CHAPTER 2
Pre-requisites & Formulation of the problems
Chapter 2

Pre-requisites and general formulation of the problems

2.1 Introduction

Chapter 2 deals with a detailed presentation with regard to the general formulations of the problems discussed in the thesis along with a brief description with regard to the software tools used for the analysis and of the methodologies employed for the research are also presented. The thesis comprises nine chapters and the brief description of the problems investigated is as follows:

Chapter 1, deals with a detailed introduction to the subject of research, together with scope and objectives and detailed literature survey pertaining to the topic.

Chapter 2, Pre-requisites & general formulation of the problem

2.2 General formulation of the problem

In this thesis there are basically nine chapters and eleven models are discussed. The first chapter deals with the Introduction and the second chapter with the pre-requisites and the general formulation of the problem. The formulation is as follows:

Chapter 3: Genetic Microarray Methods and Applications

Three models are discussed in this chapter:

Model 1: DNA Microarray and its Fabrication technique

The following three aspects are discussed in this model:

1. DNA Microarrays Technology
2. Fabrication Technique
3. Photolithography technique

Model 2: Gridding methods for microarray images

Input data: Microarray image from Stanford Microarray Database (SMD).

Image name: MicroArraySlide.jpg

Methodology:

1. Mean Horizontal Profile
2. Autocorrelation
3. Segmentation the peaks
4. Segmentation using threshold
5. Spot intensity and expression level

Tools used: MATLAB R2012a(7.14.0.739-32 bit) using Image Processing Toolbox

MATLAB is a high-level language and interactive environment for numerical computation, visualization, and programming. Using MATLAB, you can analyze data, develop algorithms, and create models and applications. The language, tools, and built-in math functions enable you to explore multiple approaches and reach a solution faster than with spreadsheets or traditional programming languages, such as C/C++ or Java. You can use MATLAB for a range of applications, including signal processing and communications, image and video processing, control systems, test and measurement, computational finance, and computational biology. More than a million engineers and scientists in industry and academia use MATLAB, the language of technical computing.

Key Features
- High-level language for numerical computation, visualization, and application development
- Interactive environment for iterative exploration, design, and problem solving
- Mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, numerical integration, and solving ordinary differential equations
- Built-in graphics for visualizing data and tools for creating custom plots
- Development tools for improving code quality and maintainability and maximizing performance
- Tools for building applications with custom graphical interfaces
- Functions for integrating MATLAB based algorithms with external applications and languages such as C, Java, .NET, and Microsoft® Excel®

Image Processing Toolbox provides a comprehensive set of reference-standard algorithms, functions, and apps for image processing, analysis, visualization, and algorithm development. You can perform image analysis, image segmentation, image enhancement, noise reduction, geometric transformations, and image registration. Many toolbox functions support multicore processors, GPUs, and C-code generation. Image Processing Toolbox supports a diverse set of image types, including high dynamic range, giga pixel resolution, embedded ICC profile, and tomographic.
Visualization functions and apps let you explore images and videos, examine a region of pixels, adjust color and contrast, create contours or histograms, and manipulate regions of interest (ROIs). The toolbox supports workflows for processing, displaying, and navigating large images.

**Key Features**

- Image analysis, including segmentation, morphology, statistics and measurement
- Image enhancement and filtering
- Geometric transformations and intensity-based image registration methods
- Image transforms, including FFT, DCT, Radon, and fan-beam projection
- Large image workflows, including block processing, tiling, and multi resolution
- Visualization apps, including Image Viewer and Video Viewer
- Multicore- and GPU-enabled functions, and C-code generation support

**Model 3: Gene Regulatory model**

**Case 1: Computational Method for the Reconstruction of Gene Regulatory Network via Microarray data.**

Reconstruction of gene regulatory network using microarray data. Evaluation of the performance of our approach which is experimentally tested on the oesophageal dataset.

**Input data:** The full data set can be downloaded from the Gene Expression Omnibus website: [http://www.ncbi.nlm.nih.gov/geo/GSE2144](http://www.ncbi.nlm.nih.gov/geo/GSE2144), with 12810 genes and 10 samples.

The main modules involve computation of interaction in the filtered genes are:

1. Discretization mapping function
2. Gene-Gene mapping function
3. Filtering functions
4. Gene-gene interaction
5. Network Analysis
6. Grid Analysis

**Tools used:** MATLAB (R2012a-7.14.0.739-32 bit) and Cytoscape 3.0.2 for network analysis.

The problem is implemented using MATLAB to obtain Gene-Gene interaction matrix. The resultant matrix is imported to Cytoscape through Excel.
Cytoscape is an open-source community software project for integrating biomolecular interaction networks with high-throughput expression data and other molecular state information for visualization and analysis. Although applicable to any system of molecular components and interactions, Cytoscape is most powerful when used in conjunction with large databases of protein-protein, protein-DNA, and genetic interactions that are increasingly available for humans and model organisms. A software “Core” provides basic functionality to layout and query the network; to visually integrate the network with expression profiles, phenotypes, and other molecular state information; and to link the network to databases of functional annotations. The Core is extensible through a plug-in architecture, allowing rapid development of additional computational analyses and features. The central organizing metaphor of Cytoscape is a network (graph), with genes, proteins, and molecules represented as nodes and interactions represented as links, i.e. edges, between nodes.

In this chapter Cytoscape is used for the following approach:

b. Layout analysis
c. Network Analysis
d. Degree analysis using Cytocluster Apps(plugin’s)
e. Drug analysis


Methodology:

1. Spot intensity of Red and Green image
2. Expression level of each spot.
3. Discretization mapping function
4. Gene-Gene mapping function
5. Filtering functions
6. Gene-gene interaction
7. Network analysis

Tools used: MATLAB (R2012a-7.14.0.739-32 bit) and Cytoscape 3.0.2 for network analysis.
Implementation is done using **MATLAB Image processing toolbox** to obtain Gene-Gene interaction matrix. The resultant matrix is imported to Cytoscape through Excel.

**Cytoscape is used for the following approach:**


b. Network Analysis

**Case 3: Reconstruction of gene regulatory network to identify prognostic molecular markers of the reactive stroma of breast and prostate cancer using information theoretic approach.**

Reconstruction of gene regulatory networks is a process of identifying gene interaction networks from experimental microarray gene expression profile through computation techniques. In this chapter, cancer-specific gene regulatory network has been reconstructed using **Information theoretic approach-Mutual Information**. The microarray database used contains 12 Gene samples each of **breast cancer** and **prostate cancer** having both normal and tumor cell information.

**Input data:** The full data set can be downloaded from the Gene Expression Omnibus website: [http://www.ncbi.nlm.nih.gov/geo/GSE26910](http://www.ncbi.nlm.nih.gov/geo/GSE26910), the dataset has information on 54765 genes under 24 different experimental conditions.

The problem is implemented using **MATLAB** to obtain Gene-Gene interaction matrix. The resultant matrix is imported to Cytoscape through Excel.

**In this chapter Cytoscape is used for the following approach:**


b. Network Analysis

c. Degree analysis using Cytocluster Apps(plugin’s)

d. Identify prognostic molecular markers

**Chapter 4: Some Challenge tasks of medical Image Processing**

**Model 1: A Modified histogram thresholding method for brain tumor detection and extraction from MRI images.**

**Methodology used are:**

This chapter concentrates on three aspects: (i). First part of the present addresses the problem of detecting the position of the tumour, that is, whether the tumour is on the left or right side of the brain. This is achieved just with the information of which part of the brain contains more numbers of the pixels whose intensity is around 255. (ii).
Second part is to extract the tumour through segmentation using histogram thresholding and. (iii). Third part performs statistical analysis on the segmented image and dimension of the tumour.

**Tools Used:** MATLAB using Image Processing Toolbox

**Model 2: A Comparative study of Different segmentation Technique for Brain tumor detection.**

Brain tumour detection is one of the challenging tasks in medical image processing. The chapter discusses in detail the segmentation process by means of the following methods:

a. Histogram clustering
b. Global thersholding
c. Watershed segmentation
d. Edge based segmentation.

Six MRI images from Radiologists were collected and the experiments were conducted for statistical analysis also, a comparative study is made.

**Chapter 5: Improvement of Traditional k-means Algorithm through the Regulation of Distance Metric parameters**

**Algorithm & approach used:**

1. K-Means Algorithm
2. Manhattan K-Means Algorithm
3. Minkowsky K-Means Algorithm(Min-Kmeans)
4. Chevbychev K-Means Algorithm (Chev-Kmeans)

**Pseudo code for k-mean algorithm:**

1. Initialize k cluster centers, \( c_k \)
2. For each \( x(i) \), assign cluster with closest center, \( x(i) \) assigned \( k=\text{arg min} \ d(x,c_k) \).
3. For each cluster, recompute center:
4. Check convergence (have cluster centers moved?)
5. If not converged, go to 2.

**Tools Used:** Design and implementation of our algorithm, using MATLAB
Chapter 6.0: Multiclass Tumor classification by using SVM classifiers

This chapter presents brain tumour classification of five classes using SVM classifier. The classes considered are Medulloblastoma, Malignant glioma, Normal cerebellum and PNET from the data set Brain_tumor1.mat), which includes 90 sample, each sample has 5920 genes. From the experiment conduct in Matlab using SVM classifier, the result was found to be Normal cerebellum has sensitivity of 1, specificity of 0.5000 and prevalence of 95.6%

Methodology: Support Vector Machine (SVM) classifier

Tools Used: Design and implementation is done using MATLAB

Chapter 7.0: Classification models for image-segmentation data

7.1: A comparative study of different classifiers on Image-segmentation data.

Algorithm for tree based classification problem:

a. Decision tree

TreeGrowth(E,F)

If stopping_code(E,F)=true then

leaf=createnode();

leaf.label=classify(E)

return leaf

else

root=createnode()

root.test_cond=find_best_split(E,F)

let V={v|v is a possible outcome of root.test_cond}

for each vєV do

Ev={e|root.test_cond(e)=v and e ε E}

child=Treegrowth(Ev,F)

add child as descendent of root and label the edge (root→child) as v.

end for

end if

return root

createnode(), function extends the decision tree by creating a new node. A node in the decision tree has either a test condition, denote as node.test_cond or a class label,
denoted as `node.label.find_best_split()`, function determines which attribute should be selected as the test condition for splitting the training records.

`classify()`, function determines the class label to be assigned to a leaf node. For each leaf node t, let p(i|t) denote the fraction of training records from class I associated with the node t.

`stopping_cond()`, function is used to terminate the tree-growing process by testing whether all the records have either the same class label or the same attribute values.

b. Perceptron learning algorithm

Let D={(x_i, y_i)|i=1,2,.....N} be the set of training examples.

Initialize the weight vector with random values, w(0)

Repeat

   For each training example (x_i, y_i) in D do
   
   Compute the predicted output y_i(k)
   
   For each weight w_j do
   
   Update the weight w_j(k+1)
   
   End for

End for

Until stopping condition is met

**Algorithm used:**

1. Decision Stump
2. Random Forest
3. REPTree
4. Simple Cart
5. Logistic Model Tree
6. LADTree
7. RandomTree
8. BayesNet
9. NaiveBayes
10. NaiveBayesUpdatable
11. Multilayer Perceptron

**Tools Used:** WEKA 3.9

**WEKA**, formally called Waikato Environment for Knowledge Learning, is a computer program that was developed at the University of Waikato in New Zealand
for the purpose of identifying information from raw data gathered from agricultural domains.

WEKA supports many different standard data mining tasks such as data preprocessing, classification, clustering, regression, visualization and feature selection. The basic premise of the application is to utilize a computer application that can be trained to perform machine learning capabilities and derive useful information in the form of trends and patterns. WEKA is an open source application that is freely available under the GNU general public license agreement. Originally written in C the WEKA application has been completely rewritten in Java and is compatible with almost every computing platform. It is user friendly with a graphical interface that allows for quick set up and operation. WEKA operates on the predication that the user data is available as a flat file or relation, this means that each data object is described by a fixed number of attributes that usually are of a specific type, normal alpha-numeric or numeric values. The WEKA application allows novice users a tool to identify hidden information from database and file systems with simple to use options and visual interfaces.

In this chapter we have used Tree based and Network based classifier algorithm for analysis and evaluation.

7.2: Knowledge Discovery in Image-segmentation data set using Decision tree classifiers

Methods for Generating Trees:

a. NBTree
b. BFTree
c. J48
d. J48graft
e. Functional Tree(FT)

Tools Used: WEKA 3.9

Chapter 8.0: Knowledge Discovery process in the Image-segmentation data

Algorithms Used:

a. ConjunctiveRule
b. JRip
c. PART
d. OneR
e. Ridor
f. NNGE

g. Decision Table

**Tools Used: WEKA 3.6**

In this chapter we have used WEKA Rule based classifier algorithm for analysis and evaluation.