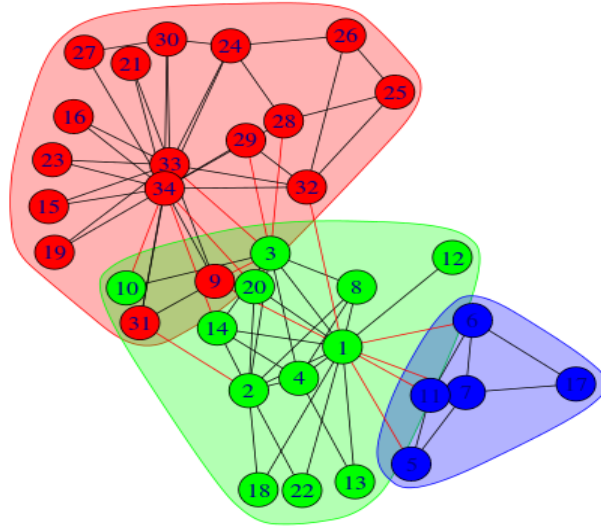


Figure 3.4: Zachary network with communities according to INF

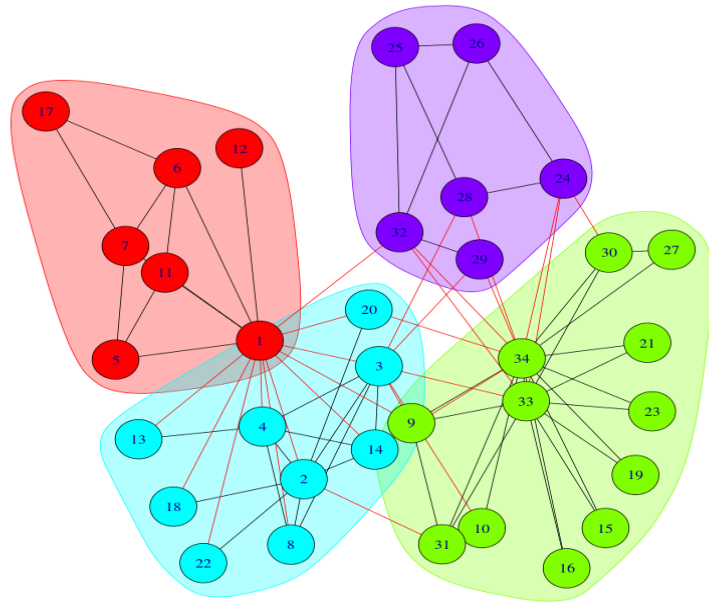


Figure 3.5: Zachary network with communities according to LEV

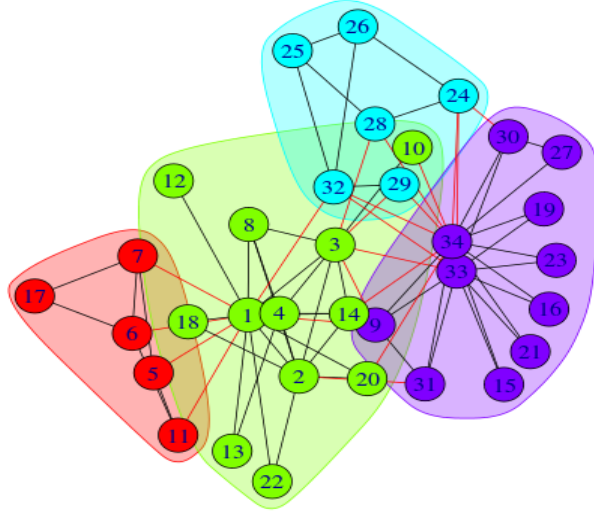


Figure 3.6: Zachary network with communities according to ML

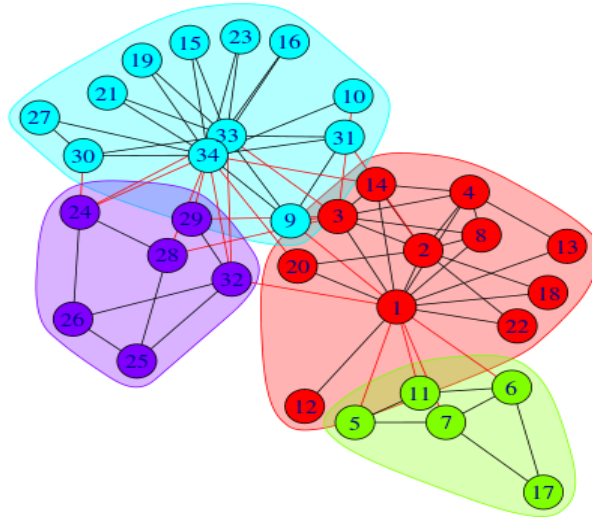


Figure 3.7: Zachary network with communities according to OPT

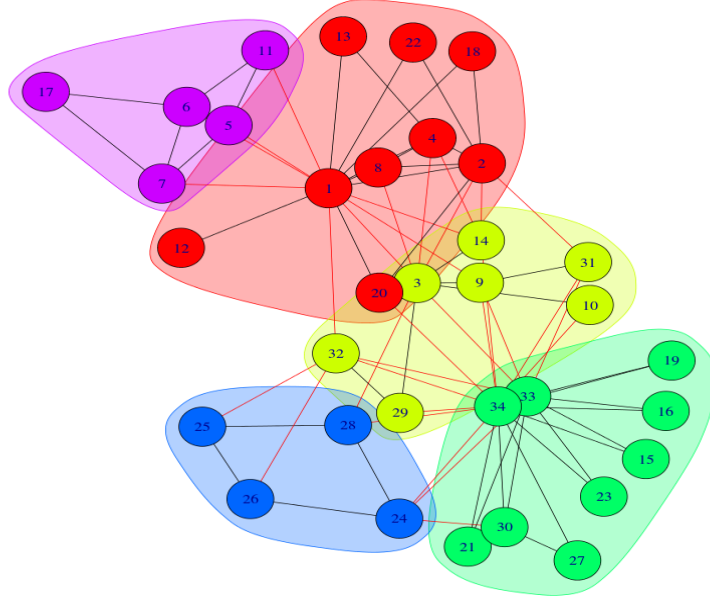


Figure 3.8: Zachary network with communities according to WT

As it is explained in the algorithm 2, the input file contains data in the format of $nodes \times algorithms$. Consider nodes one and two, their labels in FG and INF is different and remaining all other algorithms are the same. It means nodes one and two are co-clustered in five algorithms out of seven algorithms. In the same way nodes one and three are co-clustered in three algorithms out of seven algorithms. Consider nodes five, six and seven whose labels in all the algorithms are same. This means that all these three nodes are co-clustered by all algorithms (i.e. 100% of the algorithms) and hence can be considered as 100% consensus. In the same way each pair of nodes considered and checked whether they are co-clustered in all algorithms or some percentage of algorithms (could be adjusted based on the requirement). All these co-clustered pairs will be written into a file and make groups of all the nodes which are co-clustered. If any node is not co-clustered with any other node and its degree is greater than one then this node will be considered as an overlapping node. This algorithm 2 is implemented on Zachary network and five nodes (3,10,14,20,24) emerge as overlapping nodes, because some of the algorithms divide this network into more than two communities that is shown in the section. Note that in Figure 3.9 these five nodes can be seen clearly to be overlapping nodes.

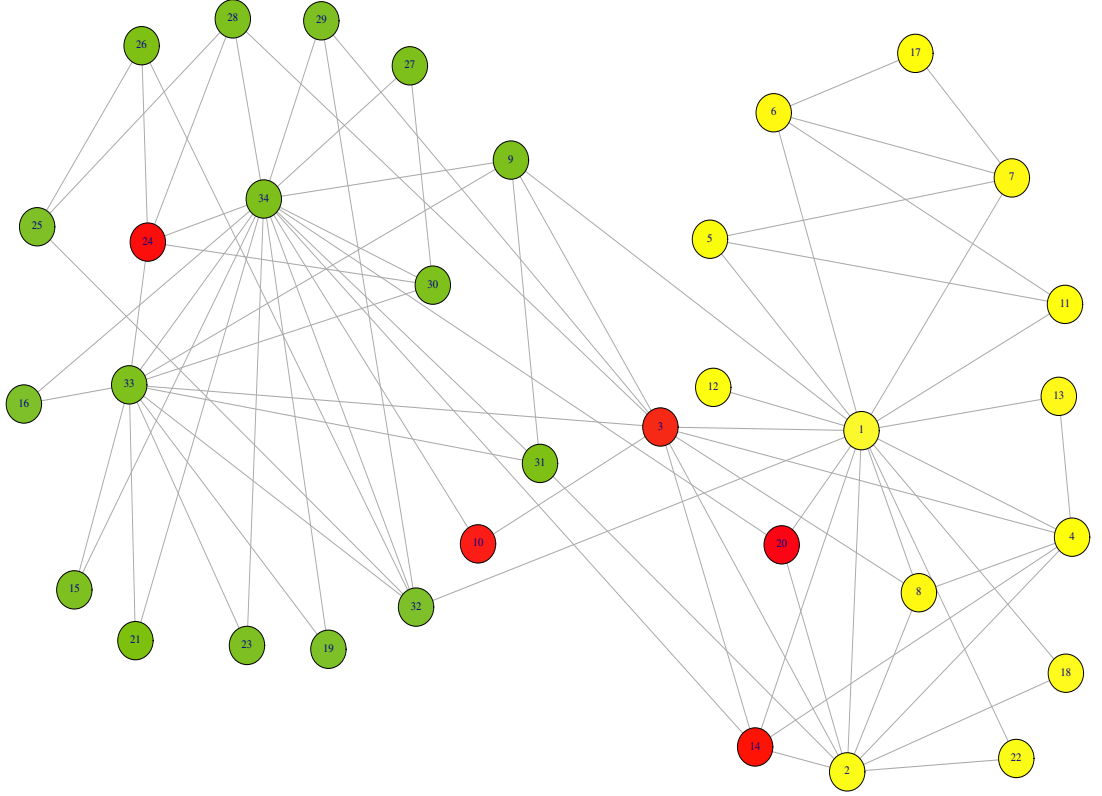


Figure 3.9: Zachary network with five overlapping nodes represented in red colour according to the above algorithm 2

In the above figure we can observe that there are five overlapping nodes (3,10,14,20,24) represented with red colour. The image source is [?]

3.3 Algorithm based on Connectivity Consensus

Unlike the previous algorithm which detects the cores of communities and annotates the rest as overlapping, here we detect directly the overlapping nodes. The intuition behind this algorithm is that an overlapping node is most probably connected to neighbours that belong to different communities. Here again the 'overlappingness' of a node is decided based on the consensus arrived at by majority of the algorithms. Zachary network nodes and its community labels are described in the section 3.2.1.

Step1: For a fixed algorithm for a node a if majority of its neighbours of node a belong to different communities.

Step2: If a node a and its neighbours do not belong to the same community by majority of the algorithms then it is to be considered that the node a is overlapping

node.

Algorithm 3 Overlapping community discovery using connectivity consensus.

Input: graph $G(V, E)$ edge list and a file consists community labels of each algorithm.

Output: overlapping nodes.

```

1:  $|V| = n, |E| = m, k =$  number of algorithms and  $edge(i, j) =$  edge between nodes
    $i$  and  $j$ .
2: calculate degree of each node.
3: for  $i = 1$  to  $n$  do
4:    $c = 0, count_1 = 0, count_2 = 0, \dots, count_k = 0$ .
5:   for  $j = 1$  to  $n$  do
6:     if  $edge(i, j) = TRUE$  then
7:       for  $l = 1$  to  $k$  do
8:         if  $label_l(i) \neq label_l(j)$  then
9:            $count_l ++$ 
10:        end if
11:      end for
12:    end if
13:  end for
14:  for  $l = 1$  to  $k$  do
15:    if  $degree(i) > 1 \&\& count_l \geq \lceil degree(i)/2 \rceil$  then
16:       $c ++$ 
17:    end if
18:  end for
19:  if  $c \geq \lceil k/2 \rceil$  then
20:    print  $i$ .
21:  end if
22: end for

```

Time complexity:

In this algorithm it has to check labels of all neighbours of one node. A node can have a maximum of $n - 1$ neighbours. To check one node's all neighbours takes $n - 1$ time, this checking should happen for all the nodes and also for all the algorithms. Therefore total time taken by this algorithm is $\mathcal{O}(n^2k)$.

3.3.1 Implementation on Zachary karate club

As explained in the algorithm, input file contains data in the format $nodes \times algorithms$. Take every individual node and check the community labels of all its neighbours for a fixed algorithm. If a minimum of 50% of its neighbours have different community labels (i.e. not co-clustered) then this node is potentially an overlapping node. Now check for all the remaining algorithms for consensus. Finally, among these, only the nodes whose degree is greater than one are considered to be overlapping nodes.

This approach when implemented on Zachary network gave again the five nodes (3,10,24,28,32) as overlapping nodes. Note that in Figure 3.10 these five nodes can be seen clearly to be overlapping nodes.

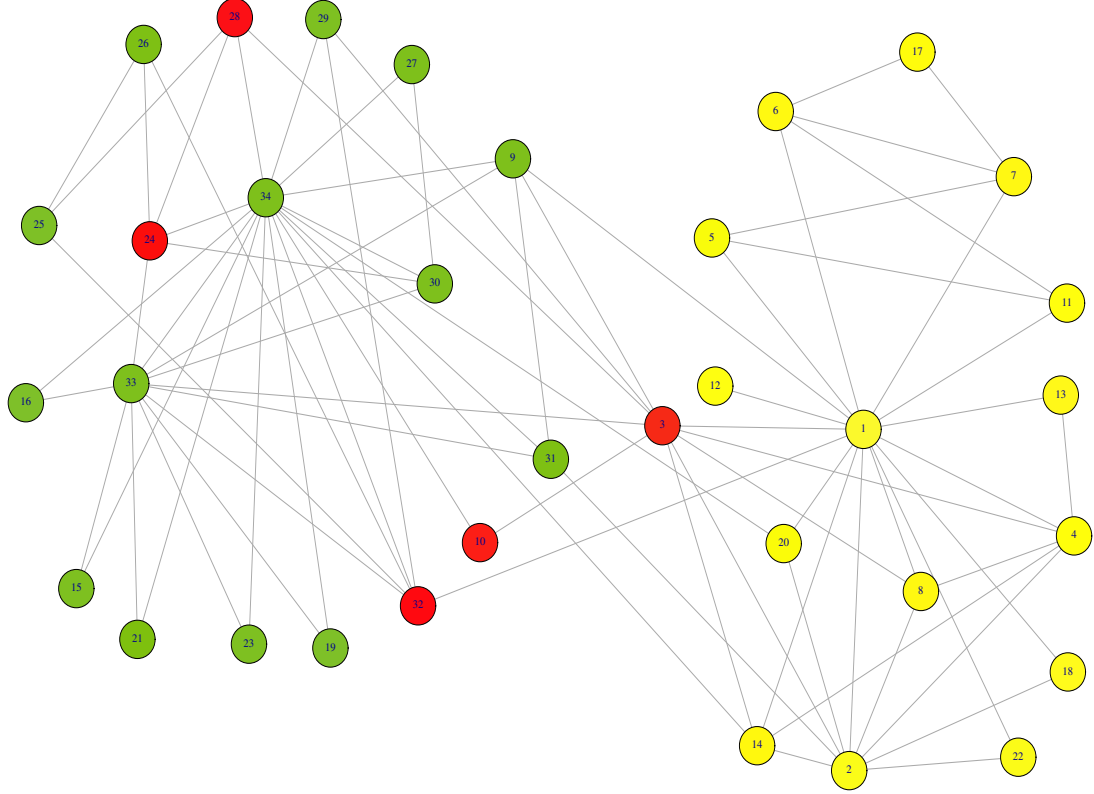


Figure 3.10: Zachary network with five overlapping nodes represented in red colour according to the above algorithm [?]

In the above figure we can observe that there are five overlapping nodes (3,10,24,28,32) represented with red colour.

3.3.2 Conclusion

Two novel algorithms for overlapping community discovery have been proposed in this chapter based on the idea of consensus clustering. As a preliminary checking, the algorithms have been implemented and tested on a small but popular benchmark data set called Zachary karate club which is known to have two communities. It should be noted that there are no benchmark data sets available for checking overlapping communities. Visually we can see that at least two nodes 3 and 10 may be considered to be overlapping for Zachary karate club. We would like to check this hypothesis with our algorithms.

The Algorithm 2 gave (3,10,14,20,24) and Algorithm 3 gave (3,10,24,28,32) as overlapping nodes of the communities. Both the algorithms detect 3 and 10 as overlapping nodes satisfying the prediction. Newman's edgebetweenness algorithm outputs four communities as seen in figure 3.2, with all the seven nodes 3, 10, 14, 20, 24, 28 and 32 detected by the two algorithms in the overlapping regions of a pair of communities. 3 and 10 are clearly between the major communities but 24 seems to be in the overlapping region of communities coloured by blue and yellow. If we observe in the figures 3.2-3.8, the communities of LEV, ML, OPT and WT node number 24 is in the boundary of that community and sharing majority of its connection with other communities as shown by four out of seven algorithms. From this explanation we can conclude that three nodes (3, 10, 24) can certainly be designated as overlapping.

The algorithms do give satisfactory results. We need to implement these algorithms on larger bench mark data sets for validation which is carried out in the next chapter.

In the next chapter we implement these algorithms on three benchmark datasets and three synthetic datasets.

Chapter 4

Implementation and Results

Implementation of the proposed overlapping community discovery algorithms, Algorithm 2 & algorithm 3 is carried out on three benchmark social network datasets namely Zachary, Collaboration and Dolphin. Though analysis of Zachary network has been done in the previous chapter, we place it here with more details and for completeness sake. Since overlapping information is not available for any of these data sets, we construct two data sets from individual facebook accounts where we know who are common friends using the author's facebook network data along with two of his friends' facebook network data. One synthetic dataset named as Synthetic dataset is also being used. These total six datasets are used for the analysis of the proposed algorithms given in the section 3.2 & section 3.3

4.1 Datasets

The datasets being considered for this study are summarized in Table 4.1 which are described in some detail in the following section.

Dataset	Nodes	Edges
Zachary network	34	78
Collaboration network	5242	14496
Dolphin	63	157
Synthetic dataset	350	6215
Shiva-Gopi facebook data	919	24279
Shiva-Swayam facebook data	1603	40203

Table 4.1: Benchmark & Synthetic datasets considered

4.1.1 Benchmark Datasets

- **Dataset1: Zachary network** [?]

Description: Zachary dataset contains information regarding interaction among members of a university karate club given as two 34×34 matrices, as a binary matrix and a weighted matrix. Binary matrix represents the presence of ties among the members of the club. Valued matrix represents the relative strength of the associations that is number of situations in and outside the club in which interactions occurred. In the year 1977 Zachary used these data and an information flow of network conflict resolution to explain the split-up of this group followin disputes among the members.

- **Dataset2: Collaboration network** [?]

Description: Collaboration network contains author collaborations with regard to General Relativity and Quantum Cosmology Arxiv GR-QC covering papers submitted submitted to General Relativity and Quantum Cosmology category in the period from January 1993 to April 2003 (124 months). If an author i co-authored a paper with author j , the graph contains an undirected edge from i to j . If the paper is co-authored by k authors this generates a completely connected (sub)graph on k nodes.

- **Dataset3: Dolphin** [?]

Description: Dolphin is an undirected graph depicting frequent associations between 62 (as per reference paper) dolphins in a community living off Doubtful Sound, New Zealand.

4.1.2 Synthetic Datasets

- **Dataset4: Synthetic dataset** [?]

Description: Synthetic dataset has been designed [?] in such a way that the network contains a community structure. This graph has four types of edges. We do not consider the type and hence remove all parallel edges. This pruned network has 350 nodes and 6215 edges.

- **Dataset5: Shiva-Gopi facebook data.**
- **Dataset6: Shiva-Swayam facebook data.**

Description: Recently Facebook has made it possible for the data to be downloaded using a facebook app called *netvizz*. Datasets 5 and 6 are collected by combining data of two facebook user accounts of the author Shiva Shankar Danturi with data of his friends Gopi Tadaka and Swayam Prakash respectively. The downloaded facebook data is in file.gdf format. Its data contains two parts, one is friends list that is in the form of

```
nodedef > name VARCHAR, label VARCHAR, sex VARCHAR,
locale VARCHAR, agerank INT
```

and another is edgelist that is in the form of

```
edgedef > node1 VARCHAR, node2 VARCHAR.
```

4.2 Results on Benchmark Datasets

The algorithms available on *RStudio* are executed for all the datasets which give modularity coefficient and number of communities as the output.

- **Zachary network**

Algorithm	Modularity (best split)	Number of Communities
Edge Betweenness	0.4012985	5
Fastgreedy	0.3806706	3
Leading Eigenvector	0.3934089	4
Multi Level	0.4188034	4
Optimal	0.4107143	4
Infomap	0.4020381	3
Walktrap	0.3532216	5

Table 4.2: Zachary network: number of communities & modularity values

On Zachary network Edge betweenness and Walktrap algorithms output five communities as shown in Figures 3.2.2, remaining all other algorithms give three

or four communities. The node 10 is considered as a separate community by Edge betweenness algorithm. In Walktrap algorithm there are three communities formed with 8, 6, 9 nodes, these nodes are forming into two communities in other algorithms. These nodes are forming into two communities of size 10, 12 by adding one more node that is node 10 which is considered a separate community in the Edge betweenness algorithm.

Algorithm	Number of overlapping nodes
Consensus Clustering	5 (3,10,14,20,24)
Connectivity Consensus	5 (3,10,24,28,32)

Table 4.3: Overlapping nodes for Zachary network

It is observed that in the above table there are five nodes in overlapping region in both the algorithms. Among these five nodes three nodes named 3, 10, 24 are common in both the algorithms, so we can say these three nodes satisfy consensus over consensus. The overlapping nodes have good number of connections to other communities as well.

- **Collaboration network**

Algorithm	Modularity (best split)	Number of Communities
Edge Betweenness	0.8490437	433
Fastgreedy	0.8137622	415
Leading Eigenvector	0.7840748	406
Multi Level	0.860752	392
Infomap	0.7943545	717
Walktrap	0.7823643	815

Table 4.4: Collaboration network: communities & modularity values

Since it is a big collaboration network with 5242 nodes and 14496 edges, it takes more time to execute in *RStudio*. Table 4.4 shows the different number of communities obtained by the different algorithms. Walktrap algorithm gives highest number of communities 815 and ML gives 392 communities with highest modularity coefficient. All the algorithms give modularity value of approximately 0.80.

Algorithm	Number of overlapping nodes
Consensus Clustering	389
Connectivity Consensus	335

Table 4.5: Overlapping nodes for Collaboration network

In the above table consensus clustering algorithm gives 389 overlapping nodes by considering 100% consensus (i.e all the algorithms). This value changes according to the consensus percentage. In case of connectivity consensus there are 335 overlapping nodes, this output is according to 50% of an individual node connectivity with consensus of 50% of algorithms. This value could be changed when we change the consensus either in connectivity or in algorithms.

There are 193 common nodes among these 389 and 335 nodes given by two algorithms 2 & 3. From this we can conclude that there are 193 nodes which are completely overlapping which satisfy the properties of both the algorithms.

- **Dolphin dataset**

Algorithm	Modularity (best split)	Number of Communities
Edge Betweenness	0.4392876	15
Fastgreedy	0.4657187	4
Leading Eigenvector	0.4986815	5
Multi Level	0.5212382	5
Infomap	0.5152745	6
Optimal	0.5252546	5
Walktrap	0.5002637	7

Table 4.6: Dolphin network: communities & modularity values

From Table 4.6, we can observe that Edge betweenness algorithm gives highest number of communities 15, other algorithms gives not more than 7. Edge betweenness algorithm outputs four communities with single node they are nodes numbers 5, 12, 13, 36 and three communities with two nodes they are nodes (33, 61), (47, 50) and (54, 62). Walktrap algorithm gives two communities having two nodes each and these are (33, 61) and (47, 50). Except for Edge betweenness and Walktrap algorithms no other algorithm forms communities with one node

or two nodes. These one node or two node communities are merged with other communities by the other algorithms. We could observe that all the modularity values are approximately 0.50.

Algorithm	Number of overlapping nodes
Consensus Clustering	9 (1,3,21,29,31,37,40,53,55)
Connectivity Consensus	10 (1,8,9,20,21,29,37,40,41,51)

Table 4.7: Overlapping nodes for Dolphin network

From Table 4.7, we can see that there are 9 nodes overlapping according to Consensus clustering with 100% consensus. This value changes when consensus percentage changes. According to the algorithm of Connectivity consensus there are 10 nodes that are overlapping. Both the algorithms agree on five common nodes 1, 21, 29, 37, 40 to be overlapping. According to the results of our algorithms we can conclude these five nodes to be certainly in the overlapping region.

4.3 Results on Synthetic Datasets

- **Synthetic dataset**

Algorithm	Modularity (best split)	Number of Communities
Edge Betweenness	0.3943921	4
Fastgreedy	0.3953974	3
Leading Eigenvector	0.3925157	3
Multi Level	0.3953667	3
Infomap	0.3953974	3
Walktrap	0.3953974	3

Table 4.8: Synthetic dataset: communities & modularity values

Edge betweenness algorithm gives four communities and remaining all the other algorithms are giving three communities for this Synthetic data set which is designed to have 3 communities confirming the definition of the data set in Section 4.1.2. Edge betweenness algorithm has a single node 310 as a separate

community. We can observe that modularity value in all algorithms is stable at approximately 0.3953.

Algorithm	Number of overlapping nodes
Consensus Clustering	1 (i.e. 310)
Connectivity Consensus	0

Table 4.9: Overlapping nodes for Synthetic dataset

It is highly interesting to note that for the synthetic data set designed to have 3 well-formed communities, the proposed algorithms also detects almost no overlapping nodes. In Table 4.9 we can observe that there is only one overlapping node according to Consensus clustering approach (with 100% consensus) because Edge betweenness algorithm considers the node 310 as a separate community. There is no overlapping node according to Connectivity consensus approach since the data set is a synthetic data set designed to have three disjoint communities.

From this we can observe that the Synthetic dataset which is designed to have three well-formed communities, has very little chance to have any overlapping nodes. In fact this shows that the Algorithms 2 and 3 are both working very well on the synthetic data set.

- **Shiva-Gopi facebook data**

Algorithm	Modularity (best split)	Number of Communities
Edge Betweenness	0.311986	92
Fastgreedy	0.3895095	7
Leading Eigenvector	0.414189	13
Multi Level	0.4264074	7
Infomap	0.3270177	21
Walktrap	0.3892167	56

Table 4.10: Shiva-Gopi facebook network: communities & modularity values

Shiva-Gopi facebook data set has 919 nodes and 24279 edges as given in Table 4.1. The author Shiva Shankar Danthuri has 301 friends and his friend Gopi Tadaka has 630 friends. There should be a total of 931 (i.e. $301 + 630$) nodes

by combining these two user networks, but we mentioned 919 nodes in the table 4.1 because there are 12 common friends in both accounts.

Among all algorithms, Edge betweenness algorithm gives highest number of communities (i.e. 92), since very high degree nodes exist, for example, 4 nodes having degrees 630, 403, 334, 330; 18 nodes whose degrees are more than 200 and there are 123 nodes whose degrees are 100 and more. We actually combined two networks with 301 nodes and 630 nodes, but the community discovery algorithms extract more than two communities because there are some nodes whose degrees exceed the number of nodes of one network.

Algorithm	Number of overlapping nodes
Consensus Clustering	74
Connectivity Consensus	69

Table 4.11: Overlapping nodes for Shiva-Gopi facebook network

In Table 4.11, we can see that there are 74 overlapping nodes by using Consensus clustering approach. In this table, Consensus clustering approach gives that there are 74 overlapping nodes because community discovery algorithms are dividing the whole network into a large number of communities instead of dividing into only two communities. This happens because there are some nodes which are densely connected in the network, their degrees are more than 50% of nodes in one account data, these nodes are also forming communities. That is the reason there are large number of communities formed in the network leading to large number of overlapping nodes. In case of Connectivity approach there are 69 overlapping nodes.

Out of these nodes, we see that the number of common nodes is 39 which can be considered to be truly overlapping agreed by both the algorithms 2 & 3. These 39 nodes share their connections approximately equally with the other communities as well as within the community. When we analyzed further, we find that not all of the 12 common friends belong to this overlapping region. Only two of these friends are found to be overlapping nodes by the first algorithm. We find that the reason for this fact is that common friends unless they are of high degree, that is have interactions with many other common friends do not come out in the

overlapping region. Hence these algorithms do retrieve non-trivial nodes which satisfy the intuition that friends having connections to many communities as overlapping.

- **Shiva-Swayam facebook data**

Algorithm	Modularity (best split)	Number of Communities
Edge Betweenness	0.6523698	198
Fastgreedy	0.6293061	10
Leading Eigenvector	0.653447	8
Multi Level	0.6712994	7
Infomap	0.6464744	47
Walktrap	0.6668866	35

Table 4.12: Shiva-Swayam facebook network: communities & modularity

In Shiva-Swayam facebook data there are 1603 nodes and 40203 edges. The user Shiva Shankar Danthuri has 301 friends and the other user Swayam Prakash has 1384 friends having 82 common friends.

Among all algorithms, Edge betweenness algorithm gives highest number of communities (i.e. 198), because there are 254 nodes whose degrees are 100 and more than 100. These nodes form new communities with its neighbourhood nodes, so that community discovery algorithms give large number of communities instead of two communities.

Algorithm	Number of overlapping nodes
Consensus Clustering	107
Connectivity Consensus	100

Table 4.13: Overlapping nodes for Shiva-Swayam facebook network

In Table 4.13 we can see that there are 107 overlapping nodes by using Consensus clustering approach and 100 nodes using the second algorithm. There are 41 common nodes among these 107 and 100 nodes given by two algorithms 2 & 3.

These 41 nodes share connections approximately equally with the other communities as well as within the community. In fact out of the 82 common friends

only one node is obtained as overlapping by both the algorithms. Similar to analysis carried out earlier, we see that the overlapping nodes obtained by our algorithms are of very high degree having many connections with friends from different communities and hence truly belong to the overlapping regions of different communities.

4.4 Conclusion

The two algorithms on overlapping community discovery tackle the problem from two different points of view: one considering nodes outside core communities as overlapping; the other detecting nodes that have connections with more than one community as overlapping. It is expected that the results are different and also it is interesting to see that there is a non-trivial intersection between the predictions. That is there is a set of overlapping nodes that is arrived at on consensus by both the algorithms. The common nodes given by both the algorithms can be considered to be in the overlapping region with a high certainty since they are sharing approximately equal connections with the same community and other communities (i.e. approximately half of their connections are with the outside of the community).

In the case of benchmark datasets of Zachary karate club and Dolphin network, we can observe that the overlapping nodes commonly given by both the algorithms 2 & 3 do belong to the overlapping region as can be seen visually since these networks are comparatively smaller than the other datasets.

In Collaboration network and three synthetic datasets, the interesting result of obtaining no overlapping nodes for Synthetic dataset is very satisfactory as the data set is designed to have three disjoint communities. The facebook results of Shiva-Gopi facebook dataset and Shiva-Swayam facebook datasets conform to the common friendships as well as nodes that may be friend-of-friend having high degree of interconnections. It is of course necessary to validate the algorithms further on any benchmark datasets that are designed and constructed for overlapping community discovery which are not yet available to the best of our knowledge.