Combining Bi-clustering Solutions for Gene Expression Data

A thesis submitted to University of Delhi

for the award of the degree of

Doctor of Philosophy

by

Geeta Aggarwal

Department of Computer Science

University of Delhi

Delhi-110007, India

Sept, 2017

Combining Bi-clustering Solutions for Gene Expression Data

Abstract of thesis submitted to University of Delhi

for the award of the degree of

Doctor of Philosophy

by

Geeta Aggarwal

Department of Computer Science

University of Delhi

Delhi-110007, India

Sept, 2017

,

Declaration

The thesis entitled "*Combining Bi-clustering Solutions for Gene Expression Data*", which is being submitted for the award of the degree of Doctor of Philosophy is a record of original and bona fide research work carried out by me in the Department of Computer Science, University of Delhi, Delhi, India.

The work presented in this thesis has not been submitted to any other university or institution for any academic award.

Geeta Aggarwal
Department of Computer Science,
University of Delhi,
Delhi, India.

Certificate

This is to certify that the thesis entitled "*Combining Bi-clustering Solutions for Gene Expression Data*" being submitted by Geeta Aggarwal in the Department of Computer Science, University of Delhi, Delhi, for the award of degree of Doctor of Philosophy is a record of original research work carried out by her under the supervision of Dr. Neelima Gupta.

The thesis or any part thereof has not been submitted to any other University or institution for any academic award.

Supervisor
Neelima Gupta
Department of Computer Science
University of Delhi
Delhi, India.

Head
Department of Computer Science
University of Delhi
Delhi, India.

Department of Computer Science
University of Delhi
Delhi-110007, India

Date:

Certificate of Originality

The research work embodied in this thesis entitled "*Combining Bi-clustering Solutions for Gene Expression Data*" has been carried out by me at the Department of Computer Science, University of Delhi, Delhi, India. The manuscript has been subjected to plagiarism check by Turnitin software. The work submitted for consideration of award of Ph.D. is original.

Name and Signature of the Candidate

Student Approval Form

Name of the Author	Geeta Aggarwal
Department	Computer Science
Degree	Ph.D.
University	University of Delhi
Supervisor	Dr. Neelima Gupta
Thesis Title	Combining Bi-clustering Solutions for Gene Expression Data
Year of Submission	2017

Agreement

1. I hereby certify that, if appropriate, I have obtained and attached hereto a written permission/statement from the owner(s) of each third party copyrighted matter to be included in my thesis/dissertation, allowing distribution as specified below.

2. I hereby grant to the university and its agents the non-exclusive license to archive and make accessible, under the conditions specified below, my thesis/dissertation, in whole or in part in all forms of media, now or hereafter known. I retain all other ownership rights to the copyright of the thesis/dissertation. I also retain the right to use in future works(such as articles or books) all or part of this thesis/dissertation or project report.

Conditions:

1. Release the entire work for access worldwide	
2. Release the entire work for ’My University’ only for 1 Year 2 Years 3 Years and after this time release the work for access worldwide	
3. Release the entire work for ’My University’ only while at the same time releasing the following parts of the work(e.g. because other parts relate to publications) for worldwide access a) Bibliographic details and Synopsis only. b) Bibliographic details, Synopsis and the following chapters only. c) Preview/Table of Contents/24 page only	
4. View Only(No Downloads)(worldwide)	

Signature of the Candidate **Signature and seal of Supervisor**

Place:

Date:

Acknowledgment

This thesis has been an educational process and I have enjoyed gaining deeper knowledge of biclustering in Gene Expression Data. I thank God Almighty for his blessings so that I could complete this herculean task without any difficulty. There are no words that can express my gratitude to my supervisor, my guide Dr. Neelima Gupta. I thank her for her invaluable guidance, constructive criticism, continuous support and patient listening. Her continuous motivation, encouragement helped in getting the best out of me. Her expertise and immense knowledge acted like a steering on the circuitous road of my study.

I am thankful to Head and all faculty members, Department of Computer Science, Delhi University for their guidance and support. My sincerest thanks go to Dr. Mukesh Aggarwal, principal PGDAV College and also to the former principal, Dr. M.M.Goyal for all the support. I would also like to thank Leena, Neha and Saurabh for helping me with the experimental results. I also express my sincerest thanks to Dr. Seema Aggarwal and Dr. Manisha Bansal both of whom gave me both academic and emotional support.

My special thanks go to other research scholars who have been one of the greatest resources to rely on. I would like to thank them for countless interesting and clarifying discussions during this research.

I am highly thankful for the support and cooperation that I got from my family and my friends. They all have expressed words of wisdom and great inspiration. Their care and love kept me going during hard times.

Last but not the least; I thank the support staff at Department of Computer Science, Delhi University and PGDAV College who helped me throughout the studies.

Geeta Aggarwal

Abstract

Advances in DNA microarray technology has led to generation of humongous gene expression data which needs to be analyzed. Clustering and biclustering have found their application in the analysis of gene expression data. Traditional clustering algorithms fail as most of the genes are responsive only to a small subset of samples/conditions rather than the entire set of samples/conditions. Also, a gene that may be responsible for more than one biological activity and hence may belong to more than one cluster. Similarly a sample may trigger the expression of genes responsible for more than one biological activity and hence may belong to more than one cluster. At the same time, there may be some genes/conditions that do not account for any biological function and thus do not belong to any. Traditional clustering algorithms typically do not allow the clusters to overlap and are exhaustive. Biclustering is a technique wherein genes and samples are clustered simultaneously so that the genes responsive only to the selected set of samples are clustered together. Biclusters are allowed to overlap both on genes as well as on conditions and they are not exhaustive. Thus Biclustering is more suitable for clustering gene expression data than traditional clustering algorithms.

Different biclustering algorithms use different heuristics and thus produce different biclustering solutions. Moreover these algorithms are sensitive to random initialization and threshold parameters. Given this, an end user often faces the problem of selecting the right algorithm for the application/data. One way to improve the robustness and quality of solutions is to combine/ensemble solutions and obtain a consensus. Several

ensemble techniques have been successfully applied for supervised classification and unsupervised clustering. Combining biclustering solutions is more challenging as compared to combining classification and clustering for several reasons: one, biclusters from two different solutions typically involve different sets of conditions, second, biclusters are non-disjoint/overlapping and thirdly, they are non-exhaustive.

We present three ensemble techniques for the biclustering problem that allow simultaneous overlap of objects as well as attributes. As different schemes/solutions may assign different labels to the same bicluster, biclusters are aligned appropriately using Hungarian method, in the first two approaches. In order to solve the label correspondence problem, one needs to solve the k dimensional bipartite matching which is known to be NP-hard for $k >= 3$. To get around this problem, one of the schemes was fixed as the reference scheme and the other solutions were aligned with it. The first algorithm, $BiETopti$ uses optimization technique to generate the consensus. For the formulation of optimization problem, global labels are defined. Through experimental studies we show that $BiETopti$ improves the quality of the biclusters as compared to those in the input solutions/schemes. The results are promising but has a limitation of having fixed number of biclusters in the input schemes. Generally this condition is difficult to meet in biclustering solutions. To overcome this limitation, another ensemble algorithm $BiETclassi$ is proposed. This algorithm makes use of classifiers such as Discriminant Analysis and Support Vector Machine for the ensembling purpose. Experiments on synthetic as well as real data sets show that $BiETclassi$ performs better than $BiETopti$ not only in terms of quality but also in terms of time. Further, Discriminant Analysis as a classifier turns out to be a better option than Support Vector Machine.

Both the above algorithms are compute intensive as they involve label correspondence followed by optimization problem (directly or indirectly) to be solved. We do away with the requirement of label correspondence in our third approach called $BiETmetaclus$. The algorithm does not require any optimization problem to be solved. It simply pools

in all the biclusters and uses statistical similarity measures like Mutual Information to form metaclusters of similar biclusters. We believe that biclusters, sharing high content of information about each other and less information with other biclusters, form a more cohesive group. Mutual Information has been considered to be a more general measure to capture linear as well as non linear associations or dependencies amongst genes. Besides, it is also robust towards noise. Finally voting is done to form the consensus. Experimental studies showed that $BiETmetaclus$ provides better biclusters than $BiETopti$ both in terms of quality as well as time. However, there is a tradeoff between quality and time amongst $BiETmetaclus$ and $BiETclassi$. $BiETclassi$ performs better than $BiETmetaclus$ in terms of quality most of the times whereas $BiETmetaclus$ is faster than $BiETclassi$.

MATLAB was used as the platform for the implementation of our work. Optimization in $BiETopti$ was done using LINGO tool. Experiments were conducted both on synthetic data sets and real data sets of *Arabidopsis Thaliana, Saccharomyces Cerevisiae, Human Breast Cancer* data and *Diffuse Large B Cell Lymphoma*. To assess the quality of biclusters on synthetic data sets, measures like biclustering error and agreement score were used whereas the biological significance of the biclusters on real data sets was validated using online biological tools *DAVID* (Database for Annotation, Visualization and Integrated Discovery) and *RSAT* (Regulatory Sequence Analysis Toolbox).

We hope that our work would help in delivering useful information to biologists in the analysis of gene expression data.

Contents

List of Tables

List of Figures

List of Algorithms

Chapter 1

Introduction

1.1 Motivation

The analysis of gene expression data (also referred to as genomics data) has become a highly popular technique for studying the biological mechanism of organisms. One can study behaviour of thousands of genes simultaneously in a single experiment in a microarray experiment. The ability to measure the expression of a whole genome under different experimental conditions with the help of microarrays has led to the generation of large scale gene expression data. Analysis of the data allows the discovery of groups of genes that share similar expression profile. Moreover it is expected that a group of genes responsible for one biological process will show similar expression profiles. Hence, analysis of the genomic data allows us to identify groups of genes that are responsible for different biological processes and also helps in discovery, validation and understanding of various diseases.

Classification and clustering have been successfully used for more than a decade for the analysis of expression data. Traditional clustering/classification algorithms [BDSY99, ESBB98, THC$^+$99, Cla99] cluster the genes based on their expression profiles under all the conditions. These traditional algorithms work well for small data sets but fare poorly

when number of experimental conditions is large. This is due to the fact that genes appear to be at equal distance from each other when the number of conditions is large. All the conditions are treated equally by these algorithms while computing similarity amongst the genes whereas only a small subset of conditions affect the cellular processes [IFB+02]. Other conditions, which do not contribute to the cellular process, add to the background noise. Thus grouping genes based on this small subset of conditions is more relevant in the study of gene expression data.

Biclustering is the term coined by Hartigan [Har72] to group objects/genes based on their expression profiles under a relevant subset of features/conditions and it was used for the first time by Cheng and Church [CC00] for the analysis of gene expression data. Biclusters are allowed to overlap, both on objects/genes as well as on features/conditions. The technique thus, has been found very useful in the analysis of gene expression data wherein genes responsible for one biological function are influenced by a subset of conditions instead of the entire set of conditions and a gene (/ a condition) may be responsible for more than one biological process.

Various biclustering algorithms exist in literature [IFB+02, PBZ+06, CC00, BDCKY03, LW07]. They lack robustness and stability with respect to random initialization and input parameters. Each algorithm aims to optimize different objective function leading to different solutions. For an end user, who has no clue about which objective function is best suitable for the application, the choice of a particular algorithm becomes difficult. Ensemble techniques come to the rescue in such situation. The principle underlying the ensemble techniques is to generate a set of models (also referred to as input schemes/input solutions/biclustering solutions/partitions in literature) and aggregate them into a single consensus model. Ensemble methods aim to provide solutions which are more robust towards random seeds and input parameters. They have been proved to be more stable and accurate than a single solution [SG02, DF03, WDH01, HY04, Die00, MO97] in the area of supervised classification and unsupervised clustering.

In this work, we extend the idea of ensemble techniques to improve the performance of biclustering solutions. However, ensembling biclustering solutions is far more challenging as compared to ensembling classification and clustering. This is due to the following reasons, first being the overlapping nature of biclusters. The overlap may be on genes, on conditions or on both. Another challenge arises from the fact that the attribute set of different biclusters may be different as biclusters are defined by a subset of attributes rather than the whole set. Also biclusters are non exhaustive in nature. Moreover different biclustering solutions may contain different number of biclusters. Most of the work done on ensembling clustering/classification assume that the clusters are non-overlapping and all the solutions contain a fixed (say k) number of clusters. Little work has been done where the clusters are allowed to overlap and since these algorithms work on the entire set of attributes, none of them needs to address the second challenge faced in biclustering. Some work has been done for the ensemble of projected clustering and co-clustering solutions. Wang et al. [WLDJ11] presented an ensemble solution for co-clustering wherein they extract block-constant biclusters generalizing the grid-style partitions to allow different resolutions in different parts of the data matrix. A pair of biclusters may overlap on objects or on features but not on both at the same time. Gullo et al. [GDT09] presented an ensemble solution for projective clustering wherein an object may belong to more than one biclusters but the total sum of the membership is one thereby meaning that if an object completely belongs to one bicluster it does not belong to any other. They project the clusters on one dimension in a fuzzy way. Biclustering is different from these problems/solutions wherein an object/feature may have a total membership more than one and a bicluster is defined by more than one feature. Also, bicluster may overlap both on objects and features simultaneously.

In this work, three approaches for ensembling biclustering solutions are proposed. The first algorithm called $BiETopti$ [AG13a, AG17] uses optimization techniques to generate consensus model. $BiETopti$ assumes that the input biclustering solutions con-

tain equal number of biclusters. In our second approach called $BiETclassi$, we drop this assumption and use classifiers to predict the labels. Both these algorithms involve expensive steps of label correspondence and optimization (directly or indirectly). In the third approach called $BiETmetaclus$ [AG13b], we do away with both of them and use the concept of metaclustering instead. Mutual Information between biclusters is used to form the metaclusters and a simple voting technique is used to form the consensus.

In a parallel work, Hanczar and Nadif (HN) [HN11] proposed the use of bagging to improve the performance of biclustering schemes. They use bootstrapping on various biclustering algorithms and use the concept of metacluster to ensemble them. All our algorithms outperform their algorithm.

1.2 Problem Definition

Let G be a set of N genes and C be a set of d samples/conditions. Let E be an $N \times d$ expression matrix where each row represents the expression of a gene under d samples. E is subjected to a biclustering algorithm which delivers a biclustering scheme/solution π_i consisting of k_i biclusters. $\pi_i = (BC_1, BC_2, ..., BC_{k_i})$, BC_j is a bicluster represented by a tuple (G_j, C_j), G_j being a subset of genes and C_j a subset of conditions. Note that in general different biclustering schemes may contain different number of biclusters. Let $\pi_1, \pi_2, ..., \pi_H$ be the H biclustering schemes so obtained and let $\lambda_i : G \times C \rightarrow 2^{\{0...k_i\}}$ be a function that yields a set of labels for each gene condition pair (g_l, c_r). Note that since the biclusters may overlap both on genes and conditions, a (gene, condition) pair may be assigned more than one label. Also, there may be a (gene, condition) pair which does not belong to any bicluster. Such a pair is given label 0. Let $\lambda_1, \lambda_2, ..., \lambda_H$ denote the H labelings of $G \times C$. The problem of bicluster ensemble is to derive a consensus function $\hat{\lambda}$, which combines the H biclusterings and deliver a biclustering $\hat{\pi}$ to achieve one or more of the following aims:

1. It improves the quality of the biclusters.

2. It is more robust and stable than its constituent schemes.

1.3 Our Contribution

There are two main steps in any ensemble algorithm [VPRS11]: *Generation* and *Consensus*. *Generation process* deals with generation or creation of a set of input partitions. There are various ways to generate the input partitions. Partitions can be generated by running different clustering/biclustering algorithms on same data set, or by running same algorithm on different samples of same data set, or by executing the same algorithm a number of times on the same data set, each time with different initialization or by changing the input parameters/threshold values. The purpose of *consensus process* is to integrate the partitions obtained in the generation step.

We propose three approaches for bicluster ensemble. In the first two approaches named $BiETopti$ and $BiETclassi$ respectively, the schemes are generated by running a biclustering algorithm several times with different initializations and the similar biclusters are aligned using Hungarian algorithm [Kuh55]. In $BiETopti$, an objective function is obtained that captures the dissimilarity of the new labels with the aligned biclusters both for genes as well as for samples. The consensus is obtained by minimizing the value of the objective function (i.e. dissimilarity). In $BiETopti$ we assume that the number of biclusters in all the biclustering solutions is same. This assumption is dropped in $BiETclassi$ wherein each bicluster is individually subjected to a classifier to refine labeling. However, label correspondence is still required to be able to do voting to generate the final consensus.

Besides label correspondence, both $BiETopti$ and $BiETclassi$, involve optimization (directly or indirectly) and hence are compute-intensive. In the third approach called $BiETmetaclus$, all the biclusters provided by various biclustering solutions are collected in a pool and metaclusters are formed based on the information they share about each

other thereby eliminating both the expensive steps of label correspondence and optimization. Voting is then used to generate the consensus model. This technique is similar to HN as both form metaclusters of similar biclusters and consensus is formed by voting. However it differs from HN, one in that the schemes are generated without bootstrapping and secondly, Mutual Information is used in place of Jacquard Index. Mutual Information has been considered to be a more general measure to capture linear as well as non linear associations or dependencies [KSG04, SKD$^+$02, BK00, MCA$^+$98]. According to Priness et al. [PMBG07], mutual information is resistant to outliers and missing data. Besides, it is also robust towards noise.

The algorithms were tested both on synthetic as well as real data sets. All the algorithms have been coded using MATLAB [MAT10]. LINGO software [LIN06] was used to solve the optimization model. Quality of biclusters in case of synthetic data sets was validated using Biclustering Error(BCE)/Error Rate and Agreement Score(AS). For real data sets, validation was done using external biological information by determining the functionality of the genes of the biclusters from Gene Ontology (GO) [ABB$^+$00] and common patterns (motifs) in the promoter regions of the genes of a bicluster with the help of biological tools, *DAVID* (Database for Annotation, Visualization and Integrated Discovery [HSL08] and *RSAT* (Regulatory Sequence Analysis Tool) available on line.

Experiments performed on synthetic data sets show that the biclusters produced by all the three approaches were better than the input solutions. The results were also compared with HN which was the only available bicluster ensemble algorithm. It was found that biclusters produced by all the three algorithms were better than HN too. For real data sets too, the biclusters produced were found to be biologically more significant than the input biclusters. Experiments show that $BiETopti$ outperforms HN. It is also experimentally seen that $BiETclassi$ and $BiETmetaclus$ both outperform $BiETopti$ both in terms of time and quality. As a result, both $BiETclassi$ and $BiETmetaclus$ also beat HN in quality and time. As compared to $BiETclassi$, $BiETmetaclus$ improves

upon the time significantly but $BiETclassi$ provides better biclusters as compared to $BiETmetaclus$. Thus there is a tradeoff between $BiETmetaclus$ and $BiETclassi$ in terms of time and quality.

1.4 Organization of Thesis

Rest of the thesis is organized as follows: In Chapter 2, an overview of the essential concepts in biology are provided for better understanding of our work. Biclustering algorithms, ensembling and related work is presented in Chapter 3. The first algorithm for ensembling biclusters, $BiETopti$ is explained in Chapter 4. Second algorithm for ensembling biclusters, $BiETclassi$ is presented in Chapter 5. The last algorithm for ensembling, $BiETmetaclus$ is explained in Chapter 6. Finally, in Chapter 7 we present concluding remarks to our work.

Chapter 2

Biological Overview

In this chapter, we will review some concepts regarding microarray bioinformatics that are key to identify the design requirements of gene expression analysis [APS06, MDPM08, RJLS10, GSS91]. These concepts will help to understand the nature of input data and to evaluate the results of the analysis process.

2.1 Cell

The basic unit of biological activity, **Cell** is the structural and functional unit of all living organisms. It contains jelly like material called protoplasm and is surrounded by a cell membrane. There are two components of protoplasm: nucleus that contains the genetic material and cytoplasm that is the semi fluid material in which cell organelles like mitochondria, ribosomes etc. float. There are two types of cells depending on the presence of nucleus i.e. prokaryotic and eukaryotic. Cells of primitive organisms (such as bacteria) which do not have a nucleus are called prokaryotic cells and those of higher organisms which have a well defined nucleus are called eukaryotic cells

Organic molecules of Cell

There are four types of basic molecules present in a cell: Proteins, Carbohydrates, Lipids and Nucleic acids. Proteins are the most diverse and complex macromolecules in the cell. They are used for structure, function and information. They are made of linearly arranged amino acids. There are twenty naturally occurring amino acids from which all proteins are composed. Carbohydrates are source of energy for the cell. Lipids are hydrophobic molecules and are constituent of the membrane of the cell and other cell organelles. DNA and RNA are the nucleic acids that encode the genetic information for synthesis of all proteins.

DNA - Deoxy-ribose Nucleic Acid

DNA contains the genetic information of a cell. It is a long sequence of nucleotides as shown in Figure 2.1. These are molecules made of an organic base, a sugar group i.e. deoxyribose and a phosphate ion. It is responsible for storage of information about an organism's inherited characteristics. DNA is the hereditary material in humans and almost all other organisms. Genetic information from parent is transferred to its offspring through DNA which is a set of blueprint needed to construct other components of cells, such as proteins and RNA molecules.

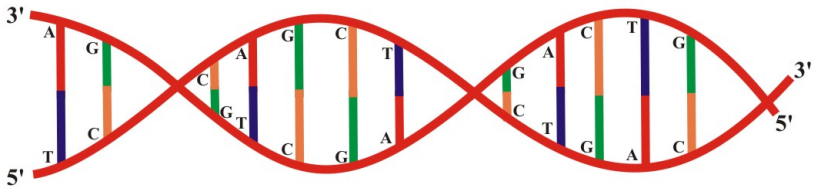

Figure 2.1: DNA - Deoxy-ribose Nucleic Acid.

Every cell in a body has the same DNA. The information in DNA is stored as a code made up of four chemical bases or nucleotides: A(adenine), T(Thymine), C(Cytosine) and

G(Guanine). The DNA molecule consists of two strands. The bases in the two strands are paired, so that A in one strand matches T in the other, and C matches G. This way the sequence of one strand completely determines that of the other, complement strand. The paired long strands are coiled into a spiral called a double helix. Each DNA strand has a polarity, from head called the 5’ end and a tail called the 3’ end. DNA has a lagging strand 3’-5’ and a leading strand 5’-3’. The 5’ end of a strand matches with a 3’ end of the other strand. The non coding part i.e. **introns** separates the coding sequences called **exons** in the DNA of eukaryotes. The structure of gene is shown in Figure 2.2. It must have Exons, start signals, stop signals and regulatory control elements. The average gene with 7-10 exons spread over 10-16 kb of DNA. **Open reading frame (ORF)** is that part

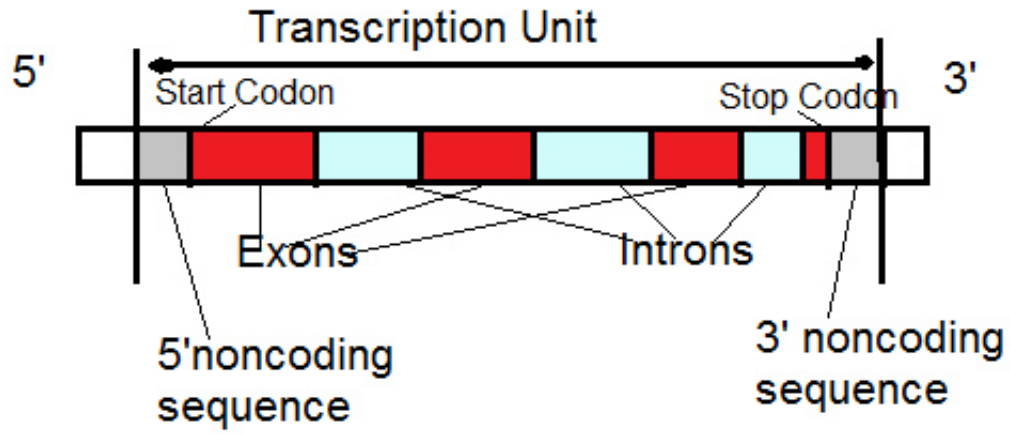

Figure 2.2: Gene.

of DNA that codes for the formation of proteins. **Upstream region** and **downstream region** of the DNA is the portion of DNA near the head region (5’ end) and the tail region (3’ end) respectively.

Property of DNA

DNA can replicate, or make copies of itself. Each strand of DNA in the double helix unzips itself and serves as a pattern for duplicating the sequence of bases. This happens

when cells divide because each new cell needs to have an exact copy of the DNA of the old cell. Most genes contain the information needed to make functional molecules called proteins.

RNA - ribose nucleic acid

RNA is a molecule which is chemically similar to DNA. Both RNA and DNA are made up of a chain of nucleotide bases, but they have slightly different chemical properties. RNA uses sugar ribose instead of deoxyribose in its backbone. RNA uses the base Uracil instead of Thymine. RNA is a single stranded structure as shown in Figure 2.3. RNA performs many functions. RNA plays a key role in the synthesis of various proteins in a cell. In some lower organisms it also acts as the carrier of genetic material. There are three

Figure 2.3: RNA - Ribose Nucleic Acid.

main types of RNA molecules. The RNA that transfers DNA code to ribosome for translation is called messenger RNA (mRNA). Transfer RNA (tRNA) helps in bringing the amino acids according to the ribosomes for protein synthesis. Ribosomal RNA (rRNA) is major component of the protein synthesizing cell organelle called ribosome. Translation, the second step in protein formation takes place in the cytoplasm. The mRNA interacts with a specialized complex called a ribosome, which reads the sequence of mRNA bases. Each sequence of three bases, called a codon, usually codes for one particular amino acid. Amino acids are the building blocks of proteins. Figure 2.4 explains the flow of information from DNA to protein.

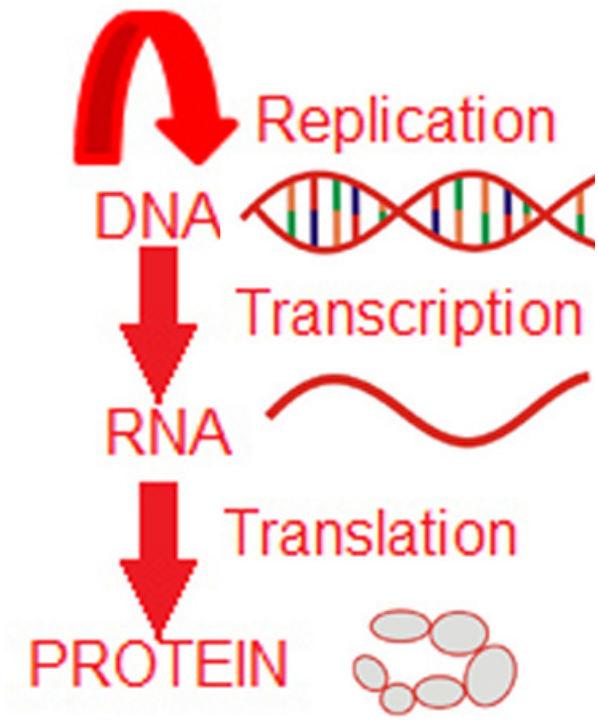

Figure 2.4: Flow of Information.

2.2 Gene and Gene Expression

Gene, the heredity factor, is the functional unit of DNA. The DNA segments that carries genetic information are called genes. It is a stretch of DNA that encodes a protein or an RNA molecule. Genes that code for protein, carry information which determine various characteristics of an organism like eye colour, hair etc. and the non protein coding genes code for RNA molecules. **Phenotype** refers to the physical characteristics of an organism i.e. what that organism looks like. The genetic encoding of its phenotype is called its **genotype**. The genotype consists of gene combination for a trait (e.g. RR Rr rr) and is shown in Figure 2.5. R is gene for red colour and is dominant whereas r the gene responsible for white colour in flowers is recessive. The physical feature resulting from a

genotype is the phenotype (e.g. Red , Red and White). Three genotypes yield only two phenotypes.

	R	r
R	RR	Rr
r	Rr	rr

Figure 2.5: Genotype: R-dominant gene r-recessive gene.

The Genome size varies from organism to organism. The yeast contains around 6000 genes whereas there are about 40,000 genes in human beings. All the genes are not expressed when subjected to a condition. The process of turning genes on and off is known as gene regulation. Gene regulation can occur at any point during gene expression, but most commonly occurs at the level of **transcription** (when the information from DNA is transferred to mRNA).

Gene Expression is the process by which :

- information from a gene sequence is manifested into structure and functions of a cell.
- genotype of an organism is manifested into its phenotype.

We say that a genetic information in gene is expressed when the protein it codes for, is synthesized and is responsible for the phenotype of an organism. Different genes or the subsets of genes may be responsible for different phenotype of an individual. A subset of genes responsible for the hair color may be different from the genes responsible for the height of an individual. The genotype of an organism influences the phenotype. The characteristics of an organism may be the result of the coordinated expression of one or several genes and their interactions with the environment. The environmental conditions influence the expression of genes. A gene may be highly expressed under some conditions and may be suppressed under some other set of conditions. Transcription

factors (TF) are proteins that bind to one (or more) specific sequence(s) of nucleotides called the **Transcription Factor Binding Site(s) (TFBS)** on the promoter region of a gene. Promoters are sequences of DNA that are the start signals for the **transcription** of mRNA. Terminators are the stop signals. The enzyme RNA Polymerase moves along the strand of the DNA. As it encounters each DNA nucleotide, it adds the corresponding complementary RNA nucleotide to a growing mRNA strand. Once the stop signal is reached the newly constructed mRNA strand is released. Finally, it leaves the nucleus and serves as a template for the synthesis of protein in the cytoplasm at the ribosome. During **Translation**, message carried in mRNA is converted into amino acids and the synthesis of the corresponding proteins at the ribosomes takes place. Amino acids are formed from the four bases (A, U, C, G). The sequence of nucleotides in the mRNA determines the sequence of amino acids in the synthesized protein. Each amino acid is actually a triplet of three nucleotide bases called a **codon**. To code for the 20 essential amino acids a genetic code must consist of at least a 3-base set (triplet) of the 4 bases. If one considers the possibilities of arranging four things 3 at a time ($4 \times 4 \times 4$), we get 64 possible code words, or codons (a 3-base sequence on the mRNA that codes for either a specific amino acid or a control word). Three codons (TAA, TAG and TGA) indicate the end of a protein sequence and are called the stop codons. The codon AUG represents methionine and is also the translational **start signal**. All others code for a particular amino acid. Most of the amino acids are encoded by more than one codon. The expression of a gene is controlled and regulated by one or more TFs and their binding to the TFBS in the promoter regions of the gene. Genes showing same expression profile or behaving similarly are said to be co expressed. Such genes are regulated by the same set of TFs and hence have common TFBSs. In other words genes having similar expression profiles, thus belonging to the same bicluster, are considered to have a **common regulatory mechanism or signature or motif** in their promoter region [HZGD05]. Binding of Transcription factors with the TFBS may be regulated by many conditions. In fact expression of one gene may be

governed by the expression of another gene. Some genes may code for a protein which in turn may act as a transcription factor and regulate the expression of some other genes. The entire network is quite complex. Also, the gene to protein correspondence is not one to one. There are genes that may code for more than one protein. Ideally measurement of gene expression should be done by measuring the amount of protein produced. However, it is often easier to measure one of the intermediate product like mRNA to infer the gene's expression level.

2.3 Microarray Experiments and Expression Matrix

A microarray is a small chip on which DNA molecules are attached in fixed grids. This chip is made up of chemically coated glass, nylon membrane or silicon. To decipher the logic of gene regulation in an organism, the simultaneous study of all the genes of an organism is important [Lan05, ZWD+04, AMK00, Kau93]. A vast amount of expression data has been collected for different organisms (healthy and diseased) during different developmental stages, changing environmental/chemical/clinical conditions, or at different time points. Biologists are facing a problem in extracting meaningful information from this humongous data. Development of efficient computational tools to analyse this data, is the need of the hour. Scientists need to know the set of genes that are responsible for a particular biological activity. The activities for which they may be interested may be the formation of a protein, genes causing stress, high blood pressure, diabetes, heart ailment, tumor or AIDS. In plants, these activities include reproduction, growth of a particular part of a plant, photosynthesis and absorption of nutrients from soil. The biologists are of view that the genes responsible for any activity get triggered under certain conditions and must have some sort of association amongst themselves. Genes with similar expression profile share something common in their regulatory mechanism was suggested by Vilo et al. [VBJ+00] and Dhaeseleer et al. [DWFS98, DLS99]. The level of a certain gene can

vary across the conditions. The conditions in the data matrix may be a time series during a biological process (e.g., the yeast cell cycle) or a collection of different tissue samples (e.g., normal versus cancerous tissues). If we are able to find the genes responsible for a disease, then conditions that affect the expression of these genes can also be found out. Also the other genes affected by these conditions can also be found out. Development of efficient computational tools for the analysis of this huge amount of data, to be able to extract biologically relevant information from it, is the most important requirement of today. If one can extract fewer genes showing a pattern, association or correlation in their expression values from the data, then the task of a biologist is greatly simplified.

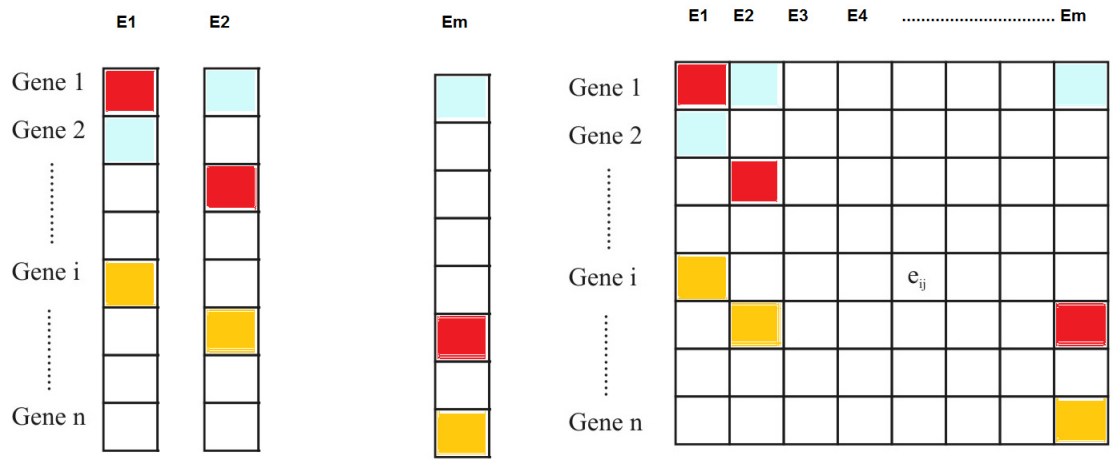

Figure 2.6: Microarray Experiments: each experiment corresponds to a condition.

The gene expression data is usually displayed as a matrix. Various experiments are performed wherein genome is subjected to different conditions as shown in Figure 2.6. The behaviour of each gene under these conditions is stored in a matrix. Row in the matrix corresponds to the expression profile of a gene and each column corresponds to a sample or a condition [CST00a]. The conditions might be different individuals, different experimental conditions of the organism, or different tissues (e.g., cancerous vs healthy) from the same individual. Figure 2.7 shows an example of a gene expression matrix. The $(ij)^{th}$ entry of the expression matrix represents the expression of i^{th} gene under j^{th}

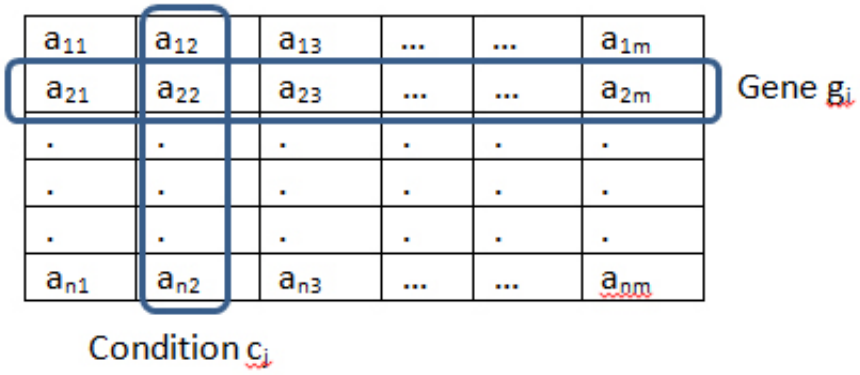

Figure 2.7: Gene Expression Data Matrix.

sample. The original data may contain noise and/or missing values that are a result of the experimental procedure. Figure 2.8 shows the heat map of gene expression data of human breast cancer.

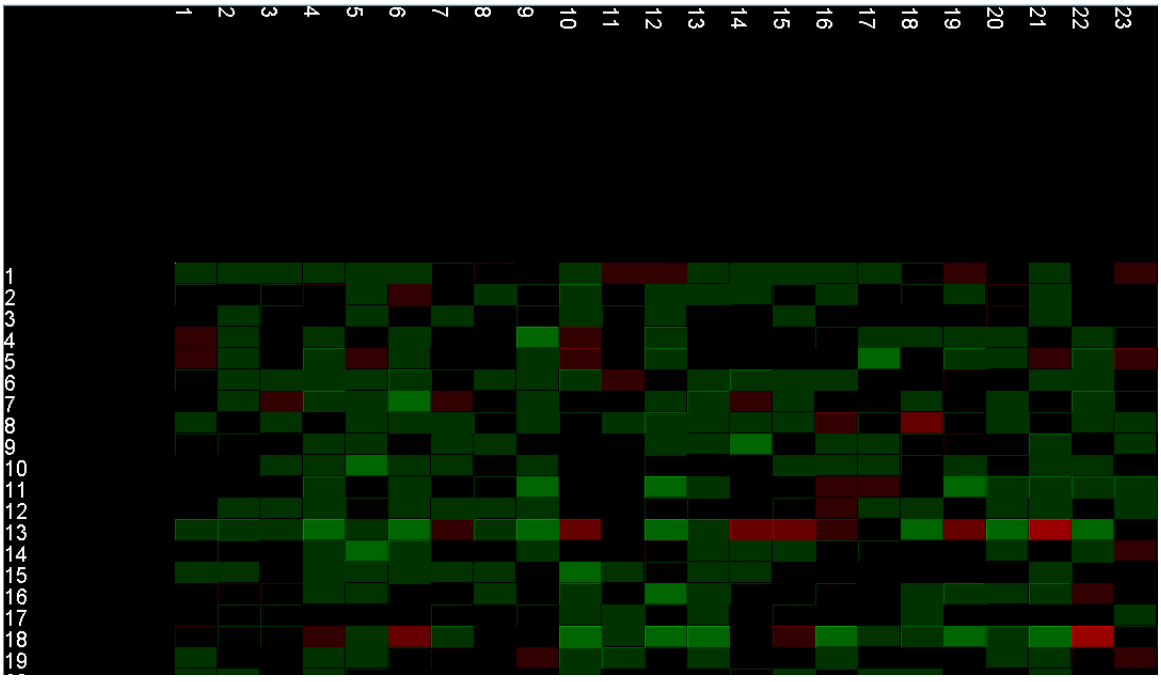

Figure 2.8: Gene Expression Data.

The motive behind studying the gene expression data is to study the functionality of genes and that the genes with similar expression patterns are likely to be involved in similar processes, and hence have similar functionality. Other than deducing function of unknown genes, gene expression analysis has proven to be helpful to identify diseases profiles, deciphering regulatory mechanisms, genotyping and drug developing. The number of genes to be analysed is very large so computational aids are needed, and the biological challenge should be formulated as a mathematical problem.

Chapter 3

Preliminaries and Related Work

In this chapter we present biclustering algorithms that have been used in our work. This is followed by the techniques used by researchers for generating ensembles in the area of classification and clustering. We also discuss the work done to generate ensembles for the closely related problems of co-clustering and projected clustering and discuss how biclustering is different from them. Parallel work due to Hanczar and Nadif [HN11] is also discussed here. Validation techniques used to assess the quality of biclusters are explained subsequently and the details of the data sets used in the study are given at the end.

3.1 Biclustering Algorithms

Several biclustering algorithms have been developed and applied to microarray analysis. Amongst these the most popular are BIMAX, CC, ISA, OPSM and xMotif. Performance of BIMAX deteriorates with increasing degree of overlap whereas OPSM is sensitive to noise. Therefore we proceeded with the rest of the three algorithms and the details of these is given here to get the insight of the same.

3.1.1 Cheng and Church (CC) Algorithm

Cheng and Church were the first ones to propose an algorithm for biclustering. They consider that biclusters follow an additive model and use a greedy iterative search to minimize the mean square residue. The algorithm identifies biclusters one by one.

Let e_{ij} be the expression of i^{th} gene under j^{th} condition of the expression matrix E with N genes and d conditions. Let (I,J) denote a bicluster where I and J are the subsets of genes and conditions. Mean square residue score of the bicluster (I,J) is then defined as:

$$MSR(I,J) = \frac{1}{|I||J|} \sum_{i \in I, j \in J} (e_{ij} - e_{iJ} - e_{Ij} + e_{IJ})^2$$

where e_{iJ} is the row mean of row i , e_{Ij} is the column mean of column j and e_{IJ} is the overall mean over the entire submatrix.

The aim is to find a bicluster that minimizes the MSR. The procedure begins with the whole matrix as a bicluster and then deletes rows and columns with the highest MSR score as long as $MSR(I,J) > \delta$, where δ is the threshold parameter. Then rows and columns with lowest score are added as long as $MSR(I,J) < \delta$. To find more biclusters, the already found bicluster is masked with random values and the process iterates till desired number of biclusters is obtained.

3.1.2 xMOTIF Algorithm

Murali and Kasif[MK03] proposed an algorithm that uses discretized expression matrix as input. The data is first discretized into a set of symbols by using a list of statistically significant intervals for each row. They aim at finding conserved gene expression motifs (xMOTIFs). xMOTIF (bicluster) is defined as a subset of genes that are simultaneously conserved across a subset of the conditions. The expression level of a gene is said to be conserved across a subset of conditions if the gene is in the same state in each of the

conditions in this subset. The obtained set of rows and columns is then called a bicluster if it contains more than an α fraction of all samples. The accuracy of the algorithm depends on how the data is discretized. xMotif is able to find biclusters with constant values and with constant rows. Data may contain several biclusters and the algorithm finds the largest such bicluster. To identify other xMOTIFs, an iterative strategy wherein samples satisfying each xMOTIF are removed, is adopted. Then the new largest xMOTIF is searched. This process continues until all samples satisfy some xMOTIF.

3.1.3 Iterative Signature Algorithm (ISA)

Bergmann et al. [BIB03] introduced Iterative Signature Algorithm (ISA) that seeks biclusters consisting of co-regulated genes and conditions. It extracts biclusters based on the assumption that genes (/conditions) belonging to a bicluster exhibit similar expression profile with high expression values. The algorithm iteratively and alternatively computes gene scores and condition scores. It generates two normalized copies of the gene expression matrix, one with normalized rows E_G (normalized on genes) and one with normalized columns E_C (normalized on conditions). Starting with an initial set of genes, it computes the condition scores by taking the product of E_G with the gene vector which has 1 for the selected genes and 0 otherwise. Conditions with scores above a threshold (t_c) are selected. It then computes the gene scores in a similar way by taking the product of E_C with the condition vector. Genes with scores above a threshold (t_g) are selected. The set of conditions and genes are refined by repeating the process iteratively unless the convergence criterion is met. The algorithm is executed for a large number of input seeds and the modules are reconstructed from the recurring fixed points using a procedure resembling agglomerative clustering by fusing the solutions that were distinct but very similar.

3.2 Overview of Ensembling

Method of ensembling consists of two main steps [VPRS11]: *generation* of a set of input models and the integration of all to obtain a *consensus*.

Generation Process: The ensembling process begins with the generation of schemes. In general, algorithms that provide more information about the data should be used for the generation step. It is difficult to know beforehand which algorithm is appropriate for the problem. It is recommended to make a diverse ensemble as more varied the set of partitions are, more information is available for the consensus function. Aggregation of same schemes is of no use. Various sources of diversity are possible which may be used to generate schemes. Different schemes may be obtained by applying different biclustering algorithms on the data set or by applying same biclustering algorithm on different samples of data set (bagging and boosting). Schemes may be generated by applying same algorithm on the data set by varying the threshold parameters/initialization.

Consensus Process: Once the schemes/solutions are generated, they have to be combined to form the consensus. Two main approaches are used to generate a consensus partition- *object co-occurrence* and *median partition*.

In the first approach, consensus partition is obtained depending upon the frequency with which two objects occur together or belongingness of an object to one cluster. One way to do this is *Relabeling and Voting* [AK10, AK08, WDH01, DWH02, FB03] and another is using *Co-association Matrix* [ACN08, Fre01, FJ05]. The Relabeling and Voting methods solve a label correspondence problem as a first step followed by voting to obtain the final consensus. Heuristic such as *bipartite matching* has been used to solve the label correspondence problem. The idea of co-association is used to avoid the label correspondence problem and they map the partitions in the cluster ensemble into an intermediate representation i.e. the co-association matrix which is formed by effectively merging the similarity matrices of all the partitions. Using co-association matrix as the similarity measure between objects, consensus partition is obtained by applying a clustering algorithm.

In the *median partition* approach, an optimization problem is solved to obtain consensus partition. The median partition is defined as the partition that maximizes the similarity with all partitions in the cluster ensemble and is defined as:

$$\hat{\pi} = \arg\max_{\pi} \sum_{j=1}^{H} sim(\pi, \pi_j)$$

where $sim(\pi_i, \pi_j)$ is a similarity between two partitions π_i, π_j. The median partition problem defined with the Mirkin distance [Mir96] has been proved to be NP-hard. Though no theoretical results are known for other similarity measures, this method is considered to be computationally expensive.

Other approaches to obtain consensus function include *graph and hyper graph algorithms [SG02], Information theoretic approaches [TJP03], finite mixture model [TJP04], LAC [DGM$^+$07], genetic algorithms [YAL$^+$06], NMF [LDJ07] and Kernel method [VPCMRS08]*. These methods either fall under object co-occurrence or median partition.

3.3 Related Work

Ensembling techniques have been used successfully in the area of classification and clustering to improve the quality of results. For the classification problems, bagging and boosting [DF03, FB03, HN11, MJ87] have been used as standard techniques to generate ensembles. Bootstrap aggregating, often abbreviated as bagging, involves having each model in the ensemble vote with equal weight. Bagging trains each model in the ensemble using a randomly drawn subset of the training set. Boosting is a general method for improving the performance of a weak learner and involves incrementally building an ensemble by training each new model instance to emphasize the training instances that previous models mis-classified. The classifiers produced by the weak learners are then combined into a single composite strong classifier in order to achieve a higher accuracy than the weak learners classifiers would have had. In some cases, boosting has been

shown to yield better accuracy than bagging, but it also tends to be more likely to overfit the training data. By far, the most common implementation of boosting is Adaboost, designed by Schapire [Sch01].

In clustering, two approaches are largely used to design consensus functions. One that establishes label correspondence between various partitions and then uses a consensus function; second that eliminates the need of label correspondence and computes the consensus function directly. Most of the work [Fre01, Fre02, HY04, KK98, SG02, TJP03] falls in the second category whereas the works of [KG07, TMbJP04] fall in the first category. Majority voting, co-association, fusion using procedure similar to agglomerative clustering, graph partitioning, statistical and information theoretic methods are several approaches that have been used in literature to design consensus functions for clustering. A detailed survey of consensus functions for clustering can be found in [GSIM09]. The adaptive clustering ensemble technique proposed in [TMbJP04] uses sampling techniques to generate individual partitions of the ensemble. Optimization techniques [GDT09, SMPX10] have also been used to generate ensembles for clustering and projective clustering.

Three heuristics (CSPA, HGPA and MCLA) based on hyper graph partitioning are proposed by Strehl and Ghosh [SG02] wherein clusters in all the schemes are represented by hyperedges in a hyper graph. In the Cluster-based Similarity Partitioning Algorithm (CSPA), a similarity matrix is formed from this hypergraph. It can be viewed as the adjacency matrix of a simple graph having objects as the nodes and edge between nodes has a weight equivalent to frequency of the two objects being grouped together in a cluster. To obtain consensus partition, METIS [KK98] algorithm is used. The Hyper Graph Partitioning Algorithm (HGPA) partitions the hyper graph directly, by eliminating the minimal number of hyper edges. It assumes that all hyper edges have the same weights. Hyper Graphs Partitioning package HMETIS [KAKS99] is used for the same. In the Meta CLustering Algorithm (MCLA), related hyperedges are grouped in metaclusters and then

collapsed. It assigns each object to the collapsed hyperedge in which it participates most strongly. METIS algorithm is used to get metaclusters.

Topchy et al. [TJP04] proposed a probabilistic model of consensus using a finite mixture of multinomial distributions in a space of constructed features. A combined clustering is found as a solution to the corresponding maximum likelihood problem using the Expectation Maximization algorithm.

Hybrid Bipartite Graph Formulation (HBGF) proposed by Fern and Brodley [FB04] is also based on graph partitioning. They model the clusters and the objects in a bipartite graph. There is an edge between an object node and a cluster node if the object belongs to the cluster. METIS is then used for consensus generation.

Dudoit and Fridlyand [DF03] use relabeling and voting to ensemble partitions. A relabeling between two clusterings/partitions is done using the Hungarian algorithm. After relabeling, voting is applied to determine cluster membership for each object.

Fred and Jain [FJ05] proposed to ensemble clustering results in a co-association matrix. The associations between sample pairs are weighted by the number of times they co-occur in a cluster. Consistent clusters are formed using a minimum spanning tree like algorithm using the co-occurrence matrix. They use co-association values and apply a hierarchical (single link) clustering to the co-association matrix. The idea of evidence accumulation for combining the result of multiple clusterings is used.

Co-clustering and projective clustering are problems that are related to biclustering. Though researchers sometimes claim that co-clustering, projective clustering and biclustering are all same but generally solutions for co-clustering do not allow clusters to overlap on objects and features whereas solutions for projective clustering allows overlapping of features but not of objects. Wang et al. [WLDJ11] and Gullo et al. [GDT09] have proposed ensemble techniques for these problems. Wang et al. presented an ensemble solution for co-clustering wherein they extract block-constant biclusters generalizing the grid-style partitions to allow different resolutions in different parts of the data matrix.

A pair of biclusters may overlap on objects or on features but not on both at the same time. Gullo et al. presented an ensemble solution for projective clustering wherein an object may belong to more than one biclusters but the total sum of the membership is one thereby meaning that if an object completely belongs to one bicluster it does not belong to any other. They project the clusters on one dimension in a fuzzy way.

Biclustering is different from these problems/solutions wherein an object/feature may have a total membership more than one and a bicluster is defined by more than one feature. Also, bicluster may overlap both on objects and features simultaneously.

Review of HN algorithm

In a parallel work, Hanczar and Nadif [HN11] proposed the use of bagging to improve the performance of biclustering schemes. HN generates schemes using bootstrapped data and then combines the clusters into metaclusters using Jacquard Index as the similarity measure. They then use voting to generate the consensus.

In another work, Hanczar and Nadif have used triclustering [HN12] to form bicluster ensemble. They represent the collection of biclusters by a 3 dimensional binary matrix with genes, conditions and all the biclusters on the three dimensions. A tricluster is defined as a set of 1's from the 3 dimensional matrix. The aim of the algorithm is to find all the triclusters from this matrix. The algorithm suffers from the anomaly that it does not fare well when the input schemes contain true biclusters. Moreover it looks at local minima whereas the global minima could be far off. They have given a graph showing that the loss function is non-increasing. However, these values of loss function are absolute rather than relative. Also, they discuss that absolute loss function may lead to a condition wherein all feature or examples be removed, and proposes the use of relative values instead. On the other hand, values of relative loss function may not necessarily be non-increasing.

3.4 Evaluating the Results

Different quality measures are used for different scenarios depending on the data and on the availability of ground truth [GVSS03] for traditional clustering. In case the ground truth is known, statistical measures like Rand Index and Jacquard Index are used to adjudge the proximity of the output to the ground truth. These measures generally count the pair of genes that behave similarly in the ground truth as well in the output i.e. they are put in the same cluster by both or in different clusters by both. These measures cannot be extended for biclustering solutions as a pair of genes may be put together in one bicluster as well as be separated at the same time in two different biclusters. Measures used in Pontes et al. [PGAR15] evaluate the quality of biclusters whereas in ensemble algorithms we are interested in determining how close our final ensemble is to input biclustering solution. We define a new measure wherein the result is aligned with the ground truth and the similarity of the two is established using membership of the genes.

3.4.1 Statistical Validation of Biclustering Solutions : Synthetic Data Set

In this section, we describe both types of evaluation metrics. We first define a statistical measure Agreement Score (AS) that captures the overlapping nature of biclusters. Let π denote the biclustering solution that needs to be compared with the ground truth π_G (the implanted biclusters). Align the biclusters of π and π_G ignoring the extra biclusters on either side and calculate the ratio of number of agreements to the sum total of agreements and disagreements for all pairwise aligned biclusters. Mathematically it can be written as:

$$AS = \frac{1}{\#\ of\ biclusters} \sum_{\langle BC_r, BC_s \rangle \in Align(\pi_G, \pi)} \frac{a_{rs} + b_{rs}}{a_{rs} + b_{rs} + c_{rs} + d_{rs}}$$

where

$$Align(\pi_G, \pi) = \{\langle BC_r, BC_s\rangle : BC_r, BC_s \, are \, aligned \, biclusters \, of \, \pi_G \, \& \, \pi \, resp.\}$$

a_{rs} is the number of genes belonging to both the biclusters BC_r and BC_s,
b_{rs} is the number of genes belonging to neither of BC_r and BC_s,
c_{rs} is the number of genes that belong to the bicluster BC_r but do not belong to the bicluster BC_s and,
d_{rs} is the number of genes that do not belong to the bicluster BC_r but belong to the bicluster BC_s.

Thus $a_{rs} + b_{rs}$ denote the agreement between the two biclusters and $c_{rs} + d_{rs}$ denote the disagreement between them.

Agreement score aims at finding the similarity between the two biclustering schemes. It takes into account gene agreement between aligned biclusters of the schemes. As a gene may contribute more than once to the similarity measure, Agreement score(AS) is normalized by dividing it by the number of biclusters. The value of AS thus lies between 0 and 1. The value 0 indicating that the two biclustering schemes do not agree at all and 1 indicating that the two biclusterings are same.

We use another measure called BCE (biclustering error) that takes both genes and conditions into account to validate biclustering solutions. The total number of misclassified values (g,c pairs) in the aligned biclusters amounts to the biclustering error between two biclustering solutions. Let π denote the biclustering solution that needs to be compared with the ground truth π_G (the implanted biclusters). Align the biclusters of π and π_G ignoring the extra biclusters on either side. BCE is calculated as follows

$$BCE = \sum_{\langle BC_r, BC_s\rangle \in Align(\pi_G, \pi)} err(BC_r, BC_s)$$

where

$$err(BC_r, BC_s) = |G_r|\,|C_r| + |G_s|\,|C_s| - 2\,|G_r \cap G_s|\,|C_r \cap C_s|$$

$$G_r, G_s \; are \; genes \; of \; BC_r \; \& \; BC_s \; respectively \; ,$$

$$C_r, C_s \; are \; conditions \; of \;\; BC_r \; \& \; BC_s \; respectively \; and$$

$$|X| \;\; denotes \; the \; cardinality \; of \; X.$$

More is the number of the genes and conditions that do not match with the classes in which they ought to be, more is the error and less significant is the biclustering solution.

3.4.2 Biological Validation of Biclustering Solutions : Real Data Set

In the absence of ground truth, external measures are used for validation of biclustering solutions. External validation methods like GO annotation term [LW07], metabolic pathways [BIB03], protein protein interaction network [PBZ$^+$06] and patterns in promoter regions [THC$^+$99] are commonly used to assess the quality of biclusters. These methods are based on the hypothesis that a group of genes that are related are responsible for some biological function in a cell. We have used gene ontology terms and motif analysis to validate our biclusters. Both these tools use p-value to find the significance of the biclusters. We start with the explanation of p-value followed by description of the tools used for validation of biclusters.

p-value

Using p-value we determine confidence in the result. It is done using hypothesis testing, wherein we use a test statistics that indicates how strongly the data we observe supports our decision. The p-value was first formally introduced by Karl Pearson but its use in statistics was popularised by Ronald Fisher.

The GO terms/motifs shared by the genes in the user's list are compared to the background distribution of the annotation. It is the probability of seeing x or more genes from

the input list of n genes annotated to a particular GO term/motif, given the proportion of genes in the whole genome annotated to that GO term/motif is F out of G. Specifically, hyper geometric distribution is used to calculate the probability of observing at least x or more genes from a functional category from an input gene list of size n given the background database consists of G genes out of which F belong to the functional category. p-value is given by

$$\sum_{j=x}^{n} \frac{\binom{F}{j}\binom{G-F}{n-j}}{\binom{G}{n}}$$

This is same as calculating the chance of getting at least x successes and can also be represented as

$$1 - \sum_{j=0}^{x-1} \frac{\binom{F}{j}\binom{G-F}{n-j}}{\binom{G}{n}}$$

It is clear that smaller the p-value, more significant is the association of the particular GO term/motif with the group of genes (i.e. it is less likely that the observed annotation of the particular GO term/motif to a group of genes occurs by chance). There may be several GO terms/motifs with different p-values associated with an input set of genes belonging to a bicluster. The best p-value for each category was used to compare the biclusters.

Gene Ontology terms analysis using DAVID Toolbox

There are three Gene Ontologies (GO) that form a common language for annotation of genes of different organisms from yeast to human. They relate genes with different biological processes across different species. The three GO ontologies are (i) **Biological process** which include biological functions to which a gene or a gene's products contribute; (ii) **Cellular component** which includes complex sub-cellular structures, locations and macro-molecular complexes like RNA polymerases where the gene products are active; (iii) **Molecular function** which defines the biochemical activities like carbo-

hydrates binding and ATPase activity, of the gene products at molecular level. A **GO term** is annotated to a gene group that is responsible for a particular biological function.

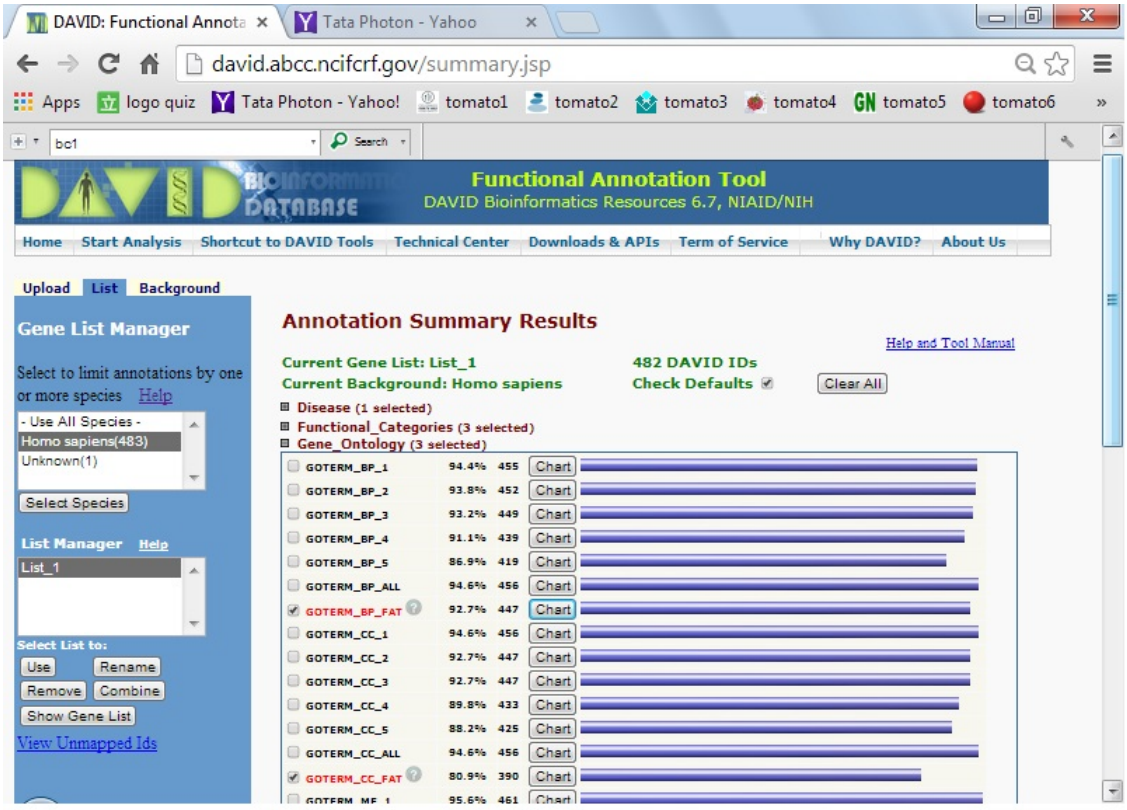

Figure 3.1: Snapshot of Functional Annotation tool of DAVID.

DAVID (Database for Annotation, Visualization and Integrated Discovery) is a free online bioinformatics resource consisting of knowledge database and analytical tool that helps in extracting biological relevance of a set of genes [HSL08]. The knowledge database integrates major public bioinformatics resources. DAVID's knowledge base collects and integrates diverse gene annotation categories, assigns a centralized internal DAVID identifier to each of them in a non redundant manner. The wide range of biological annotation coverage in the DAVID knowledge base enables a user's gene ID to be mapped across the entire database thus providing a broad coverage of gene associated annotation. Also, if a significant portion ($> 20\%$) of input gene IDs fail to be mapped to

an internal DAVID ID, another DAVID tool, the Gene ID conversion tool starts up to help in the mapping of such IDs.

The Functional Annotation tool of DAVID as shown in Figure 3.1 is used for enrichment analysis of the gene terms annotated for the input gene set. The basic principle behind the enrichment analysis is that if a biological process is active/abnormal then the co-functioning genes have a higher chance of being selected as a relevant group. To decide about the degree of enrichment, a certain background has to be setup for comparison. As per Huang et al. [HSL08] larger backgrounds e.g. the total genes in the genome as a background tends to give more significant p-values as compared to narrowed down set of genes as background. DAVID has an automatic procedure to determine the background

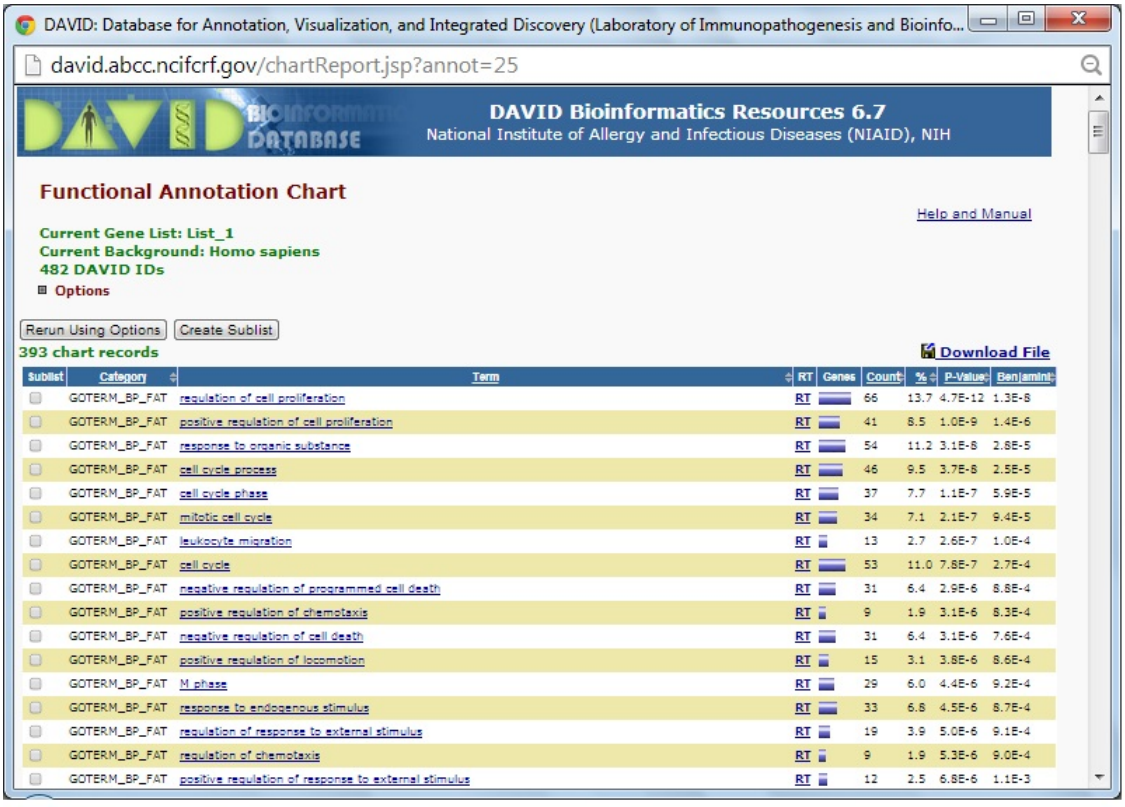

Figure 3.2: Snapshot of Functional Annotation chart.

as the global set of genes in the genome on the basis of the user's uploaded gene list. Thus

normally a user does not have to setup a population background by itself. Uploading the gene lists of the bicluster is the first step of analysis. DAVID maps a number of genes in the uploaded list to the associated biological annotation i.e. **gene ontology terms**. It then statistically examines the enrichment of gene members for each of the annotation terms by comparing the outcome to the reference background using p-values. Lower is the p-value, more statistically significant is the bicluster. Annotation terms below a certain threshold are reported as shown in Figure 3.2.

Motif analysis using RSA Toolbox

A set of genes showing similar behavior indicates that they are active or expressed together. A gene becomes active when a **transcription factor** (protein that accounts for gene regulation) binds to a **Transcription Factor Binding Site (TFBS) or motif** in the promoter region of the gene. Thus the genes responsible for one biological activity and hence belonging to a bicluster are expected to have shared elements/patterns/motifs. In order to further validate our biclusters we performed motif analysis of the genes of the biclusters using **Regulatory Sequence Analysis Toolbox (RSAT)**. Given a set of input genes, RSAT provides motifs, if any, in their regulatory region along with their p-values. **RSAT** consists of many modular tools for sequence retrieval and motif discovery. These

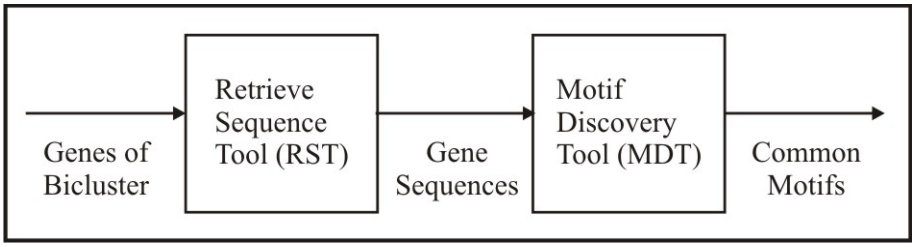

Figure 3.3: Motif analysis using RSAT.

tools can either be accessed separately or be connected in a pipeline. Two of these tools are **Retrieve Sequence Tool (RST)** and **Motif Discovery Tool (MDT)**. Figure 3.3 sum-

marizes the working of RSAT.

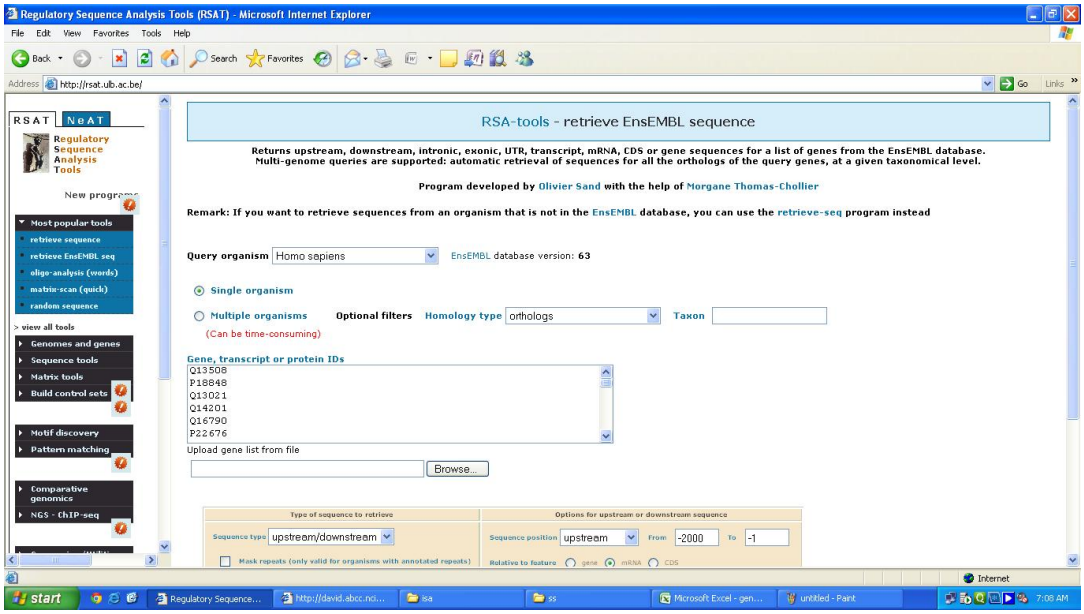

Figure 3.4: Snapshot of RSAT

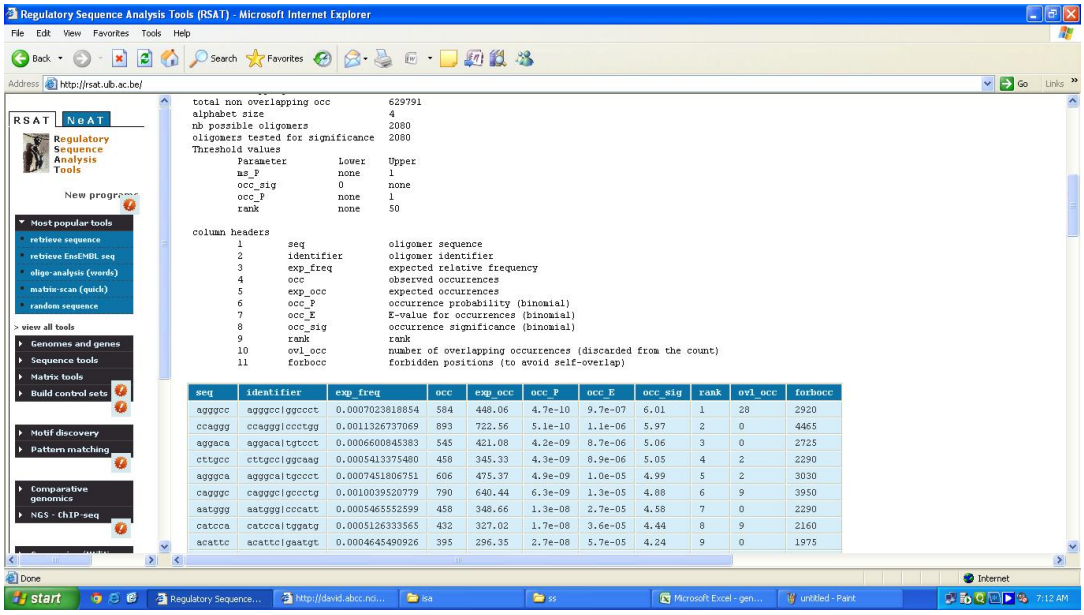

Figure 3.5: Snapshot of motif discovery tool of RSAT.

A set of genes along with the name of the organism is provided as an input to RST as shown in Figure 3.4. RST provides the sequences of the input genes as output which is then fed to MDT to extract the motifs. The output of MDT includes the motifs and their corresponding p-values as shown in Figure 3.5. The value gives the statistical significance of the motif detected. It is the expected number of times a similarity would be observed by chance in a target database of random motifs.

3.5 Data Sets

We considered synthetic data sets used by Prelic et al. in [PBZ+06]. Details of the data sets are given in Table 3.1.

Code-Data Set	Size(N*d)	# implanted biclusters(k)
DS1- Prelic(without noise)	110*110	11(overlapping)
DS2- Prelic(with noise)	100*50	10(non overlapping)

Table 3.1: Synthetic Data Sets.

The heat map of both the data sets of Prelic are shown in Figures 3.6 and 3.7.

We also worked on four real data sets for our study including a eukaryote, a plant and homosapiens. Table 3.2 gives the details of the real data sets.

Saccharomyces Cerevisiae (Yeast) is a safe, easy to grow, short generation time organism [Hun93]. As yeasts are eukaryotes and are biochemically similar to humans, they are quite popular with biologists for study purposes. Yeast data sets examines gene expression behaviour during various stress conditions. Expression profiles were normalized by subtracting the mean of each profile and dividing it by the standard deviation across the time points.

Arabidopsis Thaliana is a common weed which undergoes the same processes of growth, development, flowering etc. as most of the higher plants and yet has a small

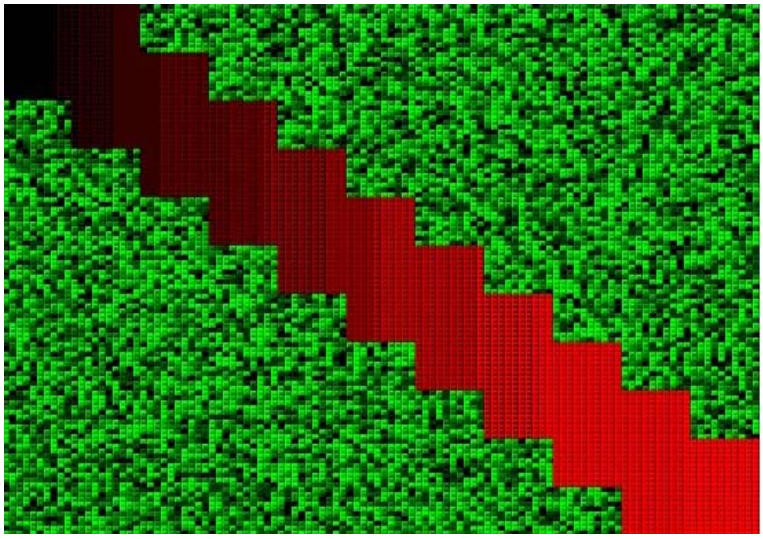

Figure 3.6: Heat map of overlapping data set of prelic-DS1.

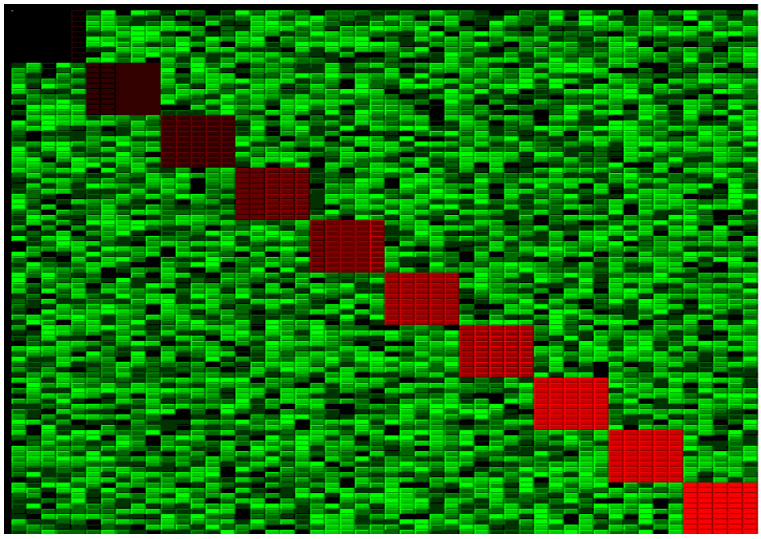

Figure 3.7: Heat map of non overlapping data set of prelic-DS2.

genome. It produces a large number of seeds and grows to a mature plant in only about six weeks. The microarray is designed to examine the expression profile of identified small coding genes and all annotated genes in various organs and various stress condition.

Diffuse Large B-cell Lymphoma data set contains the gene expression profiles of the lymphomas of patients after chemotherapy [RWC+02].

Human Breast Cancer data set aims at predictive gene signature for the outcome of a breast cancer therapy [vtVDvdV+02]. The data set contains gene expression profiles of the premalignant, preinvasive, and invasive stages of human breast cancer.

Organism (Short Name)	Genes(N)	Conditions(d)	source
Saccharomyces Cerevisiae (Yeast)	2993	173	www.tik.ee.ethz.ch/sop/bicat
Arabidopsis Thaliana (A. Thaliana)	734	69	www.tik.ee.ethz.ch/sop/bicat
Diffuse Large-B-cell Lymphoma (DLBCL)	661	180	www.bioinf.jku.at/software/fabia
Human Breast Cancer (Breast Cancer)	1213	97	www.bioinf.jku.at/software/fabia

Table 3.2: Real Data Sets.

Chapter 4

BiETopti - Biclustering Ensemble Technique using Optimization

The chapter presents our first approach towards ensembling biclustering solutions named BiETopti. Optimization technique has been used for the ensembling purpose. The algorithm starts by generating a number of schemes using a biclustering algorithm. Different schemes may assign different labels to similar biclusters. Thus the schemes are first aligned so that similar biclusters in different schemes have the same label. A pool of global labels is formed and a local label of an individual scheme is mapped to a global label. The consensus is then obtained by minimizing the dissimilarity between the obtained biclusters and the aligned input biclusters. Manhattan distance is used to capture the dissimilarity. The objective function consists of two terms: one that minimizes the dissimilarity amongst genes and the other to minimize the dissimilarity amongst conditions. The sum thus minimizes the combined dissimilarity over genes and conditions.

Experiments were performed both on synthetic data sets and real data sets to show the efficiency of our algorithm. The biclusters produced by our algorithm were found to be better than the best of the input schemes. We also compared the biclusters produced by our algorithm with those produced by Hanczar and Nadif (HN) [HN11] on their synthetic

data set. It was found that $BiETopti$ not only outperformed HN in terms of quality but it also takes lesser time. Details of our approach are presented in Section 4.1. Experimental results are discussed in Section 4.2.

4.1 BiETopti

The algorithm works in three phases. Phase I deals with the generation of input schemes. Schemes are generated by running a biclustering algorithm several times with different initializations. In phase II, biclusters of two schemes are relabeled and aligned so that similar biclusters in two schemes have the same label. A statistical measure was defined by Krumpleman and Ghosh [KG07] to capture similarity between clusters. The measure takes into account the non-disjointness of clusters but does so only amongst the genes. To see the behaviour of this method on biclusters, it was applied on two biclustering solutions as shown in Table 4.1. We observe that bicluster A1 is aligned with bicluster B1 and bicluster A2 is aligned with the bicluster B2 considering only the genes in common. This alignment is shown in Table 4.2(a). However similarity between biclusters should be computed by considering gene condition pairs. Thus, this measure is modified for our algorithm to capture the overlap of genes as well as conditions. The modified measure aligns A1 with B2 and A2 with B1 as shown in Table 4.2(b). Hungarian method is then used to relabel the biclusters so as to align similar biclusters with each other. The consensus is then generated using optimization in phase-III. Architecture of the approach is given in Figure 4.1.

Review of Hungarian Algorithm

Hungarian method is a combinatorial optimization algorithm that solves assignment problem. Assume that there are N workers to whom N jobs are to be assigned. For each pair (worker, job) we know the salary that is to be paid to the worker to perform the job.

BC#	Group of	Biclustering Solution1	Biclustering Solution2
1	genes conditions	1 2 3 4 5 1 2 3 **A1**	1 2 3 4 5 **B1**
2	genes conditions	4 5 6 4 5 **A2**	4 5 1 2 **B2**

Table 4.1: Sample Biclustering Solutions

Biclustering Solution1	Biclustering Solution2
1 2 3 4 5 1 2 3 **A1**	1 2 3 4 5 **B1**
4 5 6 4 5 **A2**	4 5 1 2 **B2**

Biclustering Solution1	Biclustering Solution2
1 2 3 4 5 1 2 3 **A1**	4 5 1 2 **B2**
4 5 6 4 5 **A2**	1 2 3 4 5 **B1**

Table 4.2: Alignment using a) Krumpleman and Ghosh method. b) Modified form of Krumpleman and Ghosh method.

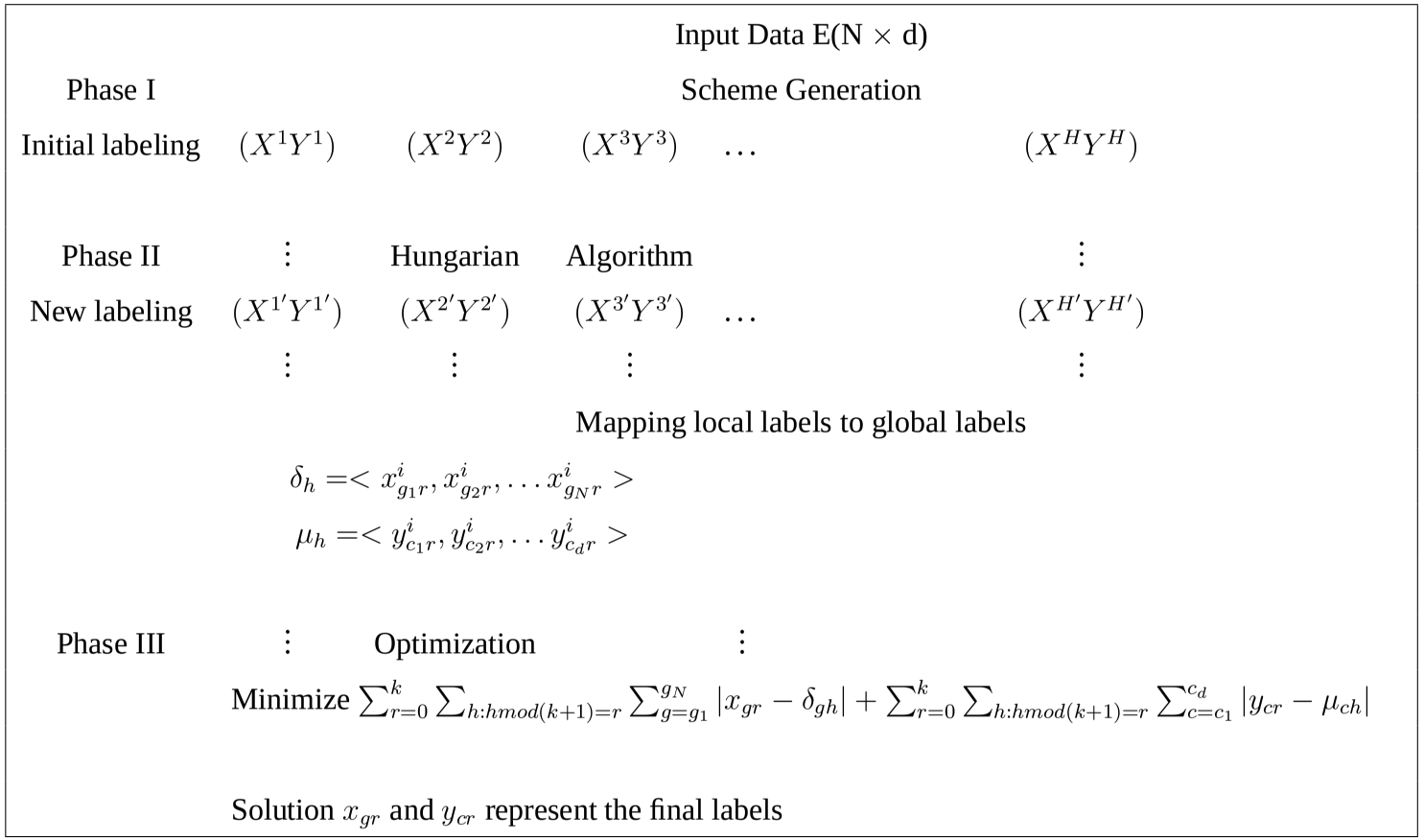

Figure 4.1: Architecture of BiETopti.

The goal is to find the lowest cost solution of getting all the jobs done by assigning each worker to exactly one job. The Hungarian algorithm does this by solving a minimum weight bipartite matching.

4.1.1 Some Notations

In this section, some notations used in the chapter are presented.

Definition 4.1.1 (Biclustering Solution/Input Scheme) *A biclustering solution π defined over expression matrix E is a triplet (k, X, Y):*

1. *k is the number of biclusters, $\{1 \dots k\}$ denote the set of bicluster labels.*
2. *$X : G \times \{1 \dots k\} \rightarrow \{0, 1\}$ is a function where $X(g, l)$ represents whether a gene g belongs to the bicluster labeled l.*
3. *$Y : C \times \{1 \dots k\} \rightarrow \{0, 1\}$ is a function where $Y(c, l)$ represents whether a condition c belongs to the bicluster labeled l.*

For the ease of presentation we will use x_{gl} to denote $X(g, l)$ and y_{cl} to denote $Y(c, l)$.

Definition 4.1.2 (Collection of Input Schemes) *Π is a collection of H input schemes, $\Pi = \{\pi_1 \dots \pi_H\}$, where each of $\pi_i = (k_i, X^i, Y^i)$ is a biclustering solution/ input scheme.*

Since the biclusters may overlap both on genes and conditions, a gene (/ condition) may be assigned more than one label. Also, there may be a gene (/ condition) which does not belong to any bicluster, such a gene (/ condition) is assigned a special label 0. Hence, an extra bicluster with label 0 is added in each input scheme. Note that the number of the biclusters increase by 1 in each scheme. Thus without loss of generality, we can assume that in each scheme, each gene and each condition belongs to at least one bicluster.

Definition 4.1.3 (Global Label Set) *is a set L of labels* $\{0, \ldots, (\sum_{i=1}^{H}(k_i + 1) - 1)\}$.

In general, different biclustering schemes may contain different number of biclusters. However, we have considered schemes with equal number of biclusters i.e. $k_i = k \; \forall \; i$. The problem of bicluster ensemble is to derive a consensus that combines the H biclustering solutions and delivers a biclustering solution $\hat{\pi} = (k, X, Y)$.

4.1.2 Phase-I: Scheme Generation

Algorithms yielding maximum information about the data are preferred. Aggregation of similar schemes is of no use. Schemes that are to be ensembled must be different as it adds to diversity which is essential for ensembling. To ensure diversity, different schemes may be produced by applying different biclustering algorithms on the same data set. Applying same biclustering algorithm on bootstrapped data also create different schemes. Schemes may be obtained by applying same algorithm on the same data by varying the threshold parameters/initialization.

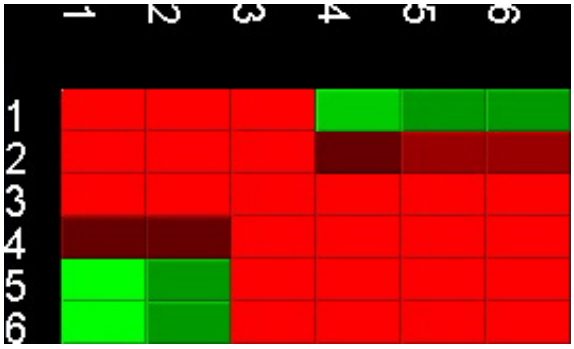

Figure 4.2: Heat Map of Toy Data.

We generate schemes without bootstrapping wherein a biclustering algorithm is applied on the data set several times, each time changing the parameters/initialization. The schemes generated are represented in the form as described in Section 4.1.1.

The algorithm is explained with the help of a toy data containing 6 genes and 6 conditions. Figure 4.2 shows the heat map of the toy data and the two possible biclustering schemes of the same are shown in Table 4.3.

BC#	Group of	Biclustering Solution1	Biclustering Solution2
1	genes	1 2 3 4	2 3 4
	conditions	1 2 3 4	2 3 4
2	genes	3 4 6	1 2 3
	conditions	3 4 5 6	1 2 3
3	genes	3 4	4 5 6
	conditions	3 4	4 5 6

Table 4.3: Two Biclustering schemes of Toy Data.

4.1.3 Phase II: Label Correspondence

In the absence of labeled data, different schemes assign different labels to genes and conditions. Therefore, we need to establish correspondence between the labels so as to align similar biclusters of different input schemes. One of the schemes is used as a reference and biclusters of other schemes are relabeled so that they are aligned with the similar biclusters of the reference scheme. Biclusters belonging to different solutions having maximum overlap with each other need to be aligned and given the same label. This was the intuition underlying the probability based alignment function proposed by Krumpleman and Ghosh [KG07] for clustering solutions. We start with review of the alignment function and suggest how it can be modified to suit the need of biclustering solutions.

Review of Krumpleman and Ghosh's (KG) alignment function

Krumpleman and Ghosh are the first ones to define a statistical measure to capture the overlap between two clusters, wherein a clusterer may produce non-disjoint clusterings, to establish cluster correspondence before applying majority voting to design the consensus function for the ensemble. The measure they used is able to capture the overlap for genes. The function defined by them measures the probability of an event occurring by chance. It is defined as the total probability of seeing the observed overlap(s) or greater between the two clusters. This value essentially measures the likelihood of the observed overlap being a random event, hence a small value indicates a small probability of seeing the observation at random and show higher similarity.

$$\sum_{t=s}^{t=min(d1,d2)} P(t) \;\; where \;\; P(s) = \frac{\binom{N}{d1}\binom{d1}{s}\binom{N-d1}{d2-s}}{\binom{N}{d1}\binom{N}{d2}}$$

Here $d1$, $d2$ are number of objects in clusters C_1, C_2 respectively. s is number of objects overlapping between C_1 and C_2 and N is total number of objects in data set.

The above formula calculates the number of ways of choosing the observed overlap s out of the total number of ways in which two clusters of sizes $d1$ and $d2$ respectively can be formed. The denominator represents the number of ways of choosing two clusters of sizes $d1$ and $d2$ respectively and the numerator finds the number of ways of choosing the observed overlap s. It is actually the product of three terms as explained here: Count number of ways in which $d1$ $1's$ can be selected from N, followed by the number of ways to choose the s overlapping points from these $d1$ $1's$ and last term counts the number of ways to place remaining $1's$ in C_1 such that they do not overlap with $1's$ in C_2.

(gene,condition) pair	Biclustering Scheme1 Labels			Biclustering Scheme2 Labels		
	BC1	BC2	BC3	BC1	BC2	BC3
(1,1)	1	0	0	0	1	0
(1,2)	1	0	0	0	1	0
(1,3)	1	0	0	0	1	0
(1,4)	1	0	0	0	0	0
(1,5)	0	0	0	0	0	0
(1,6)	0	0	0	0	0	0
(2,1)	1	0	0	0	1	0
(2,2)	1	0	0	1	1	0
(2,3)	1	0	0	1	1	0
(2,4)	1	0	0	1	0	0
(2,5)	0	0	0	0	0	0
(2,6)	0	0	0	0	0	0
(3,1)	1	0	0	0	1	0
(3,2)	1	0	0	1	1	0
(3,3)	1	1	1	1	1	0
(3,4)	1	1	1	1	0	0
(3,5)	0	1	0	0	0	0
(3,6)	0	1	0	0	0	0

(gene,condition) pair	Biclustering Scheme1 Labels			Biclustering Scheme2 Labels		
	BC1	BC2	BC3	BC1	BC2	BC3
(4,1)	1	0	0	0	0	0
(4,2)	1	0	0	1	0	0
(4,3)	1	1	1	1	0	0
(4,4)	1	1	1	1	0	1
(4,5)	0	1	0	0	0	1
(4,6)	0	1	0	0	0	1
(5,1)	0	0	0	0	0	0
(5,2)	0	0	0	0	0	0
(5,3)	0	0	0	0	0	0
(5,4)	0	0	0	0	0	1
(5,5)	0	0	0	0	0	1
(5,6)	0	0	0	0	0	1
(6,1)	0	0	0	0	0	0
(6,2)	0	0	0	0	0	0
(6,3)	0	1	0	0	0	0
(6,4)	0	1	0	0	0	1
(6,5)	0	1	0	0	0	1
(6,6)	0	1	0	0	0	1

Table 4.4: Membership Matrices.

Modifying the KG's alignment function for biclusters

Borrowing the notation from them, we extended their definition to take into account the joint overlapping of biclusters on genes and conditions. Let BC_i and BC_j be the biclusters of π_{ref} and π_l respectively. We define Bicluster Similarity Index (BSI) to compute the similarity between BC_i and BC_j. Let $N^{'}$ be the total number of (g, c) pairs. Let $d1$ and $d2$ be the number of (g, c) pairs in BC_i and BC_j respectively, i.e. $d1 = (\sum_g x_{gi})(\sum_c y_{ci})$ and $d2 = (\sum_g x_{gj})(\sum_c y_{cj})$. Let s be the number of (g, c) pairs common between BC_i and BC_j. The probability of the observed overlap being s is then given by

$$P_{ij}(s) = \frac{\binom{N^{'}}{d1}\binom{d1}{s}\binom{N^{'}-d1}{d2-s}}{\binom{N^{'}}{d1}\binom{N^{'}}{d2}} = \frac{\binom{d1}{s}\binom{N^{'}-d1}{d2-s}}{\binom{N^{'}}{d2}}$$

The above expression is the hyper-geometric distribution evaluated at s. Total probability of seeing at least s overlapping (g, c) pairs is then

$$BSI_{ij} = \sum_{t=s}^{min(d1,d2)} P_{ij}(t)$$

This can be obtained by taking a scalar product of the corresponding columns (third dimension) of the binary membership matrices of the two schemes. Binary membership matrix M is defined as a three dimensional matrix with genes, conditions and labels on the three dimensions. Hence each membership matrix is of size $N \cdot d \cdot k$ for an input scheme with k biclusters. The two binary membership matrix for the two schemes of the toy example are shown in Table 4.4. Using the notation of a 3D matrix, m_{ijr} denotes whether the gene g_i and the condition c_j belong to the r^{th} bicluster or not. Let M_{ref} and M_l be two binary membership matrices for the input schemes π_{ref} and π_l $(l = 1 \ldots H)$ respectively. The objective is to align the third dimension of M_l with that of M_{ref} such that they have maximum match. Hamming distance appears to be the most reasonable choice. However, it does not account for the different densities of 1's in different columns.

Clearly, higher the overlap, lower is the value of BSI. We compute the pairwise BSI_{ij} for all pairs of biclusters with one bicluster taken from π_{ref} and the other from π_l.

The calculation of BSI for the first bicluster of the two schemes of toy data is shown here: (Refer to BC1 column of both the schemes in Table 4.4.)

According to the formula

$$BSI_{ij} = \sum_{t=s}^{min(d1,d2)} P_{ij}(t)$$

Substituting the values $N^{'}$=36, $d1$ = 16, $d2$ = 9, s=9

$$= \sum_{t=9}^{9} P_{ij}(t)$$

$$and\ P_{ij}(s) = \frac{\binom{N^{'}}{d1}\binom{d1}{s}\binom{N^{'}-d1}{d2-s}}{\binom{N^{'}}{d1}\binom{N^{'}}{d2}} = \frac{\binom{d1}{s}\binom{N^{'}-d1}{d2-s}}{\binom{N^{'}}{d2}} \therefore = \frac{\binom{16}{9}\binom{20}{0}}{\binom{36}{9}} = .00012$$

We then define a bipartite graph with biclusters of π_{ref} as vertices on one side and biclusters of π_l as vertices on other side. BSI_{ij} is the weight of the edge between i^{th} bicluster of π_{ref} and j^{th} bicluster of π_l. This is shown in Figure 4.3.

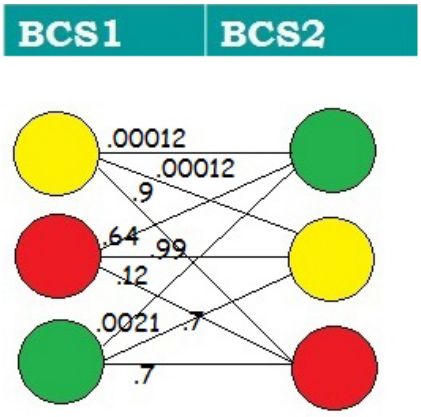

Figure 4.3: BSI between 2 schemes for the toy data. Circles correspond to biclusters in a scheme.

Hungarian method is used to compute minimum weight bipartite matching to obtain the new labels $(X^{'}Y^{'})$ for the biclusters of π_l. This is illustrated in Figure 4.4. Step by step procedure for same is given in Algorithm 1.

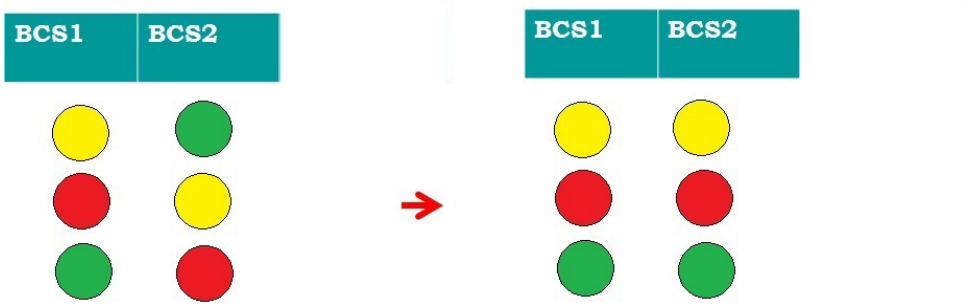

Figure 4.4: Labels before and after applying Hungarian algorithm.

Input: $(X^1Y^1), (X^2Y^2), \ldots, (X^HY^H)$: old labels of H input schemes

Output: $(X^{1'}Y^{1'}), (X^{2'}Y^{2'}), \ldots, (X^{H'}Y^{H'})$: new labels of H input schemes

`/* without loss of generality assume` π_{ref} `is` π_1`. */`

for l = 2 **to** H **do**

  for i = 1 **to** k **do**

    for j = 1 **to** k **do**

      calculate BSI_{ij} between the i^{th} and the j^{th} biclusters of π_{ref} and π_l biclustering schemes respectively;

    end

  end

  use Hungarian algorithm with BSI_{ij} as the edge weights and relabel the biclusters of π_l;

end

Algorithm 1: Label Correspondence.

4.1.4 Phase III: Generating Consensus

In this section we present our main contribution. We formulate the problem of generating a consensus from the input schemes as an optimization problem. We start by mapping the local labels to global labels as described below:

Consider the r^{th} bicluster in i^{th} input scheme. The value of r ranges between 0 and k whereas i takes values from 1 to H. The bicluster gets the global label h where h satisfies $i = \lfloor h/(k+1) \rfloor + 1,\ r = h\ mod\ (k+1)$. Figure 4.5 shows the mapping of local labels to global labels.

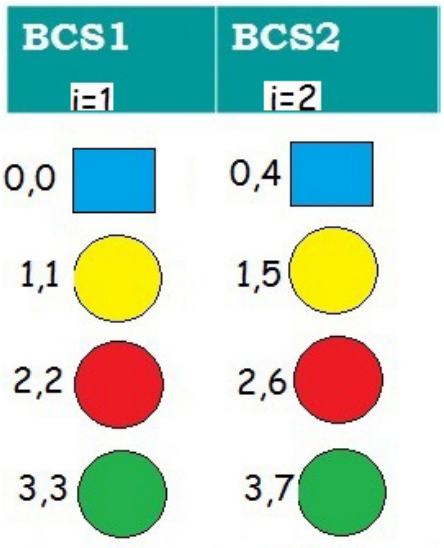

Figure 4.5: Local labels mapped to global labels (pairs (r, h) : (local label r, global label h)).

The gene-wise representation of the collection of input schemes is then given by the following $N \times 1$ vectors

$\delta_h =< \delta_{g_1 h}, \delta_{g_2 h}, \ldots \delta_{g_N h} > where\ \delta_{g_j h} = x^i_{g_j r}$

and the condition-wise representation is given by the following $d \times 1$vectors

$\mu_h =< \mu_{c_1 h}, \mu_{c_2 h}, \ldots \mu_{c_d h} > where\ \mu_{c_j h} = y^i_{c_j r}$

The δ and μ for the toy data are given in Tables 4.5 and 4.6 respectively. We next

	X^1				X^2			
$h \rightarrow$	0	1	2	3	4	5	6	7
Genes$\downarrow$	$r=0$	$r=1$	$r=2$	$r=3$	$r=0$	$r=1$	$r=2$	$r=3$
1	0	1	0	0	0	0	1	0
2	0	1	0	0	0	1	1	0
3	0	1	1	1	0	1	1	0
4	0	1	1	1	0	1	0	1
5	1	0	0	0	0	0	0	1
6	0	0	1	0	0	0	0	1

Table 4.5: Bicluster Collection Representation δ for the example (r is local label, h is global label).

	Y^1				Y^2			
$h \rightarrow$	0	1	2	3	4	5	6	7
Conditions$\downarrow$	$r=0$	$r=1$	$r=2$	$r=3$	$r=0$	$r=1$	$r=2$	$r=3$
1	0	1	0	0	0	0	1	0
2	0	1	0	0	0	1	1	0
3	0	1	1	1	0	1	1	0
4	0	1	1	1	0	1	0	1
5	0	0	1	0	0	0	0	1
6	0	0	1	0	0	0	0	1

Table 4.6: Bicluster Collection Representation μ for the example.

present the formulation of our problem. Let x_{gr} be an indicator variable that denotes whether gene g gets the label r or not. Similarly, let y_{cr} be an indicator variable that denotes whether condition c gets the label r or not. The objective is to assign labels to genes and conditions so as to minimize the dissimilarity of the obtained biclusters from the corresponding aligned biclusters. The objective function consists of two terms, one for dissimilarity over genes and the other to capture the dissimilarity over conditions. We obtain the new labeling x_{gr} and y_{cr} so as to

Minimize

$$\sum_{r=0}^{k} \sum_{h:hmod(k+1)=r} \sum_{g=g_1}^{g_N} |x_{gr} - \delta_{gh}| + \sum_{r=0}^{k} \sum_{h:hmod(k+1)=r} \sum_{c=c_1}^{c_d} |y_{cr} - \mu_{ch}|$$

subject to

$$\sum_{g=g_1}^{g_N} x_{gr} \geq 1 \quad \forall 0 \leq r \leq k \quad (1)$$

$$\sum_{c=c_1}^{c_d} y_{cr} \geq 1 \quad \forall 0 \leq r \leq k \quad (2)$$

$$\sum_{r=1}^{k} x_{gr} \geq 1 \quad \forall g \in G \quad (3)$$

$$\sum_{r=1}^{k} y_{cr} \geq 1 \quad \forall c \in C \quad (4)$$

Note that $|x_{gr} - \delta_{gh}|$ contributes 1 to the dissimilarity over genes if x_{gr} is not in agreement with δ_{gh} i.e. (if x_{gr} =1 and δ_{gh} =0) or (if x_{gr} =0 and δ_{gh} =1); first condition corresponds to the case when g gets the label r in our solution but g does not get label r in one of the input schemes. The second condition corresponds to the case when g is not assigned the label r by our solution but it got label r by one of the input schemes. And $|x_{gr} - \delta_{gh}|$ contributes 0 to the dissimilarity over genes if x_{gr} is in agreement with δ_{gh} i.e. if both x_{gr} and δ_{gh} are 1 or 0. Thus for every gene g, $\sum_h |x_{gr} - \delta_{gh}|$ counts the number of schemes with which our solution does not agree on g and $\sum_g \sum_h |x_{gr} - \delta_{gh}|$ represents the total disagreement of our solution with the input schemes over all genes.

Similarly $|y_{cr} - \mu_{ch}|$ contributes 1 to the dissimilarity over conditions if y_{cr} is not in agreement with μ_{ch} i.e. (if y_{cr} =1 and μ_{ch} =0) or (if y_{cr} =0 and μ_{ch} =1). First condition

corresponds to the case when c gets the label r in our solution but c does not get label r in one of the input schemes. The second condition corresponds to the case when c is not assigned the label r by our solution but it got label r by one of the input schemes. $|y_{cr} - \mu_{ch}|$ contributes 0 to the dissimilarity over conditions if y_{cr} is in agreement with μ_{ch} i.e. if both y_{cr} and μ_{ch} are 1 or 0. For every condition c, $\sum_h |y_{cr} - \mu_{ch}|$ counts the number of schemes which agree with our solution on c and $\sum_c \sum_h |y_{cr} - \mu_{ch}|$ represents the total agreement of our solution with the input schemes over all conditions.

The constraints (1) and (2) make sure that each bicluster has at least 1 gene and 1 condition. The constraints (3) and (4) make sure that each gene and each condition belongs to at least one bicluster. The constraints are linear whereas the objective function contains absolute terms (modulus) which can be replaced with squares of the terms and solved as a quadratic program. Though in general the problem is hard but we solve it as a relaxed LP as our coefficient matrix is totally unimodular. A matrix A is totally unimodular if every square submatrix has determinant 0, 1, or -1.

4.2 Experimental Results

In this section, we present experimental evaluation of our approach. We first compare the performance of $BiETopti$ with HN on dataset used by them and show that $BiETopti$ outperforms HN both in terms of time and quality. We then present the results of $BiETopti$ on synthetic data set of Prelic and real data sets.

4.2.1 Methodology

To compare the performance of our approach with HN, we used their code (provided to us by the authors on our request) and data set (DS0) used by them. DS0 has 200 genes and 100 conditions with two overlapping biclusters implanted in it. The implementation of HN code is in R environment [R C12]. Our approach, $BiETopti$ is implemented in

MATLAB version 7.10 (R2010a) on Intel Core i5-2430M CPU @2.40 Ghz with 4GB RAM using Windows 7 Home Basic Operating System and for the optimization part, LINGO [LIN06] tool was used.

4.2.2 Comparison with HN

First we compare the performance of $BiETopti$ with that of HN on bootstrapped data. CC algorithm was used to generate the input schemes on bootstrapped samples of data set (DS0). The ensemble formed is compared with the 'Single' to adjudge the quality of the ensemble where 'Single' is the scheme obtained when algorithm CC is run on DS0.

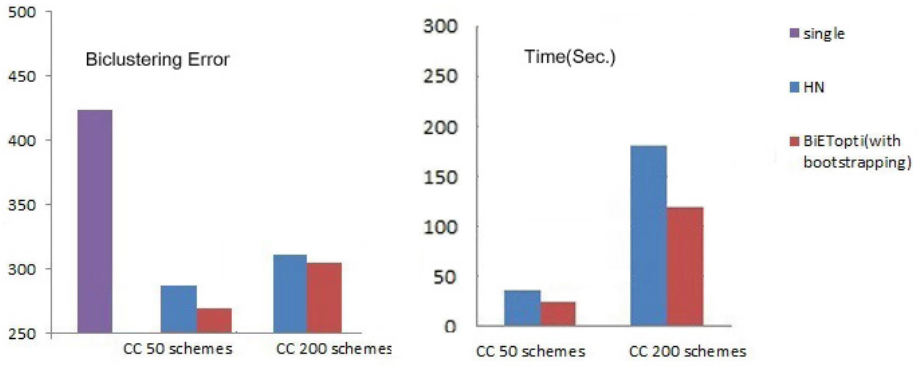

Figure 4.6: Single versus (HN and BiETopti) on DS0 using CC.

Two setups, one with 50 bootstrapped samples of the data set and other with 200 samples of the data set (as defined in their paper), were used for the purpose. Figure 4.6 compares the biclustering error (misclassified (g, c) pairs) of the biclusters generated by CC (single) and that of ensemble of $BiETopti$ and ensemble of HN. The time taken by the two algorithms is also shown in the figure. The figure shows that $BiETopti$ not only improves upon the biclustering error over the 'Single' but it also outperforms HN. In case of 50 samples, the performance of $BiETopti$ is significantly better than that of HN. Though the improvement in the quality of biclusters, in case of 200 samples, is marginal, $BiETopti$ beats HN in terms of time here significantly. Thus, $BiETopti$ not

only provides better biclusters than HN, it also saves hugely on time as compared to HN.

Next, we compare the performance of the two approaches when input schemes are generated by bootstrapped samples against those that are generated on non-bootstrapped data. Diversity in input schemes is obtained naturally when CC is run on bootstrapped data. But on non bootstrapped data, running CC with different initial values does not provide much heterogeneity in the input schemes. Thus xMotif and ISA were used for the purpose instead of CC. ISA was integrated with their code using ISA2 package [BIB03, CKB10].

Two sets of experiments were performed, one with xMotif and the other with ISA. In each set following experiments were performed:

- HN was executed on bootstrapped samples.
- $BiETopti$ was executed on bootstrapped samples.
- $BiETopti$ was executed on non bootstrapped samples also.

Figure 4.7 shows the biclustering error for all the three experiments on both the sets.

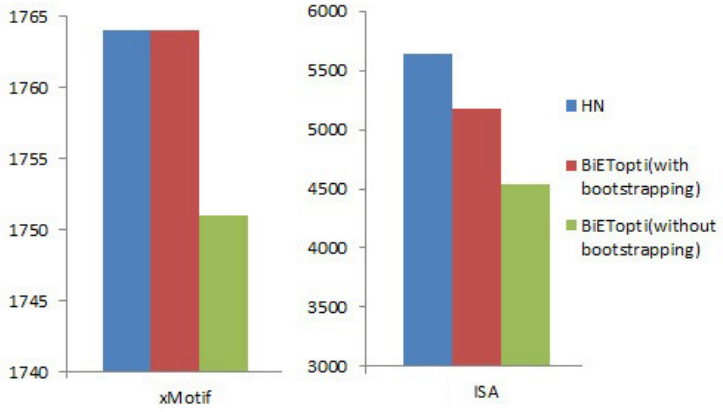

Figure 4.7: Biclustering error of HN and BiETopti (bootstrapped and non bootstrapped) for 2 setups using ISA and xMotif.

Following inferences can be drawn from the figure :

- $BiETopti$ on bootstrapped as well as non bootstrapped data outperforms HN especially with ISA.
- $BiETopti$ performs better on non bootstrapped data than on bootstrapped data.

Prelic et al. [PBZ+06] in their comparative study of different biclustering methods for gene expression have shown that ISA performs better than other biclustering algorithms. They focused on meaningfulness of the biclusters produced by different algorithms. They wanted to see if any algorithm had an edge over the other. According to them there was significant difference among these algorithms so far as performance is concerned. They show that ISA is capable of providing functionally enriched biclusters that are biologically significant. Multiple biclusters of both constant and coherently increasing values can be found using ISA. On the other hand, both CC and xMotif algorithms find large biclusters with constant expression levels and therefore not necessarily contain interesting patterns, e.g. in terms of co-regulation. The performance of CC and xMotif is significantly lower

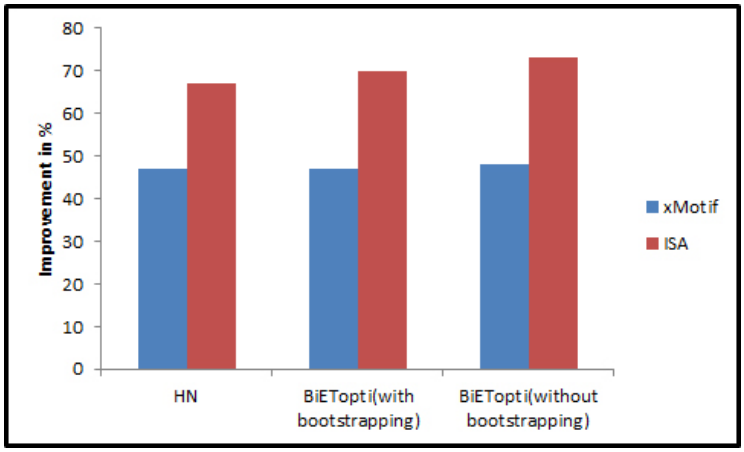

Figure 4.8: Comparing the performance of ISA and xMotif.

than that for the other biclustering methods. This was experimentally endorsed by our experiments also.

We next compare the performance of $BiETopti$ and HN w.r.t. xMotif and ISA in Figure 4.8. As the range of values of biclustering errors in the two cases is significantly

different, improvement percentage over 'Single' is used as a parameter to compare the performance. It was observed that performance of $BiETopti$ as well as HN with ISA is better as compared to that with xMotif in all the three experiments.

Thus here on, we focus on ensembles of schemes produced using ISA on non bootstrap data.

4.2.3 Results on Synthetic Data Sets

Having gained confidence in our approach on the data sets used by HN, we performed experiments on the benchmark data sets (DS1 and DS2) of Prelic et. al. using ISA as the biclustering algorithm. DS1 and DS2 are two distinct data sets: one having overlapping biclusters but no noise and the other having noise added to the data with non overlapping biclusters. Input schemes were generated by running ISA on the expression data 20 times, each time with 100 gene seed vectors. Genes and conditions not assigned to any bicluster by any of the input schemes were assigned a dummy label 0. This was done to ensure the feasibility of the input. $BiETopti$ was then executed to generate the ensemble. The whole procedure is repeated 20 times and the results are averaged over the runs.

Two sets of experiments were conducted on each data set. In the first set, the thresholds (t_g, t_c) were fixed and the schemes were generated by running ISA with different random gene seed. In the second set of experiment, t_g was varied keeping both the random gene seed and t_c fixed. The value of t_g was varied from $[-2.4, +2.0]$ in steps of 0.2. It was observed that for t_g values ranging from $[0.6, 1.6]$ schemes with biclusters identical to the implanted biclusters were obtained whereas schemes obtained for t_g varying from $[-2.4, -0.8]$ biclusters consisted essentially of all the genes and all the biclusters eventually reduced to a single bicluster after preprocessing. Ensembling such biclusters was of no help, so we focused our study on t_g varying from $[-0.6, 0.4]$ and $[1.8, 2.0]$.

We compared the performance of the algorithm with the best of the input schemes using biclustering error (BCE) and agreement score (AS). The best was obtained from the

400 schemes (20 runs of 20 schemes each).

Table 4.7 - 4.8 present the results on data set (DS1) for the first set of experiments (of varying random gene seed and having fixed t_g, t_c) using both evaluation criteria i.e. BCE and AS. The tables show that our approach improves upon the best of the input schemes both in terms of BCE and AS.

Schemes $t_g, t_c \downarrow$	**Best**	**BiETopti**
-0.50 , 2	3402	2540
-0.40 , 2	3830	3002
-0.35 , 2	3618	2652
1 , 1	5218	3580
0 , 1	5860	3768

Table 4.7: Best of input schemes vs BiETopti on DS1 for the first set of experiments using BCE.

Schemes $t_g, t_c \downarrow$	**Best**	**BiETopti**
-0.50 , 2	0.82	0.82
-0.40 , 2	0.77	0.79
-0.35 , 2	0.90	0.91
1 , 1	0.69	0.70
0 , 1	0.54	0.56

Table 4.8: Best of input schemes vs BiETopti on DS1 for the first set of experiments using AS.

The second experiment was performed for varying values of (t_g). In this experiment also, $BiETopti$ provides better biclusters than best of the input schemes. The results are displayed in Table 4.9.

Evaluation Criteria	Best	BiETopti
BCE	3180	2752
AS	0.81	0.82

Table 4.9: Best of input schemes vs BiETopti on DS1 for the second set of experiment.

Effect of noise

Noisy data set (DS2) was used to study the impact of noise on the performance of $BiETopti$. Both the experiments were repeated on (DS2).

Schemes $t_g, t_c \downarrow$	**Best**	**BiETopti**
.90, 1	2865	2431
1, .5	3012	2650
-.35,2	4187	3256

Table 4.10: Effect of noise on BiETopti (data set DS2) for the first set of experiments using BCE.

Table 4.10 shows the results for the first set of experiments using BCE as the evaluation method. Table 4.11 shows the Agreement Score (AS) of the results on DS2 for the first set of experiments.

Schemes $t_g, t_c \downarrow$	**Best**	**BiETopti**
.90, 1	0.87	0.88
1, .5	0.77	0.78
-.35, 2	0.50	0.51

Table 4.11: Effect of noise on BiETopti (data set DS2) for the first set of experiments using AS.

The results for the second set of experiment are shown in the Table 4.12 using both the evaluation methods. The tables show that $BiETopti$ was able to improve the performance of the best of input schemes even in presence of noise using any of the two evaluation measures.

Evaluation Criteria	Best	BiETopti
BCE	2588	2312
AS	0.92	0.92

Table 4.12: Effect of noise on BiETopti (data set DS2) for the second set of experiment.

Effect of number of schemes to be ensembled

To study the effect of number of schemes on the ensemble, a large number of schemes were generated by varying (t_g,t_c). t_g was varied from 0.1 to 1.0 in step of 0.1. Similarly t_c was varied from 0.1 to 1.0 in step of $.1$ to obtain 100 schemes.

#schemes ensembled	BCE of ensemble	Improvement over the Best	Time
10	2928	179	15.1
20	2900	199	30
50	2889	100	40.1
100	2867	70	75

Table 4.13: Effect of number of schemes.

Results were taken by ensembling varying number of schemes out of these 100 schemes. 4 sets of experiments were performed with $BiETopti$ by picking all the schemes (100), half of the schemes (50), one fifth of the schemes (20) and one tenth of the schemes (10). Table 4.13 shows the biclustering error and the time taken for these experiments.

Figure 4.9 shows the performance of $BiETopti$ with the varying number of input schemes. In all the cases the quality of the biclusters improves upon the best of the input schemes. It was observed that the improvement in performance increases when the number of schemes are increased up to 20 beyond which the improvement starts decreasing as shown in the figure.

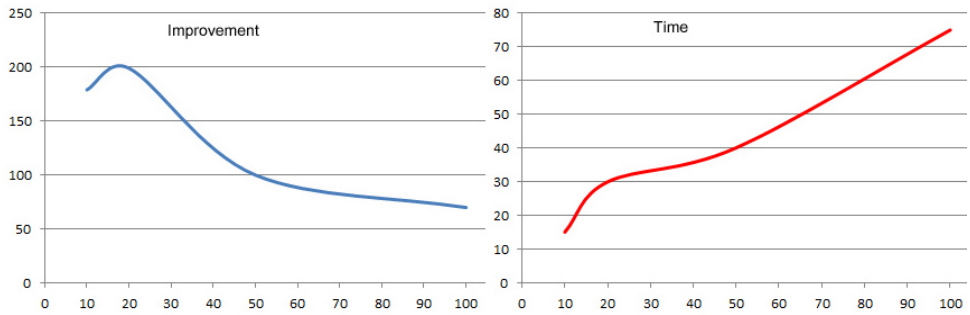

Figure 4.9: Improvement of biclustering error and time comparison to show effect of number of schemes.

At the same time, increasing the number of schemes beyond 20 increases the time substantially. Thus, for the rest of the experiments we fixed number of schemes to be 20.

Effect of changing the reference scheme

We also studied the impact of changing the reference scheme on the performance. It was observed that there was no significant change in the accuracy of the biclusters on changing the reference scheme.

4.2.4 Results on Real Data Sets

Experimental studies were performed on the data sets of Saccharomyces Cerevisiae, Arabidopsis Thaliana, DLBCL, and Human Breast Cancer. We generated input schemes by running ISA, each time with hundred different gene seed vectors. Sizes of the biclusters were kept to be comparable to eliminate the effect of size of biclusters on the p-values.

The biclusters produced by $BiETopti$ were evaluated using DAVID and RSAT tool.

Table 4.14 shows the biclusters of $BiETopti$ along with their aligned input biclusters for all the data sets. The tables show the $-log\ p$-values of GO terms of top 10 biclusters produced by $BiETopti$. To compare with the input schemes, $-log\ p$-value of the best 3 of the aligned biclusters is displayed. It was observed that the quality of the biclusters obtained was better than most of the input biclusters.

The genes of a bicluster are responsible for one biological activity and are expected to have common patterns/motifs. So to further biologically validate our biclusters, we searched for common patterns (motifs) from the promoter regions of the genes belonging to a bicluster. Table 4.15 summarizes the best p-value for the motif extracted from the gene sequences of the genes belonging to biclusters extracted by $BiETopti$ on various data sets. Again the low p-values indicate biologically significant biclusters. The table shows the motif analysis of top biclusters of $BiETopti$ along with their aligned input biclusters. It is clearly seen that the biclusters obtained by $BiETopti$ were biologically more significant than most of the input biclusters using motifs also. Promoter regions of the genes of most of the biclusters were found to have statistically significant common motif patterns.

Comparison of BiETopti with existing biclustering algorithms

Figure 4.10 shows the comparison of the biclusters produced by $BiETopti$ with the biclusters produced by existing biclustering algorithms like order-preserving sub matrix (OPSM) [BDCKY03], Cheng and Church (CC) [CC00], BIMAX [PBZ$^+$06] and ISA [BIB03].

For Yeast and A. Thaliana the biclusters for all the biclustering algorithms were taken from the BICAT site. For DLBCL and Breast Cancer, biclusters were generated by executing these algorithms in BICAT tool. Figure shows that none of the existing algorithms is said to be a clear winner in all the organisms. CC performs best amongst

Yeast : $-\log p$ value of GO terms

Best 3 of the alignment	**BiETopti**
72,72,61	74
65,65,60	65
72,56,42	57
48,48,48	49
46,43,42	47
46,31,27	46
31,31,31	32
23,23,23	28
11,11,11	12
6,6,5	6

A. Thaliana : $-\log p$ value of GO terms

Best 3 of the alignment	**BiETopti**
31,31,25	26
23,23,16	19
31,24,21	21
27,26,26	26
27,23,21	24
22,22,22	23
20,18,18	21
21,20,20	20
23,19,19	19
18,17,16	19

DLBCL : $-\log p$ value of GO terms

Best 3 of the alignment	**BiETopti**
22,16,5	20
19,16,16	19
17,16,16	16
17,16,16	16
14,2,2	13
8,8,7	9
22,8,6	8
8,8,8	8
13,7,7	7
6,6,6	6

Breast Cancer : $-\log p$ value of GO terms

Best 3 of the alignment	**BiETopti**
45,45,45	46
22,22,22	33
18,17,17	26
16,15,14	26
16,15,15	25
16,14,5	25
12,12,12	13
5,5,5	6
3,3,3	4
3,3,2	3

Table 4.14: Comparison of top 10 biclusters of $BiETopti$ with best 3 aligned input biclusters on real data sets using GO terms.

Yeast : $-\log p$ value of motifs

Best 3 of the alignment	**BiETopti**
32,24,21	32
32,22,22	32
24,23,22	23
18,15,13	20
15,15,14	20
14,14,14	15
11,10,9	13
9,9,7	12
8,5,5	9
7,7,5	10

A. Thaliana : $-\log p$ value of motifs

Best 3 of the alignment	**BiETopti**
22,18,18	45
20,20,18	29
19,18,17	23
18,18,18	18
14,12,10	18
10,9,8	12
12,11,10	11
10,10,10	11
11,9,8	10
8,7,7	8

DLBCL : $-\log p$ value of motifs

Best 3 of the alignment	**BiETopti**
28,22,16	30
19,18,16	20
18,18,18	20
17,16,16	18
14,12,10	18
10,10,8	10
12,9,9	12
10,10,9	12
13,7,7	13
6,6,6	8

Breast Cancer : $-\log p$ value of motifs

Best 3 of the alignment	**BiETopti**
16,15,12	16
15,15,14	16
15,13,12	12
12,11,11	11
10,10,10	10
9,9,8	8
7,7,6	8
5,4,3	5
3,3,3	5
3,2,1	5

Table 4.15: Comparison of top 10 biclusters of $BiETopti$ with best 3 aligned input biclusters on real data sets using common motifs.

the existing algorithms on Yeast. On A. Thaliana and DLBCL, performance of ISA is best amongst the existing algorithms. OPSM takes the lead in Breast Cancer data set. $BiETopti$ outperforms the best in each of these organisms except A. Thaliana. However in DLBCL there is a marginal difference in the performance of $BiETopti$ and the best algorithm.

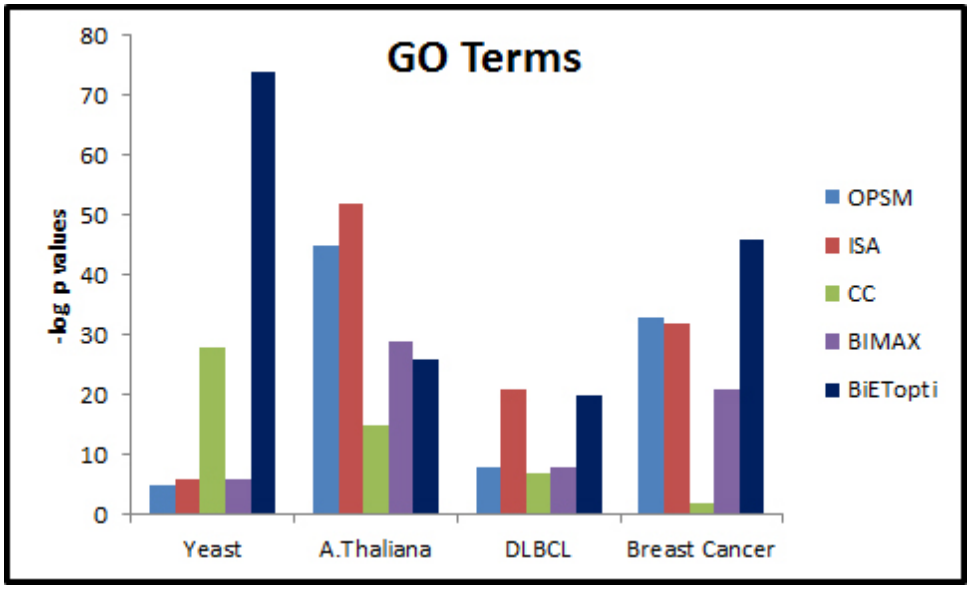

Figure 4.10: BiETopti compared with OPSM, ISA, CC and BIMAX.

Chapter 5

BiETclassi - Biclustering Ensemble Technique using Classifiers

The previous chapter explained the ensemble algorithm based on optimization technique. The results were promising but required the schemes to have equal number of biclusters. To do away with this restriction we propose another ensemble algorithm, $BiETclassi$ that makes use of classifiers for the ensembling purpose.

Traditional classifiers such as discriminant analysis and support vector machines typically label two class data. Various techniques like one-against-all are used to extend them for multi-label and multi-class classification. These techniques cannot be directly plugged in for biclusters as the set of attributes (samples) is different for different biclusters. Thus, we extend one-against-all classification methods for multi-label classification and apply to biclusters one by one. For each label (bicluster), we build a binary class problem so that the genes associated with that label are in one class and the rest are in class labeled 0. For each binary class problem, a different set of features corresponding to the conditions of the bicluster is used.

We study the performance of $BiETclassi$ on synthetic and real data sets and found that $BiETclassi$ produced biclusters that were biologically much more significant than

not only the input biclusters but also biclusters produced by $BiETopti$. Besides producing better biclusters, $BiETclassi$ also improved upon time as compared to $BiETopti$.

A brief review of classifiers used in our approach is presented in Section 5.1. The approach is explained in detail in Section 5.2 and the results of various experiments performed are shown in the Section 5.3.

5.1 Preliminaries

In this section we describe the classifiers [DHS01] used in our approach. Discriminant Analysis (DA) [Fis36] and Support Vector Machine (SVM) [CST00b] are statistical techniques used for classifying multivariate data in different classes. DA seeks a hyperplane (a discriminator variable) that best separates the scatter of the projected data points on it. It assumes that the data points are normally distributed in their respective classes on each variable.

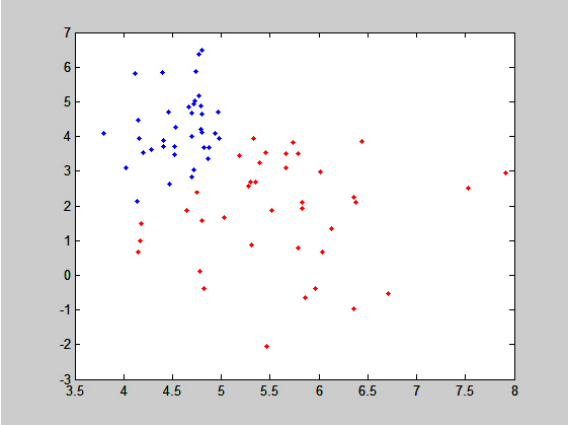

Figure 5.1: Sample Data.

Figure 5.1 shows sample data having 2 classes. Different classes are shown using red and blue colours. Distribution of the points along x-axis and y-axis is shown in Figures 5.2 and 5.3 respectively. As is visible in the figures, there is a large overlap between the

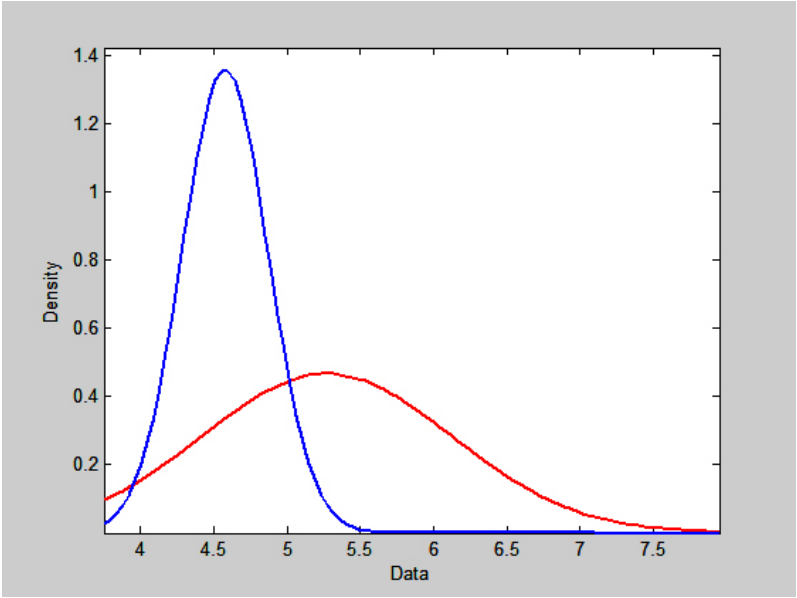

Figure 5.2: Projection of data points along x-axis.

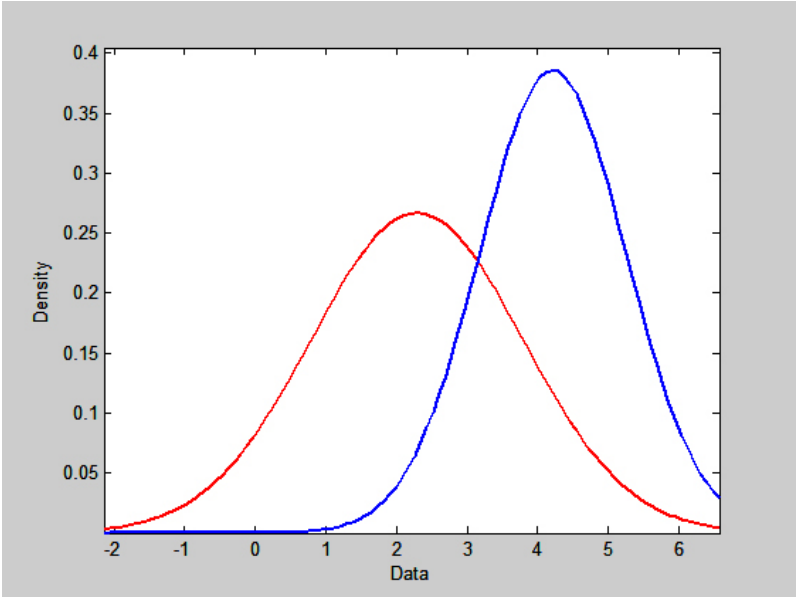

Figure 5.3: Projection of data points along y-axis.

classes thereby implying that the classes are indistinguishable on each of the attributes individually. A new discriminating dimension is calculated and the projection along this dimension is shown in Figure 5.4. It shows a decrease in the overlap and thus helps in classifying the objects in a better way.

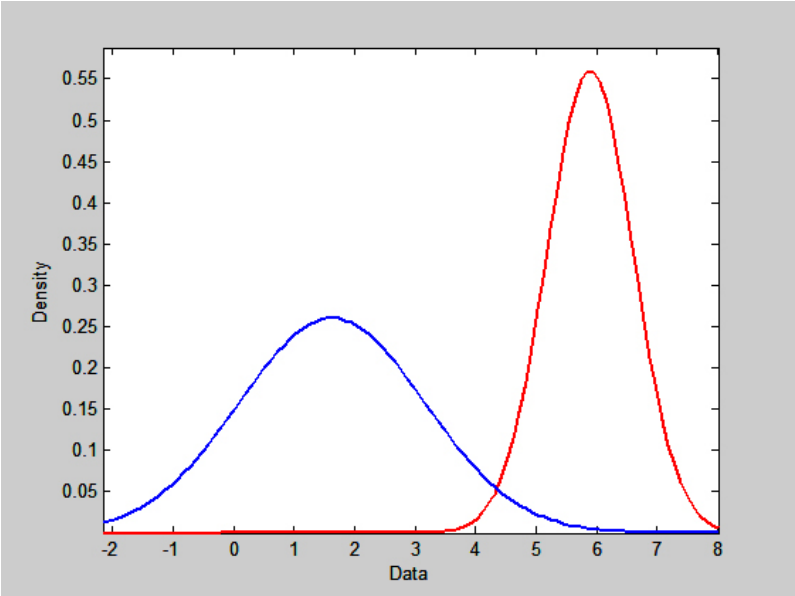

Figure 5.4: Projection of data points along the new dimension calculated by DA.

SVM seeks a hyperplane that maximizes the margin between the closest pair of points, one from each class. The two points are called the support vectors. There exist multiple hyperplanes that separate the two classes but not all of them are equally good separators. For the two dimensional data shown in Figure 5.1, The possible separators (lines here) are shown in Figure 5.5. Figure 5.6 shows one such separator along with its support vectors.

5.1.1 Discriminant Analysis

Discriminant Analysis establishes relationships between attributes for classifying objects into one of the several populations, by identifying attributes that best discriminate be-

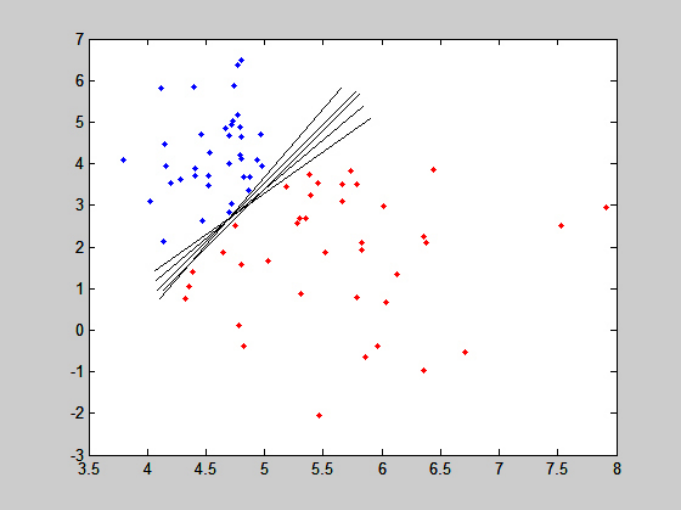

Figure 5.5: Various Lines separating the data points in two classes.

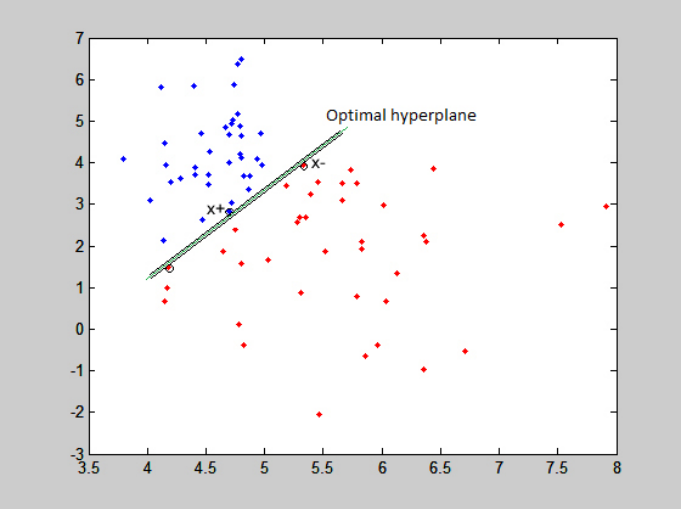

Figure 5.6: Hyperplane as calculated by Support Vector Machine.

tween the members of a group. In DA, the classes represent the dependent variables and the predictors represent the independent variables. It is a 2-step process of testing and classification. There is a matrix of total variances and covariances; likewise, there is a matrix of pooled within-group variances and covariances. The two matrices are compared to find whether or not there is any significant difference between groups and then it tries to see which of the variables have significantly different means across the groups that lead to the discrimination. This method maximizes the ratio of between-class variance to the within-class variance. In DA, data is modeled and projected onto a single dimension and class assignment is made for a given point.

Suppose that we have a set of n d dimensional samples $x_1, x_2.....x_n$. Let X denote the $n \times d$ matrix with data points along the rows and dimensions along the column. Let n_1 denote the number of samples in the subset D_1 labeled ω_1 and n_2 be the number of samples in the subset D_2 labeled ω_2. Each component (feature/attribute) of a data point is assigned a weight w_i to compute the discriminate variable. Let $\mathbf{w}$ be the $d \times 1$ vector of weights. Let $\mathbf{Y}$ be a $1 \times n$ vector defined as follows

$$(y_1, y_2, \ldots, y_n) \;=\; (\mathbf{w}^{'} \cdot \mathbf{x_1}, \mathbf{w}^{'} \cdot \mathbf{x_2}, \ldots \mathbf{w}^{'} \cdot \mathbf{x_n})$$

Here $\mathbf{w}^{'} \cdot \mathbf{x}$ is the dot product of $\mathbf{w}^{'}$ and **x**.

Further let $\mathbf{Y}$ is partitioned into the subsets Y_1 and Y_2 corresponding to D_1 and D_2 respectively. The aim is to find $\mathbf{w}$ that maximizes separation between the two classes. Geometrically, if $\|\mathbf{w}\|$ = 1 each y_i is the projection of the corresponding $\mathbf{x_i}$ onto a line in the direction of $\mathbf{w}$. Actually the magnitude of $\mathbf{w}$ is of no real significance because it merely scales y. The direction of $\mathbf{w}$ is important, however. If we imagine that the samples labeled ω_1 fall more or less into one cluster while those labeled ω_2 fall in another, we want the projections falling onto the line to be well separated. It should be abundantly clear that if the original distributions are highly overlapping, even the best $\mathbf{w}$ is unlikely to provide adequate separation and thus the method will be of little use. We now turn to the matter

of finding a good direction **w** that provides a good classification. Difference of the sample means of the projected points is a good measure of separation between the two classes. If $\mathbf{m}_i$ denote the d dimensional ($1 \times d$) sample mean and is given by

$$\mathbf{m_i} = \frac{1}{n_i} \sum_{\mathbf{x} \in D_i} \mathbf{x}$$

then the sample mean for the projected points given by

$$\begin{aligned} \overline{m_i} &= \frac{1}{n_i} \sum_{y \in Y_i} y \\ &= \frac{1}{n_i} \sum_{\mathbf{x} \in D_i} \mathbf{w}^{'} \cdot \mathbf{x} = \mathbf{w}^{'} \cdot \mathbf{m_i} \end{aligned}$$

is simply the projection of $\mathbf{m_i}$ along **w**. It follows that the distance between the projected means is

$$|\overline{m_1} - \overline{m_2}| = \left| \mathbf{w}^{'} \cdot (\mathbf{m_1} - \mathbf{m_2}) \right|$$

This distance can be made arbitrarily large by scaling the difference of sample means $\mathbf{m_1}, \mathbf{m_2}$ by $|\mathbf{w}|$. Thus to obtain good separation of the projected data we would actually want the difference between the means to be large relative to some measure of the standard deviations for each class. Rather than forming the sample variance, $scatter$ for projected samples is defined for each of the two classes and it is given by

$$\overline{s_i}^2 = \sum_{y \in Y_i} (y - \overline{m_i})^2$$

$\overline{s_1}^2 + \overline{s_2}^2$ is called the total within-class scatter of the projected samples and $(1/n)(\overline{s_1}^2 + \overline{s_2}^2)$ is an estimate of the variance of the pooled data. The fisher linear discriminant then determines the weight vector for which the discriminant function

$$J(.) = \frac{|\overline{m_1} - \overline{m_2}|^2}{\overline{s_1}^2 + \overline{s_2}^2}$$

is maximum. To obtain J(.) as a function of $\mathbf{w}$, scatter matrices $\mathbf{S_w}$ and $\mathbf{S_B}$ are defined. Let

$$\mathbf{S_i} = \sum_{\mathbf{x} \in D_i} (\mathbf{x} - \mathbf{m_i})^{'} (\mathbf{x} - \mathbf{m_i})$$

Then

$$\mathbf{S_w} = \mathbf{S_1} + \mathbf{S_2}$$

is defined to be the within class scatter matrix. Also,

$$\begin{aligned} \overline{s_i}^2 = \sum_{\mathbf{x} \in D_i} \left(\mathbf{w}^{'} \cdot \mathbf{x} - \mathbf{w}^{'} \cdot \mathbf{m_i} \right)^2 \\ = \mathbf{w}^{'} \mathbf{S_i} \mathbf{w} \end{aligned}$$

Then within-class scatter of the projected samples can be written as

$$\overline{s_1}^2 + \overline{s_2}^2 = \mathbf{w}^{'} \mathbf{S_w} \mathbf{w}$$

Similarly the between class scatter defined by separation of the projected means obeys

$$\begin{aligned} |\overline{m_1} - \overline{m_2}|^2 = \left(\mathbf{w}^{'} \cdot \mathbf{m_1} - \mathbf{w}^{'} \cdot \mathbf{m_2} \right)^2 \\ = \mathbf{w}^{'} (\mathbf{m_1} - \mathbf{m_2})^{'} (\mathbf{m_1} - \mathbf{m_2}) \mathbf{w} \\ = \mathbf{w}^{'} \mathbf{S_B} \mathbf{w} \end{aligned}$$

where

$$\mathbf{S_B} = (\mathbf{m_1} - \mathbf{m_2})^{'} (\mathbf{m_1} - \mathbf{m_2})$$

is the between class scatter matrix.

$\mathbf{S_W}$ is symmetric, positive semidefinite and usually nonsingular, if $n > d$. Like wise $\mathbf{S_B}$ is also symmetric and positive semidefinite, but because it is the outer product of two vectors, its rank is at most one. In terms of $\mathbf{S_B}$ and $\mathbf{S_W}$ the discriminating function J can be expressed as a ratio of the between-class scatter to the within-class scatter and is written as

$$J(\mathbf{w}) = \frac{\mathbf{w}^{'} \mathbf{S_B} \mathbf{w}}{\mathbf{w}^{'} \mathbf{S_W} \mathbf{w}}$$

5.1.2 Support Vector Machine

SVMs are based on the principle of Structural Risk Minimization. It maximizes the margin between the points of two classes by solving a convex quadratic programming problem. The solution to that problem gives us a hyperplane that has the maximum margin between the two classes. SVM models the boundary between the data points instead of modeling the projection of data points themselves as in DA. In support vector machines, a data point is viewed as a d dimensional vector and we want to know whether we can separate such points with a $d-1$ dimensional hyperplane. This is called a linear classifier. There are many hyperplanes that might classify the data. One reasonable choice for a good hyperplane is the one that provides the largest separation, or margin, between the two classes. Thus a hyperplane that maximizes the distance to the nearest data points on each side is aimed. If such a hyperplane exists, it is known as the maximum-margin hyperplane and the linear classifier it defines is known as a maximum margin classifier.

Let -1 and +1 be the class labels of the two classes. If $\mathbf{Y}$ is the vector of dependent variables representing the class labels then $\mathbf{Y}$ is partitioned into the subsets Y_1 and Y_2 corresponding to the two classes having labels -1 and +1. SVM seeks a hyperplane that maximizes the margin between the closest pair of points, one from each class. The two points are called the support vectors. The hyperplane separating the points is given by

$$\mathbf{w}^{'} \cdot \mathbf{x} + b = 0$$

Here $\mathbf{w}$ is the $d \times 1$ vector of direction of the hyperplane. The aim is to find $\mathbf{w}$ and b such that the separation between the two classes is maximized. Following is true for the points in the two classes.

$$\mathbf{w}^{'} \cdot \mathbf{x} + b \geq +1$$

$$\mathbf{w}^{'} \cdot \mathbf{x} + b \leq -1$$

In other words, we have

$$y^{(i)}(\mathbf{w}^{'} \cdot \mathbf{x^{(i)}} + b) \geq +1$$

Equations for the hyperplanes passing through the two support vectors $\mathbf{x}^{+}$ and $\mathbf{x}^{-}$ in the two classes are given by

$$\mathbf{w}^{'} \cdot \mathbf{x}^{+} + b = +1 \tag{5.1}$$

$$\mathbf{w}^{'} \cdot \mathbf{x}^{-} + b = -1 \tag{5.2}$$

SVM tries to maximize the distance M between these two hyperplanes where M is given by

$$M = \frac{\mathbf{w}^{'} \cdot (\mathbf{x}^{+} - \mathbf{x}^{-})}{|\mathbf{w}|} \tag{5.3}$$

From Equations 5.1 and 5.2

$$\mathbf{w}^{'} \cdot (\mathbf{x}^{+} - \mathbf{x}^{-}) = 2$$

Substituting the value in Equation 5.3, the margin is defined as

$$M = \frac{2}{|\mathbf{w}|}$$

Maximizing the margin M is then same as minimizing the following

$$M = \frac{1}{2}\mathbf{w}^{'} \cdot \mathbf{w}$$

The problem becomes a quadratic optimization problem formulated as follows: Minimize

$$\frac{1}{2}\mathbf{w}^{'} \cdot \mathbf{w}$$

subject to

$$y^{(i)}(\mathbf{w}^{'} \cdot \mathbf{x}^{(\mathbf{i})} + b) \geq +1 \; \forall \; i$$

SVM works by solving this quadratic optimization problem and finding values for **w** and b. This is illustrated in Figure 5.6.

5.2 BiETclassi

We use the 'relabeling and voting ' approach to generate the consensus. Relabeling is done twice, once to align similar biclusters using label correspondence and second time, it is done using classifiers like DA and SVM which are basically binary classifiers that work for two classes. However, in gene expression data, genes may be responsible for more than two functions. Classifiers that can handle multiple classes need to be used instead. Extensions of DA and SVM that solve the multi class problem are known to exist in literature [Bis06], but they do not allow the classes to overlap. To be able to handle overlapping biclusters, one needs to consider the multi label classification [TK07]. Classifiers that handle multi label and multi class also exist in literature [ZZ06, ZZ07] but they can not be directly applied for our problem as they work on the same set of conditions for all the labels. On the other hand in gene expression data, different samples/attributes define different biclusters. Thus we extend these techniques to suit the need of biclusters.

5.2.1 The Approach

Our algorithm works in four phases. First two phases are same as that in $BiETopti$ wherein input schemes are generated and then aligned so that similar biclusters in different schemes get the same label. One of the input schemes is taken as the reference. As the number of biclusters returned by ensemble is same as that of the reference scheme,

we choose the scheme with the largest number of biclusters as the reference scheme, in order not to miss any bicluster. In phase-III, a classifier is used to predict labels for genes for each input scheme. This is the main contribution of our approach and we discuss it in detail in Section 5.2.2. Having predicted the gene labels for each scheme, voting is used to obtain the final consensus in phase-IV. Voting is also used to obtain labels of the conditions. Figure 5.7 shows the basic architecture of our algorithm.

5.2.2 Phase III: Relabeling the Genes using a Classifier

In this phase we will discuss how to use classifiers to predict the labels of the genes. The challenge with using classifiers to predict the labels for the biclustering problem is three-fold. One, the biclusters are overlapping. Secondly the biclusters are non-exhaustive i.e. there may be genes/samples that do not belong to any bicluster. Thirdly, different biclusters are defined by a different set of attributes/samples; third being the most important.

Multi-label classifiers are used to address the first problem. In multi-label classification, an object may belong to more than one class. It is different from multi class classification, wherein objects may be categorized into more than two classes but an object belongs to one class only. We present a method to extend binary classifiers DA and SVM to handle multi class and multi label data for biclustering. There are broadly two ways of handling multi label classification [TK07, ZZ14]. The first being the problem transformation and second being the algorithm adaptation. In problem transformation, the multi-label problem is transformed into a set of binary classification problems, which can then be easily handled. In algorithm adaptation, algorithms are adapted to directly perform multi-label classification instead of transforming the problem. Various problem transformation methods exist in literature. We have used the first method wherein one binary classifier is trained for each label. For multi class classification we extend one-against-all classification methods for biclustering. One against all classification method involves training a single classifier per class. For each label (bicluster), a binary class

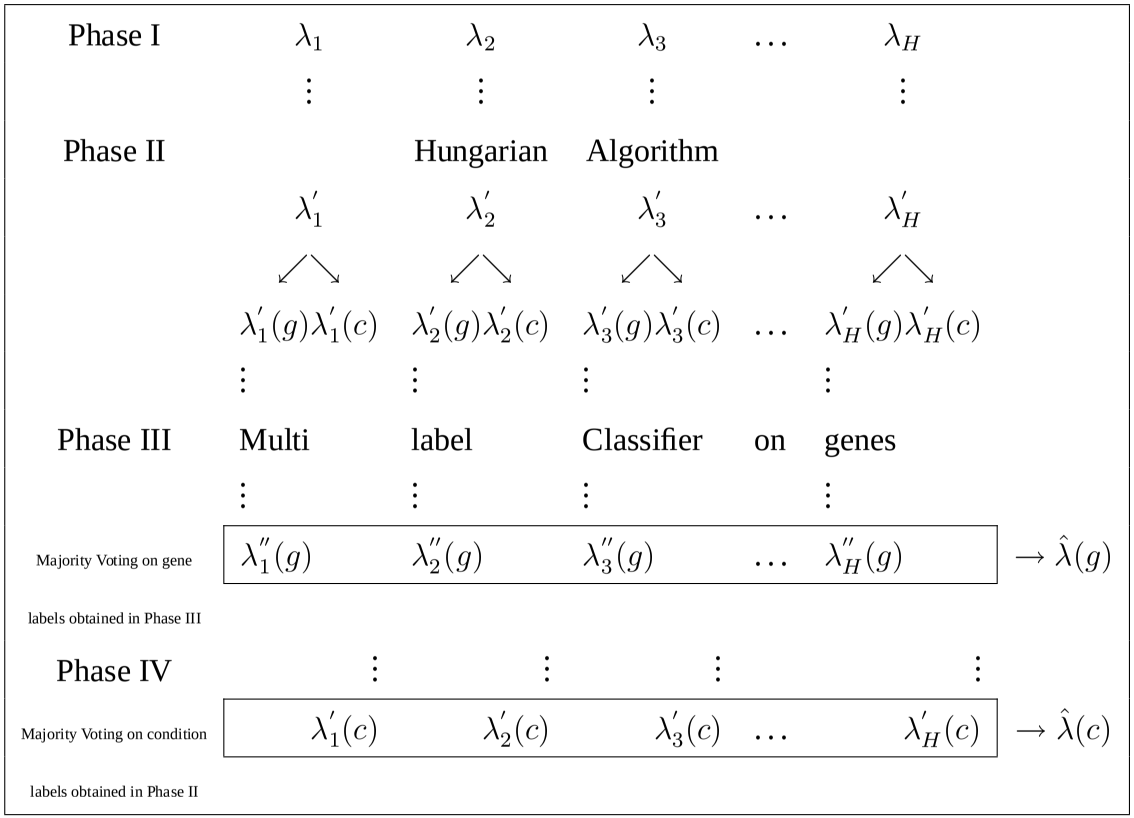

Figure 5.7: Architecture of BiETclassi.

Biclustering scheme1 (BCS1)

	Input Labels
Gene1	0
Gene2	1
Gene3	1,2
Gene4	2
Gene5	3
Gene6	3
Gene7	0

(a)

Bicluster1 - Conditions (1,2)

	Input Labels	Predicted labels PL1
Gene1	0	0
Gene2	1	1
Gene3	1	0
Gene4	0	1
Gene5	0	0
Gene6	0	0
Gene7	0	0

(b)

Bicluster2 - Conditions (2,3)

	Input Labels	Predicted labels PL2
Gene1	0	0
Gene2	0	2
Gene3	2	2
Gene4	2	2
Gene5	0	0
Gene6	0	0
Gene7	0	0

(c)

Bicluster3 - Conditions (3,4,5)

	Input Labels	Predicted labels PL3
Gene1	0	0
Gene2	0	0
Gene3	0	3
Gene4	0	0
Gene5	3	3
Gene6	3	3
Gene7	0	0

(d)

Biclustering scheme1 (BCS1)

	Predicted Label PL1	Predicted label PL2	Predicted label PL3	Union of labels	Final labels-BCS1
Gene1	0	0	0	0	0
Gene2	1	2	0	1,2	1,2
Gene3	0	2	3	2,3	2,3
Gene4	1	2	0	1,2	1,2
Gene5	0	0	3	3	3
Gene6	0	0	3	3	3
Gene7	0	0	0	0	0

(e)

Table 5.1: Working of Modified Classifier.

problem is built so that the genes associated with that label are in one class and the rest are in another class. This method is performed for every bicluster of one scheme except the one with label 0 (set of genes not belonging to any bicluster are given the label 0). For each bicluster, the genes in the bicluster are given the label of the bicluster and the rest of the genes are given the label 0. For each binary class problem, a different set of features corresponding to the conditions of the bicluster is used to take care of the third challenge. Finally, a gene is assigned the union of all the labels. This allows us to assign more than one labels to a gene. This takes care of the overlapping nature of the biclusters. The process is repeated for every bicluster of one scheme and finally the union of all the labels is taken to obtain multiple labels $\lambda''(g)$ for a gene g.

The working of the algorithm is explained in Table 5.1. The example shown here has 7 genes and 5 conditions. Data is subjected to a biclustering algorithm to get an input scheme shown in Table 5.1 (a). It shows the input labels for all the genes according to the bicluster it belongs to. All the genes of a bicluster along with its condition set are subjected to a classifier. This is done for every bicluster. The predicted labels of the genes are shown in Table 5.1 (b)-(d). Note that the set of conditions is different for different biclusters. Table 5.1 (e) shows the labels that are assigned to the genes after subjecting all the genes bicluster wise to the classifier and taking the union of all the predicted labels. Note that label 0 (refers to genes not belonging to class) is not taken into consideration for taking union. If gene is assigned label 0 by all the biclusters, then only label 0 (refers to genes not belonging to any bicluster) is assigned to it. The working is pictorially shown in Figure 5.8. This procedure is then repeated for every scheme to get the new labels λ'' for all the genes in all the schemes. Algorithm 2 summarizes the computation of new labels i.e. $\lambda''_i(g)$ for the i^{th} scheme, $i = 1 \ldots H$.

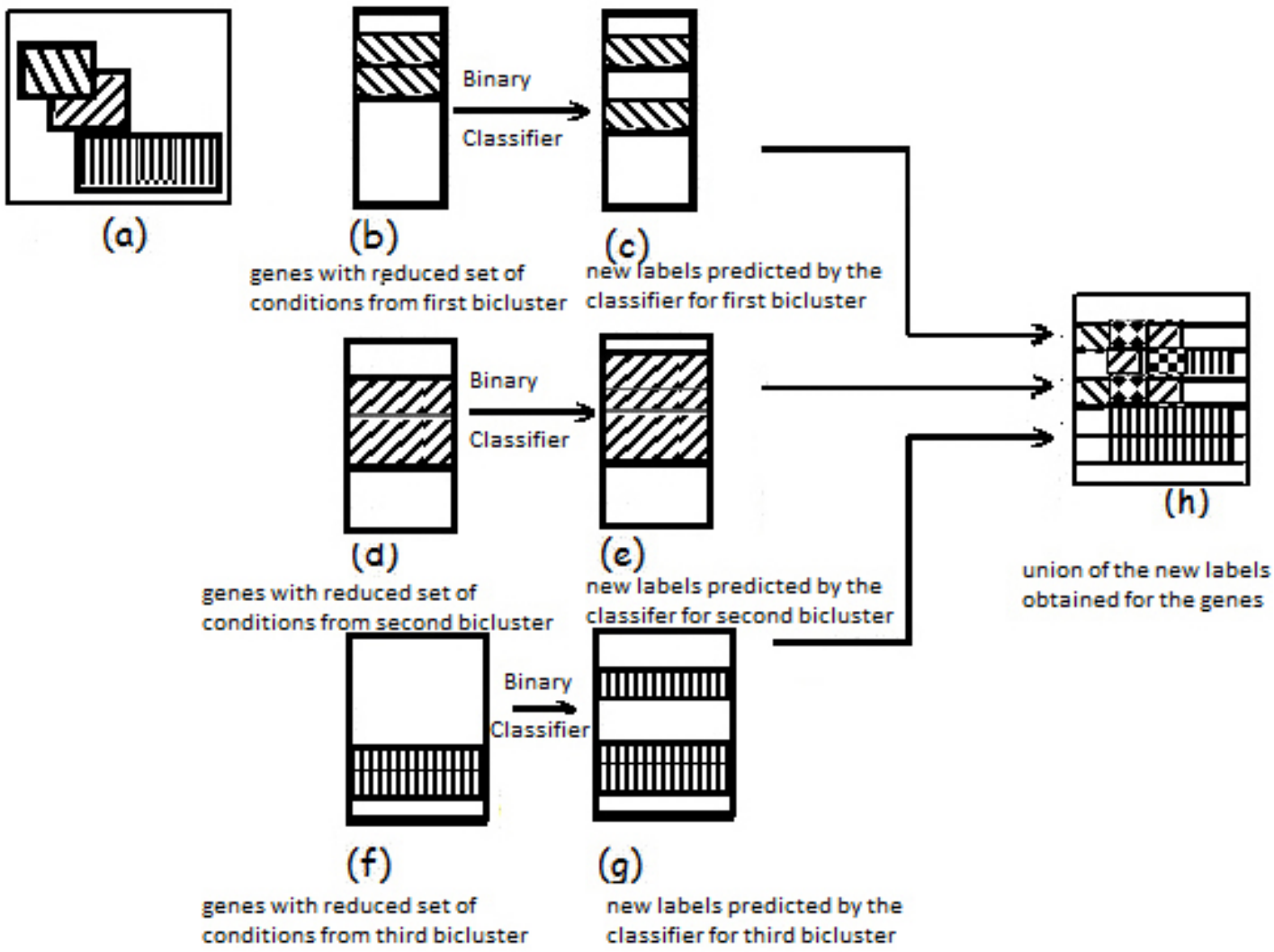

Figure 5.8: Visualization of one against all method. (a) Biclusters of an input scheme (b),(d),(f) the biclusters subjected to the classifier on reduced set of conditions/attributes (c),(e),(g) new labels predicted by the classifier for the biclusters (h) union of the new labels obtained in (c),(e) and (g).

Input: Labels $\lambda_1^{'}(g), \lambda_2^{'}(g), ..., \lambda_H^{'}(g)$

Output: Labels $\lambda_1^{''}(g), \lambda_2^{''}(g), ..., \lambda_H^{''}(g)$

for i= 1 **to** H **do**

  for j= 1 **to** k_i **do**

    Train a binary classifier (DA / SVM) with the gene vectors projected onto the conditions of the j^{th} bicluster and the class labels as $\{0, j\}$.

    for m = 1 **to** N **do**

      Predict the label $\lambda_{ij}^{''}(g_m)$ of the gene g_m.

    end

  end

  $\lambda_i^{''}(g_m) = \bigcup_j \lambda_{ij}^{''}(g_m)$

end

Algorithm 2: Predicting gene labels using a Multi-Label Classifier.

Final Gene Labels

	Final labels-BCS1	Final labels-BCS2	Final labels-BCS3	Final labels-BCS4	Final labels-BCS5	Final labels
Gene1	0	0,1	1	0	0	0
Gene2	1,2	1,2	1,2	1	2	1,2
Gene3	2,3	2	1,2	2,3	2	2
Gene4	1,2	1,3	1,3	2,3	1,3	1, 3
Gene5	3	2,3	1,2	3	2,3	3
Gene6	3	2,3	1,2	3	2	0
Gene7	0	0,1	0	1	0	0

Table 5.2: Final Consensus.

5.2.3 Phase IV: Final consensus

Voting is used in this phase to form the consensus. Voting is first done on $\lambda^{''}(g)$ to generate the final consensus labeling $\hat{\lambda}(g)$ for the genes. A label is assigned to a gene g in the final ensemble if at least τ number of schemes assign the label to g. Table 5.2 shows the labels assigned to the genes. Here, for representational purpose five schemes have been ensembled for the final consensus and τ has been taken as 80%.

Notations:

Let $\delta_g(i,l)$ =1 if $l \in \lambda_i^{''}(g),\ i = 1 \ldots H$

Let $\delta_c(i,l)$ =1 if $l \in \lambda_i^{'}(c),\ \ i = 1 \ldots H$

Input: $\lambda^{''}(g)$,$\lambda^{'}(c)$

Output: Final labelings $\hat{\lambda}(g)$,$\hat{\lambda}(c)$

for k= 1 **to** N **do**

  $\hat{\lambda}(g_k)$ = ϕ

  for l=1 **to** $max_i k_i$ **do**

    `/* for every label */`

    compute $n(g_k)$ = $\sum_i \delta_{g_k}(i,l)$

    `/* compute the number of schemes assigning label l to the` k^{th} `gene */`

    if $n(g_k) > \tau$ **then**

      $\hat{\lambda}(g_k)$ = $\hat{\lambda}(g_k) \bigcup \{l\}$

    end

  end

end

for k = 1 **to** d **do**

  $\hat{\lambda}(c_k)$ = ϕ

  for l = 1 **to** $max_i k_i$ **do**

    compute $n(c_k)$ = $\sum_i \delta_{c_k}(i,l)$

    if $n(c_k) > \tau$ **then**

      $\hat{\lambda}(c_k)$ = $\hat{\lambda}(c_k) \bigcup \{l\}$

    end

  end

end

Algorithm 3: Voting Phase.

Similarly to obtain the final labeling $\hat{\lambda}(c)$ for the conditions, voting is done on $\lambda^{'}(c)$ of the aligned biclusters. This phase is explained in Algorithm 3.

5.3 Experimental Results

Experiments were performed on synthetic as well as real data sets. To obtain a good threshold value for τ, experiments were performed on data set DS1. The results at varying threshold are shown in Table 5.3. It was found that the results improved with the increase in the threshold value, however it tends to decrease after a threshold value 80%. So we fixed the threshold value for voting at 80% for the rest of the experiments.

t_g, t_c	**50%**	**60%**	**70%**	**80%**	**90%**
-.5,2	0.83	0.84	0.84	**0.85**	0.80
-.4,2	0.84	0.84	0.84	**0.85**	0.80
-.35,2	0.96	0.96	0.96	**0.98**	0.88
1,1	0.73	0.73	0.73	**0.74**	0.69
0,1	0.56	0.56	0.56	**0.57**	0.50

Table 5.3: Effect of different voting threshold values on AS on DS1.

5.3.1 Results on Synthetic Data Sets

As in previous chapter, two sets of experiments were performed on both the synthetic data sets of Prelic (DS1 and DS2). Iterative Signature Algorithm (ISA) [BIB03] was used as the biclustering algorithm. In the first experiment the seed was changed to generate the schemes keeping (t_g,t_c) fixed whereas in the second experiment the t_g was changed to get the input schemes keeping the random gene seed and t_c constant. In both the sets of experiments the algorithm was executed on 20 input schemes. The experiments were repeated 20 times and the results were averaged over the runs.

Performance of $BiETclassi$ was compared with the best of the input schemes using BCE and AS. DA and SVM were used for prediction of labels and the algorithms are referred as $BiETDA$ and $BiETSVM$ respectively. Tables 5.4 and 5.5 compare the

Schemes → $t_g, t_c \downarrow$	**Best**	**BiETopti**	**BiETclassi**	
			BiET SVM	**BiET DA**
-0.50 , 2	3402	2540	2508	2489
-0.40 , 2	3830	3002	3002	2981
-0.35 , 2	3618	2652	2562	2087
1 , 1	5218	3580	3173	3156
0 , 1	5860	3768	3721	3712

Table 5.4: Best of input schemes vs BiETclassi on DS1 for the first set of experiments using BCE.

Schemes → $t_g, t_c \downarrow$	**Best**	**BiETopti**	**BiETclassi**	
			BiET SVM	**BiET DA**
-0.50 , 2	0.82	0.82	0.82	0.83
-0.40 , 2	0.78	0.79	0.79	0.83
-0.35 , 2	0.90	0.91	0.92	0.95
1 , 1	0.69	0.70	0.73	0.73
0 , 1	0.55	0.56	0.56	0.56

Table 5.5: Best of input schemes vs BiETclassi on DS1 for the first set of experiments using AS.

BCE and AS of the best input schemes and that of the biclusters produced by $BiETclassi$ on DS1. Tables show results for first set of experiments. The best was computed from the $400\ (20 \times 20)$ schemes. The values shown are the average of the values obtained in the

20 runs of each experiment. Biclusters produced by $BiETclassi$ are also compared with those produced by $BiETopti$.

The results for the second set of experiment are shown in Table 5.6. Results are shown both with BCE and AS.

Evaluation Criteria	**Best**	**BiETopti**	**BiETclassi**	
			BiET SVM	**BiET DA**
BCE	3180	2752	2527	2518
AS	0.81	0.82	0.84	0.85

Table 5.6: Best of input schemes vs BiETclassi on DS1 for the second set of experiment.

The following inferences can be drawn from the tables:

- $BiETclassi$ improves upon the performance of the best input schemes.
- Quality of biclusters produced by $BiETclassi$ is superior to those produced by $BiETopti$ and can be seen in Figures 5.9- 5.10 also.
- $BiETDA$ performs better than $BiETSVM$.

Effect of noise

Noisy data set (DS2) of Prelic et al. was used to study the impact of noise on the performance of $BiETclassi$. Table 5.7 shows the results for the first set of experiments using BCE. Table 5.8 gives the AS values for the same. Again, results of $BiETclassi$ are shown both with SVM and DA.

Table 5.9 gives the result for the second set of experiment for the synthetic data set DS2. The tables show that $BiETclassi$ was able to produce biclusters better than the best of the input schemes even in presence of noise. Even in noisy data set, $BiETDA$ performed better than $BiETSVM$ and also $BiETDA$ outperformed $BiETopti$.

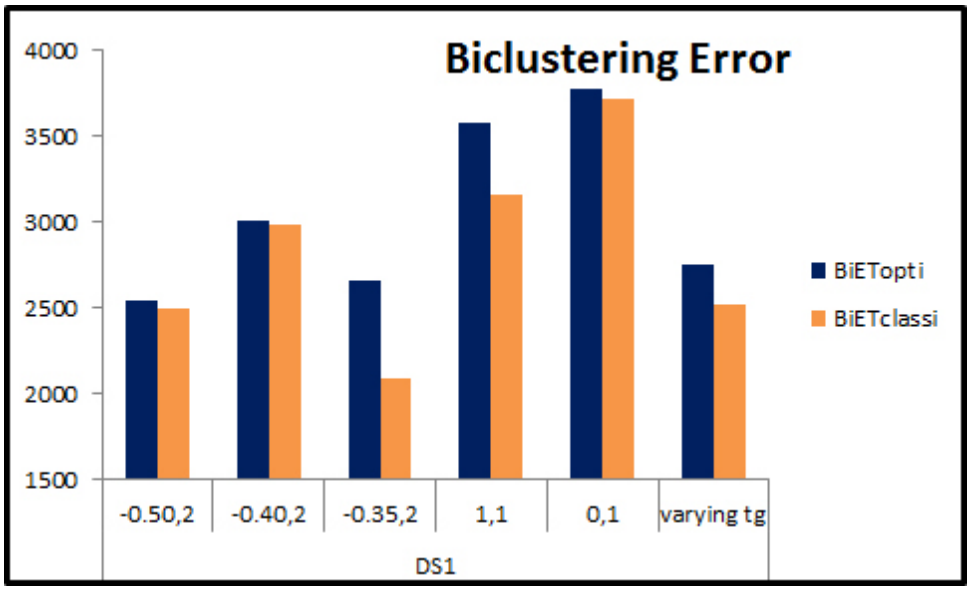

Figure 5.9: BiETclassi compared with BiETopti on DS1 using BCE.

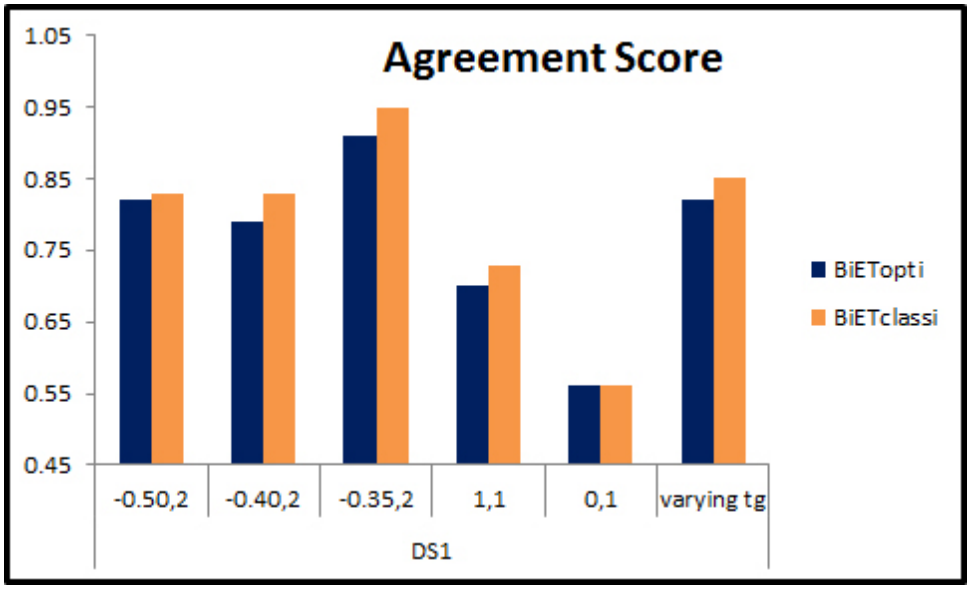

Figure 5.10: BiETclassi compared with BiETopti on DS1 using AS.

Schemes → $t_g, t_c \downarrow$	**Best**	**BiETopti**	**BiETclassi**	
			BiET SVM	**BiET DA**
.90, 1	2865	2431	2314	2300
1, .5	3012	2650	2660	2592
-.35, 2	4187	3256	3187	2891

Table 5.7: Effect of noise on BiETclassi (data set DS2) for the first set of experiments using BCE.

Schemes → $t_g, t_c \downarrow$	**Best**	**BiETopti**	**BiETclassi**	
			BiET SVM	**BiET DA**
.90, 1	0.87	0.88	0.89	0.89
1, .5	0.77	0.78	0.78	0.79
-.35, 2	0.50	0.51	0.50	0.65

Table 5.8: Effect of noise on BiETclassi (data set DS2) for the first set of experiments using AS.

Evaluation Criteria	**Best**	**BiETopti**	**BiETclassi**	
			BiET SVM	**BiET DA**
BCE	2588	2312	2113	1981
AS	0.92	0.92	0.95	0.98

Table 5.9: Effect of noise on BiETclassi (data set DS2) for the second set of experiment.

Time Comparison

The algorithms were also compared on the basis of the time taken by them.

Comparing BiETDA and BiETSVM

Tables 5.4- 5.9 show that $BiETclassi$ performs better in terms of quality when DA was used as a classifier as compared to the case when SVM was used as a classifier. Further, we computed the average time taken by the both $BiETDA$ as well as $BiETSVM$ for the experiments on both DS1 and DS2. The time was only computed for the phase-II i.e. for running the classifier, it does not include time for generating the schemes i.e. Phase-I and also time for forming the consensus i.e. Phase-III is not taken into account. $BiETDA$ took 0.29 seconds on DS1 and 0.21 seconds on DS2 whereas $BiETSVM$ took 7.6 seconds on DS1 and 5.88 seconds on DS2 thereby suggesting that $BiETDA$ was faster than $BiETSVM$.

Comparing BiETclassi and BiETopti

Table 5.10 shows the total time taken by $BiETopti$ and $BiETclassi$ algorithms. Time shown for $BiETclassi$ is with DA as classifier. Table clearly indicates that the time was reduced if ensemble was produced using classifiers instead.

Schemes $t_g, t_c \downarrow$	**Time(sec) BiETopti**	**Time(sec) BiETclassi**
-0.50 , 2	30.3	18.3
-0.40 , 2	28.5	17.7
-0.35 , 2	27	15.6
1 , 1	51	30.3
0 , 1	46.7	24.9
vary t_g	40	22.6

Table 5.10: Time of BiETclassi compared with BiETopti on DS1.

Effect of changing the reference scheme

Experiments were performed to show the effect of reference scheme on the output. Table 5.11

Ref.Scheme (#BC)	BiETDA
π**1(11)**	0.85
π**2(9)**	0.77
π**3(8)**	0.78
π**4(7)**	0.68
π**5(6)**	0.59
π**6(5)**	0.50
π**7(5)**	0.51
π**8(4)**	0.59
π**9(1)**	0.37

DS1

Ref.Scheme(#BC)	BiETDA
π**1(13)**	0.98
π**2(12)**	0.97
π**3(12)**	0.99
π**4(9)**	0.99
π**5(9)**	0.99
π**6(8)**	0.88

DS2

Table 5.11: Effect of reference scheme on AS on both the data sets DS1 and DS2.

shows the impact of changing the reference scheme on the results. It is evident that the results deteriorate as the number of biclusters in the reference scheme reduces. Last row of the table shows that if a scheme with single bicluster is included and is chosen as a reference, the performance deteriorates drastically. Study on the noisy data shows that the results are best when a reference scheme has number of biclusters close to the number (10) in this case of actual biclusters. Thus, if we have a prior knowledge of the number of biclusters in the data set, we should choose the scheme with number of biclusters closest to the actual number of biclusters as the reference. Otherwise, we choose the scheme with maximum number of biclusters as the reference scheme.

5.3.2 Results on Real Data Sets

Experimental studies were performed on the expression data set of Yeast, DLBCL, A. Thaliana and Human Breast Cancer with $BiETSVM$ and $BiETDA$. On each of these, we generated input schemes by running ISA each time with hundred different gene seed vectors. Sizes of the biclusters were kept to be comparable to eliminate the effect of size of biclusters on the p-values. Tables 5.12 - 5.13 show the top 10 biclusters obtained from $BiETclassi$ along with their aligned input biclusters which clearly show that there is a huge improvement in the quality of the biclusters obtained. Table 5.12 shows the GO terms whereas in Table 5.13 evaluation is done on the basis of motifs. As the tables show, $BiETclassi$ outperforms the best of the input schemes on the real data sets most of the times. Also the comparison with $BiETopti$ is shown to endorse that $BiETclassi$ outperforms $BiETopti$.

Comparison of BiETclassi with existing biclustering algorithms and BiETopti

Figure 5.11 shows the comparison of the biclusters produced by $BiETclassi$ with the biclusters produced by existing biclustering algorithms like order-preserving sub matrix (OPSM) [BDCKY03], Cheng and Church (CC) [CC00], BIMAX [PBZ$^+$06] and ISA [BIB03]. The performance of $BiETclassi$ is also compared with the previous ensemble algorithm, $BiETopti$. $BiETclassi$ outperforms the best in each of these organisms except A. Thaliana where OPSM and ISA perform better.

Yeast : $-\log p$-value of GO terms

Best 3 **of the alignment**	**BiET opti**	**BiETclassi**	
		BiET SVM	**BiET DA**
72,72,61	74	74	76
65,65,60	65	66	68
72,56,42	57	59	62
48,48,48	49	49	49
46,43,42	47	47	49
46,31,27	46	47	48
31,31,31	32	33	37
23,23,23	28	29	29
11,11,11	12	12	12
6,6,5	6	6	6

A. Thaliana : $-\log p$-value of GO terms

Best 3 **of the alignment**	**BiET opti**	**BiETclassi**	
		BiET SVM	**BiET DA**
31,31,25	26	27	32
23,23,16	19	23	23
31,24,21	21	24	33
27,26,26	26	27	28
27,23,21	24	27	28
22,22,22	23	23	23
20,18,18	21	21	23
21,20,20	20	20	21
23,19,19	19	22	23
18,17,16	19	21	23

DLBCL : $-\log p$-value of GO terms

Best 3 **of the alignment**	**BiET opti**	**BiETclassi**	
		BiET SVM	**BiET DA**
22,16,5	20	21	23
19,16,16	19	19	22
17,16,16	16	18	25
17,16,16	16	16	18
14,2,2	13	14	15
8,8,7	9	9	12
22,8,6	8	8	8
8,8,8	8	8	12
13,7,7	7	11	15
6,6,6	6	6	6

Breast Cancer : $-\log p$-value of GO terms

Best 3 **of the alignment**	**BiET opti**	**BiETclassi**	
		BiET SVM	**BiET DA**
45,45,45	46	48	48
22,22,22	33	35	36
18,17,17	26	27	27
16,15,14	26	30	31
16,15,15	25	25	26
16,14,5	25	26	26
12,12,12	13	13	13
5,5,5	6	7	7
3,3,3	4	4	4
3,3,2	3	3	3

Table 5.12: Comparison of top 10 biclusters of BiETclassi with best 3 aligned input biclusters on real data sets using GO Terms.

Yeast : $-\log p$-value of motifs

Best 3 **of the alignment**	**BiET opti**	**BiETclassi**	
		BiET SVM	**BiET DA**
32,24,21	32	34	35
32,22,22	32	32	33
24,23,22	23	26	26
18,15,13	20	22	23
15,15,14	20	22	22
14,14,14	15	16	18
11,10,9	13	15	15
9,9,7	12	12	13
8,5,5	9	9	9
7,7,5	10	10	10

A. Thaliana : $-\log p$-value of motifs

Best 3 **of the alignment**	**BiET opti**	**BiETclassi**	
		BiET SVM	**BiET DA**
22,18,18	45	48	50
20,20,18	29	30	30
19,18,17	23	25	27
18,18,18	18	18	18
14,12,10	18	19	19
10,9,8	12	13	13
12,11,10	11	12	13
10,10,10	11	11	11
11,9,8	10	11	12
8,7,7	8	8	8

DLBCL: $-\log p$-value of motifs

Best 3 **of the alignment**	**BiET opti**	**BiETclassi**	
		BiET SVM	**BiET DA**
28,22,16	30	32	33
19,18,16	20	21	21
18,18,18	20	20	20
17,16,16	18	19	20
14,12,10	18	19	20
10,10,8	10	10	10
12,9,9	12	13	14
10,10,9	12	12	12
13,7,7	13	14	14
6,6,6	8	8	8

Breast Cancer : $-\log p$-value of motifs

Best 3 **of the alignment**	**BiET opti**	**BiETclassi**	
		BiET SVM	**BiET DA**
16,15,12	16	17	17
15,15,14	16	18	19
15,13,12	12	15	15
12,11,11	11	13	13
10,10,10	10	10	10
9,9,8	8	9	9
7,7,6	8	8	8
5,4,3	5	5	5
3,3,3	5	6	6
3,2,1	5	6	7

Table 5.13: Comparison of top 10 biclusters of BiETclassi with best 3 aligned input biclusters on real data sets using common motifs.

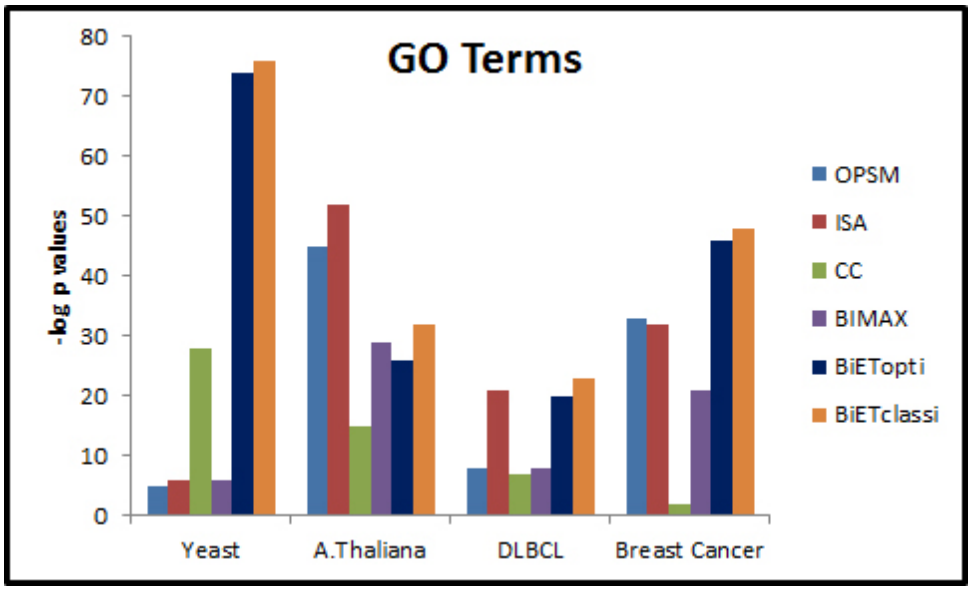

Figure 5.11: BiETclassi compared with OPSM, ISA, CC, BIMAX and BiETopti.

Chapter 6

BiETmetaclus - Biclustering Ensemble Technique using Metaclustering

Hungarian algorithm, an expensive algorithm in terms of time, needs to be invoked to align similar biclusters in different input schemes, in both $BiETopti$ as well as $BiETclassi$. Both these algorithms also require an optimization/classification problem to be solved. This chapter focuses on a technique that does away with the requirement of aligning the input schemes. Moreover there is no requirement of solving either optimization or classification problem. The technique BiETmetaclus, instead, pools in all the biclusters and then group similar biclusters in metaclusters. We propose the use of mutual information (MI) to find similarity between biclusters. It is believed that biclusters, sharing high content of information about each other and less information with other biclusters, form a more cohesive group.

Various similarity measures that have been successfully and satisfactorily used for several years, capture only the linear relationships between the objects. In particular, a vanishing correlation coefficient implies absence of only linear dependencies [HG95, PMBG07, KBG$^+$07, SKD$^+$02, SDSK03]. However, nonlinear relationships like quadratic or sinusoidal etc. may exist between the genes. Kraskov et al. [KSG04], Steur et al. [SKD$^+$02],

Butte and Kohane [BK00] and Michaels et al. [MCA+98] have shown, through their work, that mutual information is a better and general criterion for extracting complex relationships among genes. Kraskov et al. worked with yeast data and found that even though correlation coefficient between few gene pairs was zero, the mutual information between them was non zero thus indicating that other non linear dependencies exist between the genes. Steur et al. showed that higher correlation coefficient implies higher mutual information but two variables having very low values of correlation coefficient may still be related to each other. Butte and Kohane also worked with yeast data set. They hypothesized that gene pairs with high mutual information between them are also related biologically. They constructed networks of various genes having high mutual information between them and found that each network corresponded to some biological activity. They also found mutual information to be a better similarity measure as compared to linear correlation coefficient. According to Priness et al. [PMBG07], it is resistant to outliers, noise and missing data.

Metaclusters are obtained by collecting the biclusters with high pairwise mutual information. The concept of well separated seeds is used to minimize the between metacluster information. Voting is then done on metaclusters to form the final consensus. To endorse the use of mutual information as a similarity measure, we compare it with Bicluster Similarity Index (BSI) also to form the metaclusters. BSI, discussed in Section 4.1.3, is the modified form of the measure that has been successfully used to compute similarity between clusters [KG07]. The algorithms using MI and BSI are respectively called $BiETMI$ and $BiETBSI$. The results show that the biclusters produced by these algorithms are better than the input biclusters. It was also observed that the biclusters produced using MI are biologically more significant than the ones produced by the other similarity measure, BSI. Experiments also show that the time taken is greatly reduced as compared to $BiETopti$ and $BiETclassi$. $BiETopti$ algorithm takes $O((N+d)*k)^{3.5}$ time. $BiETclassi$ uses classifiers like DA and SVM having the complexity of $O(N^3)$.

The classifier is invoked $H \cdot k$ times in the algorithm. Thus the total time complexity is $O(H \cdot k \cdot N^3)$. $BiETmetaclus$ on the other hand uses similarity measures like Mutual Information and BSI. The time complexity of $BiETmetaclus$ is $O((H \cdot k)^2 \cdot Nd)$. Ignoring the small constants H and k time complexity of $BiETclassi$, $BiETmetaclus$ and $BiETopti$ is $O(N^3)$, $O(Nd)$ and $O((N+d)^{3.5})$ respectively. The value of d is generally much smaller as compared to N. Thus, $BiETmetaclus$ is much faster than both $BiETopti$ as well as $BiETclassi$.

6.1 Preliminaries

We have used two measures of similarity to group biclusters viz. MI and BSI. In this section we give a brief description of MI. The other measure BSI was discussed in Section 4.1.3.

Mutual Information

Mutual Information between two random variables X and Y is a measure of information contained in X about Y and vice versa. If given a value of X, it is easy to predict the value of Y then X contains good amount of information about Y. Clearly, if X and Y are dependent, X and Y can predict each other well and we say that the mutual information between them is high. And, if X and Y are independent, they cannot predict each others behavior and we say that the mutual information between them is zero. Mutual information is defined as a measure of divergence of the observed joint distribution of X and Y from the hypothesis that X and Y are independent and is given as:

$$MI(x,y) = -\sum_{x}\sum_{y} p(x,y) \log \frac{p(x) * p(y)}{p(x,y)}$$

As it is a function of the distribution of the variables X and Y, it does not depend on the actual values taken by X and Y, rather it depends on their probability distributions.

The unit of mutual information is defined corresponding to the base of the logarithm in the above equation i.e. nats for log_e, bits for log_2, and Hartleys for log_{10}. Mutual information is non negative and symmetrical i.e. $MI(X,Y) = MI(Y,X)$. Also, mutual information is zero if and only if X and Y are statistically independent i.e. vanishing mutual information does imply that the two variables are independent. However, it is not a true distance between distributions as it does not satisfy the triangle inequality. It does not require normalization and is robust towards noise, outliers and missing data.

Mutual information is a function of joint probability distribution and the marginal probability distribution. However, one generally does not have a prior knowledge about the distributions. Thus one needs to estimate them. Two broad classes of approaches namely Parametric and Nonparametric are used to estimate the probability distribution functions. Parametric method involves assuming a model for the probability density function and then determining the various parameters from the data. However, if the assumption is poor the results are poor. In contrast to the parametric approach no assumption about the underlying probability density function is made in the nonparametric approach. Histogram method and Kernel density estimation are two methods of estimating probability density function by the nonparametric approach.

6.2 BiETmetaclus

Our algorithm works in 3 phases. Schemes are generated in phase-I and this is same as that in the last two chapters. Phase-II deals with the formation of metaclusters and voting is done to form the consensus in phase-III.

6.2.1 Phase II: Metacluster Formation

In this step, biclusters of all the schemes are collected in a pool(G) and groups are formed on the basis of mutual information. To compute mutual information between two biclusters drawn from two different schemes, columns of the membership matrix (as described in Table 4.4) are used. Groups, called metaclusters are formed so that they share maximum information within the biclusters in a group and minimum information with the biclusters in other metaclusters. Thus metaclusters are formed with the aim to maximize within metacluster information and minimize between metacluster information.

To be able to form well separated groups, we construct a set S of seed biclusters. Initially this set is empty. The first seed bicluster BC_1 is chosen at random from G, starred and then all biclusters with high mutual information with BC_1 are grouped together to form one metacluster. Second seed bicluster BC_2 is chosen farthest from S i.e. the one that has least mutual information with BC_1. Biclusters with high mutual information with BC_2 are put in the second metacluster. Next seed bicluster is chosen farthest from S i.e. the one that has least mutual information with both BC_1 and BC_2. The process of forming metaclusters and selecting a farthest seed bicluster is repeated until no more biclusters are left to be grouped. This method of choosing the seed has also been used in [GA08] and [APW$^+$99]. Figure 6.1 shows the formation of metaclusters in pictorial form. The pseudo code for this phase is shown in the Algorithm 4.

In the next phase a representative of each metacluster is formed. The number of output biclusters is thus determined by the algorithm itself without requiring the user to specify it.

6.2.2 Phase-III: Consensus Formation

Previous phase resulted in the formation of many metaclusters, each having several similar biclusters in it. In this phase, we select one representative from each metacluster. The

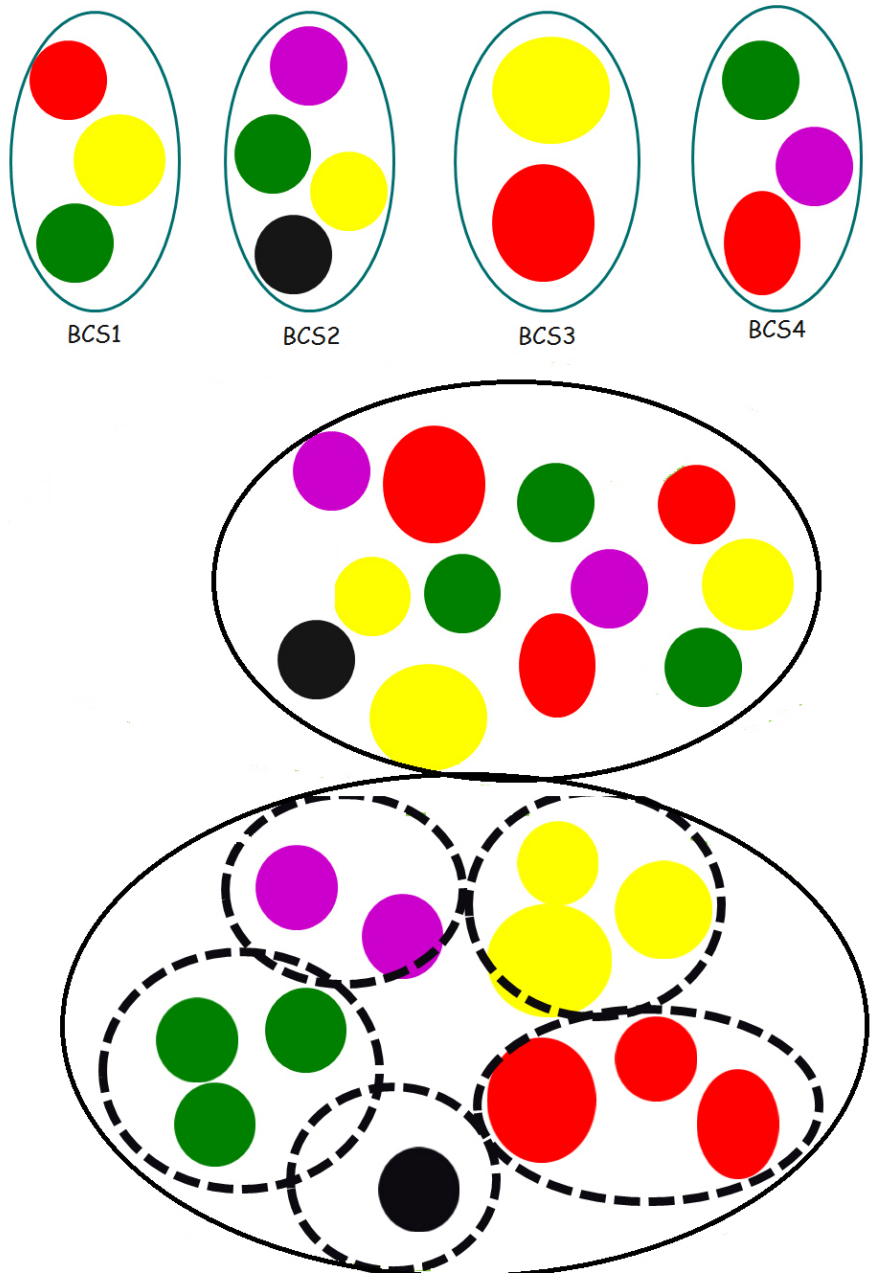

Figure 6.1: Visualization of metacluster formation.

Input: G, the set of all the biclusters in the pool.

Let S be the set of seed biclusters.

Initially S is empty.

Initially all the biclusters are ungrouped.

let $i = 1$

BC_i^* = random(), the first seed bicluster

while *no more bicluster is left to be grouped* **do**

  $G(i) =$ Group of biclusters in G with high MI with the BC_i^*

  Mark all the biclusters in $G(i)$ as grouped.

  S=$S \cup BC_i^*$

  uBC=all ungrouped biclusters

  BC_{i+1}^*=$argmin_{k \in uBC}(max_{BC^* \in S} MI(BC_k, BC^*))$

  $i = i + 1$

end

Algorithm 4: Pseudo code for metacluster formation.

Input: $G(i)$, all metaclusters formed in Phase-II

for *all groups* $G(i)$ **do**

  compute frequency of (gene, condition) pairs in all biclusters of $G(i)$ and output the representative bicluster $BC(i)$ containing the (gene, condition) pairs with frequency $\geq \eta$.

end

Algorithm 5: Pseudo code for selecting a representative from a metacluster.

bicluster that shares the maximum information with the rest of biclusters in the metacluster is a good candidate for the representative of the group. However, such a candidate has the limitation of being one of the biclusters in the metacluster. On the other hand, there may be some (gene, condition) pairs in other biclusters (of the same metacluster) that are important and should have been a part of the final bicluster. Thus, instead, we form the representative bicluster on the basis of frequency of (gene, condition) pairs. Frequency of all (gene, condition) pairs is calculated and the pairs whose value is greater than the threshold η are reported as the elements of the final bicluster. Experimentally also, it was observed that forming a representative in this manner is a better alternative than the first method. Thus, this method was used in the experiments performed. Algorithm 5 provides the pseudo code for this phase.

6.3 Experimental Results

Experiments were performed both on synthetic data sets and real gene expression data sets to show the efficiency of our approach. $BiETmetaclus$ involves formation of metaclusters which are formed by grouping similar biclusters. MI/BSI is used to find the similarity between the biclusters and in our experiments, biclusters with similarity 90% or more are grouped to form a metacluster. The process of forming the metaclusters is repeated till no more bicluster is left to be grouped. After forming the metaclusters, representative of each metacluster is formed. This is achieved by taking those (gene, condition) pairs having frequency greater than or equal to threshold value η fixed at 60%. The biclusters produced by $BiETmetaclus$ were also compared with both the previous algorithms, $BiETopti$ and $BiETclassi$.

6.3.1 Results on Synthetic Data Sets

Two sets of experiments were performed as in the previous two algorithms on both the data sets of Prelic. In the first experiment the seed was changed to generate the schemes keeping (t_g,t_c) fixed whereas in the second experiment the t_g was changed to get the input schemes keeping the random gene seed and t_c constant. In both the sets of experiments the algorithm was executed on 20 input schemes. The experiments were repeated 20 times and the results were averaged over the runs.

Schemes	**Best**	**BiETopti**	**BiETclassi**	**BiETmetaclus**	
$t_g, t_c \downarrow$			**BiETDA**	**BiETBSI**	**BiETMI**
-0.50 , 2	3402	2540	2489	2540	2540
-0.40 , 2	3830	3002	2981	2998	2990
-0.35 , 2	3618	2652	2087	2562	2426
1 , 1	5218	3580	3156	3521	3428
0 , 1	5860	3768	3712	3740	3740

Table 6.1: Best of input schemes vs BiETmetaclus on DS1 for the first set of experiments using BCE.

Schemes	**Best**	**BiETopti**	**BiETclassi**	**BiETmetaclus**	
$t_g, t_c \downarrow$			**BiETDA**	**BiETBSI**	**BiETMI**
-0.50 , 2	0.82	0.82	0.83	0.82	0.82
-0.40 , 2	0.77	0.79	0.83	0.81	0.81
-0.35 , 2	0.90	0.91	0.95	0.92	0.93
1 , 1	0.69	0.70	0.73	0.71	0.72
0 , 1	0.54	0.56	0.56	0.56	0.56

Table 6.2: Best of input schemes vs BiETmetaclus on DS1 for the first set of experiments using AS.

Table 6.1 compares the performance of $BiETmetaclus$ with the best input schemes and also that of $BiETclassi$ and $BiETopti$ on DS1 in terms of BCE. We concluded in the last chapter that $BiETDA$ performed better than $BiETSVM$ so for the comparison, only $BiETDA$ is taken. The values shown in the table are the average of the values obtained in the 20 runs of each experiment. Column 2 gives the best values, of the input schemes, over all the runs. The table shows the results for first set of experiments. Similarly Table 6.2 compares the performance of various algorithms in terms of other evaluation method, AS. The results for the second set of experiment are shown in Table 6.3.

Evaluation Criteria	**Best**	**BiETopti**	**BiETclassi**	**BiETmetaclus**	
			BiETDA	**BiETBSI**	**BiETMI**
BCE	3180	2752	2518	2732	2725
AS	0.81	0.82	0.85	0.82	0.83

Table 6.3: Best of input schemes vs BiETmetaclus on DS1 for the second set of experiment.

The following inferences can be drawn from the tables:

- $BiETmetaclus$ improves upon the performance of the best input scheme both with MI as well as BSI.

- $BiETmetaclus$ performs better than $BiETopti$ in terms of quality and is shown in Figures 6.2 and 6.3 where all the three approaches are compared.

- $BiETMI$ performs better than $BiETBSI$.

However, observe that $BiETclassi$ performs better than $BiETmetaclus$.

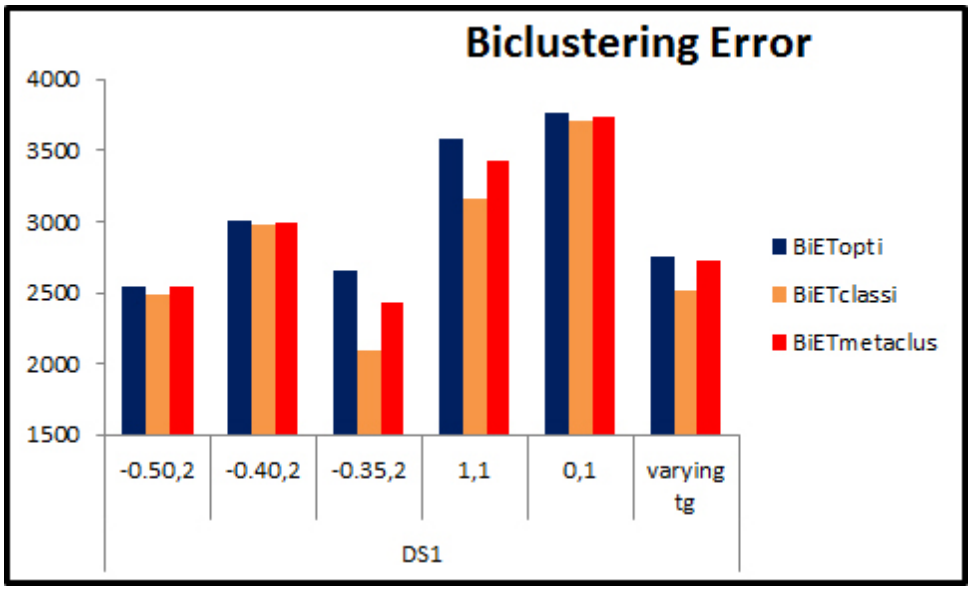

Figure 6.2: BiETmetaclus compared with BiETopti and BiETclassi on DS1 using BCE.

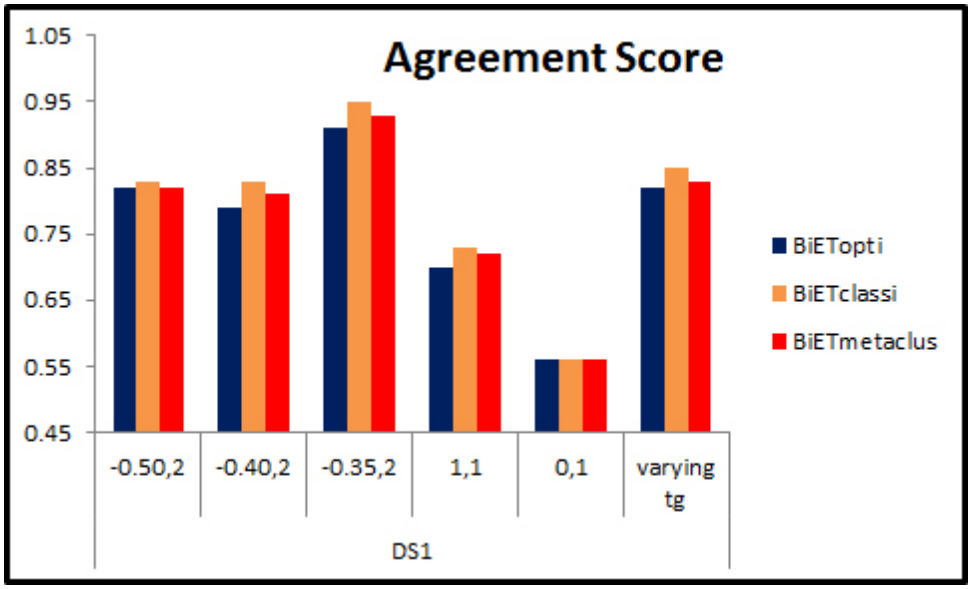

Figure 6.3: BiETmetaclus compared with BiETopti and BiETclassi on DS1 using AS.

Effect of noise

The performance of $BiETmetaclus$ was also studied on data set (DS2) of Prelic et al. to see the impact of noise. Tables 6.4- 6.5 show the results using BCE and AS for the first set of experiments. Again, results of $BiETmetaclus$ are shown with both the similarity measures used: MI and BSI. Similarly results for the second set of experiment are shown in Table 6.6. The tables show that $BiETmetaclus$ was able to extract biclusters better than the best of the input schemes even in presence of noise. Also, even in presence of noise $BiETMI$ performs better than $BiETBSI$.

Schemes	**Best**	**BiETopti**	**BiETclassi**	**BiETmetaclus**	
$t_g, t_c \downarrow$			**BiETDA**	**BiETBSI**	**BiETMI**
.90, 1	2865	2431	2300	2412	2412
1, .5	3012	2650	2592	2631	2618
-.35, 2	4187	3256	2891	3203	3195

Table 6.4: Effect of noise on BiETmetaclus (data set DS2) for the first set of experiments using BCE.

Schemes	**Best**	**BiETopti**	**BiETclassi**	**BiETmetaclus**	
$t_g, t_c \downarrow$			**BiETDA**	**BiETBSI**	**BiETMI**
.90, 1	0.87	0.88	0.89	0.88	0.88
1, .5	0.77	0.78	0.79	0.78	0.78
-.35, 2	0.50	0.51	0.65	0.52	0.54

Table 6.5: Effect of noise on BiETmetaclus (data set DS2) for the first set of experiments using AS.

Evaluation Criteria	Best	BiETopti	BiETclassi	BiETmetaclus	
			BiETDA	BiETBSI	BiETMI
BCE	2588	2312	1981	2100	2091
AS	0.92	0.92	0.98	0.95	0.96

Table 6.6: Effect of noise on BiETmetaclus (data set DS2) for the second set of experiment.

Time Comparison of BiETopti, BiETclassi and BiETmetaclus on DS1

It was observed that $BiETclassi$ is able to produce biologically better biclusters than $BiETmetaclus$ but at the cost of time as it can be clearly seen from Table 6.7. The

Schemes $t_g, t_c \downarrow$	Time(sec) BiETopti	Time(sec) BiETclassi	Time(sec) BiETmetaclus
-0.50 , 2	30.3	18.3	9.9
-0.40 , 2	28.5	17.7	8.9
-0.35 , 2	27	15.6	8.4
1 , 1	51	30.3	23.1
0 , 1	46.7	24.9	13.96
vary t_g	40	22.6	7.6

Table 6.7: Time of BiETmetaclus compared with BiETopti and BiETclassi on DS1.

time shown is the total time taken for all the approaches. Time taken for the approach $BiETclassi$ is with DA as the classifier. For $BiETmetaclus$, time has been shown wherein mutual information is used as the similarity measure. The time taken by $BiETmetaclus$ is less than time taken by $BiETclassi$. $BiETmetaclus$ wins over $BiETclassi$ as far as time is concerned. Thus, there is a tradeoff between the two approaches as far as quality and time are concerned. Note that with respect to $BiETopti$, $BiETmetaclus$ improves as regard to quality as well as time.

6.3.2 Results on Real Data Sets

Experimental studies were performed on the real expression data sets with $BiETmetaclus$. Tables 6.8- 6.9 shows the top 10 biclusters obtained from $BiETmetaclus$ along with their aligned input biclusters which clearly show that there is improvement in the quality of the biclusters obtained. Table 6.8 shows the comparison based on GO terms whereas Table 6.9 shows the comparison using motifs. The tables show that $BiETmetaclus$ outperforms not only the best of the input schemes but also the biclusters produced by $BiETopti$. Tables also show that $BiETMI$ performs better than $BiETBSI$.

Time Comparison of BiETopti, BiETclassi and BiETmetaclus on real data sets

$BiETclassi$ is able to produce biologically better biclusters on real data sets than $BiETmetaclus$ but at the cost of time as it can be clearly seen from Table 6.10. $BiETmetaclus$ wins over $BiETclassi$ as far as time is concerned. Thus, there is a tradeoff between the two approaches as far as quality and time are concerned. Note that with respect to $BiETopti$, $BiETmetaclus$ improves as regard to quality as well as time.

Comparison of BiETmetaclus with existing biclustering algorithms, BiETopti and BiETclassi

Figure 6.4 shows the comparison of the biclusters produced by $BiETmetaclus$, with the biclusters produced by existing biclustering algorithms like order-preserving sub matrix (OPSM) [BDCKY03], Cheng and Church (CC) [CC00], BIMAX [PBZ$^+$06] and ISA [BIB03]. Figure also shows comparison with the previous two ensemble algorithms, $BiETopti$ and $BiETclassi$. $BiETmetaclus$ outperforms the best of the biclustering algorithms in each of these organisms except A. Thaliana. As far as the ensemble algorithms are concerned, performance of $BiETmetaclus$ is between $BiETopti$ and $BiETclassi$.

Yeast : $-\log p$-value of GO terms

Best 3 of the input	BiETopti	BiETclassi	BiETmetaclus	
		BiETDA	BiETBSI	BiETMI
72,72,61	74	76	74	74
65,65,60	65	68	66	67
72,56,42	57	62	59	59
48,48,48	49	49	49	49
46,43,42	47	49	47	48
46,31,27	46	48	48	48
31,31,31	32	37	35	36
23,23,23	28	29	28	29
11,11,11	12	12	12	12
6,6,5	6	6	6	6

A. Thaliana : $-\log p$-value of GO terms

Best 3 of the input	BiETopti	BiETclassi	BiETmetaclus	
		BiETDA	BiETBSI	BiETMI
31,31,25	26	32	27	31
23,23,16	19	23	23	23
31,24,21	21	33	24	25
27,26,26	26	28	27	28
27,23,21	24	28	27	27
22,22,22	23	23	23	23
20,18,18	21	23	21	22
21,20,20	20	21	20	21
23,19,19	19	23	21	21
18,17,16	19	23	21	22

DLBCL : $-\log p$-value of GO terms

Best 3 of the input	BiETopti	BiETclassi	BiETmetaclus	
		BiETDA	BiETBSI	BiETMI
22,16,5	20	23	22	22
19,16,16	19	22	19	19
17,16,16	16	25	18	23
17,16,16	16	18	16	16
14,2,2	13	15	14	15
8,8,7	9	12	10	11
22,8,6	8	8	8	7
8,8,8	8	12	9	9
13,7,7	7	15	11	12
6,6,6	6	6	6	6

Breast Cancer : $-\log p$-value of GO terms

Best 3 of the input	BiETopti	BiETclassi	BiETmetaclus	
		BiETDA	BiETBSI	BiETMI
45,45,45	46	48	47	47
22,22,22	33	36	35	35
18,17,17	26	27	26	26
16,15,14	26	31	28	30
16,15,15	25	26	25	25
16,14,5	25	26	25	25
12,12,12	13	13	13	13
5,5,5	6	7	7	7
3,3,3	4	4	4	4
3,3,2	3	3	3	3

Table 6.8: Comparison of top 10 biclusters of BiETmetaclus with 3 best aligned input biclusters on real data sets using GO terms.

Yeast : $-\log p$-value of motifs

Best 3 of the input	BiETopti	BiETclassi	BiETmetaclus	
		BiETDA	BiETBSI	BiETMI
32,24,21	32	35	34	34
32,22,22	32	33	33	33
24,23,22	23	26	24	25
18,15,13	20	23	22	22
15,15,14	20	22	22	22
14,14,14	15	18	17	17
11,10,9	13	15	14	15
9,9,7	12	13	12	12
8,5,5	9	9	9	9
7,7,5	10	10	10	10

A. Thaliana : $-\log p$-value of motifs

Best 3 of the input	BiETopti	BiETclassi	BiETmetaclus	
		BiETDA	BiETBSI	BiETMI
22,18,18	45	50	48	49
20,20,18	29	30	29	29
19,18,17	23	27	25	26
18,18,18	18	18	18	18
14,12,10	18	19	18	19
10,9,8	12	13	13	13
12,11,10	11	13	12	12
10,10,10	11	11	11	11
11,9,8	10	12	11	11
8,7,7	8	8	8	8

DLBCL : $-\log p$-value of motifs

Best 3 of the input	BiETopti	BiETclassi	BiETmetaclus	
		BiETDA	BiETBSI	BiETMI
28,22,16	30	33	32	32
19,18,16	20	21	20	20
18,18,18	20	20	20	20
17,16,16	18	20	19	20
14,12,10	18	20	19	20
10,10,8	10	10	10	10
12,9,9	12	14	13	13
10,10,9	12	12	12	12
13,7,7	13	14	13	13
6,6,6	8	8	8	8

Breast Cancer : $-\log p$-value of motifs

Best 3 of the input	BiETopti	BiETclassi	BiETmetaclus	
		BiETDA	BiETBSI	BiETMI
16,15,12	16	17	16	16
15,15,14	16	19	18	19
15,13,12	12	15	15	15
12,11,11	11	13	12	13
10,10,10	10	10	10	10
9,9,8	8	9	9	9
7,7,6	8	8	8	8
5,4,3	5	5	5	5
3,3,3	5	6	5	6
3,2,1	5	7	6	6

Table 6.9: Comparison of top 10 biclusters of BiETmetaclus with 3 best aligned input biclusters on real data sets using common motifs.

Organism	BiETopti Time(sec)	BiETclassi Time(sec)	BiETmetaclus Time(sec)
Yeast	8200	801	337
A.Thaliana	565	225	165
DLBCL	647	125	86
Breast Cancer	180	155	39

Table 6.10: Comparison of time on real data sets

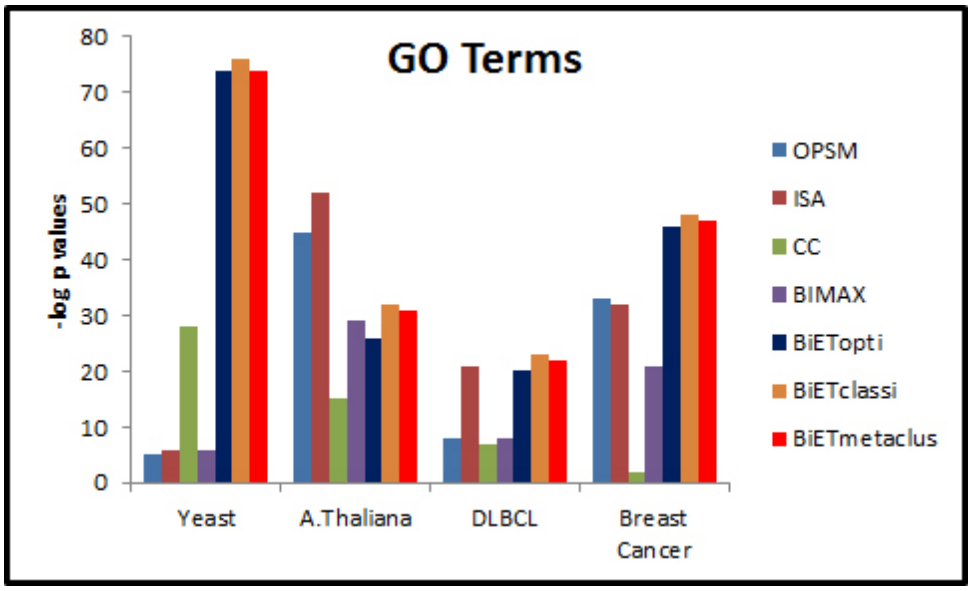

Figure 6.4: BiETmetaclus compared with OPSM, ISA, CC, BIMAX, BiETopti and BiETclassi

Chapter 7

Concluding Remarks

The task of analyzing the humongous gene expression data is greatly simplified by organizing the genes into groups that are responsible for different biological processes, thereby helping the discovery, validation and understanding of various diseases. Biclustering algorithms are particularly useful for this job as a gene may be responsible for more than one biological activity and different set of conditions may trigger different genomic activities. Several biclustering algorithms exist in literature, each one delivering a solution based on some heuristics. A solution that performs well for one heuristic may not fare well so well with respect to another. Idea of ensembling various solutions is to help an end user to obtain a solution that conforms to most of them. As the solution is obtained by combining the knowledge contained in various solutions, it is expected to be better than (or at least as good as) most of them with the advantage that the end user need not worry about which heuristic is best suited for the application at hand. Ensemble methods have also been designed with the aim to provide solutions which are more robust towards random seeds and input parameters.

We have presented three ensemble algorithms for biclustering solutions in this work. BiETopti, the first algorithm is based on an optimization technique. In order to formulate the objective function and the constraints, global labels are defined. The algorithm forces

a limitation of having fixed number of biclusters in the input schemes. To do away with this limitation, another algorithm BiETclassi, based on classifiers is designed. As an individual bicluster is subjected to a classifier on the reduced set of conditions, we do not have to worry about the number of biclusters being same in all the schemes.

The results obtained by both the algorithms are promising but both of them are compute intensive as they involve label correspondence and an optimization problem to be solved both of which require lot of computation. Thus, another algorithm, BiETmetaclus, based on the technique of metaclustering using mutual information is proposed.

Experiments were performed on synthetic data sets as well as real data sets. All the three ensemble algorithms produced biclusters that are better than the input biclusters. Comparing the three: both BiETclassi and BiETmetaclus perform better than BiETopti in terms of time as well as quality. There is a tradeoff between BiETclassi and BiETmetaclus. BiETclassi provides superior biclusters than BiETmetaclus when quality is considered and BiETmetaclus comes out to be the clear winner when time is considered as the comparing parameter.

It would be interesting to see how the benefits of both BiETclassi and BiETmetaclus can be exploited to improve upon the quality and the time simultaneously. It would be nice to see the application of these algorithms in other domains as well.

List of Publications

- Geeta Aggarwal and Neelima Gupta. BiETopti-BiClustering Ensemble Using Optimization Techniques, Proceedings of the 13th International Conference on Advances in Data Mining: Applications and Theoretical Aspects (ICDM'13), 181-192, 2013, New York, USA, Springer.

- Geeta Aggarwal and Neelima Gupta. BEMI Bicluster Ensemble Using Mutual Information. 12th International Conference on Machine Learning and Applications (ICMLA), 321-324, 2013, Miami, Florida, USA, IEEE Computer Society.

- Geeta Aggarwal and Neelima Gupta. BiETopti-BiClustering Ensemble Technique Using Optimisation, International Journal of Bioinformatics Research and Applications (IJBRA), 109-130, 2017.

Bibliography

[ABB+00] Michael Ashburner, Catherine A. Ball, Judith A. Blake, David Botstein, Heather Butler, Michael J. Cherry, Allan P. Davis, Kara Dolinski, Selina S. Dwight, Janan T. Eppig, Midori A. Harris, David P. Hill, Laurie Issel-Tarver, Andrew Kasarskis, Suzanna Lewis, John C. Matese, Joel E. Richardson, Martin Ringwald, Gerald M. Rubin, and Gavin Sherlock. Gene Ontology: Tool for the unification of biology. *Nature Genetics*, 25(1):25–29, May 2000.

[ACN08] Nir Ailon, Moses Charikar, and Alantha Newman. Aggregating inconsistent information: Ranking and clustering. *Journal of the ACM*, 55(5):1–23, November 2008.

[AG13a] Geeta Aggarwal and Neelima Gupta. BiETopti-BiClustering Ensemble Using Optimization Techniques. In *ICDM*, pages 181–192, July 2013.

[AG13b] Geeta Aggarwal and Neelima Gupta. BEMI: Bicluster Ensemble Using Mutual Information. In *ICMLA (1)*, pages 321–324, Dec 2013.

[AG17] Geeta Aggarwal and Neelima Gupta. Bietopti: biclustering ensemble technique using optimisation. *IJBRA*, 13(2):109–130, 2017.

[AK08] Hanan G. Ayad and Mohamed S. Kamel. Cumulative Voting Consensus Method for Partitions with Variable Number of Clusters. *IEEE Trans-*

actions on Pattern Analysis and Machine Intelligence, 30(1):160–173, January 2008.

[AK10] Hanan G. Ayad and Mohamed S. Kamel. On voting-based consensus of cluster ensembles. *Pattern Recognition*, 43(5):1943–1953, May 2010.

[AMK00] Tatsuya Akutsu, Satoru Miyano, and Satoru Kuhara. Inferring qualitative relations in genetic networks and metabolic pathways. *Bioinformatics*, 16(8):727–734, 2000.

[APS06] Teresa K. Atwood and David J. Parry-Smith. *Introduction to Bioinformatics*. Pearsons Education, 2006.

[APW+99] Charu C. Aggarwal, Cecilia M. Procopiuc, Joel L. Wolf, Philip S. Yu, and Jong S. Park. Fast algorithms for projected clustering. In *International Conference on Management of Data, Philadelphia, Pennsylvania, USA (SIGMOD)*, pages 61–72. ACM Press, 1999.

[BDCKY03] Amir Ben-Dor, Benny Chor, Richard M. Karp, and Zohar Yakhini. Discovering local structure in gene expression data: The order-preserving submatrix problem. *Journal of Computational Biology*, 10(3/4):373–384, 2003.

[BDSY99] Amir Ben-Dor, Ron Shamir, and Zohar Yakhini. Clustering gene expression patterns. *Journal of Computational Biology*, 6(3/4):281–297, 1999.

[BIB03] Sven Bergmann, Jan Ihmels, and Naama Barkai. Iterative signature algorithm for the analysis of large-scale gene expression data. In *Physical review. E, Statistical, nonlinear, and soft matter physics*, volume 67, March 2003.

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[BK00] Atul J. Butte and Isaac S. Kohane. Mutual information relevance networks: Functional genomic clustering using pairwise entropy measurements. *Pacific Symposium on Biocomputing*, 5:415–426, 2000.

[CC00] Yizong Cheng and George M. Church. Biclustering of Expression Data. In *Proceedings of Eighth International Conference on Intelligent Systems for Molecular Biology, La Jolla/ San Diego, CA, USA*, pages 93–103, August 2000.

[CKB10] Gabor Csardi, Zoltan Kutalik, and Sven Bergmann. Modular analysis of gene expression data with R. *Bioinformatics*, 26:1376–1377, 2010.

[Cla99] Jean-Michele Claverie. Computational methods for the identification of differential and coordinated gene expression. *Human Molecular Genetics*, 8(10):1821–1832, 1999.

[CST00a] Andrea Califano, Gustavo Stolovitzky, and Yuhai Tu. Analysis of Gene Expression Microarrays for Phenotype Classification. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology, La Jolla/ San Diego, CA, USA*, pages 75–85, August 2000.

[CST00b] Nello Cristianini and JoH Shawe-Taylor. *An Introduction to Support Vector Machines And Other Kernel-based Learning Methods*. Cambridge University Press, New York, NY, USA, 2000.

[DF03] Sandrine Dudoit and Jane Fridlyand. Bagging to improve the accuracy of a clustering procedure. In *Bioinformatics*, volume 19, pages 1090–1099, 2003.

[DGM+07] Carlotta Domeniconi, Dimitrios Gunopulos, Sheng Ma, Bojun Yan, Muna Al-Razgan, and Dimitris Papadopoulos. Locally adaptive metrics for clustering high dimensional data. *Data Mining and Knowledge Discovery*, 14(1):63–97, 2007.

[DHS01] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. Wiley, New York, second edition, 2001.

[Die00] Thomas G. Dietterich. Ensemble methods in machine learning. In *Proceedings of the First International Workshop on Multiple Classifier Systems*, MCS'00, pages 1–15, London, UK, 2000. Springer-Verlag.

[DLS99] Patrik D'Haeseleer, Shoudan Liang, and Roland Somogyi. Tutorial:Gene Expression Data Analysis and Modeling. In *Proceedings of Pacific Symposium on Biocomputing, Hawaii*, January 1999.

[DWFS98] Patrik D'haeseleer, Xiling Wen, Stefanie Fuhrman, and Roland Somogyi. Mining the gene expression matrix: Inferring gene relationships from large scale gene expression data. In *Proceedings of the Second International Workshop on Information Processing in Cell and Tissues*, IPCAT '97, pages 203–212, New York, NY, USA, 1998. Plenum Press.

[DWH02] Evgenia Dimitriadou, Andreas Weingessel, and Kurt Hornik. A combination scheme for fuzzy clustering. In *Advances in Soft Computing AFSS*, volume 2275 of *Lecture Notes in Computer Science*, pages 332–338. Springer, 2002.

[ESBB98] Michael B. Eisen, Paul T. Spellman, Patrick O. Brown, and David Botstein. Cluster analysis and display of genome wide expression patterns. *Proceedings of the National Academy of Sciences of USA*, 95(25):14863–14868, 1998.

[FB03] Bernd Fischer and Joachim M. Buhmann. Bagging for path-based clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25:1411–1415, November 2003.

[FB04] Xiaoli Z. Fern and Carla E. Brodley. Solving cluster ensemble problems by bipartite graph partitioning. In *Proceedings of the international Conference on Machine Learning*, volume 69. ACM, 2004.

[Fis36] Ronald A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, 7(7):179–188, 1936.

[FJ05] Ana L. N. Fred and Anil K. Jain. Combining multiple clusterings using evidence accumulation. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, 27:835–850, 2005.

[Fre01] Ana L. N. Fred. Finding consistent clusters in data partitions. In *Proceedings of the Second International Workshop on Multiple Classifier Systems*, MCS '01, pages 309–318, London, UK, 2001. Springer-Verlag.

[Fre02] Ana L. N. Fred. Data Clustering Using Evidence Accumulation. In *Proceedings of the 16th International Conference on Pattern Recognition*, pages 276–280, 2002.

[GA08] Neelima Gupta and Seema Aggarwal. SISA: Seeded Iterative Signature Algorithm for Biclustering Gene Expression Data. In *IADIS European Conference on Data Mining*, pages 124–128. IADIS, 2008.

[GDT09] Francesco Gullo, Carlotta Domeniconi, and Andrea Tagarelli. Projective clustering ensembles. In *Proceedings of the 2009 Ninth IEEE International Conference on Data Mining*, ICDM '09, pages 794–799, Washington, DC, USA, 2009. IEEE Computer Society.

[GSIM09] Reza Ghaemi, Nasir Sulaiman, Hamidah Ibrahim, and Norwati Mustapha. A survey : Clustering ensembles techniques. *Proceedings of World Academy of Science, Engineering and Technology,* February 2009.

[GSS91] Eldon J. Gardner, Michael J. Simmons, and D. Peter Snustad. *Principles of Genetics*. John Wiley and Sons, 1991.

[GVSS03] I. Gat-Viks, Roded Sharan, and Ron Shamir. Scoring clustering solutions by their biological relevance. *Bioinformatics*, 19(18):2381–2389, 2003.

[Har72] J. A. Hartigan. Direct Clustering of a Data Matrix. *Journal of the American Statistical Association*, 67(337):123–129, 1972.

[HG95] Hanspeter Herzel and Ivo Grosse. Measuring Correlations in symbols sequences. *Physica A: Statistical Mechanics and its Applications*, 216(4):518–542, 1995.

[HN11] Blaise Hanczar and Mohamed Nadif. Using the bagging approach for biclustering of gene expression data. In *Neurocomputing*, volume 74(10), pages 1595–1605, Amsterdam, The Netherlands, May 2011. Elsevier Science Publishers B. V.

[HN12] Blaise Hanczar and Mohamed Nadif. Ensemble methods for biclustering tasks. *Pattern Recognition*, 45(11):3938–3949, 2012.

[HSL08] Da W. Huang, Brad T. Sherman, and Richard A. Lempicki. Systematic and integrative analysis of large gene lists using DAVID bioinformatics resources. *Nature Protocols*, 4(1):44–57, December 2008. http://ncbi//david/gov.

[Hun93] Lawrence Hunter. Molecular Biology for Computer Scientists. In *Artificial intelligence and molecular biology*, pages 1–46. American Association for Artificial Intelligence, 1993.

[HY04] Xiaohua Hu and Illhoi Yoo. Cluster ensemble and its applications in gene expression analysis. In *Proceedings of the second conference on Asia-Pacific bioinformatics - Volume 29*, APBC '04, pages 297–302, Darlinghurst, Australia, 2004. Australian Computer Society, Inc.

[HZGD05] Libi Hertzberg, Or Zuk, Gad Getz, and Eytan Domany. Finding Motifs in Promoter Regions. *Journal of Computational Biology*, 12(3):314–330, 2005.

[IFB+02] Jan Ihmels, Gilgi Friedlander, Sven Bergmann, Ofer Sarig, Yaniv Ziv, and Naama Barkai. Revealing modular organization in the yeast transcriptional network. *Nature Genetics*, 31(4):370–377, 2002.

[KAKS99] George Karypis, Rajat Aggarwal, Vipin Kumar, and Shashi Shekhar. Multilevel hypergraph partitioning: Application in vlsi domain. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, pages 69–79, 1999.

[Kau93] Stuart A. Kauffman. *The origins of Order: Self organization and Selection in Evolution*. Oxford University Press, USA, first edition, June 1993.

[KBG+07] Shiraj Khan, Sharba Bandyopadhyay, Auroop R. Ganguly, Sunil Saigal, David J. Erickson, Vladimir Protopopescu, and George Ostrouchov. Relative performance of mutual information estimation methods for quantifying the dependence among short and noisy data. *Physical Review E*, 76(2):1–15, 2007.

[KG07] Chase Krumpelman and Joydeep Ghosh. Matching and visualization of multiple overlapping clusterings of microarray data. In *Proceedings Computational Intelligence and Bioinformatics and Computational Biology (CIBCB '07)*, pages 121–126, 2007.

[KK98] George Karypis and Vipin Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on Scientific Computing*, 20:359–392, December 1998.

[KSG04] Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 69:1–16, 2004.

[Kuh55] Harold W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2:83–97, 1955.

[Lan05] Eric S. Lander. Finding regulatory modules through large-scale gene-expression data analysis. *The new genomics: global views of biology*, 21:536–539, 2005.

[LDJ07] Tao Li, Chris Ding, and Michael I. Jordan. Solving consensus and semi-supervised clustering problems using nonnegative matrix factorization. In *Proceedings IEEE International Conference on Data Mining (ICDM'07)*, pages 577–582. IEEE Computer Society, 2007.

[LIN06] LINGO. Appendix II: Lingo software. In *Process Integration*, volume 7 of *Process Systems Engineering*, pages 389 – 394. Academic Press, 2006.

[LW07] Xiaowen Liu and Lusheng Wang. Computing the Maximum Similarity Bi-clusters of gene expression data. *Bioinformatics*, 23(1):50–56, 2007.

[MAT10] MATLAB. *version 7.10.0 (R2010a)*. The MathWorks Inc., Natick, Massachusetts, 2010.

[MCA+98] G. Michaels, D. Carr, M. Askenazi, S. Fuhrman, X. Wen, and R. Somogyi. Cluster Analysis and Data Visualization of Large Scale Gene Expression Data. In *Proceedings of Pacific Symposium on Biocomputing, Hawaii*, pages 42–53, January 1998.

[MDPM08] Sushmita Mitra, Sujay Datta, Theodore Perkins, and George Michailidis. *Introduction to Machine Learning and Bioinformatics*. Chapman and Hall book/CRC press, 2008.

[Mir96] Boris G. Mirkin. *Mathematical classification and clustering*. Nonconvex optimization and its applications. Kluwer Academic Publishers, Dordrecht, Boston, London, 1996.

[MJ87] Jean V. Moreau and Anil K. Jain. The bootstrap approach to clustering. In *Proceedings of the NATO Advanced Study Institute on Pattern recognition theory and applications*, pages 63–71, London, UK, 1987. Springer-Verlag.

[MK03] T. M. Murali and Simon Kasif. Extracting conserved gene expression motifs from gene expression data. In *Proceedings of Pacific Symposium on Biocomputing, Hawaii*, pages 77–88, January 2003.

[MO97] Richard Maclin and David Opitz. An empirical evaluation of bagging and boosting. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pages 546–551. AAAI Press, 1997.

[PBZ+06] Amela Prelić, Stefan Bleuler, Philip Zimmermann, Anja Wille, Peter Bühlmann, Wilhelm Gruissem, Lars Hennig, Lothar Thiele, and Eckart

Zitzler. A systematic comparison and evaluation of biclustering methods for gene expression data. In *Bioinformatics*, volume 22, pages 1122–1129, 2006.

[PGAR15] Beatriz Pontes, Ral Girldez, and Jess S. Aguilar-Ruiz. Quality measures for gene expression biclusters. *PLOS ONE*, 10(3):1–24, 03 2015.

[PMBG07] I. Priness, O. Maimon, and I. Ben-Gal. Evaluation of gene expression clustering via mutual information distance measure. *BMC Bioinformatics*, 8, 2007.

[R C12] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2012. ISBN 3-900051-07-0.

[RJLS10] Peter H. Raven, George B. Johnson, Jonathan B. Losos, and Susan R. Singer. *Biology*. Tata Mcgraw hill, seventh edition, 2010.

[RWC+02] Andreas Rosenwald, George Wright, Wing C. Chan, Joseph M. Connors, Elias Campo, Richard I. Fisher, Randy D. Gascoyne, Konrad, Erlend B. Smeland, Jena M. Giltnane, Elaine M. Hurt, Hong Zhao, Lauren Averett, Liming Yang, Wyndham H. Wilson, Elaine S. Jaffe, Richard Simon, Richard D. Klausner, John Powell, Patricia L. Duffey, Dan L. Longo, Timothy C. Greiner, Dennis D. Weisenburger, Warren G. Sanger, Bhavana J. Dave, James C. Lynch, Julie Vose, James O. Armitage, Emilio Montserrat, Armando L. Guillermo, Thomas M. Grogan, Thomas P. Miller, Michel Leblanc, German Ott, Stein Kvaloy, Jan Delabie, Harald Holte, Peter Krajci, Trond Stokke, and Louis M. Staudt. The use of molecular profiling to predict survival after chemotherapy

for diffuse large-B-cell lymphoma. *New England Journal of Medicine*, 346(25):1937–1947, 2002.

[Sch01] Robert E. Schapire. The Boosting Approach to Machine Learning: An Overview. In MSRI Workshop on Nonlinear Estimation and Classification, Berkeley, CA, USA, 2001.

[SDSK03] Ralf Steuer, Carsten Daub, Joachim Selbig, and Jürgen Kurths. Measuring Distances Between Variables by Mutual Information. *Innovations in Classification, Data Science and Information Systems*, pages 81–90, March 2003.

[SG02] Alexander Strehl and Joydeep Ghosh. Cluster ensembles – a knowledge reuse framework for combining multiple partitions. *Journal on Machine Learning Research (JMLR)*, 3:583–617, December 2002.

[SKD+02] Ralf Steuer, Jürgen Kurths, Carsten O. Daub, J. Weise, and Joachim Selbig. The Mutual Information: Detecting and evaluating dependencies between variables. *Bioinformatics*, 18(2):S231–S240, 2002.

[SMPX10] Vikas Singh, Lopamudra Mukherjee, Jiming Peng, and Jinhui Xu. Ensemble clustering using semidefinite programming with applications. *Machine Learning*, 79(1-2):177–200, May 2010.

[THC+99] Saeed Tavazoie, Jason D. Hughes, Michael J. Campbell, Raymond J. Cho, and George M. Church. Systematic determination of genetic network architecture. *Nature Genetics*, 22:281–285, 1999.

[TJP03] Alexander P. Topchy, Anil K. Jain, and William F. Punch. Combining multiple weak clusterings. In *International Conference on data Mining*, pages 331–338. IEEE Computer Society, 2003.

[TJP04] Alexander P. Topchy, Anil K. Jain, and William F. Punch. A mixture model for clustering ensembles. In *Proceedings SIAM International Conference on Data Mining*, pages 379–390. SIAM, 2004.

[TK07] Grigorios Tsoumakas and Ioannis Katakis. Multi-label classification: An overview. *International Journal on Data Warehousing and Mining*, 3:1–13, 2007.

[TMbJP04] Alexander Topchy, Behrouz Minaei-bidgoli, Anil K. Jain, and William F. Punch. Adaptive clustering ensembles. In *International Conference on pattern Recognition*, pages 272–275, 2004.

[VBJ+00] Jaak Vilo, Alvis Brazma, Inge Jonassen, Alan Robinson, and Esko Ukkonen. Mining for putative regulatory elements in the yeast genome using gene expression data. In *Proceeding of International Conference of Intelligent Systems for Molecular Biology, La Jolla/San Diego, CA USA*, pages 384–394, August 2000.

[VPCMRS08] Sandro Vega-Pons, Jyrko Correa-Morris, and Jos Ruiz-Shulcloper. Weighted cluster ensemble using a kernel consensus function. In *Progress in Pattern Recognition, Image Analysis and Applications, 13th Iberoamerican Congress on Pattern Recognition, CIARP 2008, Havana, Cuba, September 9-12, 2008. Proceedings*, volume 5197 of *Lecture Notes in Computer Science*, pages 195–202. Springer, 2008.

[VPRS11] Sandro Vega-Pons and Jos Ruiz-Shulcloper. A survey of clustering ensemble algorithms. *International Journal of Pattern Recognition and Artificial Intelligence*, 25(3):337–372, 2011.

[vtVDvdV+02] Laura J. van 't Veer, Hongyue Dai, Marc J. van de Vijver, Yudong D. He, Augustinus A. M. Hart, Mao Mao, Hans L. Peterse, Karin van der

Kooy, Matthew J. Marton, Anke T. Witteveen, George J. Schreiber, Ron M. Kerkhoven, Chris Roberts, Peter S. Linsley, Rene Bernards, and Stephen H. Friend. Gene expression profiling predicts clinical outcome of breast cancer. *Nature*, 415 (6871):530–536, 2002.

[WDH01] Andreas Weingessel, Evgenia Dimitriadou, and Kurt Hornik. Voting-merging: An ensemble method for clustering. In *Proceeding International Conference on Artificial Neural Networks*, pages 217–224. Springer Verlag, 2001.

[WLDJ11] Pu Wang, Kathryn B. Laskey, Carlotta Domeniconi, and Michael Jordan. Nonparametric bayesian co-clustering ensembles. In *Proceedings of the Eleventh SIAM International Conference on Data Mining, SDM 2011, April 28-30, 2011, Mesa, Arizona, USA*, pages 331–342. SIAM / Omnipress, 2011.

[YAL+06] Hye-Sung Yoon, Sun-Young Ahn, Sang-Ho Lee, Sung-Bum Cho, and Ju Han Kim. Heterogeneous clustering ensemble method for combining different cluster results. In *Data Mining for Biomedical applications BioDM*, volume 3916 of *Lecture Notes in Computer Science*, pages 82–92. Springer, 2006.

[ZWD+04] Xiaobo Zhou, Xiaodong Wang, Edward R. Dougherty, Daniel Russ, and Edward Suh. Gene Clustering Based on Clusterwide Mutual Information. *Journal of Computational Biology*, 11,(1):147–161, 2004.

[ZZ06] Min-Ling Zhang and Zhi-Hua Zhou. Multilabel neural networks with applications to functional genomics and text categorization. *IEEE Transactions on Knowledge and Data Engineering.*, 18(10):1338–1351, October 2006.

[ZZ07] Min-Ling Zhang and Zhi-Hua Zhou. Ml-knn: A lazy learning approach to multi-label learning. *Pattern Recognition*, 40(7):2038–2048, July 2007.

[ZZ14] Min-Ling Zhang and Zhi-Hua Zhou. A review on multi-label learning algorithms. *IEEE Transactions on Knowledge and Data Engineering*, 26(8):1819–1837, 2014.