MINING COMPARATIVE GENOMIC HYBRIDIZATION DATA

By

JUN LIU

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

2008

1
© 2008 Jun Liu
To my parents, my sister, and my little niece
ACKNOWLEDGMENTS

I would like to thank my advisers, Dr. Sanjay Ranka and Dr. Tamer Kahveci, for their long-term instructions on my research and this dissertation. I would also like to thank Dr. Michael Baudis for providing the CGH datasets and several discussions on analyzing CGH data. I would like to thank Jaaved Mohammed and James Carter for their help in implementing and testing the clustering algorithms, and Raja Appuswamy for helping develop an initial web-based interface. I would like to thank my committee members (Dr. Christopher M. Jermaine, Dr. Alin Dobra and Dr. Ravindra K. Ahuja) for their guidance and support. Finally, I would like to thank my parents and my friends for their continuous support.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>ACKNOWLEDGMENTS</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF TABLES</td>
<td>7</td>
</tr>
<tr>
<td>LIST OF FIGURES</td>
<td>8</td>
</tr>
<tr>
<td>ABSTRACT</td>
<td>9</td>
</tr>
<tr>
<td>CHAPTER</td>
<td>10</td>
</tr>
<tr>
<td>1 INTRODUCTION</td>
<td>10</td>
</tr>
<tr>
<td>1.1 Comparative Genomic Hybridization Data</td>
<td>10</td>
</tr>
<tr>
<td>1.2 Analysis of CGH Data</td>
<td>14</td>
</tr>
<tr>
<td>1.3 Contribution of This Thesis</td>
<td>15</td>
</tr>
<tr>
<td>2 RELATED WORK</td>
<td>18</td>
</tr>
<tr>
<td>2.1 Structural Analysis of Single Comparative Genomic Hybridization (CGH) Array</td>
<td>19</td>
</tr>
<tr>
<td>2.2 Clustering and Marker Detection of CGH Data</td>
<td>20</td>
</tr>
<tr>
<td>2.3 Classification and Feature Selection of CGH Data</td>
<td>21</td>
</tr>
<tr>
<td>2.4 Inferring Progression Models for CGH Data</td>
<td>25</td>
</tr>
<tr>
<td>2.5 Software for Analyzing CGH Data</td>
<td>27</td>
</tr>
<tr>
<td>3 PAIRWISE DISTANCE-BASED CLUSTERING</td>
<td>28</td>
</tr>
<tr>
<td>3.1 Method</td>
<td>28</td>
</tr>
<tr>
<td>3.1.1 Comparison of Two Samples</td>
<td>29</td>
</tr>
<tr>
<td>3.1.1.1 Raw distance</td>
<td>29</td>
</tr>
<tr>
<td>3.1.1.2 Segment-based similarity</td>
<td>30</td>
</tr>
<tr>
<td>3.1.1.3 Segment-based cosine similarity</td>
<td>31</td>
</tr>
<tr>
<td>3.1.2 Clustering of Samples</td>
<td>33</td>
</tr>
<tr>
<td>3.1.2.1 k-means clustering</td>
<td>33</td>
</tr>
<tr>
<td>3.1.2.2 Complete link bottom-up clustering</td>
<td>34</td>
</tr>
<tr>
<td>3.1.2.3 Top-down clustering</td>
<td>34</td>
</tr>
<tr>
<td>3.1.3 Further Optimization on Clustering</td>
<td>36</td>
</tr>
<tr>
<td>3.1.3.1 Combining k-means with bottom-up or top-down methods</td>
<td>36</td>
</tr>
<tr>
<td>3.1.3.2 Centroid shrinking</td>
<td>37</td>
</tr>
<tr>
<td>3.2 Results</td>
<td>37</td>
</tr>
<tr>
<td>3.2.1 Quality Analysis Measures</td>
<td>39</td>
</tr>
<tr>
<td>3.2.2 Experimental Evaluation</td>
<td>41</td>
</tr>
<tr>
<td>3.3 Conclusion</td>
<td>46</td>
</tr>
</tbody>
</table>
4 IMPROVE CLUSTERING USING MARKERS ........................................ 48
  4.1 Detection of Markers ......................................................... 48
  4.2 Prototype Based Clustering ................................................ 52
  4.3 Pairwise Similarity Based Clustering ..................................... 55
  4.4 Experimental Results ...................................................... 58
    4.4.1 Quality of Clustering ................................................ 58
    4.4.2 Quality of Markers .................................................. 59
    4.4.3 Evaluation ............................................................. 60
  4.5 Conclusion ................................................................. 65
5 CLASSIFICATION AND FEATURE SELECTION ................................. 66
  5.1 Classification with SVM ................................................... 66
  5.2 Maximum Influence Feature Selection for Two Classes .................. 70
  5.3 Maximum Influence Feature Selection for Multiple Classes .......... 74
  5.4 Datasets ......................................................................... 77
  5.5 Experimental Results ...................................................... 81
    5.5.1 Comparison of Linear and Raw Kernel ............................... 81
    5.5.2 Comparison of MIFS and Other Methods ............................ 82
    5.5.3 Consistency of Selected Features ................................... 87
  5.6 Conclusions ..................................................................... 90
6 INFERRING PROGRESSION MODELS ........................................... 93
  6.1 Preliminary ................................................................. 94
    6.1.1 Marker Detection ....................................................... 94
    6.1.2 Tumor Progression Model .............................................. 95
    6.1.3 Tree Fitting Problem .................................................. 97
  6.2 Progression Model for markers ........................................... 98
  6.3 Progression Model for cancers ........................................... 101
  6.4 Experimental Results .................................................... 105
    6.4.1 Results for Marker Models .......................................... 106
    6.4.2 Results for Phylogenetic Models .................................... 108
  6.5 Conclusions ................................................................. 113
7 A WEB SERVER FOR MINING CGH DATA .................................... 115
  7.1 Software Environment .................................................... 115
  7.2 Example: Distance-Based Clustering of Sample Dataset .............. 116
  7.3 Conclusion ................................................................. 119
8 CONCLUSION .............................................................. 120
REFERENCES .............................................................. 122
BIOGRAPHICAL SKETCH .................................................. 128
## LIST OF TABLES

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-1</td>
<td>Detailed specification of Progenetix dataset</td>
<td>38</td>
</tr>
<tr>
<td>3-2</td>
<td>Highest value of external measures for different distance/similarity measure</td>
<td>42</td>
</tr>
<tr>
<td>3-3</td>
<td>Comparison of average quality and running time of top-down methods with global and local refinement</td>
<td>46</td>
</tr>
<tr>
<td>4-1</td>
<td>Coverage measure of three clustering methods applied over three datasets</td>
<td>61</td>
</tr>
<tr>
<td>4-2</td>
<td>The NMI values of three clustering methods applied over three datasets</td>
<td>62</td>
</tr>
<tr>
<td>4-3</td>
<td>Error bar results of three clustering methods over three datasets</td>
<td>63</td>
</tr>
<tr>
<td>5-1</td>
<td>Detailed specifications of benchmark datasets</td>
<td>80</td>
</tr>
<tr>
<td>5-2</td>
<td>Comparison of classification accuracies for three feature selection methods on multi-class datasets</td>
<td>84</td>
</tr>
<tr>
<td>5-3</td>
<td>Comparison of classification accuracy for three feature selection methods on two-class datasets</td>
<td>86</td>
</tr>
<tr>
<td>5-4</td>
<td>Comparison of classification accuracy using different number of features.</td>
<td>87</td>
</tr>
<tr>
<td>5-5</td>
<td>Comparison of PMM scores of three feature selection methods</td>
<td>91</td>
</tr>
<tr>
<td>6-1</td>
<td>Name and number of cases of each cancer in the dataset.</td>
<td>106</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-1</td>
<td>Overview of CGH technique</td>
<td>11</td>
</tr>
<tr>
<td>1-2</td>
<td>Raw and normalized (smoothed) CGH data</td>
<td>13</td>
</tr>
<tr>
<td>1-3</td>
<td>Plot of a CGH dataset</td>
<td>14</td>
</tr>
<tr>
<td>3-1</td>
<td>Example of Raw distance</td>
<td>29</td>
</tr>
<tr>
<td>3-2</td>
<td>Example of Sim measure</td>
<td>31</td>
</tr>
<tr>
<td>3-3</td>
<td>Example of cosineNoGaps measure</td>
<td>32</td>
</tr>
<tr>
<td>3-4</td>
<td>Evaluation of cluster qualities using (A) NMI and (B) $F_1$-measure for different clustering methods</td>
<td>43</td>
</tr>
<tr>
<td>3-5</td>
<td>Cluster qualities of different clustering methods with Sim measure over the entire dataset. The cluster qualities are evaluated using NMI.</td>
<td>45</td>
</tr>
<tr>
<td>4-1</td>
<td>Two CGH samples X and Y with the values of genomic intervals listed in the order of positions. The segments are underlined.</td>
<td>49</td>
</tr>
<tr>
<td>4-2</td>
<td>The CGH data plot of cancer type, Retinoblastoma, NOS (ICD-O 9510/3), with 120 samples and each sample containing 862 genomic intervals</td>
<td>52</td>
</tr>
<tr>
<td>4-3</td>
<td>Comparison of GMS values of markers in clusters from two clustering approaches</td>
<td>64</td>
</tr>
<tr>
<td>5-1</td>
<td>Plot of 120 CGH cases belonging to Retinoblastoma, NOS (ICD-O 9510/3)</td>
<td>71</td>
</tr>
<tr>
<td>5-2</td>
<td>Working example of dataset re-sampler</td>
<td>79</td>
</tr>
<tr>
<td>5-3</td>
<td>Comparison of classification accuracies of SVM with linear and Raw kernels</td>
<td>82</td>
</tr>
<tr>
<td>6-1</td>
<td>Examples of Venn diagram and corresponding graph model</td>
<td>97</td>
</tr>
<tr>
<td>6-2</td>
<td>Phylogenetic trees of 20 cancers based on weighted markers</td>
<td>109</td>
</tr>
<tr>
<td>6-3</td>
<td>Phylogenetic trees of 20 cancers based on unweight markers</td>
<td>110</td>
</tr>
<tr>
<td>6-4</td>
<td>A fraction of the phylogenetic tree of sub-clusters of 20 cancers</td>
<td>112</td>
</tr>
<tr>
<td>7-1</td>
<td>Snapshot of the input interface of distance-based clustering.</td>
<td>117</td>
</tr>
<tr>
<td>7-2</td>
<td>Snapshot of the results of distance-based clustering.</td>
<td>118</td>
</tr>
<tr>
<td>7-3</td>
<td>Snapshot of the plot of clusters.</td>
<td>119</td>
</tr>
</tbody>
</table>
MINING COMPARATIVE GENOMIC HYBRIDIZATION DATA

By

Jun Liu

May 2008

Chair: Sanjay Ranka, Tamer Kahveci
Major: Computer Engineering

Numerical and structural chromosomal imbalances are one of the most prominent features of neoplastic cells. Thousands of (molecular-) cytogenetic studies of human neoplasias have searched for insights into genetic mechanisms of tumor development and the detection of targets for pharmacologic intervention. It is assumed that repetitive chromosomal aberration patterns reflect the supposed cooperation of a multitude of tumor relevant genes in most malignant diseases.

One method for measuring genomic aberrations is Comparative Genomic Hybridization (CGH). CGH is a molecular-cytogenetic analysis method for detecting regions with genomic imbalances (gains or losses of DNA segments). CGH data of an individual tumor can be considered as an ordered list of discrete values, where each value corresponds to a single chromosomal band and denotes one of three aberration statuses (gain, loss and no change). Along with the high dimensionality (around 1000), a key feature of the CGH data is that consecutive values are highly correlated.

In this research, we have developed novel data mining methods to exploit these characteristics. We have developed novel algorithms for feature selection, clustering and classification of CGH data sets consisting of samples from multiple cancer types. We have also developed novel methods and models for understanding the progression of cancer. Experimental results on real CGH datasets show the benefits of our methods as compared to existing methods in the literature.
CHAPTER 1
INTRODUCTION

Numerical and spatial chromosomal imbalances are one of the most prominent and pathogenetically relevant features of neoplastic cells [18]. Over the last decades, thousands of (molecular-) cytogenetic studies of human neoplasia have led to important insights into the genetic mechanisms of tumor development, revealing cancer to be a disease involving dynamic changes in the genome. The foundation has been set in the discovery of aberrations that produce oncogenes with dominant gain of function and tumor suppressor genes with recessive loss of function [28]. Each chromosomal region of a healthy cell has two copies of its DNA in a cell. Deviations from this normal level are called Copy Number Alterations (CNAs). Both classes of cancer genes, tumor suppressor genes and oncogenes, have been identified through DNA copy number alterations in human and animal cancer cells [36]. Detecting these aberrations and interpreting them in the context of broader knowledge facilitates the identification of crucial genes and pathways involved in biological processes and disease. The repetitive chromosomal aberration patterns reflects the supposed cooperation of a multitude of tumor relevant genes [86] in most malignant diseases. A systematic analysis of these patterns for oncogenomic pathway description requires the large-scale compilation of (molecular-) cytogenetic tumor data as well as the development of tools for transforming those data into a format suitable for data mining purposes.

1.1 Comparative Genomic Hybridization Data

Comparative Genomic Hybridization (CGH) is the first efficient approach to scanning the entire genome for variations in DNA copy number [59]. The main advantage of the CGH data is that the DNA copy numbers for the entire genome can be measured in a single experiment [70]. CGH on DNA microarray is a molecular-cytogenetic analysis method for simultaneous detecting of thousands of genes with genomic imbalances (gains or losses of DNA segments) [36]. In this technique, total genomic DNA is isolated from
Figure 1-1. Overview of CGH technique. Genomic DNA from two cell populations is differentially labeled and hybridized to a microarray. The fluorescent ratios on each array spot are calculated and normalized so that the median $\log_2$ ratio is 0. Plotting of the data for chromosome from pter to qter shows that most elements have a ratio near 0. The two elements nearest pter have ratio near -1, indicating a reduction by a factor of two in copy number. This figure is reproduced from the work by Pinkel et al [59].

Test and reference cell populations, differentially labeled and hybridized to metaphase chromosomes or, more recently, DNA microarrays. The relative hybridization intensity of the test and reference signals at a given location is then (ideally) proportional to the relative copy number of those sequences in the test and reference genomes. If the reference genome is normal, then increases and decreases in the intensity ratio directly indicate DNA copy number variation in the genome of test cells (Figure 1-1).

Raw data from CGH experiments is viewed as being continuous [57]. Pre-processing of raw CGH data comprises of all preliminary operations on the data necessary to arrive
at the quantity of interest. For CGH data, the $\log_2$ ratios undergo three pre-processing steps before arriving at the actual copy number.

- The first pre-processing step is normalization. Normalization corrects for experimental artifacts in order to make the $\log_2$ ratio’s from different hybridizations comparable [7, 19, 47, 54].

- The second step of the pre-processing, named segmentation, is motivated by the underlying discrete DNA copy numbers of test and reference samples [21, 30, 87]. Segmentation algorithms divide the genome into non-overlapping segments that are separated by breakpoints. These breakpoints indicate a change in DNA copy number. Array elements that belong to the same segment are assumed, as they are not separated by a breakpoint, to have the same underlying chromosomal copy number. Segmentation methods also estimate the mean $\log_2$ ratio per segments, referred to as states.

- As a final and last pre-processing step, referred to as calling, the DNA copy number of each segment is determined [88, 90]. Normalized CGH signal surpassing predefined thresholds is considered indicative for genomic gains or losses, respectively (Figure 1-2). At present calling algorithms cannot determine whether there are, say, three or four copies present. They can however detect deviations from the normal copy number, and classify each segment as either ‘normal’, ‘loss’ or ‘gain’. Normal status indicates there are two copies of the chromosomal segment present. Loss status indicates at least one copy is lost. Gain status indicates at least one additional copy is present. These labels are referred to calls.

The chromosomal CGH summarizes signals from many short stretches of tumor DNA hybridizing to neighboring regions. The chromosomal CGH results are annotated in a reverse in-situ karyotype format [50] describing imbalanced genomic regions with reference to their chromosomal location. CGH data of an individual patient can be considered as an ordered list of status values, where each value corresponds to a genomic interval.
Raw and normalized (smoothed) CGH data. This example shows 16 measurement points of tumor vs. control fluorescence. Runs of normalized ratio values surpassing the thresholds are considered indicative for gains or losses of genomic material in the corresponding genomic intervals (e.g. chromosomal bands). For our purposes, we use values of 1, -1, and 0 to express gain, loss and no aberration, respectively.

(e.g., a single chromosomal band. The term feature and dimension have also been used in the literature to represent the genomic interval.). Figure 1-3 shows a CGH dataset for Retinoblastoma, NOS (ICD-O 9510/3) with 120 cases (i.e., patients) each having 862 genomic intervals. Chromosomal and array CGH accounts for a significant percentage of the published analyses in cancer cytogenetics [5, 11, 24, 31, 34, 49, 82].

The Progenetix database [3] (http://www.progenetix.net) is one of the major resources for CGH data. It consisted of 15429 cases from 609 publications as of December 2006. Recently, the Progenetix database and the software tools developed for the project have shown its usefulness for the delineation of genomic aberration patterns with clear prognostic relevance in neuroblastomas [82] and for producing tumor type specific imbalance maps [45, 46].

This thesis is concerned with developing tools to help analyze CGH data.
1.2 Analysis of CGH Data

The down-stream analysis of large-scale CGH data helps cancer treatment and diagnosis and reveal the underlying genetic mechanism of cancer. For example, unsupervised clustering methods are often employed to discover previously unknown sub-categories of cancer and to identify genetic biomarkers associated with the differentiation. Classification methods can be used to separate healthy patients from cancer patients and to distinguish patients of different cancer subtypes, based on their cytogenetic profiles. These tasks help successful cancer diagnosis and treatment. Feature selection methods are employed to reveal the specific chromosomal regions that are consistently aberrant for particular cancers and thereby help in focusing investigative efforts on them. However, existing data mining methods can not be directly applied to CGH data because this data is structurally different from ordinary data. The following are important characteristics of CGH data:

1. The datasets are high dimensional. The number of intervals in the Progenetix database [1], a large publicly available database, is 862. Newer CGH datasets may
consists of 10,000 or more intervals. The size of a dataset, when compared with the
dimensionality, is relatively small. In Progenetix database, a clinico-pathological
entity usually contains tens to hundreds of samples.

2. The features in CGH data represent ordered genomic intervals on chromosomes and
their values are categorical.

3. Genomic imbalances in cancers cells correspond to runs of 5 to 15 intervals of
gains or losses for a 862 interval representation. (We will use the term segments to
represent these runs. They correspond to a few megabases to an entire chromosome).
This indicates that neighboring genomic intervals are often correlated.

1.3 Contribution of This Thesis

We have developed novel data mining methods for modeling and analysis of spatial
and temporal characteristics of imbalances in CGH datasets to address these challenges.
The contributions of this thesis are briefly described as follows:

1. We have developed novel pairwise distance-based (we will use the term distance-based
for convenience) clustering methods that effectively exploit the spatial correlations
between consecutive genomic intervals [41]. The goal of our clustering is to identify
sets of tumors exhibiting common underlying genetic aberrations and representing
common molecular causes. Our work is built in two steps. In the first step, we
measure the distance/similarity between all pairs of samples. For this purpose,
we have developed three metrics to compute the similarity/distance between two
CGH samples. In the second step, we build clusters of samples based on pairwise
similarities using variations of well known methods.
Experimental results show that segment-based similarity distance measures are
better indicators of biological proximity between pairs of samples. This measure
when combined with the top-down method produces the best clusters.

2. We have proposed the concept of markers to represent key recurrent point
aberrations that capture the aberration pattern of a set of CGH samples [42].
(See Figure 1-3. The markers are plotted in vertical lines.) We have developed a dynamic programming technique to detect markers in a group of samples. The resulting markers can be seen as the prototype of these samples. Based on the markers, we have developed several clustering strategies. Our experimental results show that the use of markers in the distance-based clustering improves the cluster qualities.

3. We have developed a novel kernel function for using SVM based methods for classifying CGH data. The classification of CGH data aims to build a model for defining a number of classes of tumors and accurately predict the classes of unknown tumors. This measure counts the number of common aberrations between any two samples. We show that this kernel measure is significantly better for SVM-based classification of CGH data than the standard linear kernel.

4. We have developed an SVM-based feature selection method called Maximum Influence Feature Selection (MIFS). It uses an iterative procedure to progressively select features. In each iteration, an SVM based model on selected features is trained. This model is used to select one of the remaining feature that provides the maximum benefit for classification. This process is repeated until the desired number of features is reached. We compared our methods against two state-of-the-art methods that have been used for feature selection of large dimensional biological data. Our results suggests that our method is considerably superior to existing methods.

5. We have developed a novel method to infer the progression of multiple cancer histological types or subtypes based on their aberration patterns. Our experimental results based on a Progenetix dataset demonstrate that cancers with similar histology coding are automatically grouped together using these methods.

We also describe a web based tool for large volume data analysis of CGH datasets [43]. The tool provides various clustering algorithms, distance measures and identifies markers that can be interactively used by researchers. It presents the results are provided in both
textual and graphical format. The tool does not require downloading or installing of software. It can be used through a web browser. A preliminary version of this tool is available at http://cghmine.cise.ufl.edu:8007/CGH/Default.html.
CHAPTER 2
RELATED WORK

One of the early techniques used to identify cytogenetic abnormalities in tumor cells is called Metaphase analysis [10, 55, 81]. The Mitelman Database of Chromosome Aberrations in Cancer (http://cgap.nci.nih.gov/Chromosomes/Mitelman) has become an invaluable resource for cytogenetic aberration data in human malignancies. This database currently contains data from more than 46,000 cancer and leukemia samples analyzed by Metaphase banding. However, utilization of the data collection for data mining purposes so far has been limited by intrinsic problems of the Metaphase banding technique as well as the specific data format.

Over the last decade, the Comparative Genomic Hybridization (CGH) [37] and array- or matrix-CGH techniques [61, 63, 71] have addressed technical problems associated with Metaphase analysis of tumor cells and are now used in many published observations. The molecular cytogenetic techniques of CGH [36] and array- or matrix-CGH [60, 62, 72] have previously been used to describe genomic aberration hot spots in cancer entities [5, 24], for the delineation of disease subsets according to their cytogenetic aberration patterns [34, 48] and for the construction of genomic aberration trees from chromosomal imbalance data [12].

In contrast to Metaphase analysis, CGH techniques are not limited to dividing tumor cells which frequently do not represent the predominant clone in the original tumor. Also, CGH is not hampered by incomplete identification of chromosomal segments, which for Metaphase analysis only recently has been addressed by SKY (Spectral Karyotyping) [84] and MFISH (Multiplex Fluorescent In-Situ Hybridization) [73] techniques. According to our own survey, chromosomal and array CGH now account for a significant percentage of published analyses in cancer cytogenetics.

In this chapter, we briefly review the data mining and related methods that have been used for analyzing CGH data.
2.1 Structural Analysis of Single Comparative Genomic Hybridization (CGH) Array

Different strategies for structural analysis of CGH data have been applied previously. Most of these analysis were aimed at the description of pseudo-temporal relationships of cytogenetic events [12, 31] or at the correlation of disease subsets with clinical parameters [48, 82]. Other CGH related data analysis have been aimed at the spatial coherence of genomic segments with different copy number levels.

Picard et al. pointed out that raw CGH signal exhibits a spatial coherence between neighboring intervals [57]. This spatial coherence has to be handled. They used a segmentation methods based on a random Gaussian model. The parameters of this model are determined by abrupt changes at unknown intervals. They developed a dynamic programming algorithm to partition the data into a finite number of segments. The intervals in each segment approximately share the same copy number on average. Further, he proposed a segmentation-clustering approach combined with a Gaussian mixture model to predict the biological status of the detected segments [58].

Fridlyand et al. used an unsupervised Hidden Markov models approach which consists of two parts [21]. In the first part, they partition the genomic intervals into the states which represent the underlying copy number of the groups of intervals. In the second part, they determine the copy number level of each individual chromosome according to whether any copy number transitions or whole chromosome gains or losses are contained in the chromosome. They derived the appropriate values of parameters in the algorithm using unpublished primary tumor data.

Pei et al. segmented each chromosome arm (or chromosome) using a hierarchical clustering tree. The clusters are identified by suppressing the False Discovery Rate (FDR) below a certain level. In addition, their algorithm provided a consensus summary across a set of intervals, as well as an estimate of the corresponding FDR. They illustrated their
method with applications on a lung cancer microarray CGH data set as well as an array
CGH data set of aneuploid cell strains [88].

Willenbrock et al. made a comparison study on three popular and publicly available
methods for the segmentation analysis of array CGH data [90]. They demonstrated that
segmented CGH data yields better results in the downstream analysis such as hypothesis
testing and classification than the raw CGH data. They also proposed a novel procedure
for calling copy number levels by merging segments across the genome.

All the above works focus on the discretization of raw CGH data. However, they do
not address the subsequent analysis, such as clustering or classification, for a large dataset
consisting of discretised (smoothed) CGH data samples.

2.2 Clustering and Marker Detection of CGH Data

Unsupervised clustering methods are often employed to discover previously unknown
sub-categories of cancer and to identify genetic biomarkers associated with the differentiation.
Towards this end, many attempts have been done on the studies of gene microarray [32].
In the following, we briefly describe some of the earlier work on clustering of CGH data.

Mattfeldt et al. applied an existing tool for the clustering of CGH data [48]. The
tool, named Genecluster, formed clusters on the basis of an unsupervised learning rule
using an artificial neural network. It was originally proposed for the clustering of gene
expression data. Mattfeldt et al applied Genecluster over a group of tens of cases from
pT2N0 prostate cancer. Based on the fact that clinically similar cases are placed into the
same clusters, they demonstrated that good clustering were found.

Beattie et al. developed a new data mining technique to discover significant
sub-clusters and marker genes in a completely unsupervised manner [4]. They used a
digital paradigm to discretise the gene microarray and transfered the data into binary
state patterns. A clustering based on Hamming distance was applied to create clusters
and identify bio-markers. Although their work is not directly based on CGH data, they
demonstrated that their method can be adapted to other categorical datasets.
Rouveirol et al. proposed two algorithms for computing minimal and minimal constrained regions of recurrent gain and loss aberrations from discretised CGH data [67]. Their algorithms can handle additional constraints describing relevant regions of copy number change. They validate their algorithms on two public array-CGH datasets.

Thus, the existing literature has not addressed clustering algorithms that exploit the important spatial and temporal characteristics of CGH data. Further, existing clustering works usually focus on small homogeneous datasets with several tens of cases. The existing marker discovery methods usually simply identify the markers, but do not explore the usage of these markers in clustering analysis, as what we propose in this thesis.

2.3 Classification and Feature Selection of CGH Data

Classification aims to build an efficient and effective model for predicting class labels of unknown data. The model is built on the training data, which consists of data points chosen from input data space and their class labels. Classification techniques has been widely used in microarray analysis to predict sample phenotypes based on gene expression patterns.

Support Vector Machine (SVM) is a state-of-art technique for classification [83]. Mukherjee et al. used an SVM classifier for cancer classification based upon gene expression data from DNA microarrays. They argued that DNA microarray problems are very high dimensional and have very few training data. This type of situation is particularly well suited for an SVM approach. Their approach achieved better performance than reported results [53].

Li et al. performed a comparative study of multiclass classification methods for tissue classification based on gene expression data [40]. They conducted comprehensive experiments using various classification methods including SVM [83] with different multiclass decomposition techniques, Naive Bayes, K-nearest neighbor and decision tree [79]. They found that SVM is the best classifier for classification of gene expression data.
To our knowledge, most existing SVM-based approaches focus on gene expression data. They usually use linear kernel due to the assumption of Golub et. al. about the additive linearity of the genes in classification [23]. For example, based on experimental results, Mukherjee et al. demonstrated that linear SVMs did as well as nonlinear SVMs using polynomial kernels. So far, there is very limited study on developing kernel functions for the classification of CGH data.

*Feature selection* is a related task that selects a small subset of discriminative features. The problem of feature selection was first proposed in machine learning. A good review can be found at [26]. Recently, feature selection methods have been widely studied in gene selection of microarray data. These methods can be decomposed into two broad classes:

1. Filter Methods: These methods select features based on discriminating criteria that are relatively independent of classification process. Several methods use simple correlation coefficients similar to Fisher’s discriminant criterion. For example, given class 1 and class -1 denoting two classes, Golub et al. used a criterion as follows [23]:

\[
P(j) = \frac{|u_1(j) - u_{-1}(j)|}{\sigma_1(j) + \sigma_{-1}(j)}
\]

where \(j\) is the gene index, \(u_1\) is the mean of class 1 for gene \(j\), \(u_{-1}\) is the mean of class -1 for gene \(j\), \(\sigma_1\) is the standard deviation of class 1 for gene \(j\), and \(\sigma_{-1}\) is the standard deviation of class -1 for gene \(j\). The genes are then ranked in descending order according to \(P(j)\) and the top values correspond to "informative" genes. Other methods adopt mutual information or statistical tests (\(t\)-test, \(F\)-test). For example, Model et al. ranked the features using a two sample \(t\)-test [51]. They assumed that the value of a feature within a class follows a normal distribution. A two sample \(t\)-test was adopted to rank the features according to the significance of the difference between the class means. In principle, their approach was similar to Fisher’s criterion because, in both methods, a large mean difference and a small
within class variance are proportional to the discriminative power of a feature. Their experimental results demonstrated that the $t$-test approach worked better than the standard Principle Component Analysis (PCA) method. Ding et al. considered the nature of feature selection for classification of multi-class data [16]. They used the $F$-statistic test which is a generalization of $t$-statistic for two class. Given a gene expression across $n$ tissue samples $g = (g_1, \cdots, g_n)$ from $K$ classes, the $F$-statistic is defined as

$$F = \left[ \sum_k n_k (\bar{g}_k - \bar{g})^2 / (K - 1) \right] / \sigma^2$$

where $\bar{g}$ is the average expression across all samples, $\bar{g}_k$ is the average within class $C_k$, and $\sigma^2$ is the pooled variance:

$$\sigma^2 = \left( \sum_k (n_k - 1) \sigma_k^2 \right) / (n - K)$$

where $n_k$ and $\sigma_k$ are the size and variance of gene expression within class $C_k$. They picked genes with large $F$-values.

Earlier filter based methods evaluated features in isolation and did not consider correlation between features. Recently, methods have been proposed to select features with minimum redundancy [14, 15, 91]. For example, Yu et al. introduced the importance of removing redundant genes in sample classification and pointed out the necessity of studying feature redundancy [91]. They proposed a filter method with feature redundancy taken into account. They combined sequential forward selection with backward elimination so that, in each step, the number of feature pairs for redundancy analysis is reduced. Their method is free of any threshold in determining feature relevance or redundancy. Their experimental results on microarray data demonstrated the efficiency and effectiveness of their method in selecting discriminative genes that improve classification accuracy.
The methods proposed by Ding et al. uses a minimum redundancy - maximum relevance (MRMR) feature selection framework [14, 15]. They supplement the maximum relevance criteria along with minimum redundancy criteria to choose additional features that are maximally dissimilar to already identified ones. By doing this, MRMR expands the representative power of the feature set and improves their generalization properties.

2. Wrapper Methods: Wrapper methods utilize a classifier as a black box to score the subsets of features based on their predictive power. Wrapper methods based on SVM have been widely studied in machine learning community [26, 64, 89]. SVM-RFE (Support Vector Machine Recursive Feature Elimination) [27], a state-of-the-art wrapper method applied to cancer research is called, uses a backward feature elimination scheme to recursively remove insignificant features from subsets of features. In each recursive step, a linear SVM is trained on the feature set. For each feature, a ranking coefficient is computed based on the reduction in the objective function if this feature is removed. The bottom ranked feature is then eliminated from the feature set. The above process is repeated until the feature set is empty. The features are sorted based on their sequence of elimination.

A number of variants also use the same backward feature elimination scheme and linear kernel. Zhang et al. proposed a method aimed for classifying two-class data [92]. It used a recursive support vector machine (R-SVM) algorithm to select important features for the classification of noisy high-throughput proteomics and microarray data. The experimental results showed that, compared to SVM-RFE, their method is more robust to outliers in the data and capable of selecting the most informative features.

Duan et al. proposed a new feature selection method that used a backward elimination procedure similar to that implemented in SVM-RFE [17]. Unlike SVM-RFE, at each step, the proposed approach trained multiple linear SVMs on
subsamples of the original training data. It then computed the feature ranking score from a statistical analysis of weight vectors of these SVMs. The experimental results showed that their method selects better gene subsets than SVM-RFE and improves the classification accuracy.

For feature selection of multiclass data, Ramaswamy et al. used an one-versus-all strategy to convert the multiclass problem into a series of two-class problems. They applied SVM-RFE to each two-class problem separately and generated a consensus sorting of all features [65].

Fu et al. also proposed a method based on the one-versus-all strategy [90]. For each two-class problem, they wrapped the feature selection into a 10-fold cross validation (CV) and selected features using SVM-RFE in each fold. They also developed a probabilistic model to select significant features from the 10-fold results. They took the union of features selected from each two-class SVM as the final set of features.

Filter methods are generally less computationally intensive than wrapper methods. However, they tend to miss complementary features that individually do not separate the data well. A recent comparison of feature selection methods for the classification of multiclass microarray data shows that wrapper methods such as SVM-RFE have better classification accuracy for large number of features, but derives lower accuracy than filter methods when the number of selected feature is small [9].

2.4 Inferring Progression Models for CGH Data

Models of tumor progression can be used to explain known clinical and molecular evidence of cancer. An earlier effort was made by Vogelstein et al [85]. They inferred a chain model of four genetic events, three of which are CNAs, for the progression of colorectal cancer. These events in the model are irreversible. That is, once an event occurs it is never undone in the future. The presence of all four events appears to be an indicator of colorectal cancer.
Desper et al. proposed a branching tree model that are more general than a path model by assuming that the recurrent CNAs are a set of genetic events that take place in some order [12]. They derived a tree model inference algorithm by utilizing the idea of maximum-weight branching in a graph. They applied the algorithm over a CGH data set for renal cancer and showed that the correct tree for renal cancer was inferred. Later, they extended their work to distance-based trees, in which events are leaves of the tree, in the style of models common in phylogenetics [13]. They proposed a novel approach to reconstruct the distance-based trees using tree-fitting algorithms developed by researchers in phylogenetics. They applied their approach over the CGH data set for renal cancer. The results showed that the distance-based models well complemented the branching tree models.

Bilke et al. proposed a graph model based on the shared status of recurrent CNAs among different stages of cancer [6]. They first identified a set of recurrent alterations and computed their shared status using statistical tests. They then constructed a Venn diagram based on these recurrent alterations. They manually converted the Venn diagram into a graph model. They found that the pattern of recurrent CNAs in neuroblastoma cancer is strongly stage dependent.

Pennington et al. developed a mutation model for individual tumor and construct an evolutionary tree for each tumor [56]. They identified the consensus tree model based on the copy number alterations shared by a substantial fraction of the population. They proved that their results are consistent with prior knowledge about the role of the genes examined in cancer progression.

All above works infer tumor progression model based on the genetic events such as recurrent CNAs. Their models describe the evolutionary relationship between these events and consequently expose the progression and development of tumors. However, these works treat every individual recurrent alterations as independent genetic events. This makes their models become very complex when applied to data sets with samples
from multiple cancers, given that each cancer type contains a set of substantially different recurrent alterations.

2.5 Software for Analyzing CGH Data

Chromosomal and array CGH recently account for a significant number of published studies in cancer cytogenetics [5, 12, 24, 31, 34, 48, 82]. Acquisition of thousands of copy number information brings forth challenges to the analysis of CGH data. Researchers have explored data mining methods for this purpose. Many of their methods focus on the structure analysis of CGH data, such as the spatial coherence of genomic segments with different copy number levels [25, 57, 58, 74]. Associated with these works, a lot of tools (both web application and stand-alone software package) are available for the analysis of CGH data, such as CGH-miner (http://www-stat.stanford.edu/~wp57/CGH-Miner/), CGH-explorer (http://www.ifi.uio.no/forskning/grupper/bioinf/Papers/CGH/), ArrayCyGHt (http://genomics.catholic.ac.kr/arrayCGH/) and CGH-plotter (http://sigwww.cs.tut.fi/TICSP/CGH-Plotter/). However, very limited efforts have been conducted for mining heterogeneous CGH datasets for more than a few hundred samples. This is the focus of our proposed work and software.
CHAPTER 3
PAIRWISE DISTANCE-BASED CLUSTERING

The goal of clustering is to develop a systematic way of placing patients with similar CGH imbalance profiles into the same cluster. Our expectation is that patients with the same cancer types will generally belong to the same cluster as their underlying CGH profiles will be similar. In this chapter, we focus on distance-based clustering. We develop three pairwise distance/similarity measures, namely Raw, Cosine and Sim. Raw measure compares the aberrations in each genomic interval separately. The other two measures take the correlations between consecutive genomic intervals into account. Cosine maps pairs of CGH samples into vectors in a high-dimensional space and measures the angle between them. Sim measures the number of independent common aberrations. We test our distance/similarity measures on three well known clustering algorithms, bottom-up, top-down and $k$-means with and without centroid shrinking. Our results show that Sim, when combined with top-down algorithm, consistently performs better than the remaining measures.

3.1 Method

Genomic aberration data from CGH experiments is usually communicated in a reverse \textit{in-situ} karyotype annotation format \cite{50}. We use this strategy and represent gain, loss, and no change with +1, -1, and 0 respectively throughout the proposal.

We propose to use three different distance-based clustering methods for CGH data and survey their performance. The key problem, however, is to compute the proximity of two CGH samples. In Section 3.1.1, we discuss the three measures we developed for such pairwise comparison. We briefly explain the three clustering algorithms we used to cluster a population of samples in section 3.1.2. Two techniques that further optimize the cluster qualities are discussed in section 3.1.3.
Figure 3-1. Example of Raw distance. \( X \) and \( Y \) are two CGH samples. The value of each genomic interval shows the status (i.e. gain loss or no change) of that interval. The distance between \( X \) and \( Y \) is \( \sum_{j=1}^{m} \text{diff}(x_j, y_j) = 9 \).

### 3.1.1 Comparison of Two Samples

Let \( X = x_1, x_2, \ldots, x_m \) and \( Y = y_1, y_2, \ldots, y_m \) be two CGH samples. Here, \( x_i \) and \( y_i \) denote the value or status of the \( i \)th genomic interval of \( X \) and \( Y \), respectively. The proximity between \( X \) and \( Y \) can be computed in terms of distance or similarity. In this section we develop three such measures of distance/similarity.

#### 3.1.1.1 Raw distance

Our first measure assumes that the genomic intervals are independent of each other. This assumption is often made in existing literature to simplify the problem of computing distances [57]. If both samples have gain (or loss) at the same genomic interval then we consider them similar at that position. Otherwise, that genomic interval contributes to the distance between them. Also, we assume that all genomic intervals have the same importance. Thus, each genomic interval contributes the same amount to the total distance. Formally, the distance is computed as \( \sum_{j=1}^{m} \text{diff}(x_j, y_j) \). Here \( \text{diff}(x_j, y_j) = 1 \) if \( x_j \neq y_j \) or \( x_j = 0 \). Otherwise \( \text{diff}(x_j, y_j) = 0 \). The similarity is obtained by subtracting the distance from \( m \), the number of genomic intervals of the CGH samples. An example is shown in Figure 3-1.

This distance function is similar to Hamming distance in principle because it compares the genomic intervals of both samples one by one. We call this distance Raw since it is computed on raw CGH data. Raw distance between two samples is small only
if the samples have gains or losses in the same positions. Raw distance ranges between $[0, m]$.

### 3.1.1.2 Segment-based similarity

This method takes the fact that consecutive genomic intervals are usually correlated. A contiguous block of gains (or losses) can be caused by a point-like aberration at a single genomic interval. We use the term *segment* to represent a contiguous block of aberrations of the same type. For example, in Figure 4-1, sample $X$ contains four segments. The first and third segments are gain type while the second and fourth segment are loss type. We call two segments from two samples *overlapping* if they have at least one common genomic interval of the same type. For example, the first segment of $X$ is overlapping with the first segment of $Y$ in Figure 4-1. Also the third segment of $X$ is overlapping with the second segment of $Y$. Next, we develop a segment-based similarity measure called $Sim$.

Given two CGH samples $X$ and $Y$, $Sim$ constructs maximal segments by combining as many contiguous aberrations of the same type as possible. Formally, the genomic intervals $x_i, x_{i+1}, \ldots, x_j$, for $1 \leq i \leq j \leq m$, define a segment if genomic intervals $x_i$ through $x_j$ are in the same chromosome, the values from $x_i$ to $x_j$ are all gains or all losses, and $x_{i-1}$ and $x_{j+1}$ are different than $x_i$. Thus, each sample translates into a sequence of segments. After this transformation, $Sim$ assumes that the segments are independent of each other and gives the same importance to all the segments regardless of the number of genomic intervals in them. $Sim$ computes the similarity between two CGH samples as the number of overlapping segment pairs. This is justified because each overlap may indicate a common point-like aberration in both samples which then led to the corresponding overlapping segments. An example is shown in the Figure 4-1. There are two important observations that follows from the definition of $Sim$. First, unlike the Raw distance measure, $Sim$ considers an overlap of arbitrary number of genomic intervals as a single match. Second, although two samples have different values for the same genomic interval, $Sim$ does not consider this as a mismatch if it is an extension of an overlap. For example,
Figure 3-2. Example of Sim measure. X and Y are two CGH samples with the values of genomic intervals shown in the order of positions. The segments are underlined. The overlapping segments are shown with arrows. Since there are two overlapping segments; one from position 3 to 4 and the other at position 10, the similarity between X and Y is 2.

in Figure 4-1, the fifth genomic intervals of sample X and Y have different values, but we still consider this position a match because it could be an extension of an overlap.

3.1.1.3 Segment-based cosine similarity

Segment-based similarity grows linearly with the number of common segments. However, the aberration patterns of some cancer types can be less complex than the others. The samples that belong to these cancer types share fewer common segments leading to small values of Sim even though the samples are almost identical. Cosine similarity of two vectors normalizes the similarity by measuring the cosine of the angle between them. This measure is the most commonly used method to compute the similarity between two directional data in vector-space model [68]. In this section, we extend the cosine similarity to measure the proximity of two CGH samples.

Let X and Y be two CGH samples. We first map X and Y to two vectors \( \hat{X} \) and \( \hat{Y} \in \mathbb{R}^g \), where \( g \) is the number of dimensions of the vectors. Usually, \( g \ll m \), where \( m \) is the number of genomic intervals of CGH samples. The mapping process is also based on segments and works as follows. First, we translate each sample into a sequence of segments. Let us define segment sequence \( G \), \( H \) that corresponds to the sample \( X \), \( Y \) respectively. Without loss of generality, we can assume that for all the genomic intervals in \( Y \), if they belong to any segment in \( H \), the genomic intervals in \( X \) at the same positions are also covered by the segments in \( G \). Here, we say that a segment covers a consecutive block of genomic intervals only if for each genomic interval, either it belongs to this
Figure 3-3. Example of cosineNoGaps measure. This figure shows the cosineNoGaps similarity between two CGH samples. X and Y are two CGH samples with the values of genomic intervals shown in the order of positions. The segments are underlined. First, X and Y are mapped to two vectors $\hat{X}$ and $\hat{Y}$ respectively. Second, the similarity between X and Y is computed as $C(\hat{X}, \hat{Y}) = 0.7071$. segment or it is of no-change status and the aberration of this segment can be extended to this genomic interval. Next, we scan the segment sequence $G$ in the ascending order of the genomic intervals. For each segment $g_i \in G$, if there exist an overlapping segment $h_j \in H$, we add a new dimension to both vectors $\hat{X}$ and $\hat{Y}$. We then assign value 1 to this dimension of $\hat{X}$ and $\hat{Y}$, indicating that the value of this dimension are exactly the same in the two vectors. If no overlapping segment $h_j \in H$ exists, we add a new dimension to both vectors with value 1 assigned to vector $\hat{X}$ and value 0 assigned to vector $\hat{Y}$, which indicates that the values of the new dimension in two vectors are orthogonal. An example of the segmenting and mapping step for this measure is shown in Figure 3-3. After the two CGH samples X and Y have been mapped to two vectors, the cosine similarity between X and Y is computed as

$$C(\hat{X}, \hat{Y}) = \frac{\sum_{i=1}^{m} \hat{x}_i \cdot \hat{y}_i}{\sqrt{(\sum_{i=1}^{m} \hat{x}_i \cdot \hat{x}_i)(\sum_{i=1}^{m} \hat{y}_i \cdot \hat{y}_i)}}.$$  

The majority of genomic intervals in CGH data have zero values (i.e. no aberration). We call a consecutive block of these genomic intervals gaps. We ignore the impact of gaps in the above cosine similarity measure. However, considering the overlapping gaps between two samples might contribute greatly to the similarity between them. We develop another variant of cosine similarity which takes the overlapping gaps into consideration. The new similarity measure changes the mapping step that translates the CGH data into vectors.
First, it extends the definition of segments to be a consecutive block of genomic intervals that share the same status, i.e. gain, loss or no change. That means, gaps are also included in the segments in this way. Then it translates the CGH data into a sequence of segments with some of the segments representing gaps. Next, a scan is performed on the segment sequence $G$. For each gap in $G$, if there exists an overlapping gap in $H$, a new dimension will be added to both vectors and a pair of value 1 will be assigned to them. Other mapping steps of gain or loss segments and computation of cosine similarity remains unchanged. Compared to the previous cosine similarity measure, this measure offers a larger similarity between two CGH samples due to the impact of overlapping gaps. Thus, we use the term $CosineGaps$ to represent it, whereas the term $CosineNoGaps$ is used to represent the previous definition. Both of these measures produce a value within a range of $[0, 1]$ indicating the similarity between two samples.

3.1.2 Clustering of Samples

With one of the aforementioned distance/similarity measures between two CGH samples, we can easily apply a distance-based clustering algorithm to group similar CGH samples together. At a high-level, the problem of clustering is defined as follows. Given a set $S$ of $n$ samples $s_1, s_2, \cdots, s_n$, we would like to partition $S$ into $k$ subsets $C_1, C_2, \cdots, C_k$, such that the samples assigned to each subset are more similar to each other than the samples assigned to different subsets. Here, we assume that two samples are similar if they correspond to the same cancer type.

As we mentioned earlier, our focus in this thesis is to evaluate the suitability of various distance/similarity measures together with clustering algorithms in the context of the CGH data clustering problem. In this section, we briefly introduce the three distance-based clustering algorithms we used in our experiments.

3.1.2.1 $k$-means clustering

$K$-means [44] is one of the simplest unsupervised learning algorithms that solve the well-known clustering problem. Its key step is to compute the distance/similarity between
a sample data and the cluster centroid, which is not necessary a real sample. Since CGH samples are represented as an array of status values, it is not trivial to compute an accurate centroid for a set of CGH samples. Here, we develop a variant of the $k$-means algorithm which is more suitable for our distance/similarity measures. Compared to the standard $k$-means, our algorithm omits the step of computing the cluster centroids, but reassigns a sample according to its average distance to all the samples in a cluster rather than the distance to the centroid of that cluster. These changes let our algorithm work for any distance/similarity measure described in Section 3.1.1.

We first partition the $n$ samples into $k$ clusters by randomly assigning each sample to one of the $k$ clusters. This random partition forms the initial cluster seeds for our $k$-means algorithm. Then we scan the $n$ samples one by one. For the $i$th sample, compute its average distance to all the samples in cluster $j$, for $1 < j < k$, and then move it to the cluster with the minimum average distance if that cluster is different from the one it already belongs to. This scanning process is repeated until there is no movement of samples during a scan or until a maximum number of iterations is reached.

3.1.2.2 Complete link bottom-up clustering

Complete link [38] clustering defines the distance between two clusters as the largest distance between a sample from the first cluster and a sample from the second cluster. The bottom-up clustering works by designating each sample as its own cluster initially. Next, each cluster is compared to each other cluster, and the closest clusters are merged. This process will continue until $k$ clusters remain.

3.1.2.3 Top-down clustering

This algorithm [75] starts by assigning all samples into one cluster. It then bisects this cluster recursively until $k$ clusters are produced, where $k$ is a user defined parameter. The bisection is performed in two phases. In the first phase, two samples are randomly selected as the seeds of two clusters. Then, for each remaining sample, its similarity to these two seeds is computed and it is assigned to the cluster whose seed has a higher
similarity to that sample. In the second phase, the clusters are refined. A refinement consists of a number of iterations. During each iteration, samples are visited one by one. Each sample \( s_i \) is then moved to all of the clusters one by one, and a criterion function is computed for each positioning of \( s_i \). The criterion function evaluates the quality of the clusters. We use the term internal measure to represent this criterion function. The formal definition of internal measure is addressed in Section 3.2.1. The sample \( s_i \) is kept in the cluster that maximizes the internal measure. This refinement process ends as soon as there is no movement of samples during an iteration or after a predefined maximum number of iterations have been performed. In our experiments, the number of iterations were typically less than 20. After the refinement is finished, the cluster with the largest number of samples is bisected similarly. Once \( k \) clusters are created, the top-down algorithm ends.

In each iteration of the refinement, \( O(n) \) time is needed to compute the change of the internal measure for each sample. This is because, the similarity between that sample and every other sample in each cluster needs to be accumulated. The time complexity of each iteration is \( O(n^2) \) as there are totally \( n \) samples. Since the total number of iterations is limited by a small constant, the complexity of refinement is \( O(n^2) \). The refinement is performed every time a new cluster is created. In the above described process the number of clusters increases by one in every stage until \( k \) clusters are created. Therefore, the overall time complexity of top-down clustering is \( O(n^2k) \).

To reduce this time complexity, we modify the top-down clustering algorithm. Essentially, the refinement process is limited to the cluster being decomposed into smaller clusters. There are two differences between the modified and the original top-down clustering. First, only the samples in the decomposed cluster are considered for refinement. Second, a sample is relocated only to the two newly created clusters rather than all the clusters. In the best case, the clusters are decomposed in a balanced fashion. The overall time complexity in this case is \( O(n^2 + 2(\frac{n}{2})^2 + \cdots + 2^{\log_2 k}(\frac{n}{2^{\log_2 k}})^2) \approx O(2n^2) \). In the worst case, a cluster with \( n \) samples could be decomposed into two clusters with \( n - 1 \) samples.
in one cluster and 1 sample in the other. If this case happens to all the bisections, the worst case time complexity could be $O(kn^2)$. Thus, with this enhanced refinement process, the average time complexity of top-down clustering is between $O(n^2)$ and $O(kn^2)$. We generally expect the time complexity to be close to $O(n^2)$, which results in a factor of $k$ improvement in time. We call this faster refinement process in the top-down clustering Local Refinement and the previous refinement process Global Refinement. It is worth noting that local refinement may produce lower quality clusters. Our experimental results described in Section 5.5 show that this deterioration is small.

### 3.1.3 Further Optimization on Clustering

In this proposal, we use two approaches to further optimize the clusters obtained by the bottom-up or top-down algorithms. We also compare the optimized results with the non-optimized results of these algorithms in Section 5.5.

#### 3.1.3.1 Combining $k$-means with bottom-up or top-down methods

Similar to the standard $k$-means, the $k$-means algorithm used in this thesis does not necessarily find the optimal clusters because it is significantly sensitive to the initial cluster seeds. This observation motivates our further optimization by choosing the results of bottom-up or top-down algorithms as the initial seeds for $k$-means. That is, after the bottom-up or top-down clustering, a $k$-means method will be invoked and the clusters produced by the bottom-up or top-down clustering will serve as the initial cluster seeds of $k$-means. The rest of the $k$-means clustering remains the same. This additional $k$-means step further refines the clusters by using the more CGH specific distance measures proposed in this thesis. We use term top-down+kmeans to represent the optimization approach that combines the top-down algorithm with the $k$-means algorithm. Similarly, we use term bottom-up+kmeans to represent the combination of the bottom-up algorithm and the $k$-means algorithm.
3.1.3.2 Centroid shrinking

The idea of centroid shrinking was first introduced by Robert et. al in [80] to improve the nearest-centroid classification. The centroids of a training set are defined as the average expression of each gene. This idea shrinks the centroids of each class towards the overall centroid after normalizing by the intra-class standard deviation for each genomic interval. This normalization has the effect of assigning more weight to the genomic interval whose status is stable within samples of the same class, and thus reduces the number of features contributing to the nearest centroid calculation. We apply this idea to achieve further optimization of clustering. The centroids of initial clusters found by the different clustering methods, i.e. bottom-up, top-down, $k$-means, bottom-up+$k$-means and top-down+$k$-means, are shrunk towards the overall centroid. Then, a standard $k$-means using Euclidean distance is invoked to re-cluster the samples using the shrunken centroids as its initial centroids.

3.2 Results

Experimental setup: We evaluated the quality and the performance of all the distance/similarity measures and the clustering methods discussed in this thesis. For evaluation of quality we used different measures belonging to two categories, external and internal measures. We discuss these measures in detail in Section 3.2.1.

We implemented all four distance measures (Raw, Sim, CosineGaps, CosineNoGaps) and five clustering algorithms ($k$-means, top-down, bottom-up, top-down + $k$-means, bottom-up + $k$-means). Thus, we had 20 different combinations. We have also implemented the centroid shrinking strategy and applied on each combination. Note that we use local refinement strategy (see Section 3.1.2.3) for top-down in our experiments unless otherwise stated.

We use a dataset consisting of 5,020 CGH samples (i.e., cytogenetic imbalance profiles of tumor samples) taken from the Progenetix database [3]. These samples belonged to 19 different histopathological cancer types with more than 100 cases and had been coded
Table 3-1. Detailed specification of Progenetix dataset. Term #cases denote the number of cases.

<table>
<thead>
<tr>
<th>ICD-O-3 code</th>
<th>#cases</th>
<th>Code translation</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000/0</td>
<td>110</td>
<td>non-neoplastic or benign</td>
</tr>
<tr>
<td>8890/3</td>
<td>118</td>
<td>Leiomyosarcoma, NOS</td>
</tr>
<tr>
<td>9510/3</td>
<td>120</td>
<td>Retinoblastoma, NOS</td>
</tr>
<tr>
<td>9391/3</td>
<td>126</td>
<td>Ependymoma, NOS</td>
</tr>
<tr>
<td>9835/3</td>
<td>128</td>
<td>Acute lymphoblastic leukemia, NOS</td>
</tr>
<tr>
<td>9180/3</td>
<td>133</td>
<td>Osteosarcoma, NOS</td>
</tr>
<tr>
<td>9836/3</td>
<td>141</td>
<td>Precursor B-cell lymphoblastic leukemia</td>
</tr>
<tr>
<td>8144/3</td>
<td>144</td>
<td>Adenocarcinoma, intestinal type</td>
</tr>
<tr>
<td>9673/3</td>
<td>171</td>
<td>Mantle cell lymphoma</td>
</tr>
<tr>
<td>8010/3</td>
<td>180</td>
<td>Carcinoma, NOS</td>
</tr>
<tr>
<td>9732/3</td>
<td>190</td>
<td>Multiple myeloma</td>
</tr>
<tr>
<td>8140/0</td>
<td>209</td>
<td>Adenoma, NOS</td>
</tr>
<tr>
<td>9500/3</td>
<td>271</td>
<td>Neuroblastoma, NOS</td>
</tr>
<tr>
<td>8170/3</td>
<td>286</td>
<td>Hepatocellular carcinoma, NOS</td>
</tr>
<tr>
<td>8523/3</td>
<td>310</td>
<td>Infiltrating duct mixed with other types of carcinoma</td>
</tr>
<tr>
<td>9680/3</td>
<td>323</td>
<td>Diffuse large B-cell lymphoma, NOS</td>
</tr>
<tr>
<td>9823/3</td>
<td>346</td>
<td>B-cell chronic lymphocytic leukemia/small lymphocytic lymphoma</td>
</tr>
<tr>
<td>8070/3</td>
<td>657</td>
<td>Squamous cell carcinoma, NOS</td>
</tr>
<tr>
<td>8140/3</td>
<td>1057</td>
<td>Adenocarcinoma, NOS</td>
</tr>
</tbody>
</table>

according to the ICD-O-3 system [22]. The subset with the smallest number of samples, consists of 110 non-neoplastic cases, while the one with largest number of samples, Adenocarcinoma, NOS (ICD-O 8140/3), contains 1057 cases. The details of this dataset is listed in Table 3-1. Each sample in the dataset consists of 862 ordered genomic intervals extracted from 24 chromosomes. Each interval is associated with one of the three values -1, 1 or 0, indicating loss, gain or no change status of that interval. In principle, our CGH dataset can be mapped to a integer matrix of size $5,020 \times 862$. We also use a small dataset with 2,510 samples by randomly selecting 50% of the entire dataset. This small dataset is generated each time an experiment is running over it.

Our experimental simulations were run on a system with dual 2.59 GHz AMD Opteron Processors, 8 gigabytes of RAM, and an Linux operating system.
3.2.1 Quality Analysis Measures

In this thesis, we hope to identify disease-related signatures of CGH data by clustering a large number of samples. We assume that samples belonging to the same cancer type are homogeneous and should be clustered together. There are a range of different cluster validation techniques that can be grouped into two categories, *external measure* and *internal measure* [29]. We use both measures to evaluate the quality of the clusters. An external measure evaluates how well the clusters separate samples that belong to different cancer types. Thus external measure can compare clusters based on different distance/similarity measure. On the other hand, an internal measure evaluates how good the clustering algorithm operates on a given distance/similarity measure. This measure ignores the cancer types of the input samples. Compared with internal measures, external measures are more reasonable in reflecting the quality of clusters as they take the cancer types into consideration. Note that internal measure is a better indicator of quality for cancer types that have multiple aberration patterns that differ significantly.

**External measure:** An external measure takes a value in $[0, 1]$ interval. Higher values of this function represent better clustering quality. An important note is that this measure is independent of the underlying distance/similarity measure. Thus, the results of different distance measures can be compared using external measure.

We use three external measures to evaluate the cluster quality. Let $n$, $m$ and $k$ denote the total number of samples, the number of different cancer types and the number of clusters respectively. Let $a_1, a_2, \cdots, a_m$ denote the number of samples that belong to each cancer type. Similarly, let $b_1, b_2, \cdots, b_k$ be the number of samples that belong to each cluster. Let $c_{i,j}, \forall i, j, 1 \leq i \leq m$ and $1 \leq j \leq k$, denote the number of samples in $j$th cluster that belong to the $i$th cancer type. The first external measure used, known as the *Normalized Mutual Information* (NMI) [93] function is computed as:
\[ NMI = \frac{\sum_{i=1}^{m} \sum_{j=1}^{k} c_{i,j} \log\left(\frac{n \cdot c_{i,j}}{a_i b_j}\right)}{\sqrt{\left(\sum_i a_i \log \frac{a_i}{n}\right)\left(\sum_j b_j \log \frac{b_j}{n}\right)}}. \]

The second external measure is F1-measure [78]. It is defined as:

\[ F_1 = \frac{1}{n} \sum_{i=1}^{m} a_i \max_j \frac{c_{i,j}}{a_i + b_j}. \]

The third external measure is known as Rand Index [78]. In order to compute the Rand Index measure for a given clustering, two values are calculated.

- \( f_{00} \) = the number of pairs of samples that have different cancer types and belong to different clusters.
- \( f_{11} \) = the number of pairs of samples that have the same cancer type and belong same cluster.

The Rand Index is then computed as:

\[ \text{Rand Index} = \frac{f_{00} + f_{11}}{\frac{n(n-1)}{2}}. \]

Unlike other external measures, NMI was computed based on mutual information \( I(X;Y) \) between a random variable \( X \), governing the cluster labels and a random variable \( Y \), governing the cancer types. It has been argued that the mutual information is a superior measure than purity or entropy [77]. Moreover, NMI is quite impartial to the number of clusters [93].

**Internal measure:** Unlike the external measure, the value of internal measure depends on the distance/similarity measure. Thus, the internal measure of different clusterings obtained by different similarity measures are not comparable. Instead, we use this measure to compare the clusters obtained by applying different clustering methods with same similarity function. In this thesis, we implement two internal measures. One is the internal measure based on compactness (cohesion) [78], the other is the internal measure based on separation.
Let $k$ denote the total number of clusters. Let $b_1, b_2, \ldots, b_k$ be the number of samples that belong to each cluster. We use $s_i$ and $C_r$ to represent $i$th sample and the $r$th cluster respectively. Let $S(s_i, s_j)$ be the function that evaluates the similarity between the $i$th and $j$th sample. The internal measure based on compactness is computed as:

$$ IC = \sum_{r=1}^{k} \sum_{i<j, s_i, s_j \in C_r} \frac{S(s_i, s_j)}{b_r}.$$ 

The internal measure based on separation is computed as:

$$ IS = \sum_{r=1}^{k} \frac{\sum_{q=1, q \neq r}^{k} \sum_{s_i \in C_r, s_j \in C_q} S(s_i, s_j)}{\sum_{r=1}^{k} \sum_{q=1, q \neq r}^{k} b_r \cdot b_k}.$$ 

Since both internal measures are computed with pairwise similarity, higher values of $IC$ and lower values of $IS$ represent better clustering quality respectively.

### 3.2.2 Experimental Evaluation

In this section, we applied the combinations of four distance/similarity measures and five clustering methods over the entire dataset and the small dataset. We compared each combination according to the qualities of clusters. The cluster results are evaluated using different external measures. Due to the space limit, we mainly report the results using NMI and $F_1$-measure in the thesis unless otherwise stated. For the small dataset that are randomly generated each time, we apply our experiments 100 times and report the results between fifth and ninety-fifth percentile as the error bar.

**Evaluation of distance measures:** The purpose of this experiment is to compare the distance/similarity measures discussed in this thesis, namely Raw, Sim, CosineNoGaps, and CosineGaps. In the experiment, we randomly select 50% of the entire dataset as a small dataset with 2,510 samples. For each distance/similarity measure, we created 2, 4, 8, 16, 32 and 64 clusters using five clustering methods: top-down, bottom-up, $k$-means, top-down + $k$-means, and bottom-up + $k$-means. This resulted in $6 \times 5 = 30$ sets of clusters per measure. We report the highest value of external measure of all these 30 sets as the best quality of a measure. We repeat this experiment for 100 times.
Table 3-2. Highest value of external measures for different distance/similarity measure. All numbers here are the medians of 100 results.

<table>
<thead>
<tr>
<th></th>
<th>Sim</th>
<th>CosineNoGaps</th>
<th>CosineGaps</th>
<th>Raw</th>
</tr>
</thead>
<tbody>
<tr>
<td>NMI</td>
<td>0.368</td>
<td>0.265</td>
<td>0.228</td>
<td>0.239</td>
</tr>
<tr>
<td>F\textsubscript{1}-measure</td>
<td>0.34</td>
<td>0.258</td>
<td>0.215</td>
<td>0.235</td>
</tr>
<tr>
<td>Rand Index</td>
<td>0.903</td>
<td>0.899</td>
<td>0.898</td>
<td>0.896</td>
</tr>
</tbody>
</table>

The median of 100 highest values for Sim, CosineNoGaps, CosineGaps and Raw are shown in the table 3-2. The results of both NMI and F\textsubscript{1}-measure show that Sim produces the highest quality compared to other distance measures. Sim obtains this quality with top-down clustering method. CosineNoGaps gives slightly better quality than the other two measures, Raw and CosineGaps. We conclude that Sim is the most suitable distance/similarity measure for clustering CGH data.

**Evaluation of clustering methods and optimizations:** The purpose of these experiment is to compare the quality of clustering algorithms with a fixed distance/similarity measure. We create 8, 16, 32 and 64 clusters using different clustering methods with and without centroid shrinking strategy. We only report the results for Sim due to the space limitations and because Sim gives the best external measure values among all distance/similarity measures.

We randomly select 50% of the entire dataset (i.e., 2,510 samples) and cluster it. We then compute the external measure for the underlying clusters. We repeat this process 100 times and compute the error bar for the external measure. The error bar indicates the interval where 5-95% of the results lie. Figure 4-3A and Figure 4-3B show the NMI and F\textsubscript{1}-measure respectively. Top-down clustering method without centroid shrinking gives the best quality consistently in both figures. The additional $k$-means step in top-down+$k$-means method deteriorates the qualities. Centroid shrinking improves the results when the quality of the clustering method is low. It hurts the quality when the quality is high, especially when top-down method is used. This can be explained as follows. The clustering quality is low when the patients with different cancer types are clustered together. This usually indicates that different samples in the same cluster can
Figure 3-4. Evaluation of cluster qualities using (a) NMI and (b) $F_1$-measure for different clustering methods. The fifth and the ninety-fifth percentile of the results are reported as the error bar.
contain gain, loss, and no-change status for the same genomic interval. Such genomic
intervals can be considered as noise. Centroid shrinking filters them out. However,
centroid shrinking has the limitation that its results can be followed by a standard
$k$-means clustering using Euclidean distance. Therefore, the underlying similarity measure
(i.e., Sim) can not be used after shrinking the centroid. Thus, we conclude that top-down
method works best in conjunction with the Sim measure. At the same time, centroid
shrinking strategy does not help the clustering using this combination. The error bars
confirm that the top-down clustering without centroid shrinking works best for Sim
measure. The error bars show that the top-down and the bottom-up methods are more
stable than the k-means method. This is because k-means is significantly sensitive to
the initial seeds that are randomly generated. The NMI value of the top-down method
increases as the number of clusters increase from 8 to 64 in Figure 4-3A. On the other
hand, the F$_1$-measure drops in Figure 4-3B. This is because F$_1$-measure favors coarser
clustering and is biased towards small number of clusters while NMI is quite impartial
to the number of clusters [93]. We don’t see the same effect for other clustering methods
because the large variance in the results of other methods, except bottom-up, hides this
effect. For bottom-up method with or without centroid shrinking, we can see that the
increase in the quality gets flattened when the number of clusters increases.

Next, we ran all the mentioned clustering methods for the entire CGH dataset (i.e.,
5,020 samples). Figure 3-5 shows the NMI for Sim. The results confirm the experiments in
Figure 4-3A: 1) Top-down clustering produces the best clusters. 2) The centroid shrinking
strategy does not have a significant impact. 3) Most of the results on the entire dataset
remain within the error intervals. The best clustering quality was obtained when 64
clusters were created. The average cluster size, i.e. number of samples in the cluster, is
78.44 and the standard deviation is 51.03.

In our experiments on the same dataset using Rand Index, we obtained slightly better
results with top-down method. The two described internal measures (compactness and
Figure 3-5. Cluster qualities of different clustering methods with Sim measure over the entire dataset. The cluster qualities are evaluated using NMI.

separation) support this conclusion that top-down clustering is the better choice (results omitted due to space limitation).

**Performance issues of top-down clustering:** In Section 3.1.2.3, we discussed two types of top-down methods, top-down method with global refinement and top-down method with local refinement. Here, we evaluate the quality and running time of these two strategies. We restrict the similarity measure to Sim as it gives the highest quality. Using each strategy, we created 2, 4, 8, 16, 32, and 64 clusters for each of the 19 cancer types. We compute the average internal measure based on compactness of all the cancer types as the quality of the clusters. We also compute the average time to create clusters as the running time.

Table 3-3 shows the average quality and running time of two different top-down methods. The first part of the table indicates that local refinement gives slightly worse qualities than the global refinement. However, the quality difference is negligible. The quality of the clusters increases as the number of clusters increases up to 32. The quality
starts to plateau or drop after this point. This indicates that, in general, as the number of clusters increases, the clusters are more compact and the intra-similarity of clusters increases. However, when the number of clusters becomes too large compared to the size of dataset, some closely similar samples will be forced into different clusters, which, instead, reduce the intra-similarity of clusters. The second part of the table indicates that the average running time for global refinement is much higher than local refinement. This observation is consistent with our analysis of time complexity in Section 3.1.2.3. Considering that local refinement gives only slightly worse qualities but runs much faster than global refinement, we use the former method throughout this chapter.

### 3.3 Conclusion

We considered the problem of clustering Comparative Genomic Hybridization (CGH) data of a population of cancer patient samples. We developed a systematic way of placing patients with same cancer types in the same cluster based on their CGH patterns. We focused on distance-based clustering strategies. We developed three pairwise distance/similarity measures, namely raw, cosine, and sim. Raw measure disregards correlation between contiguous genomic intervals. It compares the aberrations in each genomic interval separately. The remaining measures assume that consecutive genomic intervals may be correlated, Cosine maps pairs of CGH samples into vectors in a high dimensional space and measures the angle between them. Sim measure counts the number of independent common aberrations. We employed our distance/similarity measures on
three well known clustering algorithms, bottom-up, top-down, and \( k \)-means with and without centroid shrinking.

In our experiments using classified disease entities from the Progenetix database, the highest clustering quality was achieved using Sim as the similarity measure and top-down as the clustering strategy. This observation fits with the theory that contiguous runs of genomic aberrations arise around a point-like target (e.g., oncogene), and that consecutive genomic intervals cannot be considered as independent of each other.
CHAPTER 4
IMPROVE CLUSTERING USING MARKERS

We observe that Sim measure is affected from noisy aberrations in CGH data since it depends on only a pair of samples. In this chapter, we develop a dynamic programming algorithm to identify a small set of important genomic intervals called markers. The advantage of using these markers is that the potentially noisy genomic intervals are excluded from clustering. We also develop two clustering strategies using these markers. The first one, prototype-based approach, maximizes the support for the markers. The second one, distance-based approach, develops a new similarity measure called RSim and iteratively refines clusters by maximizing the RSim measure between samples in the same cluster. Our results demonstrate that the markers we found represent the aberration patterns of cancer types well and they improve the quality of clustering.

4.1 Detection of Markers

Most cancers result from genomic instability and display various genomic aberrations. A recurrent alteration is a set of aberrations common to sufficiently many CGH samples. The recurrent alterations can be used to characterize the aberration pattern of samples. Due to the correlation between adjacent genomic intervals, recurrent regions can be represented using a small number of genomic intervals within these regions. We call these genomic intervals markers. Each marker is represented by two numbers \( <p, q> \), where \( p \) and \( q \) denote the position and the type of aberration respectively.

Given a set \( S \) of \( N \) CGH samples \( \{s_1, s_2, \cdots, s_N\} \). Let \( x^j_d \) denote the status value for sample \( j \) at genomic interval \( d \), \( \forall d, 1 \leq d \leq D \), where \( D \) is the number of intervals. Let \( s_j[u, v] \) be the segment of \( s_j \) that starts at the \( u \)th interval and ends at the \( v \)th interval.

We use the term segment to represent a contiguous block of aberrations of the same type. Formally, a list of status values \( x^j_u, x^j_{u+1}, \cdots, x^j_v \), for \( 1 \leq u \leq v \leq D \) define a segment if genomic intervals \( u \) through \( v \) are in the same chromosome, the values from \( x^j_u \) to \( x^j_v \) are all gains or all losses, and \( x^j_{u-1} \) and \( x^j_{v+1} \) are different than \( x^j_u \). For example, in Figure 4-1,
Genomic Intervals 1 2 3 4 5 6 7 8 9 10 11 12
X 0 1 1 1 0 0 -1 -1 0 1 -1 -1
Y 0 0 1 1 1 0 0 0 0 1 1 1

Figure 4-1. Two CGH samples X and Y with the values of genomic intervals listed in the order of positions. The segments are underlined.

Sample X contains four segments. The first and third segments are gain type while the second and fourth segment are loss type.

Let \( \{ m_t = \langle p_t, q_t \rangle \mid 1 \leq t \leq R \} \), be a set of markers that are ordered along the genomic intervals, i.e. \( p_1 < p_2 < \cdots < p_R \). We define the support of \( s_j \) to \( m_t \) as \( \sigma(s_j, m_t) \). Here, \( \sigma(s_j, m_t) = 1 \) only if both of the following conditions are satisfied:

1. Support: There exists a segment \( s_j[u, v] \) overlapping with \( m_t \), i.e. \( u \leq p_t \leq v \) and the type of \( s_j[u, v] \) is the same as that of \( m_t \), i.e. \( x^j_u = q_t \).

2. Uniqueness: There exists no other marker \( m_{t'} \), \( t' < t \), in the same chromosome band such that \( u \leq p_{t'} \leq v \), \( x^j_u = q_{t'} \) and \( \sigma(s_j, m_{t'}) = 1 \). Otherwise, \( \sigma(s_j, m_t) = 0 \). We say \( s_j \) supports \( m_t \) or \( m_t \) covers \( s_j \) if \( \sigma(s_j, m_t) = 1 \). We define the support value of marker \( m_t \) as the sum of its support from all the samples. Formally, \( \text{Support}(m_t) = \sum_{j=1}^{N} \sigma(s_j, m_t) \).

The intuition behind these two conditions is as follows. 1. Support: The support value of a marker counts the number of samples that share the same aberration type as the type of this marker. Large support value for a marker implies that that marker is important in characterizing the aberration pattern of samples. The first condition ensures that a sample supports a marker only if it has the same aberration as that marker in the position specified by that marker. 2. Uniqueness: The aberrations in the same segment may correspond to a single aberration that spread to neighboring genomic intervals [78]. The second condition forces a segment overlapping with multiple markers of the same aberration type to support only one of those markers.
Marker detection problem can be formally defined as follows. Given a set of CGH samples $S = \{s_1, s_2, \cdots, s_N\}$ and a positive integer $R$, the goal is to find the set of $R$ markers $M = \{m_1, m_2, \cdots, m_R\}$, $p_1 < p_2 < \cdots < p_R$, such that the sum of support of markers in $M$, i.e., $\sum_{t=1}^{R} \text{Support}(m_t)$, is maximized. Next, we develop a dynamic programming algorithm to solve this problem optimally.

Let $O(d, r) = \sum_{t=1}^{r} \text{Support}(m_t)$, for $1 \leq p_t \leq d \leq D, \forall t \in [1, r]$ denote the largest possible support that $r$ markers can get from the genomic intervals in the range $[1..d]$. Here, $[1..d]$ denote the integers $1, 2, \cdots, d$. $O(1, 1) = \text{support of the single marker at the first genomic interval}$. The value of $O(d, r)$ in general, where $1 \leq r \leq R$, $r \leq d \leq D$, can be computed as the maximum of two possible cases.

- **Case 1:** There exists a marker $m_r$ at the $d$th genomic interval. In this case, $O(d, r)$ can be computed as the maximum sum of support of $m_r$ and the optimal value of locating $r-1$ markers. Formally, it can be written as $O(d, r) = \max_{b-1 \leq d \leq d-1} \{O(dt, r-1) + \text{Support}(m_r)\}$, where $b$ denotes the least genomic interval that is in the same chromosome as the $d$th genomic interval. Note that $O(dt, r-1)$ may correspond to different set of $r-1$ markers for different values of $dt$. The type of marker $m_r$ can be either gain or loss. We choose the marker type as the one that leads to a larger $\text{Support}(m_r)$ value.

- **Case 2:** There is no marker at the $d$th genomic interval. $O(d, r) = O(d-1, r)$ in this case.

Thus, $O(d, r)$ can be computed using the following recursive equation, $O(d, r) =$

$$
\max \begin{cases}
O(d-1, r-1) + \text{Support}(m_r) \\
O(d-2, r-1) + \text{Support}(m_r) \\
\cdots \\
O(b-1, r-1) + \text{Support}(m_r) \\
O(d-1, r)
\end{cases}
$$

if $m_r$ appears at interval $d$

otherwise

50
The marker set $M$ that leads to $O(D, R)$ corresponds to $R$ markers with the largest sum of supports. We call the above approach a *fixed* number of markers approach.

An important feature of the dynamic programming approach is that optimal solutions to subproblems are retained so as to avoid recomputing their values [67]. We construct a $D \times R$ matrix with cell $(d, r)$ storing the optimal value of $O(d, r)$. An iterative program is implemented to fill this matrix. For each cell $(d, r)$, we need to revisit cell $(g, r - 1)$, $b - 1 \leq g \leq d - 1$, which takes constant time that is proportional to the average length of chromosome. Besides, we need $O(N)$ time to compute $Support(m_r)$. So the time complexity of filling one cell is $O(N)$. The overall time complexity of filling the whole matrix would be $O(DNR)$.

**Adaptive number of markers:** The above approach finds the best combination of $R$ markers when $R$ is given. However, the number of markers, usually, is not known in advance. We modify the approach to adaptively determine the number of markers.

Generally, markers are genomic intervals that are supported by sufficiently many segments contained in a set of samples. We, here, define the segment coverage of markers as ratio of segments that support the markers to the total number of segments contained in the samples. Formally, given $R$ markers found in a set $S$ of CGH samples, we define $SC(R) = \frac{O(D, R)}{\delta}$, where $\delta$ denotes the total number of segments in $S$. We adaptively determine the number of markers as below. Given a threshold $\sigma$, where $0 < \sigma \leq 1$, we find the minimum $R$ markers whose segment coverage is greater or equal to $\sigma$. That is, $R = \text{argmin}_R (SC(R) \geq \sigma)$. Here, $\sigma$ indicates the fraction of segments that are relevant to the aberration patterns of samples. Therefore, it determines the number of markers that appropriately capture the aberration patterns. The value of $\sigma$ is given by the users.

The number of markers chosen is no longer fixed. Figure 4-2 presents an example of applying adaptive number of markers approach over CGH data of cancer type, Retinoblastoma, NOS (ICD-O 9510/3) [22], which contains 120 samples with each sample including 862 genomic intervals. In this case, the parameter $\sigma$ is set to 0.5 and four
Figure 4-2. The CGH data plot of cancer type, Retinoblastoma, NOS (ICD-O 9510/3), with 120 samples and each sample containing 862 genomic intervals. The genomic intervals with gain and loss status are plotted with green and red colors respectively. The genomic intervals with no change status are not plotted. Four markers are found at genomic interval 52, 69, 287 and 690 using adaptive number of marker approach with $\sigma$ set to 0.5. The markers are shown using vertical lines. The types of four markers are gain, gain, gain and loss respectively.

Markers are founded at genomic interval 52, 69, 287 and 690 with types gain, gain, gain and loss respectively. Please note that two gain markers are found at adjacent genomic intervals, 52 and 69, because they are in different chromosomes and one segment can support both of them at the same time.

4.2 Prototype Based Clustering

Given a set of CGH samples, our marker detection technique finds a set of markers that characterize the aberration pattern of the samples in that set. These markers can be considered as the prototype of the samples. In this section, we develop a prototype-based clustering algorithm to partition the dataset into subsets such that each subset has a prototype common to the samples in that subset. It is similar to the $k$-means algorithm.
in spirit. It starts with a random partition of samples. It then iteratively maximizes an objective function, which we call cohesion function, in two steps. We discuss the cohesion function later. The two steps are described as follows.

- **refinement step:** For each cluster, the dynamic programming technique developed in Section 4.1 is used to identify the optimal set of markers. These markers serve as the prototype of this cluster.

- **reassignment step:** Each sample is assigned to the cluster whose prototype is supported the most by that sample.

Essentially, both steps optimize a cohesion function alternatively until the value of this function does not change, i.e. converges to a stable value. Next, we discuss the cohesion function in detail and prove that our clustering algorithm converges.

Given a set of CGH samples \( S = \{s_1, s_2, \ldots, s_N\} \). Let \( K \) denote the number of clusters. Let \( c_i \) denote the \( i \)th cluster. A clustering of samples can be represented by an encoding function \( f : s_j \rightarrow [1..K] \) that maps sample \( s_j \) to cluster \( c_{f(s_j)} \). Let \( R \) be the number of markers in each cluster. Let \( M_i = \{m_{i,1}, m_{i,2}, \ldots, m_{i,t}\} \) denote the set of markers (i.e., prototype) for cluster \( i \), where \( 1 \leq i \leq K \), \( 1 \leq t \leq R \) and \( m_{i,t} \) denote the \( t \)th marker in the \( i \)th cluster. Each marker \( m_{i,t} \) is a tuple \(< p_{i,t}, q_{i,t} >\), where \( p_{i,t} \) and \( q_{i,t} \) denote the position and type of this marker respectively. Let \( \phi \) denote the set of prototypes \( M_1, M_2, \ldots, M_K \). We define a cohesion function of clustering as below

\[
\text{cohesion}(f, \phi) = \sum_{i=1}^{K} \sum_{s_j \in c_i} \sum_{t=1}^{R} \sigma(s_j, m_{i,t}),
\]

where \( \sigma(s_j, m_{i,t}) \) is defined as same as in Section 4.1. Essentially, the cohesion function computes the intra-cluster similarity between samples and cluster prototypes.

In the refinement step, we optimize the cohesion function by refining the prototype (markers) of clusters given that samples are partitioned into \( K \) clusters. For cluster \( i \), let \( O_i(d, r) = \sum_{t=1}^{r} \text{Support}(m_t) = \sum_{t=1}^{r} \sum_{s_j \in c_i} \sigma(s_j, m_{i,t}) \) denote the largest support that \( t \) markers can get from the genomic intervals in the range \([1..d]\). Thus the optimal cohesion
function can be written as \(\text{cohesion}(f, \phi^*) = \sum_{i=1}^{K} O_i(D, R)\) where \(\phi^*\) denotes the set of refined prototypes and \(\phi^* = \text{Argmax}_\phi(\text{cohesion}(f, \phi))\). The dynamic programming technique described in Section 4.1 is used to compute \(O_i(D, R)\) for \(1 \leq i \leq K\) and, in this way, the cohesion function is optimized.

In the reassignment step, we reassign the sample \(s_j\) to the \(i\)th cluster whose prototype \(M_i\) is supported the most by \(s_j\), i.e. \(M_i\) covers the largest number of segments in \(s_j\). This is because, otherwise, the cohesion function could always be improved by letting \(f(s_j) = i\). Formally, \(f^* = \text{Argmax}_f(\text{cohesion}(f, \phi))\) where \(f^*\) denotes the new encoding function after reassignment step.

**Proof of convergence:** It can be seen that both steps, refinement and reassignment step, are connected through the cohesion function they alternatively optimize. The dynamic programming in refinement step optimize the cohesion with respect to \(\phi\). At iteration \((h)\) we have

\[
\text{cohesion}(f^{(h)}, \phi^{(h+1)}) \geq \text{cohesion}(f^{(h)}, \phi^{(h)})
\]

On the other hand, the reassignment step optimize the cohesion based on \(f\), i.e.

\[
\text{cohesion}(f^{(h+1)}, \phi^{(h+1)}) \geq \text{cohesion}(f^{(h)}, \phi^{(h+1)})
\]

Put together, our algorithm generates a sequence \((f^{(h)}, \phi^{(h)}), h \geq 0\), that increase the cohesion function as

\[
\text{cohesion}(f^{(h+1)}, \phi^{(h+1)}) \geq \text{cohesion}(f^{(h)}, \phi^{(h)})
\]

Since the maximum value of cohesion function is finite given a set of samples, our algorithm converges to a stable value at the end.

It is worth noting that our algorithm is a \(k\)-means-type algorithm. Let \(x_1, x_2, \cdots, x_N \in \mathbb{R}^D\) be a finite number of samples. To partition patterns into \(K\) partitions, \(2 \leq K \leq N\), a \(k\)-means-type algorithm tries to solve the following mathematical problem:

\[
P: \text{minimize } f(W, Z) = \sum_{i=1}^{K} \sum_{j=1}^{N} w_{ij} D(x_j, z_i)
\]
subject to $\sum_{i=1}^{K} w_{ij} = 1, j = 1, 2, \cdots, N$, $w_{ij} = 0$ or 1 for $i = 1, 2, \cdots, K$ and $j = 1, 2, \cdots, N$

where $W = [w_{ij}]$ is a $K \times N$ matrix, $Z = [z_1, z_2, \cdots, z_K]$ and $z_i \in \mathbb{R}^D$ is the center of cluster $i$, $D(x_j, z_i)$ is some similarity measure between $x_j$ and $z_i$. If we define $D(x_j, z_i) = -\sum_{t=1}^{R} \sigma(x_j, m_{i,t})$, the maximization of cohesion function is equivalent to problem $P$. It has been proved that a $k$-means-type algorithm converges to a partial optimal solution of problem $P$ in a finite number of iterations [69].

Under certain conditions, the $k$-means-type algorithm may fail to converge to a local minimum. Let $(W^*, Z^*)$ be a partial optimal solution of problem $P$ and $A(W^*) = \{Z : Z$ minimizes $f(W^*, Z), Z \in \mathbb{R}^{DK}\}$. A sufficient condition for $W^*$ to be a local minimum of problem $P$ is that $A(W^*)$ is a singleton [69]. Next, we show that practically the prototype-based clustering algorithm usually converges to a local optimum. In prototype-based clustering algorithm, $A(W^*)$ represents the sets of markers identified in each cluster for a certain clustering results $W^*$. When the number of markers is small as compared to the number of all intervals, each chromosome arm often contains at most one marker. Since alterations in different chromosome arms are independent to each other, the markers identified in a chromosome arm should be the one with the largest support and no other markers is identified in the same chromosome arm. Therefore, the optimal marker in each chromosome arm can be a singleton if we assume no two markers have the same support value. This makes the set of markers for each cluster a singleton. Therefore, the prototype-based clustering algorithm often converges to a local optimum.

### 4.3 Pairwise Similarity Based Clustering

Pairwise similarity based algorithms partition the samples into clusters so that the similarity between samples from the same cluster are larger than the similarity between samples of different cluster. This generally requires calculating the similarity between two samples. In Chapter 3, we proposed a segment-based similarity measure, called Sim,
for CGH data. In this section, we develop a new similarity measure, called $RSim$, which avoids noisy aberrations with the help of markers.

Let $D$ denote the number of genomic intervals of each sample. Let $s_i = x_{i1}, x_{i2}, \ldots, x_{iD}$ and $s_j = x_{j1}, x_{j2}, \ldots, x_{jD}$ be two CGH samples. Here, $x_{id}$ and $x_{jd}$ denote the value or status of the $d$th genomic interval of $s_i$ and $s_j$, respectively.

We, first, summarize the $Sim$ measure we developed for computing the similarity of two CGH data in Chapter 3. We call two segments from two samples overlapping if they have at least one common genomic interval of the same type. $Sim$ constructs maximal segments by combining as many contiguous aberrations of the same type as possible. Thus, each sample translates into a sequence of segments. For example, in Figure 4-1, $s_i$ and $s_j$ are two samples that have four and two segments respectively. After this transformation, $Sim$ computes the similarity between two CGH samples as the number of overlapping segment pairs. This is because each overlap may indicate a common point-like aberration in both samples which potentially led to the overlapping segments. In Figure 4-1, the first segment of $s_i$ is overlapping with the first segment of $s_j$. Similarly the third segment of $s_i$ is overlapping with the second segment of $s_j$. $Sim$ computes the similarity between two samples based on the genomic aberrations local to these samples. Thus, $Sim$ can not distinguish the true aberrations from noisy ones. As a result, $Sim$ is a local measure that is easily biased by the noise. Next, we develop a new approach that addresses this limitation.

We propose to employ markers to eliminate the contribution of noise to the pairwise similarity. We develop a refined $Sim$ measure, called $RSim$, as follows. Let $M = \{m_1, m_2, \ldots, m_R\}$, $p_1 < p_2 < \cdots < p_R$, denote the set of markers that are globally identified in all samples. The markers imply the important genomic intervals that are associated with the aberration patterns of samples. Given two CGH samples $s_i$ and $s_j$, $RSim$ computes the similarity between them as the number of overlapping segments pairs, such that these segments satisfy both of the following two conditions:
1. At least one of the markers in $M$ is contained in both segments.

2. Both of the overlapping segments have the same aberration type as the marker they both contain.

Formally, let $x^i_u, x^i_{u+1}, \ldots, x^i_v$ and $x^j_w, x^j_{w+1}, \ldots, x^j_v$ be a pair of segment from samples $s_i$ and $s_j$, respectively. RSim counts this pair of segments as one only if

1. there exists a marker $m_t = <p_t, q_t>$, $m_t \in M$, such that $u \leq p_t \leq v$ and $u' \leq p_t \leq v'$, and

2. the aberration type of both segments is the same as that of $m_t$, i.e. $x^i_u = x^j_w = q_t$.

Unlike Sim, RSim does not consider the overlapping segments that do not intersect with any marker. This is because RSim considers such segments as noise. For example, assume that there are two markers in Figure 4-1, one at the 3rd and the other at the 11th genomic interval. Then RSim measure for $s_i$ and $s_j$ is computed as one, whereas Sim measure is two.

An important observation on RSim is as follows. As the number of markers approaches to the number of genomic intervals in the CGH data, RSim becomes equivalent to the Sim measure. This is because all segments in the CGH data will overlap with a marker and contribute to the similarity. Thus Sim is a special case of RSim when noisy aberrations are not eliminated.

Our previous work in Chapter 3 showed that Sim works best when combined with topdown algorithm compared to other popular clustering algorithms such as bottom-up and $k$-means. In this chapter, we propose to use the topdown clustering method with RSim as the pairwise similarity measure for pairs of CGH samples.

Note that it is possible to extend the Raw measure in Chapter 3 by taking the markers into account. The extended Raw measure works as follows. For each pair of samples, we compute the similarity between them as the number of genomic intervals that meet the following two conditions. First, both samples have the same aberration type (gain or loss) at this interval. Second, one marker of the same type as both samples
appears at this interval. Our experiments (results omitted due to space limit) show that although the markers improve the original Raw measure, RSim is always superior to this measure.

4.4 Experimental Results

Dataset: With more than 12,000 cases [1], the largest resource for published CGH data can be found in the Progenetix database [3] (http://www.progenetix.net). We use three different datasets, dissimilarDS, interDS and similarDS, taken from Progenetix databases. Each dataset contains more than 800 CGH samples (i.e. cytogenetic imbalance profiles of tumor samples) from four different histopathological cancer types. Each sample has been coded according to the ICD-O-3 system [22] and consists of 862 ordered genomic intervals extracted from 24 chromosomes. In principle, each dataset can be mapped to an integer matrix of size $N \times 862$, where $N$ denotes the number of samples. The difference of these three datasets are the divergence of aberration patterns in distinct cancer types. In dissimilarDS, the samples of different cancer types contain diverse aberration patterns that are easily distinguished from each other. The samples in similarDS contain similar aberration patterns. The interDS dataset is at an intermediate degree. The choices of these datasets are based on a visual inspection of the matrices for each of the cancer types.

System specifications: Our experimental simulations were run on a system with dual 2.59 GHz AMD Opteron Processors, 8 gigabytes of RAM, and an Linux operating system.

4.4.1 Quality of Clustering

Measuring the quality of clustering is a challenging task as it is an unsupervised learning task. There are a number of internal and external cluster validation techniques that are described in the literature. In the following, we describe two measures that can be used for evaluating the clustering.

1. Coverage Measure (CM):

An internal measure evaluates the quality of clusters if the class labels of samples are not known a priori. One possible internal measure is the cohesion function defined in
Section 4.2. This function measures the total support of markers over each cluster. We use the term Coverage Measure (CM) to denote this measure. Markers with high support potentially convey some meaningful biological information and potentially can serve as the first step for further analysis, such as the identification of new oncogenes and tumor suppressor genes. A group of markers can be considered as biologically relevant if they cover most of the segments in all the patients.

2. Normalized Mutual Information (NMI): One way to measure the quality of clustering is to see if each cluster predominantly consists of samples from a single cancer type. This clearly makes the assumption that this information is available (as was the case for datasets used in this chapter) and those samples from the same cancer type will generally be similar to each other. The external measure, known as the Normalized Mutual Information, described in Section 3.2.1 can be used for this purpose.

4.4.2 Quality of Markers

Measuring the quality (or the biological relevance) of the markers identified for the clusters is also important. Here, we develop a measure to address this. We combine all the markers found in each cluster. For each marker in combined marker set, we first compute the ratio of the samples that support it among the samples in each cancer type. Thus, if there are $T$ cancer types in the dataset, $T$ values are computed for each marker. We define the maximum of these ratios as the biological relevance of this marker. This is because a larger ratio of support from one cancer type indicates that the marker better captures the aberration pattern of that cancer type. Therefore, this marker is biologically relevant in that cancer type. We use the term Global Maximum Support (GMS) to represent this measure. Formally, let $\mathcal{M} = \{m_1, m_2, \ldots, m_r\}$ denote the set of markers. Let $C_i, 1 \leq i \leq T$ denote the set of samples that belong to $i$th cancer type, where $T$ is the number of different cancer types in the dataset. GMS measure for marker $m_i \in \mathcal{M}$ is computed as:

$$GMS(m_i) = \max_{1 \leq i \leq T} \left\{ \frac{1}{|C_i|} \sum_{s_j \in C_i} \sigma(s_j, m_i) \right\}$$
Note that GMS differs greatly from CM for the CM measure is computed over clusters identified by the underlying clustering strategy, whereas GMS is computed over the cancer types.

4.4.3 Evaluation

We tested the prototype-based approach and two similarity-based approaches, RSim and Sim [78] over three datasets, dissimilar DS, interDS and similar DS. For each clustering method and dataset, we created 6, 8 and 10 clusters. For each number of clusters, we tried different number of markers, i.e. 4, 6, 8, 10 and 12 markers, per cluster in the prototype-based approach. This is because biologists have pointed out that a total of 4-7 genetic alterations can be estimated for the malignant transformation of a cell [41]. Thus, we estimated that the number of aberrations common to the samples of one cluster could be around 10. For the consistency, we also employed different numbers of markers for each clustering of 6, 8 and 10 clusters in RSim. Here, the number of markers is determined as the product of number of clusters and the number of markers per cluster used in the prototype-based approach. For example, 24, 36, 48, 60 and 72 markers were found to create 6 clusters using RSim. We compared three methods according to the qualities of clusters. We evaluated the cluster qualities using both NMI and CM measures. To evaluate the cluster qualities using CM, for both RSim and Sim, we identified the markers for each resulting cluster. The number of markers per cluster is the same as that used in the prototype-based approach. We compute the error bars for part of the results. We also used GMS to evaluate the biological relevance of markers found in prototype-based approach and RSim approach.

The CM results are shown in Table 4-1. The CM values monotonically increase as the number of clusters or number of markers increase. Thus, we compare three clustering methods for the same number of clusters and markers. We observe that prototype-based approach has 8 to 34% better coverage than RSim and 15 to 41% better coverage than Sim. This is because the cohesion function optimized in prototype-based approach is
Table 4-1. Coverage measure for three clustering methods applied over three datasets. Here, \( K \) denotes the number of clusters. Proto denotes the prototype-based approach.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>( K )</th>
<th>Alg</th>
<th>( 4K )</th>
<th>( 6K )</th>
<th>( 8K )</th>
<th>( 10K )</th>
<th>( 12K )</th>
</tr>
</thead>
<tbody>
<tr>
<td>dissimilarDS</td>
<td>8</td>
<td>RSim</td>
<td>1585</td>
<td>1925</td>
<td>2254</td>
<td>2516</td>
<td>2817</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sim</td>
<td>1470</td>
<td>1883</td>
<td>2209</td>
<td>2484</td>
<td>2719</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proto</td>
<td>1839</td>
<td>2214</td>
<td>2613</td>
<td>2928</td>
<td>3149</td>
</tr>
<tr>
<td>interDS</td>
<td>8</td>
<td>RSim</td>
<td>1011</td>
<td>1473</td>
<td>1821</td>
<td>2114</td>
<td>2373</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sim</td>
<td>1004</td>
<td>1382</td>
<td>1706</td>
<td>1993</td>
<td>2246</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proto</td>
<td>1443</td>
<td>1860</td>
<td>2189</td>
<td>2519</td>
<td>2779</td>
</tr>
<tr>
<td>similarDS</td>
<td>8</td>
<td>RSim</td>
<td>1627</td>
<td>2061</td>
<td>2400</td>
<td>2694</td>
<td>3044</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sim</td>
<td>1530</td>
<td>1956</td>
<td>2316</td>
<td>2625</td>
<td>2889</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Proto</td>
<td>1946</td>
<td>2438</td>
<td>2908</td>
<td>3274</td>
<td>3551</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>RSim</td>
<td>1734</td>
<td>2153</td>
<td>2506</td>
<td>2831</td>
<td>3197</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Sim</td>
<td>1636</td>
<td>2100</td>
<td>2492</td>
<td>2822</td>
<td>3116</td>
</tr>
</tbody>
</table>

The results also show that RSim is superior to Sim most of the time. This is because the use of markers in RSim filters out the noise that are irrelevant to the aberration patterns. As a result, the markers in each cluster are supported by more samples in RSim as compared to Sim.

The NMI results are shown in Table 4-2. Since Sim produces clusters without finding markers, we list its results on a separate column. The results show that all three methods perform better on dissimilarDS than interDS and similarDS. This is because the aberration patterns of distinct cancer types in dissimilarDS are divergent. Thus, it is harder to cluster interDS and similarDS datasets. From the results, we observe that
Table 4-2. The NMI values of the three clustering methods are applied over three datasets. Here, \( K \) denotes the number of clusters. Proto, denotes the prototype-based approach.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>( K )</th>
<th>Sim</th>
<th>Alg</th>
<th>4K</th>
<th>6K</th>
<th>8K</th>
<th>10K</th>
<th>12K</th>
</tr>
</thead>
<tbody>
<tr>
<td>dissimilarDS</td>
<td>6 0.41</td>
<td>Proto</td>
<td>0.31</td>
<td>0.31</td>
<td>0.29</td>
<td>0.29</td>
<td>0.24</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSim</td>
<td>0.45</td>
<td>0.49</td>
<td>0.50</td>
<td>0.49</td>
<td>0.50</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8 0.47</td>
<td>Proto</td>
<td>0.32</td>
<td>0.28</td>
<td>0.30</td>
<td>0.25</td>
<td>0.28</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSim</td>
<td>0.48</td>
<td>0.49</td>
<td>0.49</td>
<td>0.51</td>
<td>0.49</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10 0.43</td>
<td>Proto</td>
<td>0.35</td>
<td>0.33</td>
<td>0.37</td>
<td>0.34</td>
<td>0.29</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSim</td>
<td>0.46</td>
<td>0.44</td>
<td>0.47</td>
<td>0.50</td>
<td>0.49</td>
<td></td>
</tr>
<tr>
<td>interDS</td>
<td>6 0.34</td>
<td>Proto</td>
<td>0.06</td>
<td>0.08</td>
<td>0.08</td>
<td>0.08</td>
<td>0.11</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSim</td>
<td>0.35</td>
<td>0.34</td>
<td>0.38</td>
<td>0.40</td>
<td>0.39</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8 0.32</td>
<td>Proto</td>
<td>0.07</td>
<td>0.10</td>
<td>0.09</td>
<td>0.10</td>
<td>0.11</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSim</td>
<td>0.34</td>
<td>0.34</td>
<td>0.36</td>
<td>0.38</td>
<td>0.36</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10 0.32</td>
<td>Proto</td>
<td>0.10</td>
<td>0.09</td>
<td>0.10</td>
<td>0.15</td>
<td>0.12</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSim</td>
<td>0.35</td>
<td>0.36</td>
<td>0.36</td>
<td>0.37</td>
<td>0.36</td>
<td></td>
</tr>
<tr>
<td></td>
<td>6 0.39</td>
<td>Proto</td>
<td>0.14</td>
<td>0.13</td>
<td>0.06</td>
<td>0.07</td>
<td>0.07</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSim</td>
<td>0.29</td>
<td>0.41</td>
<td>0.39</td>
<td>0.39</td>
<td>0.40</td>
<td></td>
</tr>
<tr>
<td>similarDS</td>
<td>8 0.36</td>
<td>Proto</td>
<td>0.14</td>
<td>0.09</td>
<td>0.11</td>
<td>0.07</td>
<td>0.05</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSim</td>
<td>0.36</td>
<td>0.38</td>
<td>0.38</td>
<td>0.38</td>
<td>0.37</td>
<td></td>
</tr>
<tr>
<td></td>
<td>10 0.33</td>
<td>Proto</td>
<td>0.08</td>
<td>0.08</td>
<td>0.10</td>
<td>0.06</td>
<td>0.07</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RSim</td>
<td>0.35</td>
<td>0.37</td>
<td>0.35</td>
<td>0.37</td>
<td>0.37</td>
<td></td>
</tr>
</tbody>
</table>

RSim and Sim always beat prototype-based approach in terms of the NMI values. This observation, together with the conclusion from results in Table 4-1, indicates that NMI measure has no apparent relationship with CM. This is because NMI computes the quality based on the class labels of samples. On the other hand, CM evaluates the compactness of samples in each cluster based on chromosomal aberration patterns and completely ignores the class labels. Therefore, we conclude that the pairwise similarity-based clustering approaches are more suitable to external measures, such as NMI, while the prototype-based approach works better for the Coverage Measure. RSim usually has the best NMI results using ten markers. When compared to Sim, RSim usually has better NMI values. This indicates that the use of markers in refining the pairwise similarity also leads to a better clustering in terms of NMI. Given that RSim has better CM (see Table 4-1) and NMI values than Sim (see Table 4-2), we conclude that RSim is a better pairwise similarity measure than Sim.
We compute the error bars of experimental results as follows. We randomly sample 50% of each dataset and cluster it using three methods, prototype-based, RSim and Sim, described in the thesis. To reduce the amount of computations, we choose one configuration from different combinations of parameters in the experiments. For each dataset, we create 8 clusters. We identify 10 markers per cluster in the prototype-based approach and 80 markers in RSim approach. We then compute both NMI and CM values for the resulting clusters using 10 markers per cluster. We repeat this process 100 times and compute the error bar for the values of NMI and CM. The error bar indicates the interval where 5-95% of the results lie. Table 4-3 shows the results with error bars. Note that the results of CM are roughly half the results shown in Table 4-1 because the calculation of CM depends on the dataset and we sample 50% of each dataset in the experiments. The results show that RSim is superior to Sim most of the time in terms of both NMI and CM. Moreover, among the three clustering methods, RSim and prototype-based approach works the best for NMI and CM measures, respectively. These observations are compatible to those we obtained from Table 4-1 and 4-2. Therefore, the error bars confirm our earlier conclusions.
Figure 4-3. Comparison of GMS values of markers in clusters from two clustering approaches. Plots of global maximum support of markers found (A) In dissimilarDS, (B) In interDS and (C) In similarDS. The solid line indicates the results of markers generated by RSim approach. The dashed line indicates the results of markers generated by prototype-based approach (denoted as model).

Next experiment compares the GMS values for RSIM and prototype-based clustering approaches. For each dataset, we created eight clusters and identified ten markers per cluster. This is because these results are among the best results of each dataset in Table 4-2. We then sort the markers in descending GMS value order. We plot the sorted results of both RSim and prototype-based approach (Figure 4-3). The plots show that the maximum global support of markers found by RSim is always comparable to or better than those found by the prototype-based approach.
4.5 Conclusion

We considered the problem of clustering Comparative Genomic Hybridization (CGH) data of a population of cancer patient samples. There are three main contributions of our work:

1. We developed a dynamic programming algorithm to identify the optimal set of important genomic intervals called markers. The advantage of using these markers is that the potentially noisy genomic intervals are excluded in the computation of pairwise similarity between samples.

2. We developed two clustering strategies using these markers. The first one, prototype-based approach, maximizes the support for the markers. The second one, similarity-based approach, develops a new similarity measure called RSim. It computes the pairwise similarity between samples by removing the noisy aberrations. We demonstrated the utility of such a measure in improving the quality of clustering using the classified disease entities from the Progenetix database. Our results show that the markers we found represent the aberration patterns of cancer types well.

3. We developed several measures for comparing markers and different clustering methods. Our experimental results show that optimizing for the coverage measure may not lead to better values of NMI and vice versa.
CHAPTER 5
CLASSIFICATION AND FEATURE SELECTION ALGORITHMS FOR MULTI-CLASS CGH DATA

Classification is the task of learning a target function that maps each attribute set to one of the predefined class labels [79]. Typical classification tasks for cancer research include separating healthy patients from cancer patients and distinguish patients of different cancer subtypes, based on their cytogenetic profiles. These tasks help successful cancer diagnosis and treatment. An important technique related to classification is feature selection. The goal of feature selection is to select a small number of discriminative features, i.e. genomic intervals in CGH data, for accurate classification. In this chapter, we propose novel SVM-based methods for classification and feature selection of CGH data. For classification, we developed a novel similarity kernel that is shown to be more effective than the standard linear kernel used in SVM. For feature selection, we propose a novel method based on the new kernel that recursively selects features with the maximum influence on an objective function. We compared our methods against the best wrapper based and filter based approaches that have been used for feature selection of large dimensional biological data. Our results on datasets generated from the Progenetix database, suggests that our methods are considerably superior to existing methods.

5.1 Classification with SVM

Support Vector Machine (SVM) is a state-of-art technique for classification [83]. It has been shown to have better accuracy and computational advantages over their contenders [27] and has been successfully applied for many biological classification problems. The basic technique works as follows. Consider a set of points that are presented in a high dimensional space such that each point belongs to one of two classes. An SVM computes a hyperplane that maximizes the margin separating the two classes of samples. The optimal hyperplane is called decision boundary. Formally, let $x_1, x_2, \cdots, x_n$ and $y_1, y_2, \cdots, y_n$ denote $n$ training samples and their corresponding class labels respectively. Let $y_i \in \{-1, 1\}$ denote labels of two classes. The decision boundary of
a linear classifier can be written as \( w \cdot x + b = 0 \) where \( w \) and \( b \) are parameters of the model. By rescaling the parameters \( w \) and \( b \), the margin \( d \) can be written as \( d = 2 \frac{2}{\|w\|^2} \). The learning task in SVM can be formalized as the following constrained optimization problem:

\[
\min_w \left\{ \frac{\|w\|^2}{2} \right\}
\]

subject to \( y_i (w \cdot x_i + b) \geq 1, i = 1, 2, \cdots, n \).

The dual version of the above problem corresponds to finding a solution to the following quadratic program:

Maximize \( J \) over \( \alpha_i \):

\[
J = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j
\]

subject to \( \alpha_i \geq 0, \sum_{i=1}^{n} \alpha_i y_i = 0 \), where \( \alpha_i \) is a real number.

The decision boundary can then be constructed from the solutions \( \alpha_i \) to the quadratic program. The resulting decision function of a new sample \( z \) is

\[
D(z) = w \cdot z + b
\]

with \( w = \sum_i \alpha_i y_i x_i \) and \( b = y_i - w \cdot x_i \). Usually many of the \( \alpha_i \) are zero. The training samples \( x_i \) with non-zero \( \alpha_i \) are called support vectors. The weight vector \( w \) is a linear combination of support vectors. The bias value \( b \) is an average over support vectors. The class label of \( z \) is obtained by considering the sign of \( D(z) \).

Standard SVM methods find a linear decision boundary based on the training examples. They compute the similarity between sample \( x_i \) and \( x_j \) using the inner product \( x_i^T x_j \). However, the simple inner product does not always measure the similarity effectively for all applications. For some applications, a non-linear decision boundary is more effective for classification. The basic SVM method can then be extended by transforming samples to a higher dimensional space via a mapping function \( \Phi \). By doing this, a linear decision boundary can be found in the transformed space if a proper function \( \Phi \) is used.
However, the mapping function $\Phi$ is often hard to construct. The computation in the transformed space can be expensive because of its high dimensionality. A kernel function can be used to overcome this limitation. A kernel function is defined as $K(x_i, x_j) = \Phi(x_i)^T\Phi(x_j)$, where $x_i$ and $x_j$ denote the $i$th and $j$th sample respectively. It really computes the similarity between $x_i$ and $x_j$. With the help of kernel function, an explicit form of the mapping function $\Phi$ is not required.

We introduce a new measure called Raw that captures the underlying categorical information in CGH data and then show how to incorporate it into the basic SVM method. CGH data consists of sparse categorical values (gain, loss and no change). Conceptually, the similarity between CGH samples depends on the number of aberrations (gains or losses) they both share. Raw calculates the number of common aberrations between a pair of samples. Given a pair of samples $a = a_1, a_2, \cdots, a_m$ and $b = b_1, b_2, \cdots, b_m$. The similarity between $a$ and $b$ is computed as $Raw(a, b) = \sum_{i=1}^{m} S(a_i, b_i)$. Here $S(a_i, b_i) = 1$ if $a_i = b_i$ and $a_i \neq 0$. Otherwise $S(a_i, b_i) = 0$.

The main difference between $Raw(a, b)$ and $a^T \cdot b$ is the way they deal with different aberrations in the same interval. For example, if two samples $a$ and $b$ have different aberrations at the $i$th interval, i.e. $a_i = 1, b_i = -1$ or $a_i = -1, b_i = 1$, the inner product calculates this pair as $a_i \times b_i = -1$ while $Raw$ calculates $S(a_i, b_i) = 0$. The similarity value between $a$ and $b$ computed by Raw is always greater than or equal to the inner product of $a$ and $b$. We propose to use $Raw$ function as the kernel function for the training as well as prediction.

Using SVM with the Raw kernel amounts to solving the following quadratic program:

Maximize $J$ over $\alpha_i$:

$$J = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j Raw(x_i, x_j)$$

subject to $\alpha_i \geq 0, \sum_{i=1}^{n} \alpha_i y_i = 0$. 

68
Accordingly, the resulting decision function of a new sample $z$ is

$$D(z) = \sum_i \alpha_i y_i \text{Raw}(x_i, z) + b$$

The main requirement for the kernel function used in nonlinear SVM is that there exists a mapping function $\Phi$ such that the kernel function computed for a pair of samples is equivalent to the inner product between the samples in the transformed space $[79]$. This requires that the underlying kernel matrix is "semi-positive definite". Formally, a kernel $K$ is a symmetric function taking two arguments of an arbitrary set $\mathcal{X}$ where the data stems from, i.e., $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$. For given data points $(x_i)_{i=1}^n \in \mathcal{X}^n$, the kernel matrix $M := (K(x_i, x_j))_{i,j=1}^n$ can be defined. If for all $n$, all sets of data points and all vectors $v \in \mathbb{R}^n$ the inequality $v^T M v \geq 0$ holds, then $K$ is called semi-positive definite. We now prove that our Raw kernel satisfies this requirement.

The mapping function $\Phi$ is defined as follows: $a \in \{1, 0, -1\}^m \rightarrow b \in \{1, 0\}^{2m}$, where,

$$a_i = 1, b_{2i-1} b_{2i} = 01$$

$$a_i = -1, b_{2i-1} b_{2i} = 10$$

$$a_i = 0, b_{2i-1} b_{2i} = 00$$

With this transformation, it is easy to see that the Raw kernel can be written as the inner product of $\Phi(x)$ and $\Phi(y)$, i.e. $\text{Raw}(x, y) = \Phi(x)^T \cdot \Phi(y)$. This is because Raw only counts the number of common aberrations in computing the similarity between two samples (if both the values are 0, they are not counted).

We define a $2m$ by $n$ matrix $u$ whose $j$th column vector corresponds to $\Phi(x_j)$, i.e. $u := [ \Phi(x_1) \ \Phi(x_2) \ \ldots ]$. The Raw kernel matrix can be written as
Now we have $v^T M v = v^T (u^T u) v = (uv)^T uv = \|uv\|^2 \geq 0, \forall v \in \mathbb{R}^n$. Therefore, the Raw kernel is semi-positive definite.

It is worth noting that we have developed other similarity measures such as Sim for the clustering of CGH data in Chapter 3. Although Sim works better than Raw in clustering, it can not work as kernel function in SVM because it is not semi-positive definite.

5.2 Maximum Influence Feature Selection for Two Classes

An important characteristic of CGH data is that neighboring features are strongly correlated (Figure 5-1). When a compact set of features are selected, these highly correlated features may cause "redundancy" in the predictive power. For example, assume we have a training dataset with four cancer types and we want to select two features for classification. If the $i$th feature is ranked high for separating samples of the first cancer from others, the $(i + 1)$th or $(i - 1)$th feature may be ranked high too for the same effect. However, selecting both $i$th and $(i + 1)$th (or $(i - 1)$th) feature can not improve the classification performance much. On the other hand, if another feature, say $j$th feature,
Figure 5-1. Plot of 120 CGH cases belonging to Retinoblastoma, NOS (ICD-O 9510/3). The X-axis and Y-axis denote the genomic intervals and the samples respectively. We plot the gain and loss status in green and red respectively.

well separates samples of the third cancer from others but has a lower ranking than the 
(i + 1)th (or (i − 1)th) feature, we should select the ith and jth feature instead.

Typical wrapper methods based on backward feature elimination, such as SVM-RFE [27], have poor effect in discriminating highly correlated features, especially when a small set of features are selected. Filter methods, such as MRMR [15], address this problem by selecting features with minimal redundancy. However, due to the difficulty in selecting complementary features, filter methods often produce lower predictive accuracy compared to wrapper methods. In this paper, we propose a novel non-linear SVM-based wrapper method called Maximum Influence Feature Selection (MIFS) for the classification of multiclass CGH data.

When the number of features is very large, an exhaustive search of all possible feature subsets is computationally intractable. We use a greedy search strategy to progressively add features into a feature subset. To find the next feature to add, we use criteria similar
to the one suggested by Guyon et al [27]. The basic idea is to compute the change in the objective function caused by removing or adding a given feature. In our case, we select the feature that maximizes the variation on the objective function. The added feature is the one which has the most influence on the objective function. This is unlike the backward elimination scheme that removes the feature that minimizes the variation on the objective function [27, 64].

The feature that has the most influence on the objective function is determined as follows. Let $S$ denote the feature set selected at a given algorithm step and $J(S)$ denote the value of the objective function of the trained SVM using feature set $S$. Let $k$ denote a feature that is not contained in $S$. The change in the objective function after adding a candidate feature is written as $DJ(k) = |J(S \cup \{k\}) - J(S)|$. In the case of SVM, the objective function that needs to be maximized (under the constraint $0 \leq \alpha_i$ and $\sum_i \alpha_i y_i = 0$) is:

$$J(S) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j \text{Raw}(x_i, x_j)$$

For each feature $k$ not in $S$, we compute the new objective function $J(S(+k))$. To make the computation tractable, we assume no change in the value of the $\alpha$’s after the feature $k$ is added. Thus we avoid having to retrain a classifier for every candidate feature [27]. The new objective function with feature $k$ added is:

$$J(S \cup \{k\}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j \text{Raw}(x_i(+k), x_j(+k))$$

where $x_i(+k)$ means training sample $i$ with feature $k$ added.

Therefore, the estimated (this is because we are not retraining the classifier with the additional feature) change of objective function is:

$$DJ(k) = \frac{1}{2} \left| \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j \text{Raw}(x_i, x_j) - \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j \text{Raw}(x_i(+k), x_j(+k)) \right|$$
We add the feature that has the largest difference $DJ(k)$ to the feature set. The iterative procedure for MIFS is formally defined as follows:

**Input:** Training samples $\{x_1, x_2, \ldots, x_n\}$ and class labels $\{y_1, y_2, \ldots, y_N\}$, $y_i \in \{1, -1\}$, initial feature set $S$, predetermined number of features $r$

1. **Initialize:** Ranked feature list $RL = S$, candidate feature set $L = D - S$ ($D$ is the set of all features)

2. **While** $|S| < r$

   (a) Train an SVM using training samples with features in $RL$;

   (b) Compute the change of objective function $DJ(k)$ for each candidate feature $k \in L$

   (c) Find the feature $e$ with the largest $DJ(k)$, e.g. $e = \arg\max(DJ(k))$

   (d) Update $RL = [RL, e]$ and $L = L - \{e\}$

3. **Return:** Ranked feature list $RL$

This algorithm can be generalized to add more than one feature in Step 2.d to speed up computations when the number of features $r$ is large.

**Time Complexity** The training time complexity for linear SVM is dominated by the time for solving the underlying quadratic program. The conventional approach for solving the quadratic program takes time cubic in the number of samples and linear in the number of features [48]. Recent work has shown that the empirical time complexity for training a linear SVM is about $O(n^{1.7}m)$ [33], where $n$ and $m$ denote the number of samples and number of features respectively. Based on this, the conventional and empirical time complexity for this algorithm is $O(n^3r^2)$ and $O(n^{1.7}r^2)$ respectively.

The above method requires a set of features $S$ to be non-empty. To start the method, we need to derive the first feature to be added to this set. One possibility is to compute $J(\{k\})$ for every feature $k$ by training a separate SVM for each feature $k$. We can, then, select the feature with the largest value as the starting feature. However, this can be computationally very expensive. Another approach is to use the most discriminating
feature (such as done by standard filter based methods that rank features according to their individual predictive power). Specifically, the mutual information $I$ of two variables $r$ and $s$ is defined as

$$I(r, s) = \sum_{i,j} p(r_i, s_j) \log \frac{p(r_i, s_j)}{p(r_i)p(s_j)}$$

where $p(r, s)$ is their joint probabilities; $p(r)$ and $p(s)$ are the respective marginal probabilities. If we look at the $k$th feature as a random variable, we use mutual information $I(k, y)$ between class labels $y = \{y_1, y_2, \cdots, y_n\}$ and the feature variable $k$ to quantify the relevance of $k$th feature for the classification task. We choose the feature $k$ with the maximum $I(k, y)$ as our starting feature. We have found that using such methods is satisfactory. Our preliminary experimental results showed that Multiple Selection is not sensitive to the initial feature chosen.

5.3 Maximum Influence Feature Selection for Multiple Classes

The feature selection method proposed in Section 5.2 only works for two-class problems. We derive the multiclass version using a one-versus-all approach as follows.

- **First step.** Let $C \geq 3$ denote the number of classes. For each $i$, $1 \leq i \leq C$, a binary SVM that separates the $i$th class from the rest is trained based on the selected feature set $S$.

- **Second step.** For each binary SVM, we compute $DJ(k)$ for every feature $k$ not in $S$. We rank all the candidate features based on the value of $DJ$. The larger value the value of $DJ(k)$, the smaller is its rank of $k$ (smaller is better). As a result, we obtain $C$ ranked lists of features with each ranked list corresponding to one of the $C$ SVMs. Equivalently, each candidate feature corresponds to a ranking vector containing its rankings in these $C$ ranked lists. For example, a feature can be ranked as the first in the first list; third in the second list; 20th in the third list, 15th in the fourth list. The vector that is used for ranking this feature is $[1, 3, 20, 15]$.

- **Third step.** A feature that ranks low in one list may rank high in another. We are interested in those features that are most informative in discriminating one
class from the rest even if they are quite uninformative in other classifications. We achieve this as follows. We first sort the ranking vector of each candidate feature in an ascending order. If we regard each element of the ranking vector as a digit, each ranking vector could represent a $C$ digit number. The smallest ranking (the first element) represents the most significant digit. We use a least significant digit radix sort algorithm to sort all the ranking vectors and, accordingly, produce a global ranking of features. For example, assume we have three features, $k_1$, $k_2$ and $k_3$ whose rankings in four binary SVMs are $[1, 3, 20, 15]$, $[8, 4, 7, 6]$ and $[5, 1, 30, 4]$ respectively. The vectors show that $k_1$ ranks top in separating class one from others and ranks third in separating class two from others etc. We first sort each ranking vector in an ascending order. The resulting vectors are $[1, 3, 15, 20]$, $[4, 6, 7, 8]$ and $[1, 4, 5, 30]$ respectively. Next, we apply a radix sort algorithm over the three vectors. The resulting order of vectors changes to $[1, 3, 15, 20]$, $[1, 4, 5, 30]$, $[4, 6, 7, 8]$, which corresponds to the order of features: $k_1$, $k_3$, $k_2$. Therefore, we have a global ranking of the three features.

The lowest ranked feature is added into $S$. The above three step process is used iteratively to determine the next feature. This process stops when a predetermined number of features are selected or $S$ contains all the features. Also, with the set $S$, the features are ranked based on the order of addition into this set. The iterative procedure for MIFS is formally defined as follows:

**Input:** Training samples $\{x_1, x_2, \ldots, x_n\}$ and class labels $\{y_1, y_2, \ldots, y_N\}$, $1 \leq y_i \leq C$, initial feature set $S$, predetermined number of features $r$

1. **Initialize:** Ranked feature list $RL = S$, candidate feature set $L = D - S$ ($D$ is the set of all features)

2. **While** $|S| < r$
   (a) **For** $i = 1$ to $C$
i. Construct new class labels \( \{y_1', y_2', \ldots, y_n'\} \), \( y_j' = 1 \) if \( y_j = i \), otherwise \( y_j' = -1 \);

ii. Train an SVM using training samples with features in \( RL \);

iii. Compute the change of objective function \( DJ(k) \) for each candidate feature \( k \in L \)

iv. Sort the sequence of \( DJ(k), k \in L \) in descending order; create a corresponding ranked list of candidate features;

(b) Compute the ranking vectors for all the features in \( L \) from \( C \) ranked lists;

(c) Sort the elements of each ranking vector in an ascending order;

(d) Perform a radix sort over all ranking vectors to produce a global ranking of features in \( L \);

(e) Find the top ranked feature \( e \) and update \( RL = [RL, e] \) and \( L = L - \{e\} \)

3. **Return:** Ranked feature list \( RL \)

This algorithm can be generalized to add more than one feature in Step 2.e to speed up computations when the number of features \( r \) is large.

**Time Complexity** The conventional and empirical time complexity for this algorithm is \( O(n^3r^2C) \) and \( O(n^{1.7}r^2C) \) respectively as a one-versus-all strategy is used to train \( C \) SVMs in each iterative step.

In the above algorithm, we generate a global ranking of features based on their rankings in each binary SVM. Another ”ranking scheme” can be derived based on the sum of the value that feature brings to each classifier, i.e. \( \sum_{i=1}^{C} (n_i \times |J_i(S \cup \{k\}) - J_i(S)|) \) where \( J_i(S) \) is the corresponding objective function for SVM that discriminating class \( i \) from the rest and \( n_i \) is the number of samples in class \( i \). This ranking scheme gives comparable results to the one described above. For this reason, the results concerning this scheme are not reported in the experimental section.
5.4 Datasets

The Progenetix database [3] (http://www.progenetix.net) consists of more than 12,000 cases [1]. We use a dataset consisting of 5020 CGH samples (i.e. cytogenetic imbalance profiles of tumor samples) taken from Progenetix (Table 3-1). These samples belong to 19 different histopathological cancer types that have been coded according to the ICD-O-3 system [22]. The subset with the smallest number of samples, consists of 110 non-neoplastic cases, while the one with largest number of samples, Adenocarcinoma, NOS (ICD-O 8140/3), contains 1057 cases. Each CGH sample consists of 862 ordered genomic intervals extracted from 24 chromosomes.

Testing the performance (predictive accuracy and run time) of the proposed methods, requires evaluating them over datasets with different properties such as 1) number of samples contained in the dataset, 2) number of cancer types contained in the dataset, and 3) the similarity level between samples from different cancer types, which indicating the difficulty of classification. Currently, there are no standard benchmarks for normalized CGH data that take the three properties into account. We propose a method to select subsets from the Progenetix database in a principled manner to create datasets with desired properties. The dataset sampler accepts the following three parameters as input: 1) Approximate number of samples (denoted as $N$) 2) Number of cancer types (denoted as $C$) 3) Similarity range (denoted as $[\delta_{\text{min}}, \delta_{\text{max}}]$) between samples belonging to different cancer types. An outline of the proposed dataset sampler is as follows:

1. For each cancer type, partition all the samples belonging to this cancer type into several disjoint groups using clustering. Each cluster corresponds to the different aberration patterns for a given cancer type.

2. Compute the pairwise similarity between pairs of groups obtained in the first step.

3. Construct a complete weighted graph where each vertex denotes a group of samples and the weight of an edge equals to the similarity between two groups that are connected by this edge.
One can use this graph to find a set of samples of a given size $N$ (by choosing a subset of groups that sum to $N$), given number of cancer types, and based on level of similarity between groups (by only considering groups that have a similarity within the range of $[\delta_{\text{min}}, \delta_{\text{max}}]$). The advantage of the above dataset sampler is that a large number of datasets can be created with variable number of samples and cancer types as well as variable level of similarities between the chosen cancer types. This allows for testing the accuracy and performance of a new method across a variety of potential scenarios.

Figure 5-2 shows an example of how such a dataset sampler works. Consider a dataset containing 1,000 CGH samples - 400 samples belonging to cancer type $c_1$ and the other 600 samples belonging to cancer type $c_2$. Assume that each cancer type is clustered into 2 clusters. This results in 4 groups of CGH samples, which are denoted as $g_i$, $1 \leq i \leq 4$. Let the size of $g_1$, $g_2$, $g_3$ and $g_4$ be 150, 250, 450, and 150 respectively. The pairwise similarity between any two groups is shown in the Figure. Using this, one can construct a weighted graph where each vertex denotes a group and the weight of each edge equals to the similarity between two groups that are connected by this edge. Suppose that a dataset needs to be sampled with $N = 400$, $C = 2$, $\delta_{\text{min}} = 0.025$ and $\delta_{\text{max}} = 0.035$. The graph can be parsed to find out that $g_2$ and $g_4$ satisfy the three conditions and a new dataset can be sampled by combining the samples in $g_2$ and $g_4$.

We used our dataset resampling scheme to select datasets at four different similarity levels from the Progenetix dataset. We denote the similarity levels as Best, Good, Fair, and Poor. The samples in Best has the highest similarity and those in Poor have the lowest similarity. For each similarity level, we created three datasets with two, four, six, and eight cancer types respectively. Thus, in total, we have sixteen datasets. For convenience, we use the similarity level followed by the number of cancer types to denote a dataset. For example Best6 denotes the dataset with similarity level Best (i.e., homogeneous samples) and contains six cancer types. The number of samples in each two-class dataset and multi-class dataset is around 500 and 1,000 respectively. Note that
Figure 5-2. Working example of dataset re-sampler. $c_i$ and $g_j$ denote the $i$th cancer type and the $j$th group of samples, respectively. In the first step, the samples are partitioned in each cancer type into two disjoint groups. In the second step, pairwise similarity metrics are computed. In the third step, a complete weighted graph is generated.

There is no topological relations between different datasets because we generate all datasets in separate runs. For example, any sample in $best4$ is not necessarily contained in $best6$ or $best8$.

The sampling of the sixteen datasets are explained as follows. Let $N$ and $C$ denote the number of samples and number of cancer types in the resampled datasets respectively. In our experiments, we choose $N = 500$ and $C = 2$ for two-class dataset. We choose $N = 1000$ and $C = 4, 6$ and $8$ for multi-class dataset respectively. For each value of $C$, we sample four datasets with four different levels of similarity.

In the first step, a clustering algorithm is applied to each cancer type to partition all the samples belonging to this cancer type into several disjoint groups. Each cluster corresponds to the different aberration patterns for a given cancer type. We use the RSim clustering method for this purpose. The number of clusters for each cancer type is determined adaptively as follows. For the $i$th cancer, let $S_i$ denote the number of samples
Table 5-1. Detailed specifications of benchmark datasets. Term \#cases and \( C \) denote the number of cases and number of cancer types respectively.

<table>
<thead>
<tr>
<th>Name</th>
<th>( C )</th>
<th>#cases</th>
<th>similarity level</th>
<th>ICD-O-3 code of cancer types</th>
</tr>
</thead>
<tbody>
<tr>
<td>Best2</td>
<td>2</td>
<td>478</td>
<td>([0.030, 1.000])</td>
<td>80703, 81403</td>
</tr>
<tr>
<td>Good2</td>
<td>2</td>
<td>466</td>
<td>([0.018, 0.030])</td>
<td>80103, 81703</td>
</tr>
<tr>
<td>Fair2</td>
<td>2</td>
<td>351</td>
<td>([0.008, 0.018])</td>
<td>80103, 96733</td>
</tr>
<tr>
<td>Poor2</td>
<td>2</td>
<td>373</td>
<td>([0.000, 0.008])</td>
<td>98233, 96803</td>
</tr>
<tr>
<td>Best4</td>
<td>4</td>
<td>1160</td>
<td>([0.035, 1.000])</td>
<td>95003, 85233, 80703, 81403</td>
</tr>
<tr>
<td>Good4</td>
<td>4</td>
<td>790</td>
<td>([0.020, 0.035])</td>
<td>98363, 81703, 96803, 81403</td>
</tr>
<tr>
<td>Fair4</td>
<td>4</td>
<td>800</td>
<td>([0.010, 0.020])</td>
<td>95103, 96733, 96803, 81403</td>
</tr>
<tr>
<td>Poor4</td>
<td>4</td>
<td>800</td>
<td>([0.000, 0.010])</td>
<td>85233, 96803, 98233, 81403</td>
</tr>
<tr>
<td>Best6</td>
<td>6</td>
<td>1100</td>
<td>([0.030, 1.000])</td>
<td>81443, 95003, 81703, 85233, 80703, 81403</td>
</tr>
<tr>
<td>Good6</td>
<td>6</td>
<td>850</td>
<td>([0.017, 0.030])</td>
<td>91803, 81443, 96733, 80103, 97323, 81403</td>
</tr>
<tr>
<td>Fair6</td>
<td>6</td>
<td>880</td>
<td>([0.007, 0.017])</td>
<td>95103, 81400, 98233, 96803, 80703, 81403</td>
</tr>
<tr>
<td>Poor6</td>
<td>6</td>
<td>810</td>
<td>([0.000, 0.007])</td>
<td>85233, 98233, 80703, 81400, 81403, 96803</td>
</tr>
<tr>
<td>Best8</td>
<td>8</td>
<td>1000</td>
<td>([0.030, 1.000])</td>
<td>80103, 97323, 81400, 95003, 81703, 85233, 80703, 81403</td>
</tr>
<tr>
<td>Good8</td>
<td>8</td>
<td>830</td>
<td>([0.018, 0.030])</td>
<td>88903, 93913, 91803, 96733, 80103, 81703, 80703, 81403</td>
</tr>
<tr>
<td>Fair8</td>
<td>8</td>
<td>750</td>
<td>([0.006, 0.018])</td>
<td>95103, 80103, 97323, 81703, 96803, 98233, 80703, 81403</td>
</tr>
<tr>
<td>Poor8</td>
<td>8</td>
<td>760</td>
<td>([0.000, 0.006])</td>
<td>00000, 81400, 81703, 85233, 96803, 98233, 80703, 81403</td>
</tr>
</tbody>
</table>

in it. The number of cluster is computed as \( \left\lfloor \frac{S_{i\text{N/C}}} {N/C} \right\rfloor \). Therefore, the number of each cluster is around \( N/C \) on average, which makes the size of resampled dataset close to \( N \).

In the second step, the similarities between any pair of clusters are computed and sorted in an ascending order. The 25, 50 and 75 percentile of the sorted similarity sequence are chosen to divide the sequence into four segments with about equal length. Each segment corresponds to a similarity level. We denote the four similarity levels as Poor, Fair, Good, and Best.

In the third step, the minimum and maximum similarity in each segment are chosen as the parameters \( \delta_{\text{min}} \) and \( \delta_{\text{max}} \) respectively. The datasets of different similarity levels are sampled by the dataset resampler accordingly. We list the detailed specifications of our datasets in Table 5-1.

It is worth noting that the actual number of cases may not equal to the parameter \( N \) in the resampled datasets. This is because, the clustering algorithm may generate clusters...
with unbalanced sizes. When we combine these clusters together, the actual size may be larger or smaller than \( N \).

## 5.5 Experimental Results

In this section, we describe the experimental comparison of our methods with SVM-RFE and MRMR. We developed our code using MATLAB and ran our experiment on a system with dual 2.59 GHz AMD Opteron Processors, 8 gigabytes of RAM, and a Linux operating system.

### 5.5.1 Comparison of Linear and Raw Kernel

In this section, we compare the Raw kernel to linear kernel for the classification of CGH data. We perform the experiments over the sixteen datasets using a 5-fold cross validation (CV). For each dataset, we randomly divided the data set into five disjoint subsets about equal size. For each fold, we keep one subset as the test data set and the other four sets as the training examples. We train two SVMs over the training examples using linear and Raw kernel respectively. We then use each SVM to predict the class labels of the set aside examples respectively. We compute the predictive accuracy of each SVM as the ratio of number of correctly classified samples to the number of test dataset examples. Next, we choose another subset as set aside examples and the rest as training examples. We repeat this procedure until each subset has been chosen as set aside examples. As a result, we have five values of predictive accuracy corresponding to each kernel respectively. We compute the average of the five values as the average predictive accuracy for each kernel in 5-fold CV.

We use the DAGSVM (Directed Acyclic Graph SVM) provided by MATLAB SVM Toolbox [8] for the classification of multiclass data. All other parameters of SVM are set to the standard values that are part of the software package and existing literature.

The results are presented in Figure 5-3. X-axis lists the sixteen different datasets. Y-axis denotes the value of average predictive accuracy in 5-fold CV. For the sixteen datasets, Raw kernel outperforms linear kernel in fifteen datasets (except best8). On
average, Raw kernel improves the predictive accuracy by 6.4% over sixteen datasets compared to linear kernel. For the best8 dataset, the difference between Raw and Linear is is less than 1%. These results demonstrate that SVM based on Raw kernel works better for the classification of CGH data as compared to linear SVM.

The remaining set of experimental results are limited to the Raw kernel (unless stated explicitly).

5.5.2 Comparison of MIFS and Other Methods

In this section, our method, MIFS, is compared against MRMR (a filter based approach) and SVM-RFE (a wrapper based approach). MRMR is shown to be more effective than most filter methods, such as methods based on standard mutual information, $F$-statistic or $t$-statistic [15]. The MIQ scheme of MRMR, i.e. the divisive combination of relevance and redundancy, is used because it outperforms MID scheme consistently. SVM-RFE is a popular wrapper method for gene selection and cancer classification. It is shown to be better than filter methods such as those based on ranking coefficients similar to Fisher’s discriminant criterion. SVM-RFE is also shown to be more effective than
wrapper methods using RFE and other multivariate linear discriminant functions, such as Linear Discriminant Analysis and Mean Squared Error (Pseudo-inverse) [27].

For each method, a 5-fold cross validation is used. In each fold, the feature selection method is applied over the training examples. Multiple sets of features with different sizes (4, 8, 16 features etc) are selected. For each set of features, an SVM is trained on the training examples with only the selected features. The predictive accuracy of this SVM is determined using the test (set aside) examples with the same set of features. These steps are repeated for each of the 5-folds to compute the average predictive accuracy.

To test the predictive accuracy of features selected by different methods, DAGSVM with Raw kernel is used as it is found to be more effective than other methods. Since the SVM-RFE presented in the literature only works for two-class data, it is extended to multiclass data using the same "ranking scheme" that we use to extend MIFS (as described in Section 5.3). The linear kernel is used in SVM-RFE for feature selection purpose.

The experimental results for multi-class dataset and two-class dataset are shown in Table 5-2 and Table 5-3 respectively. In these tables, the predictive accuracy of features selected by three methods, MIFS, MRMR and SVM-RFE, over each dataset are compared. For each feature selection method, the results for 4, 8, 16, 40, 60, 80, 100, 150, 250 and 500 features over each dataset are presented. The results are averaged over the 5-folds and reported in columns 3 to 12. In the 13th column, the average predictive accuracies of SVM built upon 862 features, i.e. no feature selection, are reported. The average predictive accuracies of the twelve datasets are reported in the last three rows. We mainly describe the key findings of multi-class datasets in Table 5-2.

**Comparison between MIFS and MRMR** The results in Table 5-2 show that, when the number of features is less than or equal to sixteen, there is no clear winner between MIFS and MRMR. Although, MIFS is slightly better than MRMR based on the average results of the twelve datasets, neither of the two methods are predominantly better than
Table 5-2. Comparison of classification accuracy for three feature selection methods on multi-class datasets. The three methods are MIFS, MRMR and SVM-RFE (denoted as RFE). The average results over twelve datasets are reported in the last three rows.

<table>
<thead>
<tr>
<th>DS</th>
<th>Method</th>
<th>Number of Features</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>4 8 16 40 60 80 100 150 250 500 862</td>
</tr>
<tr>
<td>poor4</td>
<td>MIFS</td>
<td>0.696 0.765 0.811 0.819 0.814 0.819 0.821 0.824 0.814 0.815</td>
</tr>
<tr>
<td></td>
<td>MRMR</td>
<td>0.734 0.772 0.778 0.794 0.791 0.799 0.814 0.814 0.819 0.802 0.809</td>
</tr>
<tr>
<td></td>
<td>RFE</td>
<td>0.567 0.644 0.681 0.706 0.746 0.771 0.794 0.814 0.821 0.821</td>
</tr>
<tr>
<td>poor6</td>
<td>MIFS</td>
<td>0.527 0.590 0.615 0.622 0.640 0.654 0.659 0.645 0.649 0.633</td>
</tr>
<tr>
<td></td>
<td>MRMR</td>
<td>0.542 0.576 0.588 0.589 0.581 0.596 0.61 0.596 0.610 0.635 0.633</td>
</tr>
<tr>
<td></td>
<td>RFE</td>
<td>0.337 0.370 0.431 0.531 0.551 0.564 0.578 0.593 0.608 0.635</td>
</tr>
<tr>
<td>poor8</td>
<td>MIFS</td>
<td>0.338 0.394 0.433 0.469 0.470 0.488 0.496 0.513 0.530 0.486</td>
</tr>
<tr>
<td></td>
<td>MRMR</td>
<td>0.335 0.408 0.454 0.467 0.469 0.482 0.470 0.474 0.489 0.465 0.472</td>
</tr>
<tr>
<td></td>
<td>RFE</td>
<td>0.259 0.274 0.303 0.390 0.423 0.435 0.457 0.456 0.456 0.475</td>
</tr>
<tr>
<td>fair4</td>
<td>MIFS</td>
<td>0.621 0.687 0.755 0.784 0.802 0.816 0.816 0.816 0.809 0.808 0.806</td>
</tr>
<tr>
<td></td>
<td>MRMR</td>
<td>0.598 0.685 0.728 0.777 0.796 0.789 0.784 0.777 0.783 0.786 0.798</td>
</tr>
<tr>
<td></td>
<td>RFE</td>
<td>0.466 0.527 0.608 0.693 0.753 0.753 0.771 0.786 0.787 0.806</td>
</tr>
<tr>
<td>fair6</td>
<td>MIFS</td>
<td>0.587 0.698 0.754 0.814 0.822 0.825 0.827 0.820 0.820 0.820 0.807</td>
</tr>
<tr>
<td></td>
<td>MRMR</td>
<td>0.593 0.698 0.767 0.772 0.786 0.807 0.802 0.807 0.801 0.804 0.792</td>
</tr>
<tr>
<td></td>
<td>RFE</td>
<td>0.504 0.640 0.696 0.761 0.775 0.780 0.781 0.780 0.797 0.816</td>
</tr>
<tr>
<td>fair8</td>
<td>MIFS</td>
<td>0.536 0.641 0.684 0.700 0.736 0.733 0.727 0.735 0.732 0.713</td>
</tr>
<tr>
<td></td>
<td>MRMR</td>
<td>0.540 0.653 0.681 0.721 0.707 0.712 0.715 0.704 0.698 0.695 0.720</td>
</tr>
<tr>
<td></td>
<td>RFE</td>
<td>0.398 0.528 0.616 0.677 0.687 0.688 0.702 0.700 0.701 0.709</td>
</tr>
<tr>
<td>good4</td>
<td>MIFS</td>
<td>0.586 0.673 0.763 0.773 0.782 0.78 0.783 0.774 0.778 0.767</td>
</tr>
<tr>
<td></td>
<td>MRMR</td>
<td>0.609 0.681 0.755 0.761 0.779 0.780 0.780 0.770 0.772 0.761 0.755</td>
</tr>
<tr>
<td></td>
<td>RFE</td>
<td>0.543 0.610 0.656 0.711 0.718 0.740 0.732 0.735 0.767 0.749</td>
</tr>
<tr>
<td>good6</td>
<td>MIFS</td>
<td>0.455 0.551 0.593 0.645 0.709 0.716 0.724 0.697 0.700 0.694</td>
</tr>
<tr>
<td></td>
<td>MRMR</td>
<td>0.427 0.532 0.621 0.667 0.680 0.690 0.677 0.687 0.675 0.664 0.696</td>
</tr>
<tr>
<td></td>
<td>RFE</td>
<td>0.339 0.437 0.517 0.597 0.638 0.653 0.660 0.682 0.674 0.698</td>
</tr>
<tr>
<td>good8</td>
<td>MIFS</td>
<td>0.373 0.477 0.567 0.659 0.674 0.676 0.665 0.673 0.666 0.655</td>
</tr>
<tr>
<td></td>
<td>MRMR</td>
<td>0.336 0.461 0.527 0.615 0.634 0.647 0.644 0.646 0.649 0.661 0.652</td>
</tr>
<tr>
<td></td>
<td>RFE</td>
<td>0.258 0.346 0.424 0.508 0.530 0.581 0.605 0.624 0.632 0.654</td>
</tr>
<tr>
<td>best4</td>
<td>MIFS</td>
<td>0.650 0.754 0.763 0.817 0.829 0.832 0.829 0.821 0.838 0.820</td>
</tr>
<tr>
<td></td>
<td>MRMR</td>
<td>0.667 0.757 0.775 0.785 0.789 0.793 0.798 0.791 0.784 0.802 0.803</td>
</tr>
<tr>
<td></td>
<td>RFE</td>
<td>0.596 0.659 0.708 0.753 0.766 0.789 0.776 0.791 0.803 0.817</td>
</tr>
<tr>
<td>best6</td>
<td>MIFS</td>
<td>0.497 0.568 0.699 0.731 0.767 0.765 0.763 0.770 0.750 0.755</td>
</tr>
<tr>
<td></td>
<td>MRMR</td>
<td>0.497 0.568 0.688 0.730 0.731 0.725 0.746 0.739 0.748 0.740 0.750</td>
</tr>
<tr>
<td></td>
<td>RFE</td>
<td>0.449 0.499 0.587 0.667 0.710 0.712 0.727 0.729 0.736 0.749</td>
</tr>
<tr>
<td>best8</td>
<td>MIFS</td>
<td>0.427 0.543 0.635 0.726 0.737 0.733 0.735 0.732 0.735 0.727</td>
</tr>
<tr>
<td></td>
<td>MRMR</td>
<td>0.434 0.563 0.652 0.704 0.700 0.714 0.712 0.700 0.693 0.704 0.707</td>
</tr>
<tr>
<td></td>
<td>RFE</td>
<td>0.342 0.429 0.532 0.641 0.648 0.687 0.694 0.723 0.719 0.724</td>
</tr>
<tr>
<td>Avg</td>
<td>MIFS</td>
<td>0.524 0.612 0.673 0.713 0.732 0.736 0.737 0.734 0.735 0.723</td>
</tr>
<tr>
<td></td>
<td>MRMR</td>
<td>0.518 0.606 0.664 0.696 0.702 0.709 0.710 0.707 0.707 0.706 0.716</td>
</tr>
<tr>
<td></td>
<td>RFE</td>
<td>0.422 0.497 0.563 0.636 0.662 0.679 0.69 0.700 0.708 0.721</td>
</tr>
</tbody>
</table>
other. However, when the number of features is greater than sixteen, MIFS outperforms MRMR in almost all cases. This is because, as the number of features increases, features that individually are not discriminating may increase the predictive power when combined with the selected features. Although, MRMR tries to address this deficiency of filter based method by incorporating minimum redundancy, it is inferior to the method described in this paper for CGH datasets. Further, If we compare the best predictive accuracy obtained for a given dataset (given in bold) by using MIFS to that of MRMR, we observe that MIFS always gives a better value.

**Comparison between MIFS and SVM-RFE** The results in Table 5-2 show that MIFS outperforms SVM-RFE in almost all cases. Clearly, as the number of features increases, the gap between MIFS and SVM-RFE drops. They become comparable in terms of predictive accuracy when the number of features reaches several hundreds (we do not report these results due to the space limitations). We believe that a forward scheme is better because it first adds the highest discriminating features followed by features that individually are not discriminating, but improve the classification accuracy when used in combination with the discriminating features. On the other hand, a backward elimination scheme (RFE) often selects ”redundant” features but excludes complementary features that individually do not discriminate the data well. This is exemplified by a simple example of a classification problem with three features $k_1$, $k_1$ and $k_2$. Feature $k_1$ works much better than $k_2$ in discriminating the data. Assume that we want to select two features. A typical RFE scheme will first remove $k_2$ because it influences objective function least. The two selected features would be $k_1$ and $k_1$. On the other hand, the proposed forward selection scheme (FS) will choose $k_1$ followed by $k_2$ because choosing another $k_1$ does not change the objective function at all. Therefore, the two features selected by FS scheme lead to a better predictive accuracy as compared to those selected by RFE scheme.
Table 5-3. Comparison of classification accuracy for three feature selection methods on two-class datasets. The three methods are MIFS, MRMR and SVM-RFE (denoted as RFE). The average results over four datasets are reported in the last three rows.

<table>
<thead>
<tr>
<th>DS</th>
<th>Method</th>
<th>Number of Features</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>poor2</td>
<td>MIFS</td>
<td>0.807</td>
</tr>
<tr>
<td></td>
<td>MRMR</td>
<td>0.791</td>
</tr>
<tr>
<td></td>
<td>RFE</td>
<td>0.775</td>
</tr>
<tr>
<td>fair2</td>
<td>MIFS</td>
<td>0.744</td>
</tr>
<tr>
<td></td>
<td>MRMR</td>
<td>0.741</td>
</tr>
<tr>
<td></td>
<td>RFE</td>
<td>0.675</td>
</tr>
<tr>
<td>good2</td>
<td>MIFS</td>
<td>0.818</td>
</tr>
<tr>
<td></td>
<td>MRMR</td>
<td>0.798</td>
</tr>
<tr>
<td></td>
<td>RFE</td>
<td>0.758</td>
</tr>
<tr>
<td>best2</td>
<td>MIFS</td>
<td>0.854</td>
</tr>
<tr>
<td></td>
<td>MRMR</td>
<td>0.852</td>
</tr>
<tr>
<td></td>
<td>RFE</td>
<td>0.793</td>
</tr>
<tr>
<td>Avg</td>
<td>MIFS</td>
<td>0.806</td>
</tr>
<tr>
<td></td>
<td>MRMR</td>
<td>0.795</td>
</tr>
<tr>
<td></td>
<td>RFE</td>
<td>0.750</td>
</tr>
</tbody>
</table>

Comparison on two-class datasets The results in Table 5-3 show that, unlike results in Table 5-2, although MIFS is slightly better than MRMR based on the average results of four datasets, there is no clear winner that beats the other for every one of the four datasets. This may indicate that MIFS and MRMR are comparable in terms of classification accuracy for two-class datasets. The results also show that MIFS outperforms SVM-RFE in most cases when number of features are less than 250. As the number of features increases, the gap between MIFS and SVM-RFE drops. They become comparable when number of features reaches 250. This consists with our conclusion on multi-class datasets.

Using MIFS for feature selection The results in Table 5-2 and Table 5-3 shows that using 40 features result in classification accuracy that is comparable to using all the features. Also, using 80 features derived from MIFS scheme results in comparable or better classification accuracy as compared to all the features. This is significant as beyond data reduction, the proposed scheme can lead to better classification. To support this hypothesis, we generated four new datasets using our dataset resampler. The
resulting four datasets (newds1 to newds4) contain 4, 5, 6 and 8 classes respectively. The number of samples in the four datasets are 508, 1021, 815 and 649. We applied the MIFS method over these datasets. We compare the classification accuracies obtained by using all 862 features to those using only 40 and 80 selected features. The results are shown in Table 5-4. These results substantiate our hypothesis that using around 40 features (roughly 5% of all features) can generate comparable accuracy to using all the features. Also, using around 80 features (roughly 10% of all the features) can result in comparable or better prediction than all the 862 features.

It is worth noting that the other two methods, typically have lower or comparable accuracy when a smaller number of features is used.

### 5.5.3 Consistency of Selected Features

To test the classification performance of a feature selection method, a multiple folds cross validation is usually used. In each fold, a set of features is selected based on the training examples. A classifier trained on training examples with the selected features is used to test the predictive accuracy of these features on testing examples. However, the feature sets selected in different folds are often different. A criterion is needed to evaluate how consistent the features are selected across multiple folds. This criterion is important because an algorithm selecting features with a low consistency may indicate that this algorithm is sensitive to the training examples and easily subject to overfitting. Further, consistently selected features help identify the most important chromosomal regions that are particularly relevant to cancers. In this section, we develop a novel measure called
Pairwise Maximum Matching (PMM) to evaluate the consistency of features selected across multiple folds for CGH data.

An important property of CGH data is that neighboring features are usually highly correlated as a pointlike genomic aberration can expand to the neighboring intervals. Due to the difference in the training examples, these highly correlated features can be alternatively selected in different folds. For example, assume that two sets of features are selected in two folds. The 53rd and 54th feature are only selected in the first and second set respectively. Although these two features are different, they should be considered matching because both the 53rd and 54th features are highly correlated and represent the same aberration pattern of interest.

We first define the correlation between two features. Given a set of $n$ CGH samples $\{x_1, x_2, \ldots, x_n\}$. Let $x_d^i$ denote the value (1, -1 or 0 for gain, loss or no aberration respectively) for the $i$th sample at the $d$th feature, $\forall d, 1 \leq d \leq D$, where $D$ is the number of genomic intervals. The number of samples that has aberrations at the $d$th feature can be computed as $B(d) = \sum_{i=1}^{n} |x_d^i|$. In principle, the correlation between neighboring features are caused by contiguous runs of gain or loss status. We use the term segment to represent a contiguous run of aberrations of the same type in a sample. Intuitively, two features are highly correlated when a large amount of segments intersect with both features. Let $x_i[u, v]$ denote a segment of $x_i$ that starts at the $u$th interval and ends at the $v$th interval, i.e. $\{x_u^i, \ldots, x_v^i\}$, such that $x_u^i = x_{u+1}^i = \cdots = x_v^i \neq 0$, $x_{u-1}^i \neq x_u^i$. Let $k$ and $k't \leq k, k't \leq D$ denote two features. We define that $C_i(k, k't) = 1$ if there exists a segment $x_i[u, v]$ in the $i$th sample that intersects with $k$ and $k't$, i.e. $u \leq k, k't \leq v$. The correlation coefficient between $k$ and $k't$ is computed as

$$Cor(k, k') = \frac{\sum_{i=1}^{n} C_i(k, k't)}{\max(B(k), B(k't))}.$$
The value of this coefficient is between $[0, 1]$. It identifies the fraction of segments that intersects with both two features. Given a user specified threshold $\epsilon$, two features $k$ and $k'$ are defined highly correlated if and only if $Cor(k, k') \geq \epsilon$.

Next, we explain how to evaluate the consistency of two sets of features. Let $K = \{k_1, \cdots, k_r\}$ and $K' = \{k'_1, \cdots, k'_r\}$ denote the two sets of $r$ features. We create a bipartite graph as follows. For each feature in the two sets, a vertex is added in the graph. For each feature $k_i, 1 \leq i \leq r$, if there exists a feature $k'_j$ such that $k'_j$ is highly correlated to $k_i$, e.g. $Cor(k_i, k'_j) \geq \epsilon$, an edge connecting the corresponding vertex of $k_i$ and $k'_j$ is added. Let $V_1$ and $V_2$ denote the set of vertices corresponding to $K$ and $K'$ respectively. It can be seen that every edge in the graph connects a vertex in $V_1$ and one in $V_2$. Therefore, the resulting graph is a bipartite graph. A maximum matching $M$ found in this graph is a set of edges that identify pairs of features (or highly correlated features) selected in both sets. We score the consistency between $K$ and $K'$ as $T(K, K') = \frac{|M|}{r}$, where $|M|$ denote the number of edges in $M$.

For multiple sets of features $\{K_1, \cdots, K_f\}$, The PMM measure is computed as the average score of each pair of feature sets:

$$PMM = \frac{2 \sum_{i=1}^{f} \sum_{j=i+1}^{f} T(K_i, K_j)}{f \times (f - 1)}$$

where $K_i$ and $K_j$ denote the $i$th and $j$th feature sets respectively.

We use the above approach to evaluate the consistency of features selected by three methods (MIFS, MRMR and SVMRFE) for the twelve multi-class datasets. For each dataset, each method selects five sets of features because a 5-fold cross validation is used. The PMM scores of the five sets of features selected by different methods on different datasets are reported in Table 5-5. The number of features are specified as 20, 50 and 100. The parameter $\epsilon$ is set to 0.8.

To show the significance of the PMM scores, a random test is performed as follows. For each dataset, five sets of features are randomly selected and the PMM score is
computed. The number of features and the value of parameter $\epsilon$ are exactly the same as above. This process is repeated for one million times. The mean value, the first percentile and ninety-ninth percentile of the one million scores are reported in Table 5-5.

The results show that both MRMR and MIFS outperforms SVMRFE considerably in terms of PMM scores. The PMM scores of MRMR is often slightly better than those of MIFS. Also, the PMM scores of both MRMR and MIFS are significantly greater than the ninety-ninth percentile of random scores. This indicates that the features selected by MRMR and MIFS in multiple folds are significantly consistent. On the other hand, the scores of SVMRFE are often within the range of the first and the ninety-ninth percentile of the random scores. This indicates that SVMRFE works poor in consistently selecting features in multiple folds. It is worth noting that the gap between MIFS or MRMR and random approach decreases as the number of features increases. This is because the more features are selected, the larger is the chance to find a pair of matching features in two random sets. As a result, the PMM scores of random approach increase too. Since the results show that using about 10% of all features already provides a good classification performance, we limit the comparison of PMM scores to small numbers of features (less than or equal to 100).

5.6 Conclusions

Recurrent chromosomal alterations provide cytological and molecular positions for the diagnosis and prognosis of Cancer. Comparative Genomic Hybridization (CGH) is one of the important mapping techniques that has been shown to be useful for understanding these alterations in cancerous cells.

In this chapter, we develop novel SVM based methods for classification and feature selection of CGH data. For classification, we developed a novel similarity kernel that is shown to be more effective than the standard linear kernel used in SVM. For feature selection, we propose a novel method based on the new kernel that iteratively selects features that provides the maximum benefit for classification. We compared our methods
Table 5-5. Comparison of PMM scores of three feature selection methods. Term \( r \), 99% and 1% denote the number of selected features, the ninety-ninth percentile and the first percentile respectively.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>( r )</th>
<th>SVMRFE</th>
<th>MRMR</th>
<th>MIFS</th>
<th>mean</th>
<th>99%</th>
<th>1%</th>
</tr>
</thead>
<tbody>
<tr>
<td>best4</td>
<td>20</td>
<td>0.32</td>
<td>0.87</td>
<td>0.68</td>
<td>0.29</td>
<td>0.41</td>
<td>0.19</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.51</td>
<td>0.91</td>
<td>0.82</td>
<td>0.49</td>
<td>0.57</td>
<td>0.42</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.65</td>
<td>0.91</td>
<td>0.86</td>
<td>0.64</td>
<td>0.69</td>
<td>0.58</td>
</tr>
<tr>
<td>best6</td>
<td>20</td>
<td>0.45</td>
<td>0.94</td>
<td>0.83</td>
<td>0.31</td>
<td>0.43</td>
<td>0.21</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.58</td>
<td>0.91</td>
<td>0.84</td>
<td>0.52</td>
<td>0.6</td>
<td>0.44</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.68</td>
<td>0.91</td>
<td>0.88</td>
<td>0.66</td>
<td>0.72</td>
<td>0.60</td>
</tr>
<tr>
<td>best8</td>
<td>20</td>
<td>0.41</td>
<td>0.93</td>
<td>0.71</td>
<td>0.31</td>
<td>0.43</td>
<td>0.20</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.49</td>
<td>0.91</td>
<td>0.85</td>
<td>0.51</td>
<td>0.6</td>
<td>0.41</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.67</td>
<td>0.90</td>
<td>0.87</td>
<td>0.65</td>
<td>0.72</td>
<td>0.60</td>
</tr>
<tr>
<td>fair4</td>
<td>20</td>
<td>0.28</td>
<td>0.82</td>
<td>0.83</td>
<td>0.22</td>
<td>0.31</td>
<td>0.13</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.44</td>
<td>0.86</td>
<td>0.76</td>
<td>0.40</td>
<td>0.47</td>
<td>0.33</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.54</td>
<td>0.88</td>
<td>0.81</td>
<td>0.57</td>
<td>0.62</td>
<td>0.52</td>
</tr>
<tr>
<td>fair6</td>
<td>20</td>
<td>0.46</td>
<td>0.86</td>
<td>0.81</td>
<td>0.25</td>
<td>0.37</td>
<td>0.16</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.54</td>
<td>0.87</td>
<td>0.81</td>
<td>0.44</td>
<td>0.54</td>
<td>0.35</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.63</td>
<td>0.91</td>
<td>0.83</td>
<td>0.59</td>
<td>0.65</td>
<td>0.54</td>
</tr>
<tr>
<td>fair8</td>
<td>20</td>
<td>0.47</td>
<td>0.82</td>
<td>0.77</td>
<td>0.25</td>
<td>0.36</td>
<td>0.14</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.53</td>
<td>0.85</td>
<td>0.80</td>
<td>0.45</td>
<td>0.52</td>
<td>0.36</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.68</td>
<td>0.88</td>
<td>0.84</td>
<td>0.60</td>
<td>0.67</td>
<td>0.53</td>
</tr>
<tr>
<td>good4</td>
<td>20</td>
<td>0.42</td>
<td>0.89</td>
<td>0.83</td>
<td>0.29</td>
<td>0.40</td>
<td>0.18</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.54</td>
<td>0.87</td>
<td>0.86</td>
<td>0.49</td>
<td>0.57</td>
<td>0.41</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.61</td>
<td>0.89</td>
<td>0.87</td>
<td>0.64</td>
<td>0.70</td>
<td>0.59</td>
</tr>
<tr>
<td>good6</td>
<td>20</td>
<td>0.34</td>
<td>0.84</td>
<td>0.76</td>
<td>0.31</td>
<td>0.41</td>
<td>0.20</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.52</td>
<td>0.89</td>
<td>0.75</td>
<td>0.51</td>
<td>0.59</td>
<td>0.42</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.68</td>
<td>0.90</td>
<td>0.88</td>
<td>0.65</td>
<td>0.71</td>
<td>0.58</td>
</tr>
<tr>
<td>good8</td>
<td>20</td>
<td>0.35</td>
<td>0.74</td>
<td>0.78</td>
<td>0.27</td>
<td>0.39</td>
<td>0.18</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.51</td>
<td>0.86</td>
<td>0.80</td>
<td>0.47</td>
<td>0.55</td>
<td>0.38</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.66</td>
<td>0.87</td>
<td>0.85</td>
<td>0.63</td>
<td>0.68</td>
<td>0.57</td>
</tr>
<tr>
<td>poor4</td>
<td>20</td>
<td>0.36</td>
<td>0.75</td>
<td>0.73</td>
<td>0.20</td>
<td>0.33</td>
<td>0.11</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.46</td>
<td>0.85</td>
<td>0.74</td>
<td>0.38</td>
<td>0.45</td>
<td>0.31</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.56</td>
<td>0.87</td>
<td>0.82</td>
<td>0.54</td>
<td>0.60</td>
<td>0.49</td>
</tr>
<tr>
<td>poor6</td>
<td>20</td>
<td>0.25</td>
<td>0.79</td>
<td>0.62</td>
<td>0.19</td>
<td>0.29</td>
<td>0.11</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.44</td>
<td>0.83</td>
<td>0.68</td>
<td>0.51</td>
<td>0.45</td>
<td>0.29</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.58</td>
<td>0.85</td>
<td>0.81</td>
<td>0.52</td>
<td>0.57</td>
<td>0.47</td>
</tr>
<tr>
<td>poor8</td>
<td>20</td>
<td>0.33</td>
<td>0.74</td>
<td>0.68</td>
<td>0.22</td>
<td>0.33</td>
<td>0.13</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>0.46</td>
<td>0.80</td>
<td>0.74</td>
<td>0.40</td>
<td>0.49</td>
<td>0.34</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.60</td>
<td>0.83</td>
<td>0.80</td>
<td>0.56</td>
<td>0.62</td>
<td>0.51</td>
</tr>
<tr>
<td>Average</td>
<td>50</td>
<td>0.50</td>
<td>0.87</td>
<td>0.79</td>
<td>0.45</td>
<td>0.54</td>
<td>0.37</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>0.63</td>
<td>0.88</td>
<td>0.84</td>
<td>0.61</td>
<td>0.66</td>
<td>0.55</td>
</tr>
</tbody>
</table>
against the best wrapper based and filter based approaches that have been used for feature selection of large dimensional biological data. Our results on datasets generated from the Progenetix database, suggests that our methods are considerably superior to existing methods. Further, unlike other methods proposed in the literature, our methods can improve the overall classification error by using a small fraction (around 10%) of all the features.
Cancer is classified into multiple histological types, each of which consists of multiple subtypes. Genomic aberrations may differ between histologically identical tumors (e.g. gastro-esophageal, depending on location [76]), different histological subtypes have different changes (e.g. in renal cell carcinomas,[35]), and different patterns may appear in the same histologic subtype (and may point towards different mechanisms; e.g. see complex re-arrangements vs. whole-chromosome gains/losses [52]). Tumor evolution process leaves characteristic signatures of inheritance along the pathways of progression and present a method to infer models of tumor progression by an identification of these signatures in genome-wide data of mutations [6]. Evidences have shown that patterns of recurrent Copy Number Alterations (CNAs) are observed for a broad range of cancers or subtypes of the same cancer.

To our knowledge, most existing works infer tumor progression models based on genetic events such as recurrent Copy Number Alterations (CNAs). Their models describe the evolutionary relationship between events and consequently expose the progression and development of tumors. However, most existing works focus on the progression of individual recurrent alterations. This approach leads to very complex models when multiple cancers are concerned, given that each cancer contains a set of recurrent alterations. A promising approach seems to consider the whole set of alterations of a cancer and infer a model based on the alteration patterns of different cancers. Such models effectively utilize the molecular characters of cancers and easily extend to large scale analysis. In this chapter, we have developed novel graph based computational methods that derive relationships within a histological type or between histological subtypes.
6.1 Preliminary

In this section, we briefly introduce some preliminary knowledge related to this work. In Section 6.1.1, we review the concept of markers that define the key recurrent CNAs in a cancer. In Section 6.1.2, we introduce an approach proposed by Bilke et al. which is extended later for inferring the progression of markers. In Section 6.1.3, we demonstrate a known tree fitting problem that infers progenetic models for cancers.

6.1.1 Marker Detection

Due to the correlation between neighboring genomic intervals [42], recurrent alterations usually accumulate together and forms a region of recurrent alterations, which we call recurrent region. Given a set of samples that belong to the same cancer, a marker is an independent key recurrent alteration representing a recurrent region. We proposed a dynamic programming algorithm to identify the best $R$ markers for a set of CGH cases. We demonstrated that our markers capture the aberration patterns well and improve the clustering of CGH cases [42].

Next, we briefly introduce some notations of markers. Each marker $m$ in a cancer is represented by two numbers $<p, q>$, where $p$ and $q$ denote the position and the aberration type respectively. The aberration type of a marker is either gain or loss, denoted by 1 or -1 respectively. Given a set $S$ of $N$ CGH cases $\{s_1, s_2, \ldots, s_N\}$. Let $x^j_d$ denote the alteration value (1, -1 or 0 for gain, loss or no aberration respectively) for case $j$ at the $d$th feature, $\forall d, 1 \leq d \leq D$, where $D$ is the number of genomic intervals. We use the term segment to represent a contiguous run of aberrations of the same type in a case. Let $s^j[u, v]$ be the segment of $s_j$ that starts at the $u$th interval and ends at the $v$th interval. Formally, $s^j[u, v]$ denotes a continuous run of intervals $\{x^j_u, \ldots, x^j_v\}$, for $c^j_s \leq u \leq v \leq c^j_e$, where $x^j_u = x^j_{u+1} = \ldots = x^j_v \neq 0$, $x^j_{u-1} \neq x^j_u$, $x^j_{v+1} \neq x^j_v$, $c^j_s$ and $c^j_e$ denote the starting and ending intervals of a chromosome in $s_j$ respectively.

Let $m = <p, q>$ be a marker. We denote the independent support of $s_j$ to $m$ as $\delta(s_j, m)$. Here, $\delta(s_j, m) = 1$ if and only if $x^j_p = q$. Otherwise, $\delta(s_j, m) = 0$. We define the
total independent support value of marker \( m_t \) as the sum of its support from all the cases. Formally, 
\[
S_{\text{upt}}(m) = \sum_{j=1}^{N} \delta(s_j, m).
\]
We will use term support to denote \( S_{\text{upt}}(m) \) in this paper. Please note that this support is not the same as what we proposed in our previous work for marker identification [42].

6.1.2 Tumor Progression Model

Bilke et al. proposed an approach of inferring a tumor progression model for Neuroblastoma (NB) with four different subtypes from CGH data [6]. They described the relationship between different subtypes based on the recurrent alterations shared by these subtypes. Their idea first identified a set of recurrent alterations. Each recurrent alteration belonged to one of the following three categories: common (shared by all the subtypes), shared (shared by two or more subtypes) and unique (distinct to only one subtype). They proposed a statistical model to identify recurrent alterations and compute the shared status of these alterations. Each shared status was a set of subtypes that contain this recurrent alteration.

The shared status of recurrent alterations can be described using a Venn diagram. For example, Figure 6-1 shows two Venn diagrams of two sets, represented by two overlapping circles. Let \( S_1 \) and \( S_2 \) denote the left and right circle respectively. There are three distinct areas (denoted as sections) marked by \( A, B \) and \( C \) in each Venn diagram. Each section represent a possible logical relationship between the two sets. For example, section \( A \) and \( C \) represent \( S_1 - S_2 \) and \( S_1 \cap S_2 \) respectively. A section is called non-empty if it contains some members. Each non-empty section is marked by a distinct color in Figure 6-1. The component of a non-empty section is defined to be the sets whose members are contained in this section. For example, the components of section \( A \) and \( C \) are \( \{S_1\} \) and \( \{S_1, S_2\} \) respectively. In general, the number of distinct sections \( S \) in a Venn diagram of \( K \) sets is given by 
\[
S = \sum_{n=1}^{K} \frac{K!}{(K-n)!n!}.
\]
which is also the number of different shared status of a recurrent alteration between \( K \) cancer subtypes. Since each section can be empty or non-empty, there are totally \( 2^S \) distinct Venn diagrams for \( K \) sets.
The authors built a Venn diagram of four sets for the four different subtypes of NB. Each shared status corresponds to a distinct section in this Venn diagram. By computing the shared status of each recurrent alteration, one can determine if a section in the Venn diagram is empty or not. As a result, the structure of the Venn diagram is determined. The authors proposed a graph model based on the structure of Venn diagram to infer the progression of four different subtypes of NB. The graph model satisfies the following three conditions:

1. All alterations found in a parent genotype must be present in the offspring with a similar frequency. The daughter generation acquires additional alterations.
2. Unobservable intermediate genotypes are possible, but the model with the smallest number of genotypes is utilized.
3. All genotypes arise from a common ancestor (i.e. the model has a root).

The resulting graph is a directed acyclic graph with each vertex corresponding to a non-empty section in the Venn diagram. An edge connects from a vertex $u$ to a vertex $v$ if (1) the set of cancer subtypes that contain the recurrent alterations of $u$ is a superset of that of $v$ and (2) there is no other vertex $w$ such that the set of cancer subtypes that contain the recurrent alterations of $w$ is a superset of that of $v$ and a subset of that of $u$.

The number of vertices in the resulting graph is bounded by $\min\{S, T\}$, where $T$ is the number of recurrent alterations. For example, the graph models corresponding to the two Venn diagrams in Figure 6-1 is shown on the right of the figure.

The authors demonstrate that, with the help of such a model, it is possible to identify trances of tumor progression in CGH data. However, their approach has several limitations.

- First, their methods of calculating the shared status of each recurrent alteration is very computational expensive. The time complexity is exponential to the number of cancers $K$. 

96
Figure 6-1. Examples of Venn diagram and corresponding graph model. Each Venn diagram (left) consists of two sets. The corresponding graph models are shown on the right. The three sections in the Venn diagram are denoted as $A$, $B$ and $C$ respectively. The main difference between example (a) and (b) is that, in example (b), section $A$ is empty, i.e. it contains no members. Therefore, the corresponding graph model of (b) consists of only two vertices, $C$ and $B$.

- This method can model the progression of markers. It, however, can not model the evolutionary relationship among different cancer types.

In addition to these limitations, Bilke et al do not provide a systematic algorithm for mapping the Venn diagram to the graph model automatically. These limitations make it impractical to use their method for large scale datasets composed of many cancers.

6.1.3 Tree Fitting Problem

Phylogenetics is one of the approaches commonly used to infer evolutionary relationships between genes or species of organisms. Central to most studies of phylogeny is the concept of a phylogenetic tree, typically a graphical representation of the evolutionary relationship among three or more genes or organisms [39].
A broad range of phylogenetic tree construction methods have been proposed. Among them, an important category is called distance matrix method. The tree construction problem of distance matrix method can be described as follows. Let \( L \) denote the set of samples and the set of real numbers. A distance matrix, \( D \), on \( L \) is a \( |L| \times |L| \) matrix, where each entry \( D(i, j) \) of this matrix denotes the distance between the \( i \)th and the \( j \)th sample based on a predefined distance function. Let \( T \) denote a phylogenetic tree built upon \( L \), \( T = (V, E) \).

Each leaf level node of \( T \) corresponds to a sample in set \( L \). Also, there is a node at the leaf level of \( T \) for each sample in \( L \). In other words, there is a bijection between the leaf level nodes and the samples in \( L \). The rest of the nodes in \( V \) are the internal nodes of the tree. Each edge in \( E \) is assigned a positive real number that denotes the weight of that edge. This is also termed the length of the edge in the literature. For any pair of leaf nodes \( i, j \in V \), define \( P_{ij} \) as the path in \( T \) between \( i \) to \( j \). The length of a path is the sum of the weights of the edges on that path. We create a new distance matrix \( D_T \), where each entry \( D_T(i, j) \) is the length of the path between \( i \) and \( j \). The distance matrix method aims the following: Given a distance matrix \( D \), find a tree \( T \) such that \( D_T \) is a good approximation to \( D \).

The tree fitting problem has been widely studied in molecular phylogenetics. Some of the leading distance matrix methods for tree construction include the unweighted-pair-group method with arithmetic mean (UPGMA) \([39]\) and Neighbor Joining \([39]\).

### 6.2 Progression Model for markers

In this section, we extend Bilke’s approach \([6]\) to infer progression models for markers of multiple cancer types. Markers are the independent key recurrent alterations that characterize the aberration pattern of a cancer type (Figure 4-2). Studies of the evolution of markers would be of obvious value to define gene loci relevant for the early diagnosis or treatment of cancer. It helps to answer questions about which marker tend to occur in many cancers, which markers are likely to occur together etc. The main difference between
our approach and the previous work is that we focus on markers instead of every recurrent alteration.

We compute the shared status of markers as follows. A marker identified in one cancer represent a recurrent alteration region in this cancer. However, for any two or more cancers containing the same recurrent region, they may not have markers identified at the same position due to the noise in the aberration patterns. Therefore, markers in different cancers representing the same recurrent region should be considered shared by these cancers.

First, we define the correlation between a marker and its neighboring intervals. Let \( C \) denote a set of cases belonging to the same cancer. Let \( m = < p, q > \) and \( d, 1 \leq d \leq D \) denote a marker in \( C \) and a genomic interval respectively. For each case \( s_j \in C \), we define \( E_j(d, m) = 1 \) if there exists a segment \( s_j[u, v] \) overlapping with both intervals \( d \) and \( p \), i.e. \( u \leq d, p \leq v \) and \( x^j_u = q \), otherwise, \( E_j(d, m) = 0 \). The function \( E_j(d, m) \) indicates that the alterations at \( d \) and \( p \) belong to the same segment in \( s_j \) and can be caused by the same point-like genomic alteration. We compute the correlation between \( d \) and \( m \) as
\[
Cor(d, m) = \frac{\sum_{j=1}^{|C|} E_j(d, m)}{\text{Supt}(m)}
\]
where \(|C|\) denotes the size of \( C \) and \( \text{Supt}(m) \) denotes the support value of marker \( m \) in \( C \). A large value of \( Cor(d, m) \) implies that intervals \( p \) and \( d \) belong to the same recurrent region that is represented by marker \( m \).

Next, we define that a marker \( m = < p, q > \) in cancer \( C_i \) is shared by \( C_j \) if and only if the following condition is reached: there is a marker \( m_t = < p_t, q_t > \) in cancer \( C_j \) such that \( q_t = q \) and \( Cor(p, m_t) > \epsilon \), where \( \epsilon \) is a user-defined threshold. The larger is the value of \( \epsilon \), the harder for a marker shared among multiple cancers. Intuitively, this definition indicates that a marker \( m_i \) in \( C_i \) is shared by another cancer \( C_j \) if and only if there exists a marker \( m_j \) in \( C_j \) such that \( m_j \) is highly correlated with \( m_i \) if \( m_i \) is also a marker in \( C_j \). To compute the shared status of a marker in \( C_i \), we visit every cancer other than \( C_i \). This makes the time complexity linear to the number of cancers \( K \). We denote
the shared status $S(m)$ of a marker $m$ as the set of cancers that share this marker, i.e. $S(m) \in \mathcal{P}\{C_1, \cdots, C_K\}$, where $\mathcal{P}$ denotes the power set operation.

We propose an algorithm that generates a progression model for $K$ cancers based on markers. Our algorithm consists of three steps:

- **First step:** We identify an optimal set of $R$ markers for each cancer using our marker identification program. These markers represent significant recurrent alterations specific to each cancer.

- **Second step:** For each marker in each cancer, we compute the shared status of this marker using the method we described above. Please note that markers in different cancers may have the same position and type. We treat these markers as a single marker and compute its shared status once.

- **Third step:** The logical relationship between $K$ cancers corresponds to a Venn diagram of $K$ sets. There are totally $S = \sum_{n=1}^{K} \frac{K!}{(K-n)!n!}$ distinct sections in this Venn diagram. Given a marker $m$ with shared status $S(m)$, the section corresponding to $S(m)$ is non-empty. We mark all the non-empty sections in the Venn diagram based on the shared status of all markers. We then convert the Venn diagram to a graph model as follows. We create a vertex $V$ for each non-empty section and associate it with the markers whose shared status corresponds to this section. We define the height of this vertex, denoted as $H(V)$, as the number of components in the corresponding section. We visit the vertices in the descending order of their heights. For each pair of vertices $V_i$ and $V_j$, $H(V_i) < H(V_j)$, we create an edge from $V_j$ to $V_i$ if both of the following conditions hold:
  1. The component set of the section corresponding to $V_i$ is a true subset of that of $V_j$.
  2. There is no other vertex $V_k$ such that the component set of the section corresponding to $V_k$ is a superset of that of $V_i$ and a subset of that of $V_j$.
We analyze the time complexity of this algorithm as follows. The time complexity of the first step is \( O(DNR) \) as analyzed in our previous work [42], where \( D \) and \( N \) denote the number of genomic intervals and number of cases of all \( K \) cancers respectively. The time complexity of the second step is \( O(TNR) \), where \( T \) is the cardinality of set consisting of the union of all markers. In the third step, the number of vertices is bounded by \( \min\{S, T\} \). Since \( T \leq K \times R \), the time complexity of this step is \( O(K^2R^2) \) in the worst case. Since we have \( D \geq T \), the overall time complexity is \( O(DNR) + O(K^2R^2) \).

In general, we have \( D \gg R, N \gg K^2 \), the overall time complexity can be written as \( O(DNR) \).

The graph created by our algorithm can be used to describe the hierarchical or evolutionary relationship between markers representing multiple stages between a single cancer type or among the markers of different cancer types. We term a node as a root node if it does not have any incoming edges. The nodes that are close to a root (there can be multiple roots) denote the aberrations that started in earlier stages. From this perspective, markers are not equally important. The markers that are parents of other markers in the hierarchical representation are common to multiple cancers. Thus, difference at parent marker positions should contribute more to the distance between different cancers than the child markers.

### 6.3 Progression Model for cancers

The aberration pattern defines the molecular characteristics of a cancer. We assume that cancers with similar aberration patterns are close to each other in the evolutionary history. The proper identification of the similarities between cancers will expose the underlying mechanism of cancer development and benefit the diagnosis and treatment of cancers.

Phylogenetic tree is a simple and efficient model that infers evolutionary relationship among multiple cancers. A key challenge of using existing distance matrix method for tree construction is to find a biologically meaningful distance function between
cancers. Next, we propose a novel measure for computing the distance between cancers based on their aberration patterns. Since markers are a set of recurrent alterations that characterize the aberration patterns of a cancer, our distance measure computes the distance between cancers based on their markers. Formally, let $C_i$ and $C_j$ denote two cancers. Let $M_i = \{m_{i,1}, \cdots, m_{i,R}\}$ and $M_j = \{m_{j,1}, \cdots, m_{j,R}\}$ denote the corresponding $R$ markers identified in $C_i$ and $C_j$ respectively, where $p_{i,1} < p_{i,2} < \cdots < p_{i,R}$ and $p_{j,1} < p_{j,2} < \cdots < p_{j,R}$. Please note that $p_{i,k}$ may not equal to $p_{j,k}$ for any $1 \leq k \leq R$. To compute the distance between $C_i$ and $C_j$, we first align the markers in $M_i$ to those in $M_j$. The goal of this alignment is to map $M_i$ and $M_j$ into two high dimensional vectors $\hat{M}_i$ and $\hat{M}_j \in \mathbb{R}^g$, where $g \leq 2R$ is the number of dimensions of the new vectors, such that the new vectors contain unified format of aligned markers in $C_i$ and $C_j$ respectively.

We say that a pair of markers $m_{i,k}$ and $m_{j,r}$ are overlapping if they satisfy either one of the following two conditions:

1. Both markers appear at the same interval and have the same type, i.e. $p_{i,k} = p_{j,r}$ and $q_{i,k} = q_{j,r}$

2. Both markers represent the same region of recurrent alterations, i.e. $Cor(p_{i,k}, m_{j,r}) > \epsilon$ and $Cor(p_{j,r}, m_{i,k}) > \epsilon$, where $\epsilon$ is a user-defined threshold.

In Section 6.2, we argue that markers are not equally important in the progression of cancers. A marker that is common to many cancers usually represents a fundamental characteristic of cancers. Therefore, we assume that markers shared by many cancers are more important than those shared by a few cancers. The intuition behind this reasoning can be explained as follows. A marker that triggers most of the cancers has survived the evolution of cancer progression with high likelihood. The markers that are cancer specific have most likely appeared later in the evolutionary history and created the underlying cancer alteration pattern. As a result, the deviation in genomic alterations corresponding to older markers corresponds to larger distance between two cancer types as the age of the genomic alteration increases. We incorporate this idea into the mapping process. We
assign weights to markers in each cancer. The weight of a marker is the number of cancers that share this marker. Let $W_i = \{w_{i,1}, \cdots, w_{i,R}\}$ and $W_j = \{w_{j,1}, \cdots, w_{j,R}\}$ be the vectors of weights for markers in $M_i$ and $M_j$. Here, $w_{i,k}$ and $w_{j,k}$ denote the weights the $k$th marker in $M_i$ and $M_j$.

The mapping process works as follows. Each time we pick up a pair of markers from $M_i$ and $M_j$. We add a pair of new dimensions to $\hat{M}_i$ and $\hat{M}_j$ respectively. The values of the added dimensions are determined by three attributes of markers: support, weight and type. Let $\Delta(m_{i,k}) = Supt(m_{i,k}) \times w_{i,k} \times q_{i,k}$. If the two markers are overlapping, the values added into $\hat{M}_i$ and $\hat{M}_j$ are $\Delta(m_{i,k})$ and $\Delta(m_{j,r})$ respectively. If two markers are not overlapping, we focus on the marker at a smaller genomic interval. Without loss of generality, we can assume $p_{i,k} < p_{j,r}$. There is no marker at interval $p_{i,k}$ in $C_j$. However, we need to compute the information of this interval across both cancers so that the difference of this interval can be taken into account. So we assume that there is a ”hypothetical” marker at $p_{i,k}$ in $C_j$. This marker is of the same type and weight as $m_{i,k}$. However, the support of this marker is computed based on the samples in $C_j$. Let $m' = (p', q')$ in $C_j$ denote this ”hypothetical” marker. We have $p' = p_{i,k}$, $q' = q_{i,k}$ and $w' = w_{i,k}$. Please note that $Supt(m')$ depends on the alteration pattern in $C_j$ and may not equal to $Supt(m_{i,k})$. We add the two values, $\Delta(m_{i,k})$ and $\Delta(m')$, into $\hat{M}_i$ and $\hat{M}_j$ respectively. Next, we choose another pair of markers and repeat the above procedure until all the markers have been processed.

The algorithm of the mapping process of two sets of markers is implemented as follows.

**Inputs:** $M_i = \{m_{i,1}, \cdots, m_{i,R}\}$ and $M_j = \{m_{j,1}, \cdots, m_{j,R}\}$ where $p_{i,1} < p_{i,2} < \cdots < p_{i,R}$ and $p_{j,1} < p_{j,2} < \cdots < p_{j,R}$. $W_i = \{w_{i,1}, \cdots, w_{i,R}\}$ and $W_j = \{w_{j,1}, \cdots, w_{j,R}\}$ are the vectors of weights for markers in $M_i$ and $M_j$

1. **Initialize:** $\hat{M}_i = \hat{M}_j = []; k = r = 1$;
2. **while** $k \leq R$ and $r \leq R$
(a) if $m_{i,k}$ and $m_{j,r}$ are overlapping
$$\hat{M}_i = [\hat{M}_i, \Delta(m_{i,k})]; \hat{M}_j = [\hat{M}_j, \Delta(m_{j,r})]; k = k + 1; r = r + 1;$$
(b) else if $p_{i,k} < p_{j,r}$
Create a "hypothetical" marker $m_t$ same as $m_{i,k}$ in $C_j$; $\hat{M}_i = [\hat{M}_i, \Delta(m_{i,k})]; \hat{M}_j = [\hat{M}_j, \Delta(m_t)]; k = k + 1$
(c) else if $p_{i,k} > p_{j,r}$
Create a "hypothetical" marker $m_t$ same as $m_{j,r}$ in $C_i$; $\hat{M}_i = [\hat{M}_i, \Delta(m_t)]; \hat{M}_j = [\hat{M}_j, \Delta(m_{j,r})]; r = r + 1$
(d) else
$$\hat{M}_i = [\hat{M}_i, \Delta(m_{i,k})]; \hat{M}_j = [\hat{M}_j, \Delta(m_{j,r})]; k = k + 1; r = r + 1$$

3. while $k \leq R$
Create a "hypothetical" marker $m_t$ same as $m_{i,k}$ in $C_j$; $\hat{M}_i = [\hat{M}_i, \Delta(m_{i,k})]; \hat{M}_j = [\hat{M}_j, \Delta(m_t)]; k = k + 1$

4. while $r \leq R$
Create a "hypothetical" marker $m_t$ same as $m_{j,r}$ in $C_i$; $\hat{M}_i = [\hat{M}_i, \Delta(m_t)]; \hat{M}_j = [\hat{M}_j, \Delta(m_{j,r})]; r = r + 1$

Outputs: $\hat{M}_i, \hat{M}_j$ Once we have the aligned vectors $\hat{M}_i$ and $\hat{M}_j$, we use Extended Jaccard coefficient $[79]$ to compute the similarity between the two vectors. Extended Jaccard coefficient is widely used as a similarity measure in vector spaces. It retains the sparsity property of the cosine similarity while allowing discrimination of collinear vectors. For example, given two vectors $\hat{M}_i = [0.1, 0.3]$ and $\hat{M}_j = [0.2, 0.6]$, the cosine similarity does not discriminate the difference between them and the similarity value is computed as 1. However, in our case, $\hat{M}_i$ and $\hat{M}_j$ are different because they denote recurrent alterations in $C_i$ and $C_j$ with different frequencies. The Extended Jaccard coefficient is computed as follows.

$$EJ(\hat{M}_i, \hat{M}_j) = \frac{\hat{M}_i \cdot \hat{M}_j}{\|\hat{M}_i\|^2 + \|\hat{M}_j\|^2 - \hat{M}_i \cdot \hat{M}_j}$$
The Extended Jaccard similarity of any two vectors is within the range of [0, 1]. It is easy to convert Extended Jaccard similarity to distance by subtracting it from 1, i.e. \( D(C_i, C_j) = 1 - EJ(\hat{M}_i, \hat{M}_j) \). We compute the distance \( D(C_i, C_j) \) for any \( 1 \leq i, j \leq R \). As a result, we construct the distance matrix for \( K \) cancers. We apply existing distance matrix method, such as UPGMA, to construct the phylogenetic tree.

### 6.4 Experimental Results

**Dataset:** With 15127 cases from 571 publications as of Dec 2007, Progenetix is the largest resource for published chromosomal CGH data [3] ([http://www.progenetix.net/](http://www.progenetix.net/)). For the purpose of this paper, we use a dataset with 5918 clearly malignant epithelial neoplasias (ICD-O-3 xxxx/2 and xxxx/3), a descriptive overview of which had been published previously [2]. From the biomedical perspective, this dataset could be divided into 22 clinico-pathological disease categories. Additional entities consisting of less than 40 cases each were summarily moved to an 'other' category.

As result of the Progenetix database format transformation, for each case the genomic imbalance status for 862 ordered intervals had been extracted from the karyotype annotation. This information represents the whole genome copy number status information, in the maximum resolution feasible for cytogenetic methods. The value of each interval is 1, -1 or 0, indicating the gain, loss and no change status. The target data set can be represented as a 2-dimensional matrix with 5918 rows, with 862 columns representing the imbalance status for each genomic interval. Additional columns contain clinical information categories.

Although these cases are important for the evaluation of overall genomic instability, due to our focus on aberration patterns 875 cases without any CNAs were deemed non-informative for our purposes and removed prior to further analysis. Also, the categories 'cholangio' and 'squamous skin' were removed due to the limited number of informative cases (11 and 15, respectively). We also excluded cases sub-summarized in
Table 6-1. Name and number of cases of each cancer in the dataset.

<table>
<thead>
<tr>
<th>Diagnosis</th>
<th>no. of cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>head-neck squamous cell carcinoma (HNSCC)</td>
<td>309</td>
</tr>
<tr>
<td>non-small cell lung carcinoma (NSCLC)</td>
<td>242</td>
</tr>
<tr>
<td>small cell lung carcinoma (SCLC)</td>
<td>63</td>
</tr>
<tr>
<td>bladder carcinoma</td>
<td>140</td>
</tr>
<tr>
<td>breast carcinoma</td>
<td>640</td>
</tr>
<tr>
<td>cervical carcinoma</td>
<td>210</td>
</tr>
<tr>
<td>colorectal adenocarcinoma (CRC)</td>
<td>392</td>
</tr>
<tr>
<td>esophagus carcinoma (ES)</td>
<td>206</td>
</tr>
<tr>
<td>gastric carcinoma</td>
<td>477</td>
</tr>
<tr>
<td>hepatocellular adenocarcinoma (HCC)</td>
<td>334</td>
</tr>
<tr>
<td>melanocytic (MEL)</td>
<td>81</td>
</tr>
<tr>
<td>nasopharynx carcinoma (NPC)</td>
<td>149</td>
</tr>
<tr>
<td>neuroendocrine ca. and carcinoid (NE)</td>
<td>114</td>
</tr>
<tr>
<td>ovarian carcinoma</td>
<td>388</td>
</tr>
<tr>
<td>pancreas adenocarcinoma (PAC)</td>
<td>64</td>
</tr>
<tr>
<td>prostate carcinoma</td>
<td>416</td>
</tr>
<tr>
<td>renal carcinoma (RCC)</td>
<td>163</td>
</tr>
<tr>
<td>thyroid carcinoma</td>
<td>154</td>
</tr>
<tr>
<td>uterus carcinoma</td>
<td>42</td>
</tr>
<tr>
<td>vulva carcinoma</td>
<td>47</td>
</tr>
</tbody>
</table>

the 'other' category (386 cases). The remaining 20 entities with 4631 cases are used for analysis in this paper. The details of the dataset is shown in Table 6-1.

**System specifications**: We developed our code using MATLAB and ran our experiment on a system with dual 2.59 GHz AMD Opteron Processors, 8 gigabytes of RAM, and a Linux operating system.

6.4.1 Results for Marker Models

In this experiment, we infer a progression model for markers using the dataset in Table 6-1. We perform each step one by one and discuss the results of each step as follows.

In the first step, we identify an optimal set of 20 markers for each cancer. Please note that we exclude 100 (peri) centromeric intervals because 1) they mostly consist of repetitive sequence (ALU repeats etc.) without encoding genes; 2) they have technical or interpretation difficulties. The markers are identified from the remaining 762 intervals.
An existing work by Baudis has identified the imbalance hot spots in clinico-pathological entities in the same dataset [2], using an ‘average profile’ based approach. We compared our markers to the reported imbalance hot spots for validation test. Due to the limitation of space, here we only present the comparison results for HNSCC disease category.

- Imbalance hot spots identified by Baudis [2]:
  
  gains: 3q26 (59.2%), 8q24 (40.8%), 11q13 (31.9%, many specific high-level), 5p (26.5%), Xq, 1q, 7q(21), 12p, 17
  
  losses: 3p (30.1%), 18q(22) (22.4%), 9p (22.4%), 11q24 (19.2%), 4, 5q, 8p, 13

- Markers identified by our method:
  
  gains: 3q26.2 (57.2%), 8q24.3 (41%), 11q13.4 (31.9%), 5p14.3 (26.5%), Xq28 (23%), 7q21.3 (20.9%), 12p13.1 (17.7%), 17q25.3 (17.7%), 20q12 (17.7%), 19p13.11 (16.8%), 1q31.3 (16.2%), 18p11.23 (15.9%)
  
  losses: 3p26.3 (30.7%), 18q23 (22.7%), 9p23 (22.4%), 11q25 (19.2%), 4p14 (18%), 5q21.3 (15.3%), 8p23.3 (16.2%), 13q21.33 (16.5%)

In the above results, markers or hot-spots are listed with detailed locus and frequency information. Gains and losses are evaluated separately. The hot-spots or markers are sorted in descending frequency of occurrence. We identify markers as individual intervals while Baudis identified the regional hot-spots from summary data. Our results are highly compatible to reported results if we consider a marker as a representative of a region. We successfully identify all the hot spots identified by Baudis. We also identify additional hotspots (e.g., 18q23) that has significant support.

In the second step, for each disease entity, we compute the shared status of each marker identified in this cancer using the method we described in Section 6.2. We set the threshold \( \epsilon \) to 0.8. To compare with the reported most frequent imbalances over all cancers, we analyze the markers that are in the same regions. The comparisons of imbalance with top frequencies are shown as follows.

- Most frequent imbalances reported by Baudis [2]:
+8q: ubiquitously high (exception NE and thyroid)

- Markers identified by our method and their shared status:
  +8q23.1, +8q23.2, +8q23.3: 19 cancers (exception thyroid)
  +8q24.13, +8q24.23, +8q24.3: 18 cancers (exception NE and thyroid)

- Most frequent imbalances reported by Baudis [2]:
  -13q: occurring in most carcinoma types (exception cholangio and SQS)

- Markers identified by our method and their shared status:
  -13q21.1, -13q21.2, -13q21.33: 18 cancers (exception CRC, gastric, cholangio and SQS)
  -13q22.3: 15 cancers (exception SCLC, CRC, prostate, thyroid, gastric, cholangio and SQS)

The results show that our approach discovers the most frequent markers in a consistent way to Baudis’ work. Please note that markers are individual intervals instead of chromosomal regions. Additionally to the markers reported by Baudis et al. as top-scorers in the different entities, our method detected other regions, for example +17q and +7p which both are shared by more than 12 cancers types.

In the third step, we build a graph model based on the shared status of markers. The model contains 119 vertices and 385 edges, which makes it hard to fit in this thesis. The model conveys useful information about the importance of markers. We use this information in our next experiments in Section 6.4.2.

### 6.4.2 Results for Phylogenetic Models

In this experiment, we infer progression models for cancers using the distance-based approach described in Section 6.3. We compute the distance matrix $D$ of 20 cancers in Table 6-1 based on the markers reported in Section 6.4.1. We use UPGMA algorithm in PHYLIP package [20] to generate the phylogenetic tree. To demonstrate the use of computing the distance between cancers based on the importance of markers, we generate two phylogenetic trees. For the first tree, we compute the distance matrix by assigning
Figure 6-2. Phylogenetic trees of 20 cancers based on weighted markers. The tree is generated by taking the importance of markers into account. We mark different cancers using different colors and capitalized letters based on their overall histological compositions. The legend is shown at top right side.
Figure 6-3. Phylogenetic trees of 20 cancers based on unweight markers. The tree is generated by giving equal weights to markers. We mark different cancers using different colors and capitalized letters based on their overall histological compositions.
the weight of each marker as the number of cancers that share this marker. The resulting
tree is shown in Figure 6-2. For the second tree, we compute the the distance matrix by
assigning the weight of each marker to 1. The resulting tree is shown in Figure 6-3.

The leaf nodes of the trees correspond to cancers (e.g. clinico-pathological cancer
entities). We mark these cancers using different colors as well as capitalized letters based
on the major histological composition of cases in this cancer. Each color corresponds to
a capitalized letter. Different colors (letters) encode different histological compositions of
cancers. The internal nodes are denoted by numbers and represent hypothetical cancers.
Since these intermediate cancers may contain daughter branches from completely different
histological, they have to be viewed as as common biological feature sets rather than truly
occurring clinico-pathological cancer entities. The lengths of the branch are proportional
to the difference between pairs of neighboring nodes.

In both trees, some cancers of the same histological composition are closely organized
in the same subtree. However, the tree in Figure 6-2 shows a higher correlation of
histological composition and subtree assignment compared to the tree in Figure 6-3.
This correlation would be in concordance with the view that cancer clones may arise
from tissue-specific cancer stem cells [66], with a similar regulatory program targeted by
genomic aberrations in related tissues.

Each clinico-pathological cancer entity may contain multiple subtypes with heterogeneous
aberration patterns. To infer a progression model for cancer subtypes, we first divide
each cancer into several (two or four) clusters. By doing this, we hope that cases with
similar aberration patterns can be grouped in the same cluster. We use RSim clustering
method [42] for this purpose. We determine the number of clusters by visually inspecting
the aberration patterns of the cancer. For each cluster of the same cancer, we compute
its quantity of imbalance as the ratio of intervals with imbalances in all CGH cases. We
sort the clusters of each cancer in the ascending order of their imbalance quantities. We
name each cluster by concatenating the cancer name and its ranks. For example, if we
divide HNSCC into four clusters, ‘HNSCC1’ and ‘HNSCC4’ are clusters with the least and most quantity of imbalance respectively. Please note that we do not perform clustering on entities uterus and vulva because they both contain less than 50 cases. As a result, we divide the 20 cancers into 58 clusters of cases. We compute the distance matrix $D$ for the 58 clusters based on the weighted markers in them. We apply UPGMA to construct a phylogenetic tree over these cancer clusters. A part of the tree is shown in Figure 6-4.

Figure 6-4 shows a fraction of the phylogenetic tree (left side) for 58 clusters. This fraction contains a subtree whose seven leaf nodes correspond to the cluster of six different cancer types. The name of these clusters all ends with 3 or 4, which indicates that cases in these clusters contain a large amount of imbalances. (SCLC2 is also the cluster with the highest quantity of imbalance in SCLC because SCLC only contains two clusters.)
The plots of the seven clusters are on the middle and in the same order as the leaf nodes. The X-axis and Y-axis denote the index of genomic intervals and cases respectively. Those intervals with gain and loss imbalance are plotted in green and red respectively. The markers in each cluster are denoted by vertical dotted lines. By visually inspecting the plots, we observe that these seven clusters are indeed similar in their aberration patterns. For example, many cases present loss aberrations around intervals 150, 200, 600 and 750 and gain aberrations around intervals 180, 240 and 400 etc. In contrast to our overall observation of a high concordance of histological origin and marker profile, the histological composition of these clusters is varying and includes small cell carcinomas, adenocarcinomas and squamous cell carcinomas.

6.5 Conclusions

While the computational analysis of genomic imbalance profiles has led to evolutionary models for aberrations in single cancer entities, a large scale analysis across heterogeneous cancer types remains a challenging subject. Recently, the descriptive analysis of oncogenic summary data was able to point towards a concordance of imbalance profiles from entities from similar histological categories. However, the analysis of average imbalance profiles will not be able to capture the diversity of aberration complexity in the different entities.

We have developed an automatic method to infer a graph model for the markers of multiple cancers. We demonstrated the use of this model in determining the importance of markers in cancer evolution. We also developed a new method to measure the evolutionary distance between different cancers based on their markers. We used this measure to create an evolutionary tree for multiple cancers.

With the application of our modeling approach to a set of more than 4600 epithelial neoplasias (carcinomas) with genomic imbalances, we can draw some preliminary conclusions:
• Marker determination and marker dependent subset generation are powerful tools for structuring large CGH data sets.

• Phylogenetic modeling of 58 cancer subtypes with unique genomic marker sets shows a high concordance between branch association and histological subtype.

• Cancer subtypes with a high level of genomic instability have overall similar imbalance patterns, which may reflect their origin from earlier, less determined progenitor cells and/or tissue independent mechanisms responsible for high-order genomic instability.

While our approach as described here used rough histological group classification as a reference, a refined data set combined with different reference qualities (e.g. clinical parameters) should provide a significant contribution to the overall perception of genomic instability in cancer development.
CHAPTER 7
A WEB SERVER FOR MINING CGH DATA

Data mining analysis on a large number of CGH samples helps biologist understand the intrinsic mechanism of tumor progression. For example, clustering methods are often employed to discover previously unknown sub-categories of cancer and to identify genetic biomarkers associated with the differentiation. Accurate classification of patients to their cancer types based on their genomic imbalances is crucial in successful cancer diagnosis and treatment. A public tool for data mining analysis of CGH data is of great use to cancer studies.

An ideal tool for end-users and large-volume data analysis is a web-based application: a web browser is the only client software. End users do not require to download and install extra software. In addition, the backend server can distribute the intensive computing jobs to clusters or multicore CPUs to reduce the execution time. In this chapter, we discuss a web application developed based on our previous work [42, 78] to fulfill these requirements.

7.1 Software Environment

We have developed a web based data mining tool for mining CGH data using our algorithms for clustering, marker detection and marker selection. It has the following features:

• It allows data import from tab-delimited text files in Progenetix (http://www.progenetix.de) format
• It can perform data clustering using multiple algorithms and distance measures, perform detection of important markers and perform selection of discriminative markers that help build a reliable classifier
• It provides results in textual and Graphical formats
• It provides multiple metrics for evaluation of results

The application is developed using Microsoft Internet Information Server (http://www.microsoft.com). The web user interface is developed using Microsoft ASP.NET
and C# (http://www.microsoft.com). When a user submit a request, the front-end C# program calls the executable files to perform the computation. The results are written to HTML files stored at the server side. The front-end program polls these results and return them to users.

The underlying algorithms are developed using MATLAB (http://www.mathworks.com). A MATLAB compiler is used (http://www.mathworks.com/products/compiler/) to generate executable files.

A preliminary version of this tool is available at (http://128.227.162.207:8007/CGH/Default.html). New features and algorithms are constantly being added to this tool.

### 7.2 Example: Distance-Based Clustering of Sample Dataset

In this section, we briefly describe the clustering of a small dataset using our web based tool to demonstrate its functionality.

The programs accepts tab-delimited text files with both CGH data and genomic interval information. We follow the format of Progenetix, the largest source for published CGH data (with more than 12,000 cases) (http://www.progenetix.de). The Progenetix format is a chromosomal band specific matrix suitable for mining experiments. It currently supports a resolution of 862 genomic intervals from 23 chromosomes. Data files consist of a header row followed by rows of data. Progenetix also provides online tools that can convert other formats of CGH data to the matrix format.

Our web tool provides five distance based clustering algorithms: topDown, bottomUp, $k$-means, topDown+$k$-means and bottomUp+$k$-means. The first three are well known clustering algorithms in the literature. The last two algorithms are the combination of topDown or bottomUp with $k$-means. They work as follows. They first find clusters using the top down (or bottom up) clustering algorithm. They, then, feed these clusters into $k$-means algorithm as the initial cluster. They aim to avoid the poor results obtained by the $k$-means algorithm due to the random initial clusters. The distance-based clustering...
tool provides four distance measures, Raw, Sim, cosineGaps and cosineNoGaps as described in Chapter refchap:paper1.

Users can choose any one of the 20 combinations of the five algorithms and four distance measures as shown in Figure 7-1. The default algorithm and distance measure is topDown and Sim respectively because this combination produces the best clusters according to our experimental results. The interface allows the user to upload a database file containing all the samples. It also allows the user to specify the number of clusters. Clustering is usually a computational intensive task. Depending on the database, this process can take several minutes. The web server allows users to provide their email addresses so that they can be notified when the results are ready. The server stores the results in a temporary html file and emails the link for this file. The user can browse this file later. If the user chooses to keep the browser open, the server automatically refreshes it with the page that contains the results.
Figure 7-2. Snapshot of the results of distance-based clustering.

Figure 7-2 provides the snapshot of the resulting page after applying the clustering algorithm to the sample dataset. In this example the dataset contains 391 samples from two cancer types, namely Retinoblastoma, NOS (ICD-O code = 9510/3) and Neuroblastoma, NOS (ICD-O code = 9500/3). The numbers of samples in the two cancer types are 120 and 271 respectively. The server identifies the clusters using the parameters provided by the user. It also provides clustering by applying the centroid shrinking technique [80] to refine the clusters. The results page consists of three parts.

1. **Quality of clustering.** We provide a table that reports the quality of the clusters according to a number of internal and external measures [79].

2. **Cluster membership.** We list the index of samples that belong to each cluster. We also provide a link to a downloadable text file that contains this information.

3. **Cluster Plot.** We plot the clusters as JPEG files and embed them in the results page as thumbnails. Users can click the thumbnails to view the full-size pictures. In the plot shown in Figure 7-3, the X-axis denotes the index of genomic intervals, the Y-axis denotes the samples that grouped by clusters. Different clusters are separated
Figure 7-3. Snapshot of the plot of clusters.

by the horizontal lines. The genomic intervals with gain and loss imbalances are plotted in green and red, respectively. Those with no imbalance are not plotted.

In both the query and result pages, all field names are clickable. Clicking on a field name brings the help page that contains a description of that field.

7.3 Conclusion

Our web server employs novel data mining methodologies for clustering and classification of CGH datasets as well as algorithms for identifying important markers that are potentially cancer signatures. It also provides a visualization of the dataset and the results. The developed software will help in understanding the relationships between genomic aberrations and cancer types.
CHAPTER 8
CONCLUSION

Comparative Genomic Hybridization (CGH) is a molecular-cytogenetic analysis method for simultaneous detecting of a number of chromosomal imbalances, which are one of the most prominent and pathogenetically relevant features of human cancer. Along with the high dimensionality (around 1000), a key feature of CGH data is that the consecutive value are highly correlated. The aim of this thesis is to develop novel data mining methods that exploit these characteristics in mining a population of CGH samples. In particular, this thesis has following contributions:

1. Novel distance measures are investigated for the clustering of CGH data. Three pairwise distance/similarity measures, namely raw, cosine, and sim, are proposed. The first one ignores the correlation, while the latter two can effectively leverage this correlation. These distance/similarity measures are tested on CGH data using three main clustering techniques. The results show that Sim consistently performs better than the remaining measures since it can effectively utilize the correlations between consecutive intervals in the underlying data.

2. A dynamic programming algorithm is developed to identify a small set of important genomic intervals called markers. The recurrent imbalance profiles of samples can be captured using a set of markers. Two novel clustering strategies are developed. Both methods utilize markers to exclude noisy intervals from clustering. The experimental results demonstrate that the markers found represent the aberration patterns of CGH data very well and they improve the quality of clustering significantly.

3. Novel SVM based methods for classification and feature selection of CGH data are developed. For classification, a novel similarity kernel is proposed. It is shown to be more effective than the standard linear kernel used in SVM. For feature selection, a novel method based on the new kernel is proposed. It iteratively selects features that provides the maximum benefit for classification. Our methods are compared against
the state-of-the-art wrapper and filter methods that have been used for feature selection of large dimensional biological data. Our results on datasets generated from the Progenetix database, suggests that our methods are considerably superior to existing methods.

4. A graph model is proposed to infer the progression of markers (key recurrent CNAs). With this model, the importance of markers in cancer evolution can be derived. A new distance measure is proposed for computing the distance between cancers based on their aberration patterns. Existing distance matrix method is employed along with the new measure for inferring progression model of multiple cancers. The results show that cancers with similar histological compositions are well grouped together.

These methods are evaluated using large repositories of datasets that are publicly available. These methods are also encapsulated into a web service that can be used for analyzing and visualizing CGH data.

In the present study, our work is based on chromosomal CGH data annotated in a reverse in-situ karyotype format [50]. In the future, we will extend our work to support other CGH formats, such as aCGH data, and other datasets such as Gene expression array data, SNP data and proteomics data.
REFERENCES


126


BIOGRAPHICAL SKETCH

Jun Liu was born in 1976 in Nanjing, China. He grew up mostly in Nanjing, China. He earned his B.S. and M.E. degrees in computer science from Nanjing University in 1998 and 2000, respectively. He earned his Ph.D. in computer engineering from the University of Florida (Gainesville, Florida, USA) in 2008.