Genetic Algorithm Based Feature Selection for the Classification of Breast Masses in Mammograms

A PROJECT REPORT

Submitted by

Rakesh Prasanna R (910619205042)

In partial fulfilment for the award of the degree

Of

BACHELOR OF TECHNOLOGY

IN

INFORMATION TECHNOLOGY

K.L.N.COLLEGE OF ENGINEERING, POTTAPALAYAM
(An Autonomous Institution, Affiliated to University, Chennai)

ANNA UNIVERSITY: CHENNAI 600 025

APRIL 2023

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **"Genetic Algorithm Based Feature Selection for the Classification of Breast Masses in Mammograms"** is the bona-fide work of **"Mr. Rakesh Prasanna R (Reg. No. 910619205042)"** who carried out the project work under my supervision.

SIGNATURE

Dr. P.Ganesh Kumar
B.E(I&C),M.E(APPL.ELECS.),Ph.D.
HEAD OF THE DEPARTMENT

INFORMATION TECHNOLOGY

K.L.N.COLLEGE OF ENGINEERING
(An ISO 9001-2008 Certified Institution)
POTTAPALAYAM, SIVAGANGAI,
TAMIL NADU, INDIA.

SIGNATURE

Mr.V.Aravinda Rajan,
B.E(ECE),M.E(COMMN.SYS).,
ASSISTANT PROFESSOR

INFORMATION TECHNOLOGY

K.L.N.COLLEGE OF ENGINEERING
(An ISO 9001-2008 Certified Institution)
POTTAPALAYAM, SIVAGANGAI,
TAMIL NADU, INDIA.

Submitted for the Practical Examination held on _____

Internal Examiner **External Examiner**

Acknowledgement

Any work would be unfulfilled without a word of thanks. We hereby take pleasure in acknowledging the persons who guided me throughout our work.

First and foremost, thanks are to the omnipotent for providing us with his abundant blessings all throughout. We all extend our heartfelt thanks to **Er.K.N.K.KARTHIK, B.E.,** President of our college and **Dr.A.V.RAMPRASAD, M.E., Ph.D.,** Principal for provisioning us with the all required.

We esteem our self to articulate our sincere thanks to **Dr.P.Ganesh Kumar, B.E(I&C),M.E(APPL.ELECS.),Ph.D.** Head of the Information Technology for leading us towards the zenith of success.

We express our grateful thanks to our **Project Guide Mr.V.Aravinda Rajan,, B.E(ECE),M.E(COMMN.SYS).,** and **Project Coordinator Dr.J.S.Kanchana, B.E(CSE), M.E(CSE), Ph.D.,**for their invaluable guidance and motivation. Their assistance and advices had been very helpful throughout our project. I would like to thank all teaching and non-teaching staffs of our department who have been the sources of encouragement and ideas. I thank them for lending their support whenever needed.

Abstract

Cancer is one of the most dangerous diseases to humans, and yet no permanent cure has been developed for it. Breast cancer is one of the most common cancer types. According to the National Breast Cancer foundation, in 2020 alone, more than 276,000 new cases of invasive breast cancer and more than 48,000 non-invasive cases were diagnosed in the US. It is very necessary to detect cancer at early stages. In this work, a 3-class classification system is proposed to classify the breast masses in mammogram images into benign, malignant, and normal. The proposed algorithm consists of five modules: pre-processing of images, segmentation of Region of Interest (RoI), feature extraction, feature selection, and classification. In this, feature selection plays a major role, because an optimal set of features will increase the classification accuracy. Genetic Algorithm (GA) is used for selecting the optimal set of features from the extracted feature vector. To evaluate the performance of GA, a t-test method is also applied. To evaluate the performance, three classifiers are used and the results are compared. They are multiSVM, kNN, and Naive Bayes classifiers. Six combinations are evaluated. GA+multiSVM, GA+kNN, GA+Naive Bayes, t-test+multiSVM, t-test+kNN, and t-test+Naive Bayes. It is observed that the GA+kNN combination outperformed for accuracy. Mammogram images used for experimentation are collected from the publicly available dataset, Digital Database for Screening Mammography (DDSM).

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. INTRODUCTION

1.1 INTRODUCTION CHAPTER 1

Breast cancer is cancer that develops in breast cells. Typically, the cancer forms in either the lobules or the ducts of the breast. Lobules are the glands that produce milk, and ducts are the pathways that bring the milk from the glands to the nipple. Cancer can also occur in the fatty tissue or the fibrous connective tissue within your breast. The uncontrolled cancer cells often invade other healthy breast tissue and can travel to the lymph nodes under the arms. The lymph nodes are a primary pathway that helps the cancer cells move to other parts of the body. In its early stages, breast cancer may not cause any symptoms. In many cases, a tumor may be too small to be felt, but an abnormality can still be seen on a mammogram. If a tumor can be felt, the first sign is usually a new lump in the breast that was not there before. However, not all lumps are cancer. Each type of breast cancer can cause a variety of symptoms. Many of these symptoms are similar, but some can be different.

If the cells are not cancerous, the tumor is benign. It won't invade nearby tissues or spread to other areas of the body. A benign tumor is less worrisome unless it is pressing on nearby tissues, nerves, or blood vessels and causing damage. Benign tumors may need to be removed by surgery.

Malignant means that the tumor is made of cancer cells, and it can invade nearby tissues. Some cancer cells can move into the bloodstream or lymph nodes, where they can spread to other tissues within the body this is called metastasis.

Mammography is an imaging modality that uses low energy x-rays specifically for taking images of breast tissue. Mammography practice utilizes standardized views of the breasts for the assessment of breast lesions. It is also used as a screening tool for the detection of early breast cancer in asymptomatic women. Each breast is examined separately and compressed against the film to obtain maximum visualization of masses or calcifications. Early detection of breast cancer allows early treatment and increased rates of survival.

Images from mammography are taken in two angles or views. They are Cranial-Caudal (CC) view and MedioLateral-Oblique (MLO) view. Cranial-Caudal is a view of an image taken above of the breast. While MedioLateral-Oblique is an oblique or angled view.

1.2 Problem Statement

In this work, the problem addressed is the selection of an optimal set of features for accurate classification of breast masses needs improvement. So the better optimizing technique is analyzed and proposed for selecting the optimal features from extract features. One major problem lies with a large number of features, is very difficult to determine which feature or combination of features achieves better classification accuracy rate. Therefore, it is important to select a suitable and optimized set of features. It improves the prediction accuracy and decreases the computational cost. It can distinguish between different types of mammograms.

1.3 Problem Objective

The breast masses in mammogram images are of three types. They are Benign, Malignant, Normal. The main aim of this work is to classify mammograms with a high accuracy rate. To increase the accuracy two different methods are used for selecting the optimal feature and three classifiers are used for the classification of mammogram images. Totally six combinations take place, and the results are evaluated and compared.

1.4 Scope of the Project

1.4.1 Existing System

A Benchmark datasets were used for the experiments. Several ensembles of different ML-based classifiers were also tested for the classification of BC. SVM outperforms both datasets compared to all ML classifiers. ANN from DL classifiers when used individually. For the ensembling method, (SVM + LR + NB + DT) performs well without and with upsampling on the diagnosis dataset (SVMCLRCRFCNB) outperforms all other combinations on the prognosis dataset when ANN is used as a final layer. The performance was also analyzed using a different number of K-fold. .

1.4.2 Proposed System

This work proposed a Genetic algorithm and t-test to select the optimal set of features among the extracted features from all collected images. DDSM mammogram images set is used in this work. The images in the collected dataset are pre-processed. Pre-processing step is achieved by using Adaptive Histogram Equalization (AHE). Region Growing algorithm is used to segment the dense part from pre-processed image. GLCM is used to extract the features. Features are based on texture property. Totally, twenty one features are extracted in this work. Feature selection plays a major role in classification.

1.5 SOFTWARE LIFE CYCLE MODEL

The software life cycle is represented pictorially and diagrammatically by a software life cycle model, also known as a process model. A life cycle model depicts every procedure needed to move a software product through each stage of its life cycle. It also captures the organisational framework in which these techniques are to be used. In other words, a life cycle model depicts the different tasks carried out on a piece of softw1.5 are from conception until retirement. The necessary development activities may be scheduled according to phases in various life cycle models. Therefore, no matter whether life cycle model is used, all of the fundamental tasks are included, even if they may be carried out in different sequences depending on the life cycle model, basically it involves 7 stages.

Stage 1: Planning and requirement analysis

The level of the SDLC that is most crucial and essential is requirement analysis. With input from all the stakeholders, domain experts, and SMEs in the industry, the senior team members carry it out. At this point, planning is also done for the requirements for quality assurance and for the identification of project-related risks. A meeting is scheduled with the client by the business analyst and project manager to obtain all the necessary information, such as what the customer wants to construct, who will be the end user, and what the product's goal is. A fundamental knowledge or understanding of the product is crucial before constructing it.

Stage 2: Defining Requirements

The process of representing, documenting, and getting the project stakeholders to approve the software requirements follows the completion of the requirement analysis. This is done by using the "SRS" document, which contains all the product requirements that must be created and developed during the project life cycle.

Stage 3: Designing the Software

The knowledge of the software project's needs, analysis, and design will all be revealed in the upcoming phase. This phase is the result of the previous two, such as requirement collection and client input.

Stage 4: Developing the project

The actual development phase of the SDLC starts here, and programming is created. Coding represents the start of design implementation. Programming tools including compilers,

Interpreters, debuggers, and other similar tools are used to generate and implement the code, and developers must adhere to the coding standards outlined by their management.

Stage 5: Testing

Following the generation of the code, it is compared to the requirements to ensure that the solutions are satisfying the demands identified and acquired during the requirements stage. Unit testing, integration testing, system testing, and acceptability testing are carried out at this level.

Stage 6: Deployment

When the software has been certified and no defects or mistakes have been reported, it is put into use. The software may then be delivered as is or with proposed improvements in the object portion depending on the assessment. The maintenance of the software starts once it has been deployed.

Stage 7: Maintenance

When the customer begins utilising the technologies that have been designed, the true problems and ongoing needs become apparent. Maintenance is the process when the developed product is given attention.

1.5.1 Types

The common life cycles are

1.5.1.1 Waterfall model
1.5.1.2 Incremental model
1.5.1.3 Spiral model
1.5.1.4 Big bang model
1.5.1.5 Prototyping model
1.5.1.6 RAD model
1.5.1.7 Agile model

1.5.2 Agile model

Agile methodology is a software development approach that emphasizes flexibility, collaboration, and continuous improvement throughout the development process. Unlike the traditional Waterfall model, which follows a linear and sequential approach, the Agile

Methodology involves working in short iterations and delivering working software at the end of each iteration.

The Agile methodology is based on four core values:

1. Individuals and interactions over processes and tools.
2. Working software over comprehensive documentation.
3. Customer collaboration over contract negotiation.
4. Responding to change over following a plan.

Overall, the Agile methodology promotes flexibility, collaboration, and continuous improvement, which can result in faster delivery of working software, improved quality, and increased customer satisfaction.

1.5.3 Reason for choosing the Agile model.

Agile methodology is a practice which promotes continues interaction of development and testing during the SDLC process of any project. The entire project is divided into various small incremental builds which are way more convenient for us and all these builds are provided in iteration and each iteration lasts for 10 to 14 days.

Another reason to use agile model is that it will respond to change over a following plan, it is difficult to think in advance which software requirements will persist and which will change. It is equally difficult to predict how priorities will change as the project proceeds. Analysis, design, development, and testing are not as predictable (from a planning point of view).

While building an software design and development are interleaved i.e., both activities should be performed in tandem so that design models are proven as they are created. It is difficult to think about how much design is necessary before construction is used to test the configuration and more importantly, agile model can accommodate even with fewer developers.

1.6 Project Plan

Project is broken down into various steps which makes easier for us to complete the task within the stipulated amount of time.

Task 1: To collect the mammogram images from DDSM Database.

Task 2: The collected images are pre-processed to remove the artifacts present in the images. Adaptive Histogram Equalization algorithm is used for pre-processing.

Task 3: The Region of Interest is extracted with the help of the Region Growing algorithm.

Task 4: From the cropped image the GLCM features are extracted based on the textural property.

Task 5: The genetic algorithm and t-test are used to select the optimal features.

Task 6: Classification is done by using KNN, SVM, and Naïve Bayes classifiers.

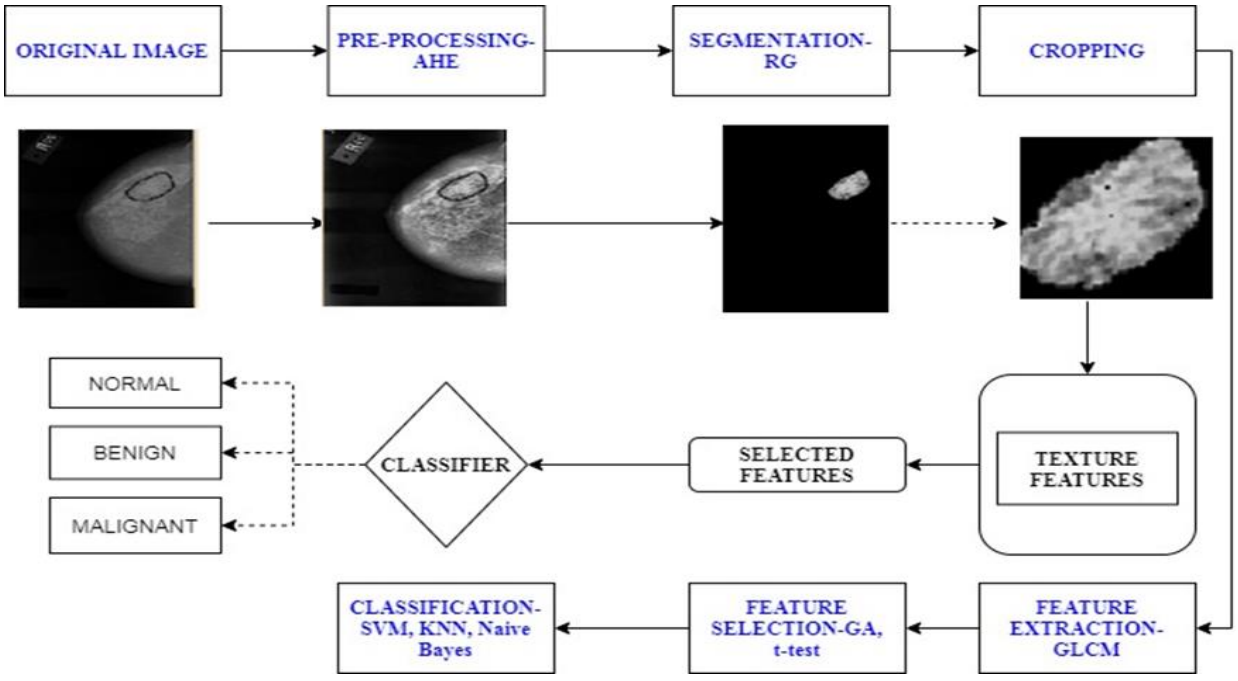


Figure 1.6.1: detailed project plan

1.7 Summary

In this section we have briefly mentioned about the overview of the project, objective and the impact it makes in the medical field and even more than that a clear picture of the project step by step is given from scratch.

We have used the agile model for SDLC (Software Development Life Cycle) as it accommodate fewer developers and can be able to accept many changes and works dynamically and also it requires minimal prior planning.

2. LITERATURE REVIEW

2.1 Introduction CHAPTER 2

In this chapter contains all the literature survey that was the motivation for creating such project and has to make this project into more efficient and effective manner so that a large number of industries can be benefitted from it.

2.1 Literature Review

[1] USMAN NASEEM1, JUNAID RASHID2, LIAQAT ALI3, JUNGEUN KIM 4, QAZI EMAD UL HAQ 5, MAZHAR JAVED AWAN 6, AND MUHAMMAD IMRAN 7 "An Automatic Detection of Breast Cancer Diagnosis and Prognosis Based on Machine Learning Using Ensemble of Classifiers "VOLUME 10, 2022.

Breast cancer (BC) is the second most prevalent type of cancer among women leading to death. This Paper presented an ensemble of machine learning-based methods for breast cancer diagnosis and prognosis using an ensemble of machine learning classifiers. A comprehensive comparison of the performance of various machine learning and ensemble machine learning-based classifiers were discussed. It evaluated different sampling methods to address the class imbalance issue in our datasets. The proposed method outperforms various state-of-the-art methods for the detection of breast cancer

[2] ZEXIAN HUANG ,DAQI CHEN, "A Breast Cancer Diagnosis Method Based on VIM Feature Selection and Hierarchical Clustering Random Forest Algorithm", IEEE Access VOLUME 10, 2022

Accurate detection and effective treatment are of vital significance to lower the death rate of breast cancer. In this paper, a Hierarchical Clustering Random Forest (HCRF) model was developed. By measuring the similarity among all the decision trees, the hierarchical clustering technique is used to carry out clustering analysis on decision trees. The representative trees are selected from divided clusters to construct the hierarchical clustering random forest with low similarity and high accuracy. Variable Importance Measure (VIM) method was used to optimize the selected feature number for the breast cancer prediction. HCRF algorithm with VIM as a feature selection method reaches the best accuracy when compared to Decision Tree, Adaboost and Random Forest on both the WDBC and WBC datasets

[3] Mohammed Amine Naji a, Sanaa El Filalib Kawtar Aarikac, EL Habib Benlahmard, Rachida Ait Abdelouhahide, Olivier Debauche, "Machine Learning Algorithms For Breast Cancer Prediction And Diagnosis" International Workshop on Edge IA-IoT for Smart Agriculture (SA2IOT) August 9-12, 2021, Leuven, Belgium

To predict and diagnosis breast cancer using Machine Learning algorithms and find out most effective with respect to confusion matrix, accuracy and precision. Breast Cancer Wisconsin Diagnostic dataset from University of Wisconsin Hospitals Madison Breast Cancer Database.The features of dataset are computed from a digitized image of a breast cancer sample obtained from fine-needle aspirate (FNA) . Five main algorithms which are: SVM, Random Forests, Logistic Regression, Decision Tree, K-NN . All algorithms have been programmed in Python using scikit-learn library in Anaconda environment. Support Vector Machine achieved a higher efficiency of 97.2%, Precision of 97.5%, AUC of 96.6% and outperforms all other algorithms.

[4] Sweta Bhise , Simran Bepari , Shrutika Gadekar, "Breast Cancer Detection using Machine Learning Techniques", International Journal of Engineering Research & Technology (IJERT) ISSN: 2278-0181 Vol. 10 Issue 07, July-2021

This paper presents a Machine Learning model to perform automated diagnosis for breast cancer. The system was experimented on BreaKHis 400X Dataset. Recursive Feature Elimination (RFE) for feature selection. This method employed CNN as a classifier model. Also, five algorithms SVM, Random Forest, KNN, Logistic Regression, Naïve Bayes classifier have been compared. The performance of the system is measured on the basis of accuracy and precision. It was observed that CNN outperforms the existing methods when it comes to accuracy, precision and also size of the data set.

[5] KOSMIA LOIZIDOU, GALATEIA SKOUROUMOUNI,CHRISTOS NIKOLAOU, "Automatic Breast Mass Segmentation and Classification Using Subtraction of Temporally", 4 November 2022; date of current version 10 November 2022.

.

Cancer remains a major cause of morbidity and mortality globally, with 1 in 5 of all new cancers arising in the breast. The introduction of mammography significantly decreased their mortality rates. In this study, subtraction of temporally sequential digital mammograms and machine learning are proposed for the automatic segmentation and classification of masses. The performance of the algorithm was evaluated on a dataset with 320 images from 80 patients with precisely annotated mass locations by two radiologists. Ninety-six features were extracted and ten classifiers were tested in a leave-one-patient-out and k-fold cross-validation process. Using Neural Networks, the detection of masses was accuracy.

2.3 Summary

This chapter explains the literature survey that has been carried out in order to make the projectmore effective.

3. SYSTEM ANALYSIS

3.1 Introduction CHAPTER 3

System analysis is the process of studying a system to identify its components, their interrelationships, and how they work together to achieve specific goals or objectives. It is the first step in system development, and it involves gathering information about the current system, defining the requirements of the new system, and proposing solutions to any problems or inefficiencies that are identified.

The system analysis process is critical to the success of any system development project. It helps to ensure that the new system meets the needs of the stakeholders, is efficient and effective, and is aligned with the goals and objectives of the organization.

3.2 Requirement Analysis

Requirement analysis is the process of identifying, documenting, and prioritizing the needs and expectations of stakeholders for a software development project. It involves gathering, analysing, and defining the requirements that the software must meet to satisfy stakeholders and achieve the project's goals.

3.2.1 Functional Requirements

Functional Requirement defines a function of software system and how system must behave when presented with specific inputs or conditions. These may include calculations, data, manipulation and processing and other specific functionality. In this system following are the functional requirement: -

- Training dataset must be loaded

3.2.2 Non-Functional Requirements

Non-functional requirements are the requirements which are not directly concerned with the specific function delivered by the system. They specify the criteria that can be used to judge the operation of a system rather than specific behaviours. They may relate to emergent system properties. Non-functional requirements for this system are specified as follows:

- Responsiveness of the system needs to be appropriate since timely retrieval of sensitive health data is essential.
- The software utilised should be portable so that medical institutions can easily expand to their inter-connected hospitals, spread across locations.

- Privacy of sensitive data should always be maintained and must not be misused in any manner. Researchers requesting for data must be from authorized sources. Proof of consent is a responsibility that is borne by the medical institution where the health data is generated.

3.2.3 Hardware Requirements

PROCESSOR	: Pentium IV & above
RAM	: 4 GB

3.2.4 Software Requirements

OPERATING SYSTEM	: Windows 7
MATLAB VERSION	: Matlab R2014a

3.2.5 Module Specification

The proposed system consists of five modules. They are listed as follows:

1. Pre-processing
2. Segmentation
3. Feature extraction
4. Feature selection
5. Classifier Comparison Analysis

3.2.5.1 Pre-Processing

Pre-processing is a process that is used to boost the precision and interpretability of an image. Image pre-processing is a challenging task in the CAD system. In medical image processing, pre-processing of an image is very important so that the extracted image does not have any impurities, and it is accomplished to be better for the forthcoming process such as segmentation, feature extraction, etc. Only the correct segmentation of the tumor will yield an accurate result. Accurate detection leads to precise feature extraction and in turn, gives perfect classification. The accurate tumor segmentation is promising, only if the image is pre-processed clearly. Images are collected from the Digital Database for Screening Mammography (DDSM) database.

The pre-processing step applied to the collected dataset images. It can be achieved by using AHE. Histogram works on an entire image. In contrast, AHE works on a small region of an image called tiles. It enhances the contrast of each tile. The range of gray-level in a low contrast image got increased. It takes the dataset image as input. Split the input image into regions r_1, r_2, … , r_n. The process has been applied for each contextual region. Extract a single region from an image. Create a histogram for the region. Clip the histogram using the clip limit. Then create a transformation function for a region R. Defined as cumulative distribution function (CDF) of a given probability density function (PDF) of a gray-levels in a given image. Apply four-level mapping to the pixel, which got extracted from the region. Then interpolate the gray level between the pixel results. The output image is named as pre-processed image.

3.2.5.2 Segmentation

From the pre-processed image, the dense part is segmented as an ROI. In this work, a single seed region growing algorithm is proposed to implement image segmentation. Here, the segmentation process separates the tumor areas from the background tissue in mammograms. The main goal of segmentation is to simplify and/or change the representation of an image. It will reduce the computation time. The proposed method is used to segment the selected region in the original image. It is a simple region-based image segmentation method. This approach is the opposite of the split and merges approach. Start by choosing an arbitrary seed pixel. The selected seed pixel is compared with neighboring pixels based on the intensity value and it is checked whether the pixel is inside or outside the image. The region is grown from the seed pixel by adding the neighboring pixels that are similar in intensity property. While adding the

pixel to the region, it is necessary to check whether the pixel belongs to some other regions. When the growth of one region stops we simply choose another seed pixel. This whole process is continued until all pixels belong to some regions. Steps are given below:

1. Start by choosing an arbitrary seed pixel and compare it with neighboring pixels.
2. The region is grown from the seed pixel by adding similar neighbouring pixels.
3. When the growth of one region stops we simply choose another seed pixel.
4. This whole process is continued until all pixels belong to some regions.

A cropping operation has been applied on a segmented image to extract the ROI which contains the abnormalities, excluding the unwanted portions of the image. This process is performed by referring to the center of the abnormal area as the center of ROI and taking the approximate radius (in pixels) of a circle enclosing the abnormal area. The main advantages of this automatic segmentation technique are:

- It can correctly separate the regions that have the same properties we define.
- It can provide the original images which have clear edges with good segmentation results.

The concept is simple. A small number of seed points are used to represent the property and to grow the region.

3.2.5.2 Feature Extraction

In image processing, the processing of large data is time-consuming and less efficient for classification. For reducing time, the input data is transformed into a reduced set of a feature vector. This transformation process is called the feature extraction process. This feature vector contains relevant information and is used as an input vector for classification. Features can be classified based on color, texture, and shape. In this, we are mainly concerned about the texture features and for extraction of features Gray Level Co-occurrence Matrix (GLCM) is used. These are powerful tools for feature extraction. GLCM considers the relation between two neighboring pixels, the first pixel is known as a reference and the second is known as a neighbour pixel. GLCM is a matrix where the number of rows and columns is equal to the number of grey levels, G, in the image. The matrix element p (i, j|Δx, Δy) is the relative frequency with which two pixels, are separated by a pixel distance (Δx, Δy), occurs within a given neighborhood, one with intensity ‘i’ and neighbor one with an intensity ‘j’. The matrix element p(i, j|d, Θ) contains the second-order statistical probability values for changes between the gray levels ‘i’ and ‘j’ at a particular distance ‘d’ and at a particular angle ‘Θ’. Using a large number of intensity levels are G implies storing a lot of temporary data, i.e. G ×

G matrix. The co-occurrence matrix is a two-dimensional array in which both rows and columns represent a set of possible image values. In this work, fourteen features are got extracted from the segmented image. The features are Entropy, Dissimilarity, Contrast, Inverse difference, correlation, Homogeneity, Autocorrelation, Cluster Shade, Cluster Prominence, Maximum probability, Sum of Squares, Sum Average, Sum Variance, Sum Entropy, Difference variance, Difference entropy, Information measures of correlation (1), Information measures of correlation (2), Maximal correlation coefficient, Inverse difference normalized, Inverse difference moment normalized.

3.2.5.4 Feature Selection

The features are extracted from the textures of ROIs. This helps the classifier to distinguish the breast tissues as normal, benign, or malignant. One major problem lies with a large number of features, is very difficult to determine which feature or combination of features achieves better classification accuracy rate. Therefore, it is important to select a suitable and optimized set of features. It improves the prediction accuracy and decreases the computational cost. It can distinguish between different types of mammograms. For selecting the features, a genetic algorithm and a t-test are used.

Genetic algorithms are used to find optimal values of some function. An initial set of features is created and their corresponding fitness values are calculated. This set of features is referred to as a population and each features as an individual. The individuals with the best fitness values are combined randomly to produce offsprings which make up the next population. To do so, individuals are selected and undergo cross-over and also are subject to random mutations. This process is repeated again and again and many generations are produced that should create better and better solutions. The optimal features, which are selected by using a genetic algorithm are Contrast, correlation, dissimilarity, Maximum probability, Sum Average, Sum Entropy, Difference variance, Information measures of correlation (1), Inverse difference normalized.

The most common type of t-test, namely the student t-test, is often used to assess whether the means of two classes are statistically different from each other by calculating a ratio between the difference of two class means and the variability of the two classes. The t-test has been used to rank features for microarray. These uses of the t-test are limited to two-class problems. For multi-class problems, calculated a t-statistics value for each feature of each class by evaluating the difference between the mean of one class and the mean of all the classes, where the difference is standardized by the within-class standard deviation.

$$t_{ic} = \frac{\overline{x_{ic}} - \overline{x_i}}{M_c(S_i + S_0)}$$

$$M_c = \sqrt{\frac{1}{n_c} + \frac{1}{N}}$$

Here, t_i is the t-statistics value for the i^{th} feature of the c^{th} class; $\overline{x_{ic}}$ is the mean of the i^{th} feature in the c^{th} class, and $\overline{x_i}$ is the mean of i^{th} feature for all classes; S_i is the within-class standard deviation and S_0 is set to be the median value of S_i for all the features.

$$S_i = N - C \sum c = 1C \sum j \in c(x_{ij} - \overline{x_{ic}})$$

x_{ij} refers to the i^{th} feature of the j^{th} sample; N is the number of all the samples in the C classes; nc is the number of samples in class c;

Extending the t-score for feature i to be the greatest t-score for all classes for feature i:

$$t_i = max \frac{|\overline{x_{ic}} - \overline{x_i}|}{(M_c * S_i)}$$

The feature with the greatest t-score is considered as an optimal feature. The features selected by using the t-test are Contrast, Cluster Shade, Cluster Prominence, dissimilarity, Homogeneity, Maximum probability, Difference variance, Difference entropy, energy, entropy.

3.2.5.5 Classification

Classification is the process to assign a class label to the testing data as normal, benign, and malignant. Optimal features, which are selected using GA is used to classify the mammogram images. The first phase of the classification problem is to train the model. The second phase is to test the given data set. In this work, 979 images are collected. In that, 550 images features are considered as a training set, and 429 images features are considered as a testing set. SVM, KNN, Naïve Bayes classifiers are used to classify the mammogram images. The performance of the classifier is evaluated with the help of a confusion matrix. The confusion matrix is a table that shows the output and actual class classification accomplished by the classifier.

SVM is a supervised machine learning algorithm that can be used classification. In the SVM algorithm, each data item is a plot as a point in n-dimensional space (where n is many features) with the value of each feature being the value of a particular coordinate. Then, the classification is done by finding the hyper-plane that differentiates each class very well.

KNN is a type of supervised machine language algorithm that can be used for both

classifications as well as regression predictive problems. However, it is mainly used for classification predictive problems in the industry. It uses ‘feature similarity’ to predict the values of data points which means new data points will be assigned a value based on how closely it matches the points in the training set.

Naive Bayes is a statistical classification technique based on Bayes Theorem. It is one of the simplest supervised learning algorithms. Naive Bayes classifiers have high accuracy and speed on large datasets. A Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

3.3 Summary

This chapter explains details of system analysis, function and nonfunctional requirements along with hardware and software requirements. Apart from that a clear picture about each modules is explained clearly with the certain proof required for it.

4. SYSTEM DESIGN

4.1 Introduction CHAPTER 4

System design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specific requirements. It involves analysing the requirements, determining the system's objectives and constraints, and developing a high-level architecture for the system. System design includes defining the components that make up the system and the interactions between them. It also involves defining the data structures and algorithms used to process the data.

The system design phase is critical as it sets the foundation for the development, testing, and deployment of the system. It ensures that the system is designed to meet the desired requirements and that the architecture is flexible and scalable to accommodate future changes and enhancements.

4.2 Data Flow Diagram

Data Flow Diagram (DFD) is a graphical representation of how data flows through a system, showing the inputs, outputs, and processes involved. It is commonly used in software engineering and business analysis.

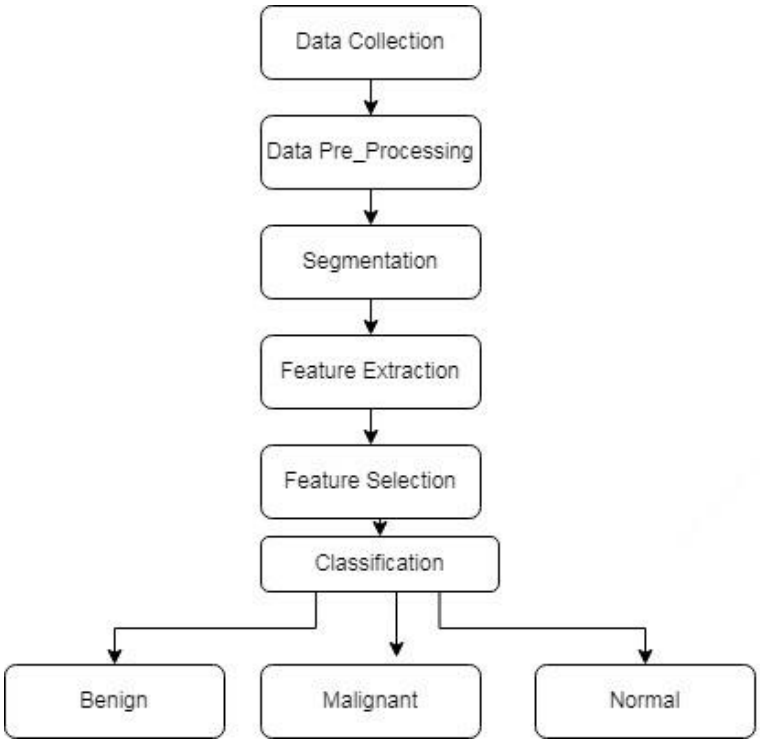


Figure 4.2.1: Data Flow Diagram

4.3 Object Oriented Analysis and Design

Object Oriented Analysis and Design (OOAD) is a software engineering approach that involves modelling a system as a collection of objects with their attributes and behaviours. It focuses on developing reusable and modular software systems.

4.3.1 Detailed Design

Detailed Design in UML is the process of creating a detailed blueprint for the software system based on the system design. It includes creating detailed class diagrams, sequence diagrams, state machine diagrams, and other UML diagrams to describe the behaviour and structure of the system. The detailed design phase also involves designing the user interface, database schema, and other technical details of the system. The output of this phase is a detailed design document that serves as a guide for the development team. Detailed design is crucial for developing high-quality, maintainable, and scalable software systems.

4.3.1.1 Use Case Diagram

A use case diagram is a type of UML diagram that provides a visual representation of the interactions between actors and the system. It is used to depict the system's functions, their relationships, and how they are related to actors who interact with the system. Use case diagrams are commonly used in software engineering to model and design systems.

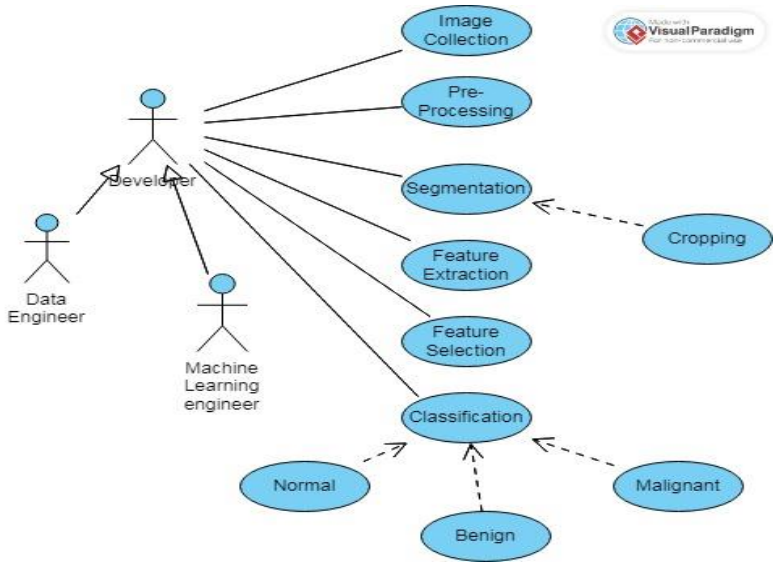


Figure 4.3.1.1.1: Use Case diagram

4.3.1.2 Activity Diagram

An activity diagram is a type of UML diagram that depicts the flow of activities or processes in a system. It is used to model business processes, software processes, and workflows. Activity diagrams show the activities, decisions, and branching paths involved in a process, making it easier to understand and analyse the system's behavior.

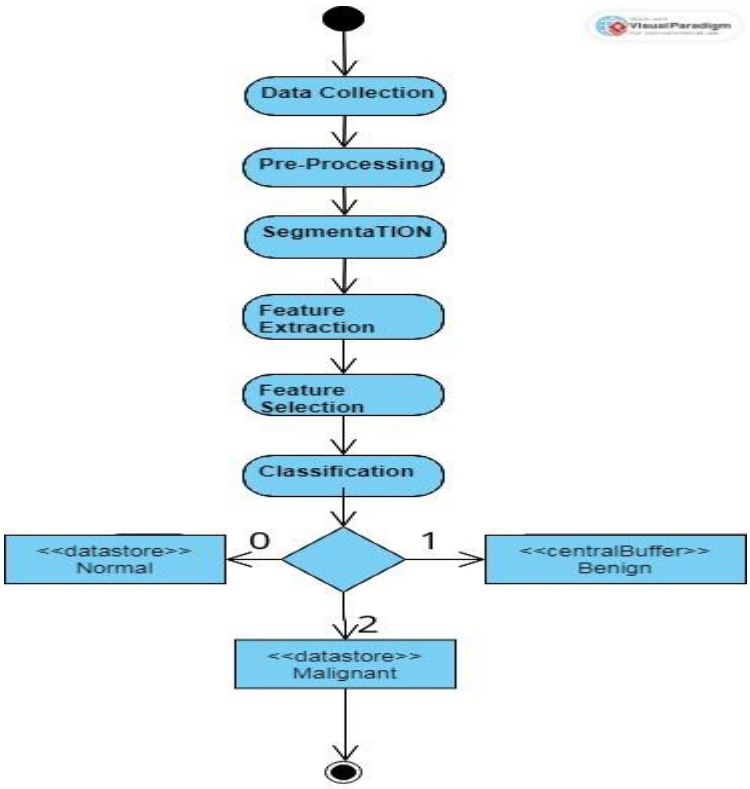


Figure 4.3.1.2.1: Activity diagram

4.3.1.3 Class Diagram

A class diagram is a type of UML diagram that describes the structure of a system by showing the classes, their attributes, methods, and relationships with other classes. It is a static representation that is commonly used in software engineering to model and design object-oriented systems.

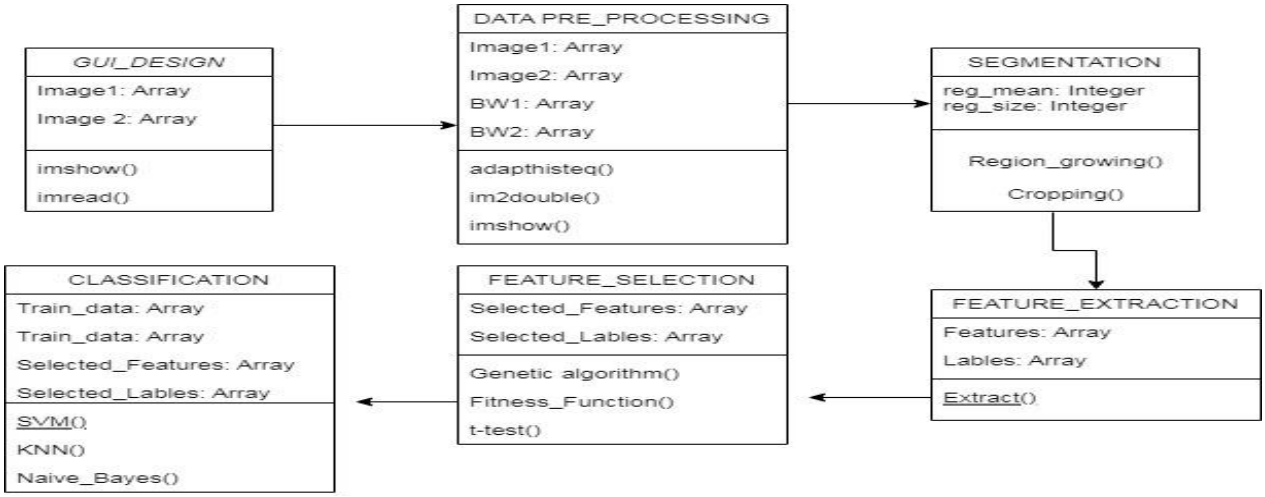


Figure 4.3.1.3.1: Class diagram

4.3.1.4 Sequence Diagram

A sequence diagram is a type of UML diagram that shows the interactions between objects or actors in a system, including the order in which messages are sent and received. It depicts the sequence of events in a system, making it easier to understand and analyse the system's behavior. Sequence diagrams are commonly used in software engineering to model and design systems that involve multiple objects and actors.

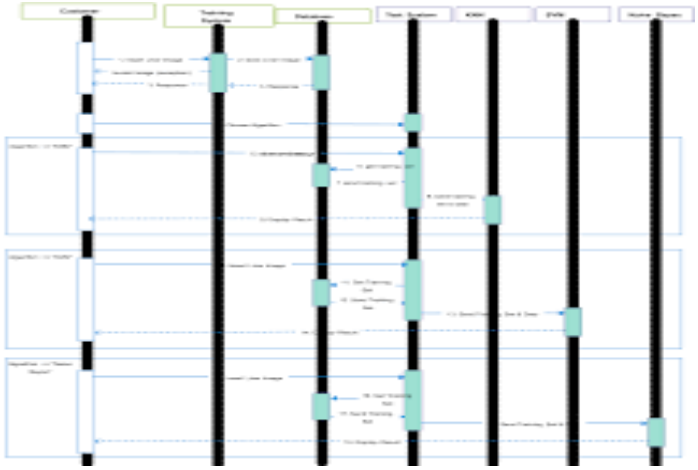


Figure 4.3.1.4.1: Sequence diagram

4.4 SUMMARY

This chapter explains in depth about the overall system design underlying the complete project and even explains all the scenarios and the system will react to it.

5. IMPLEMENTATION

```
function varargout = genetic(varargin)
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @genetic_OpeningFcn, ...
                   'gui_OutputFcn',  @genetic_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end
 if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
function genetic_OpeningFcn(hObject, ~, handles, varargin)
handles.output = hObject;
 guidata(hObject, handles);
 function varargout = genetic_OutputFcn(~, ~, handles)
varargout{1} = handles.output;
 function edit1_Callback(hObject, eventdata, handles)
function edit1_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'), get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function load_Callback(hObject, eventdata, handles)
global im1 im2
axes(handles.axes2);
[fn1,pn1]=uigetfile('*.jpg','select cc view');
im1=imread(strcat(pn1,fn1));
imshow(im1);
axis equal;
```

```
axis off;
title('CC VIEW','fontsize',15);
axes(handles.axes6);
[fn2,pn2]=uigetfile('*.jpg','select mlo view');
im2=imread(strcat(pn2,fn2));
imshow(im2);
axis equal;
axis off;
title('MLO VIEW','fontsize',15);
```

5.2 PRE-PROCESSING

```
function preprocessing_Callback(hObject, eventdata, handles)
global im1 im2 bw1 bw2
bw1=adapthisteq(im1);
axes(handles.axes7);
imshow(bw1);
bw2=adapthisteq(im2);
axes(handles.axes5);
imshow(bw2);
function segmentation_Callback(hObject, eventdata, handles)
global  bw1 bw2 c1 c2
I = im2double(bw1);
[J,K] = regiongrowing(I);
axes(handles.axes10);
imshow(K);
title('SEGMENTED REGION','fontsize',15);
c1 = imcrop(K);
axes(handles.axes12);
imshow(c1);
title('CROPPED','fontsize',15);
I = im2double(bw2);
[J,K] = regiongrowing(I);
axes(handles.axes11);
```

```
imshow(K);
title('SEGMENTED REGION','fontsize',15);
c2 = imcrop(K);
axes(handles.axes13);
imshow(c2);
title('CROPPED','fontsize',15);
function featureextraction_Callback(hObject, eventdata, handles)
global c1 c2
I = im2double(c1);
GLCM2 = graycomatrix(I,'Offset',[2 0;0 2]);
stats = glcmfeatures1(GLCM2,0);
statsTable = struct2table(stats)
I = im2double(c2);
GLCM2 = graycomatrix(I,'Offset',[2 0;0 2]);
stats = glcmfeatures1(GLCM2,0);
statsTable = struct2table(stats)
```

5.2.1 SEGMENTATION BASED ON REGION GROWING

```
function [J,K]=regiongrowing(I,x,y,reg_maxdist)
if(exist('reg_maxdist','var')==0), reg_maxdist=0.2; end
if(exist('y','var')==0), figure, imshow(I,[]); [y,x]=getpts; y=round(y(1)); x=round(x(1)); end
J = zeros(size(I));
K = zeros(size(I));
Isizes = size(I);
reg_mean = I(x,y);
reg_size = 1;
neg_free = 10000; neg_pos=0;
neg_list = zeros(neg_free,3);
pixdist=0;
neigb=[-1 0; 1 0; 0 -1;0 1];
while(pixdist<reg_maxdist&&reg_size<numel(I))
  for j=1:4,
       xn = x +neigb(j,1); yn = y +neigb(j,2);
```

```
        ins=(xn>=1)&&(yn>=1)&&(xn<=Isizes(1))&&(yn<=Isizes(2));
        if(ins&&(J(xn,yn)==0))
              neg_pos = neg_pos+1;
              neg_list(neg_pos,:) = [xn yn I(xn,yn)];
       J(xn,yn)=0;
       K(xn,yn)=I(xn,yn);
         end
        end
    % Add a new block of free memory
    if(neg_pos+10>neg_free), neg_free=neg_free+10000; neg_list((neg_pos+1):neg_free,:)=0; end
     % Add pixel with intensity nearest to the mean of the region, to the region
    dist = abs(neg_list(1:neg_pos,3)-reg_mean);
    [pixdist, index] = min(dist);
    J(x,y)=2; reg_size=reg_size+1;
     % Calculate the new mean of the region
    reg_mean= (reg_mean*reg_size + neg_list(index,3))/(reg_size+1);
    % Save the x and y coordinates of the pixel (for the neighbour add proccess)
    x = neg_list(index,1); y = neg_list(index,2);
    % Remove the pixel from the neighbour (check) list
    neg_list(index,:)=neg_list(neg_pos,:); neg_pos=neg_pos-1;
end
J=J>1;
```

5.3 GLCM BASED FEATURE EXTRACTION

```
function [out] = GLCM_Features1(glcmin,pairs)
 if ((nargin > 2) || (nargin == 0))
   error('Too many or too few input arguments. Enter GLCM and pairs.');
elseif ( (nargin == 2) )
    if ((size(glcmin,1) <= 1) || (size(glcmin,2) <= 1))
       error('The GLCM should be a 2-D or 3-D matrix.');
    elseif ( size(glcmin,1) ~= size(glcmin,2) )
        error('Each GLCM should be square with NumLevels rows and NumLevels cols');
    end
```

```
elseif (nargin == 1) % only GLCM is entered
    pairs = 0;
    if ((size(glcmin,1) <= 1) || (size(glcmin,2) <= 1))
       error('The GLCM should be a 2-D or 3-D matrix.');
    elseif ( size(glcmin,1) ~= size(glcmin,2) )
       error('Each GLCM should be square with NumLevels rows and NumLevels cols');
    end
end
format long e
if (pairs == 1)
    newn = 1;
    for nglcm = 1:2:size(glcmin,3)
        glcm(:,:,newn)  = glcmin(:,:,nglcm) + glcmin(:,:,nglcm+1);
        newn = newn + 1;
    end
elseif (pairs == 0)
    glcm = glcmin;
end
size_glcm_1 = size(glcm,1);
size_glcm_2 = size(glcm,2);
size_glcm_3 = size(glcm,3);
% checked
out.autoc = zeros(1,size_glcm_3);
out.contr = zeros(1,size_glcm_3);
out.corrm = zeros(1,size_glcm_3);
out.corrp = zeros(1,size_glcm_3);
out.cprom = zeros(1,size_glcm_3);
out.cshad = zeros(1,size_glcm_3);
out.dissi = zeros(1,size_glcm_3);
out.energ = zeros(1,size_glcm_3);
out.entro = zeros(1,size_glcm_3);
out.homom = zeros(1,size_glcm_3);
out.homop = zeros(1,size_glcm_3);
out.maxpr = zeros(1,size_glcm_3);
```

```
out.sosvh = zeros(1,size_glcm_3);
out.savgh = zeros(1,size_glcm_3);
out.svarh = zeros(1,size_glcm_3);
out.senth = zeros(1,size_glcm_3);
out.dvarh = zeros(1,size_glcm_3);
out.denth = zeros(1,size_glcm_3);
out.inf1h = zeros(1,size_glcm_3);
out.inf2h = zeros(1,size_glcm_3);
out.indnc = zeros(1,size_glcm_3);
out.idmnc = zeros(1,size_glcm_3);
glcm_sum  = zeros(size_glcm_3,1);
glcm_mean = zeros(size_glcm_3,1);
glcm_var  = zeros(size_glcm_3,1);
u_x = zeros(size_glcm_3,1);
u_y = zeros(size_glcm_3,1);
s_x = zeros(size_glcm_3,1);
s_y = zeros(size_glcm_3,1);
p_x = zeros(size_glcm_1,size_glcm_3);
p_xplusy = zeros((size_glcm_1*2 - 1),size_glcm_3);
p_xminusy = zeros((size_glcm_1),size_glcm_3);
hxy  = zeros(size_glcm_3,1);
hxy1 = zeros(size_glcm_3,1);
hx   = zeros(size_glcm_3,1);
hy   = zeros(size_glcm_3,1);
hxy2 = zeros(size_glcm_3,1);
for k = 1:size_glcm_3 % number glcms
   glcm_sum(k) = sum(sum(glcm(:,:,k)));
   glcm(:,:,k) = glcm(:,:,k)./glcm_sum(k);
glcm_mean(k) = mean2(glcm(:,:,k));
glcm_var(k)  = (std2(glcm(:,:,k)))^2;
   for i = 1:size_glcm_1
      for j = 1:size_glcm_2
         out.contr(k) = out.contr(k) + (abs(i - j))^2.*glcm(i,j,k);
         out.dissi(k) = out.dissi(k) + (abs(i - j)*glcm(i,j,k));
```

```
            out.energ(k) = out.energ(k) + (glcm(i,j,k).^2);
            out.entro(k) = out.entro(k) - (glcm(i,j,k)*log(glcm(i,j,k) + eps));
            out.homom(k) = out.homom(k) + (glcm(i,j,k)/( 1 + abs(i-j) ));
            out.homop(k) = out.homop(k) + (glcm(i,j,k)/( 1 + (i - j)^2));
            out.sosvh(k) = out.sosvh(k) + glcm(i,j,k)*((i - glcm_mean(k))^2);
            %out.invdc(k) = out.homom(k);
            out.indnc(k) = out.indnc(k) + (glcm(i,j,k)/( 1 + (abs(i-j)/size_glcm_1) ));
            out.idmnc(k) = out.idmnc(k) + (glcm(i,j,k)/( 1 + ((i - j)/size_glcm_1)^2));
            u_x(k)       = u_x(k) + (i)*glcm(i,j,k); % changed 10/26/08
            u_y(k)       = u_y(k) + (j)*glcm(i,j,k); % changed 10/26/08
             end
        end
      out.maxpr(k) = max(max(glcm(:,:,k)));
end
for k = 1:size_glcm_3
 for i = 1:size_glcm_1
   for j = 1:size_glcm_2
            p_x(i,k) = p_x(i,k) + glcm(i,j,k);
            p_y(i,k) = p_y(i,k) + glcm(j,i,k); % taking i for j and j for i
            if (ismember((i + j),[2:2*size_glcm_1]))
               p_xplusy((i+j)-1,k) = p_xplusy((i+j)-1,k) + glcm(i,j,k);
            end
            if (ismember(abs(i-j),[0:(size_glcm_1-1)]))
               p_xminusy((abs(i-j))+1,k) = p_xminusy((abs(i-j))+1,k) +...
                  glcm(i,j,k);
            end
        end
    end

end
% computing sum average, sum variance and sum entropy:
for k = 1:(size_glcm_3)
     for i = 1:(2*(size_glcm_1)-1)
        out.savgh(k) = out.savgh(k) + (i+1)*p_xplusy(i,k);
```

```
        % the summation for savgh is for i from 2 to 2*Ng hence (i+1)
        out.senth(k) = out.senth(k) - (p_xplusy(i,k)*log(p_xplusy(i,k) + eps));
    end
end
% compute sum variance with the help of sum entropy
for k = 1:(size_glcm_3)
 for i = 1:(2*(size_glcm_1)-1)
        out.svarh(k) = out.svarh(k) + (((i+1) - out.senth(k))^2)*p_xplusy(i,k);
    end
end
% compute difference variance, difference entropy,
for k = 1:size_glcm_3
    for i = 0:(size_glcm_1-1)
        out.denth(k) = out.denth(k) - (p_xminusy(i+1,k)*log(p_xminusy(i+1,k) + eps));
        out.dvarh(k) = out.dvarh(k) + (i^2)*p_xminusy(i+1,k);
    end
end
% compute information measure of correlation(1,2) [1]
for k = 1:size_glcm_3
    hxy(k) = out.entro(k);
    for i = 1:size_glcm_1
     for j = 1:size_glcm_2
            hxy1(k) = hxy1(k) - (glcm(i,j,k)*log(p_x(i,k)*p_y(j,k) + eps));
            hxy2(k) = hxy2(k) - (p_x(i,k)*p_y(j,k)*log(p_x(i,k)*p_y(j,k) + eps));
        end
        hx(k) = hx(k) - (p_x(i,k)*log(p_x(i,k) + eps));
        hy(k) = hy(k) - (p_y(i,k)*log(p_y(i,k) + eps));
    end
    out.inf1h(k) = ( hxy(k) - hxy1(k) ) / ( max([hx(k),hy(k)]) );
    out.inf2h(k) = ( 1 - exp( -2*( hxy2(k) - hxy(k) ) ) )^0.5;
end
corm = zeros(size_glcm_3,1);
corp = zeros(size_glcm_3,1);
for k = 1:size_glcm_3
```

```
    for i = 1:size_glcm_1
        for j = 1:size_glcm_2
            s_x(k)  = s_x(k)  + (((i) - u_x(k))^2)*glcm(i,j,k);
            s_y(k)  = s_y(k)  + (((j) - u_y(k))^2)*glcm(i,j,k);
            corp(k) = corp(k) + ((i)*(j)*glcm(i,j,k));
            corm(k) = corm(k) + (((i) - u_x(k))*((j) - u_y(k))*glcm(i,j,k));
            out.cprom(k) = out.cprom(k) + (((i + j - u_x(k) - u_y(k))^4)*...
                glcm(i,j,k));
            out.cshad(k) = out.cshad(k) + (((i + j - u_x(k) - u_y(k))^3)*...
                glcm(i,j,k));
        end
    end
    s_x(k) = s_x(k) ^ 0.5;
    s_y(k) = s_y(k) ^ 0.5;
    out.autoc(k) = corp(k);
    out.corrp(k) = (corp(k) - u_x(k)*u_y(k))/(s_x(k)*s_y(k));
    out.corrm(k) = corm(k) / (s_x(k)*s_y(k));
end
```

5.4 GENETIC ALGORITHM BASED FEATURE SELECTION

Main.m

```
(Method 1)GA version 1
close all; N=10; T=100; CR=0.8; MR=0.01;
[sFeat,Sf,Nf,curve]=jGA(feat,label,N,T,CR,MR);
% Plot convergence curve
figure(); plot(1:T,curve); xlabel('Number of Iterations');
ylabel('Fitness Value'); title('GA'); grid on;
%% (Method 2) GA version 2
% Assume the chromosomes with CR probabilities are used in selection process
close all; N=10; T=100; CR=0.8; MR=0.01;
[sFeat,Sf,Nf,curve]=jGA2(feat,label,N,T,CR,MR);
% Plot convergence curve
figure(); plot(1:T,curve); xlabel('Number of Iterations');
ylabel('Fitness Value'); title('GA2'); grid on;
```

fitness function.m

```
function fitness=jFitnessFunction(feat,label,X)
% Parameter setting for k-value of KNN
k=5;
% Parameter setting for number of cross-validation
kfold=2;
% Error rate
fitness=jwrapperKNN(feat(:,X==1),label,k,kfold);
end
% Perform KNN with k-folds cross-validation
function ER=jwrapperKNN(feat,label,k,kfold)
Model=fitcknn(feat,label,'NumNeighbors',k,'Distance','euclidean');
C=crossval(Model,'KFold',kfold);
% Error rate
ER=kfoldLoss(C);
End
```

jGA.m

```
function [sFeat,Sf,Nf,curve]=jGA(feat,label,N,T,CR,MR)
fun=@jFitnessFunction;
% Number of dimensions
D=size(feat,2);
% Initial population
X=zeros(N,D); fit=zeros(1,N);
for i=1:N
  for d=1:D
    if rand() > 0.5
      X(i,d)=1;
    end
  end
end
% Fitness
for i=1:N
  fit(i)=fun(feat,label,X(i,:));
```

```
end
% Pre
curve=inf; t=1;
figure(1); clf; axis([1 100 0 0.5]); xlabel('Number of Iterations');
ylabel('Fitness Value'); title('Convergence Curve'); grid on;
%---Generations start----------------------------------------------------
while t <= T
   % Convert error to accuracy (inverse of fitness)
 Ifit=1-fit;
 % Get probability
 Prob=Ifit/sum(Ifit);
 % {1} Crossover
 X1=zeros(1,D); X2=zeros(1,D); z=1;
 for i=1:N
  if rand() < CR
    % Select two parents
    k1=jRouletteWheelSelection(Prob); k2=jRouletteWheelSelection(Prob);
    % Store parents
    P1=X(k1,:); P2=X(k2,:);
    % Random select one crossover point
    ind=randi([1,D]);
    % Single point crossover between 2 parents
    X1(z,:)=[P1(1:ind),P2(ind+1:D)];
    X2(z,:)=[P2(1:ind),P1(ind+1:D)]; z=z+1;
  end
 end
 % Union
 Xnew=[X1;X2]; Nc=size(Xnew,1); Fnew=zeros(1,Nc);
 % {2} Mutation
 for i=1:Nc
  for d=1:D
    if rand() <= MR
      % Mutate from '0' to '1' or '1' to '0'
      Xnew(i,d)=1-Xnew(i,d);
```

```
    end
   end
   % Fitness
   Fnew(i)=fun(feat,label,Xnew(i,:));
  end
  % Merge population
  XX=[X;Xnew]; FF=[fit,Fnew];
  % Select N best solution
  [FF,idx]=sort(FF); X=XX(idx(1:N),:); fit=FF(1:N);
  % Best chromosome
  Xgb=X(1,:); fitG=fit(1); curve(t)=fitG;
  % Plot convergence curve
  pause(0.000000001); hold on;
  CG=plot(t,fitG,'Color','r','Marker','.'); set(CG,'MarkerSize',5);
  t=t+1;
end
% Select features based on selected index
Pos=1:D; Sf=Pos(Xgb==1); Nf=length(Sf); sFeat=feat(:,Sf);
end
%---Call Function-------------------------------------------------------
function Route=jRouletteWheelSelection(Prob)
% Cummulative summation
C=cumsum(Prob);
% Random one value, most probability value [0~1]
P=rand();
% Route wheel
for i=1:length(C)
   if C(i) > P
   Route=i; break;
  end
end
end
```

jGA2.m

```
function [sFeat,Sf,Nf,curve]=jGA2(feat,label,N,T,CR,MR)
```

```
fun=@jFitnessFunction;
% Number of dimensions
D=size(feat,2);
% Initial population
X=zeros(N,D); fit=zeros(1,N);
for i=1:N
  for d=1:D
    if rand() > 0.5
      X(i,d)=1;
    end
  end
end
% Number of crossover
CR=round(CR*N);
% Fitness
for i=1:N
  fit(i)=fun(feat,label,X(i,:));
end
% Pre
X1=zeros(CR,D); X2=zeros(CR,D); Fnew=zeros(1,2*CR); curve=inf; t=1;
figure(1); clf; axis([1 100 0 0.5]); xlabel('Number of Iterations');
ylabel('Fitness Value'); title('Convergence Curve'); grid on;
%---Generations start----------------------------------------------------
while t <= T
    % Convert error to accuracy (inverse of fitness)
  Ifit=1-fit;
  % Get probability
  Prob=Ifit/sum(Ifit);
  % {1} Crossover
  for i=1:CR
    % Select two parents
    k1=jRouletteWheelSelection(Prob); k2=jRouletteWheelSelection(Prob);
    % Store parents
    P1=X(k1,:); P2=X(k2,:);
```

```
    % Random select one crossover point
    ind=randi([1,D]);
    % Single point crossover between 2 parents
    X1(i,:)=[P1(1:ind),P2(ind+1:D)]; X2(i,:)=[P2(1:ind),P1(ind+1:D)];
  end
  % Union
  Xnew=[X1;X2];
  % {2} Mutation
  for i=1:2*CR
    for d=1:D
      if rand() <= MR
     % Mutate from '0' to '1' or '1' to '0'
        Xnew(i,d)=1-Xnew(i,d);
      end
    end
    % Fitness
    Fnew(i)=fun(feat,label,Xnew(i,:));
  end
  % Merge population
  XX=[X;Xnew]; FF=[fit,Fnew];
  % Select N best solution
  [FF,idx]=sort(FF); X=XX(idx(1:N),:); fit=FF(1:N);
  % Best chromosome
  Xgb=X(1,:); fitG=fit(1); curve(t)=fitG;
  % Plot convergence curve
  pause(0.000000001); hold on;
  CG=plot(t,fitG,'Color','r','Marker','.'); set(CG,'MarkerSize',5);
  t=t+1;
end
% Select features based on selected index
Pos=1:D; Sf=Pos(Xgb==1); Nf=length(Sf); sFeat=feat(:,Sf);
end
%---Call Function-------------------------------------------------------
function Route=jRouletteWheelSelection(Prob)
```

```
% Cummulative summation
C=cumsum(Prob);
% Random one value, most probability value [0~1]
P=rand();
% Route wheel
for i=1:length(C)
   if C(i) > P
   Route=i; break;
  end
end
end
```

5.5 CLASSIFIER COMPARISON ANALYSIS

5.5.1 multiSVM

```
function [result] = multisvm(TrainingSet,GroupTrain,TestSet,Testlabels)
u=unique(GroupTrain);
numClasses=length(u);
result = zeros(length(TestSet(:,1)),1);
%build models
for k=1:numClasses
   %Vectorized statement that binarizes Group
   %where 1 is the current class and 0 is all other classes
   G1vAll=(GroupTrain==u(k));
   models(k) = svmtrain(TrainingSet,G1vAll);
end
%classify test cases
for j=1:size(TestSet,1)
   for k=1:numClasses
      if(svmclassify(models(k),TestSet(j,:)))
         break;
      end
   end
   result(j) = k;
end
 confMat=myconfusionmat(Testlabels,result);
```

```
 disp('confusion matrix:')
 disp(confMat)
 conf=sum(result==Testlabels)/length(result);
 disp(['accuracy = ',num2str(conf*100),'%'])
```

5.5.2 KNN

```
function [predicted_labels,nn_index,accuracy] = cknn(k,data,labels,t_data,t_labels)
if nargin < 4
   error('Too few input arguments.')
elseif nargin < 5
   t_labels=[];
   accuracy=0;
end
if size(data,2)~=size(t_data,2)
   error('data should have the same dimensionality');
end
if mod(k,2)==0
   error('to reduce the chance of ties, please choose odd k');
end
%initialization
predicted_labels=zeros(size(t_data,1),1);
ed=zeros(size(t_data,1),size(data,1)); %ed: (MxN) euclidean distances
ind=zeros(size(t_data,1),size(data,1)); %corresponding indices (MxN)
k_nn=zeros(size(t_data,1),k); %k-nearest neighbors for testing sample (Mxk)
%calc euclidean distances between each testing data point and the training
%data samples
for test_point=1:size(t_data,1)
   for train_point=1:size(data,1)
      %calc and store sorted euclidean distances with corresponding indices
      ed(test_point,train_point)=sqrt(...
         sum((t_data(test_point,:)-data(train_point,:)).^2));
   end
   [ed(test_point,:),ind(test_point,:)]=sort(ed(test_point,:));
end
```

```
%find the nearest k for each data point of the testing data
k_nn=ind(:,1:k);
nn_index=k_nn(:,1);
%get the majority vote
for i=1:size(k_nn,1)
    options=unique(labels(k_nn(i,:)'));
    max_count=0;
    max_label=0;
    for j=1:length(options)
        L=length(find(labels(k_nn(i,:)')==options(j)));
        if L>max_count
            max_label=options(j);
            max_count=L;
        end
    end
    predicted_labels(i)=max_label;
end
%calculate the classification accuracy
if isempty(t_labels)==0
    accuracy=length(find(predicted_labels==t_labels))/size(t_data,1);
end
confMat=myconfusionmat(t_labels,predicted_labels);
disp('confusion matrix:')
disp(confMat)
conf=sum(predicted_labels==t_labels)/length(predicted_labels);
disp(['aaaccuracy = ',num2str(conf*100),'%'])
```

5.5.3 Naive Bayes

```
% NAIVE BAYES CLASSIFIER
clear
tic
disp('--- start ---')
distr='normal';
distr='kernel';
```

```
% read data
nav = dataset('xlsfile', 'nav2.xlsx');
X = double(nav(:,1:10));
Y = double(nav(:,11));
% Create a cvpartition object that defined the folds
c = cvpartition(Y,'holdout',.439);
% Create a training set
 x = double(nav(1:550,2:10));
 y = double(nav(1:550,11));
% test set
u= double(nav(551:979,2:10));
v = double(nav(551:979,11));

yu=unique(y);
nc=length(yu); % number of classes
ni=size(x,2); % independent variables
ns=length(v); % test set
% compute class probability
for i=1:nc
   fy(i)=sum(double(y==yu(i)))/length(y);
end
switch distr

   case 'normal'

      % normal distribution
      % parameters from training set
      for i=1:nc
         xi=x((y==yu(i)),:);
         mu(i,:)=mean(xi,1);
         sigma(i,:)=std(xi,1);
      end
      % probability for test set
      for j=1:ns
```

```
            fu=normcdf(ones(nc,1)*u(j,:),mu,sigma);
            P(j,:)=fy.*prod(fu,2)';
        end
    case 'kernel'
        % kernel distribution
        % probability of test set estimated from training set
        for i=1:nc
            for k=1:ni
                xi=x(y==yu(i),k);
                ui=u(:,k);
                fuStruct(i,k).f=ksdensity(xi,ui);
            end
        end
        % re-structure
        for i=1:ns
            for j=1:nc
                for k=1:ni
                    fu(j,k)=fuStruct(j,k).f(i);
                end
            end
            P(i,:)=fy.*prod(fu,2)';
        end
    otherwise

        disp('invalid distribution stated')
        return
end
% get predicted output for test set
[pv0,id]=max(P,[],2);
for i=1:length(id)
    pv(i,1)=yu(id(i));
end
% compare predicted output with actual output from test data
confMat=myconfusionmat(v,pv);
```

```
disp('confusion matrix:')
disp(confMat)
conf=sum(pv==v)/length(pv);
disp(['accuracy = ',num2str(conf*100),'%'])
```

3.3 Summary

This chapter explains details of system analysis, function and nonfunctional requirements along with hardware and software requirements. Apart from that a clear picture about each modules is explained clearly with the certain proof required for it.

6. Testing

6.1 Introduction CHAPTER 6

This chapter gives a clear picture about the various testing performed for our project. Testing is one of the crucial part for a software development because it is the only way to find whether the project meets all the requirements without any error and checks that it can also be used out of the environment also.

6.2 Software Testing

Software testing is a process of evaluating a software system or application to detect any defects or errors in the system. It involves validating that the system meets its functional and non-functional requirements and performs as expected. The goal of software testing is to identify and eliminate any issues in the system before it is released to end-users. Testing is an essential part of the software development lifecycle and ensures the quality and reliability of the software.

The main objective of software testing are:

1. To ensure that the software meets its functional and non-functional requirements.
2. To detect and fix defects or errors in the software before it is released to end-users.
3. To improve the quality and reliability of the software.
4. To enhance the user experience by ensuring the software is easy to use and navigate.
5. To increase the efficiency of the software by identifying and removing any performance bottlenecks.
6. To ensure the software is secure and not vulnerable to attacks.
7. To reduce the overall cost of software development by identifying issues early in the development process.

6.3 Basic Types of Testing

There are several types of software testing, including:

1. **Unit Testing**: Testing individual components of the software to ensure they function as intended.
2. **Integration Testing**: Testing how multiple components of the software interact with each other.
3. **System Testing**: Testing the entire system to ensure it meets functional and non-functional requirements.

4. **Acceptance Testing**: Testing the software with real-world scenarios to ensure it meets user requirements.
5. **Regression Testing**: Testing previously working functionality after changes to ensure no new issues have been introduced.
6. **Performance Testing**: Testing the system's performance under varying load conditions.
7. **Security Testing**: Testing the system's security and vulnerability to attacks.

6.4 Testing Techniques

The testing technique used in our projects are:

- Unit testing
- Integration testing
- System testing
- Performance testing

6.4.1 Unit Testing

Unit testing is a type of software testing where individual units or components of the software are tested in isolation to ensure they function as expected. Each unit is tested separately from the rest of the system to identify and isolate any defects or errors. The goal of unit testing is to verify that each unit performs as intended and meets its functional requirements. This type of testing is typically performed by developers during the development phase and helps to ensure the quality and reliability of the software.

Reason to use unit testing:

1. Catching defects early: Unit testing helps identify defects or errors in the code early in the development process, reducing the overall cost of fixing them later.
2. Ensuring code quality: Unit testing helps ensure the code meets the expected requirements and follows coding standards, increasing the overall quality of the software.
3. Facilitating change: Unit testing helps identify the impact of changes to the codebase, making it easier to make changes without breaking the software.
4. Encouraging modularity: Unit testing encourages developers to create modular code that can be tested in isolation, making it easier to identify and fix issues.

5. Enabling automation: Unit testing can be automated, reducing the overall time and effort required to test the code and ensuring consistent testing.
6. Providing documentation: Unit tests serve as documentation of the expected behaviour of the code, making it easier for new developers to understand the codebase.

6.4.2 Integration Testing

Integration testing is a type of software testing where multiple components or modules of the software are combined and tested together as a group to ensure they work correctly and seamlessly. The goal of integration testing is to identify any defects or errors that occur when different components are combined, and to ensure that they interact with each other as intended. This type of testing is typically performed after unit testing and before system testing. Integration testing helps to identify issues early in the development process, reducing the overall cost and time required for testing and development.

Reason to use integration testing:

1. Verifying component interaction: Integration testing helps ensure that different components or modules of the software work together as intended and interact correctly.
2. Identifying defects early: Integration testing helps identify defects or errors that occur when different components are combined, reducing the overall cost of fixing them later.
3. Ensuring reliability: Integration testing helps ensure that the software is reliable and functions correctly as a whole.
4. Improving overall quality: Integration testing helps improve the overall quality of the software by identifying and fixing issues early in the development process.
5. Facilitating change: Integration testing helps identify the impact of changes to the codebase on the software as a whole, making it easier to make changes without breaking the software.
6. Meeting user expectations: Integration testing helps ensure that the software meets the functional and non-functional requirements and meets user expectations.

6.4.2 System Testing

System testing is a type of software testing that is performed on a complete, integrated system to evaluate its compliance with functional and non-functional requirements. It verifies that the software system meets its intended purpose, functionality, performance, and reliability. System

testing is usually conducted after integration testing is completed, and is performed on the entire system, as opposed to individual components or modules. The goal of system testing is to identify defects and errors that occur when the system is tested as a whole, and to ensure that the system meets the user's requirements and expectations.

Reason to use system testing:

1. Ensure that the system meets functional and non-functional requirements: System testing verifies that the software system meets the functional and non-functional requirements specified in the requirements document.
2. Identify defects early: System testing helps identify defects or issues that occur when the system is tested as a whole, reducing the overall cost of fixing them later.
3. Validate system functionality: System testing validates that the system works as intended and all its features are functioning correctly.
4. Ensure reliability and stability: System testing helps ensure that the software system is reliable, stable, and performs correctly under different load conditions.
5. Validate security and compatibility: System testing validates that the system is secure and compatible with other systems and platforms.
6. Meet user expectations: System testing helps ensure that the software system meets the user's requirements and expectations, and delivers the desired user experience.

6.4.2 Performance Testing

Performance testing is a type of software testing that is designed to evaluate the system's responsiveness, speed, stability, and scalability under various workloads. It measures the performance of the system and identifies performance bottlenecks and issues that can impact user experience. Performance testing can help determine how much load the system can handle before it becomes unstable, how fast it can perform under different load conditions, and how quickly it can recover from failures. This type of testing is typically conducted using specialized tools that simulate real-world usage scenarios and generate metrics and reports that can be used to optimize the system's performance.

Reason to use performance testing:

1. Ensure that the system meets functional and non-functional requirements: System testing verifies that the software system meets the functional and non-functional requirements specified in the requirements document.

2. Identify defects early: System testing helps identify defects or issues that occur when the system is tested as a whole, reducing the overall cost of fixing them later.
3. Validate system functionality: System testing validates that the system works as intended and all its features are functioning correctly.
4. Ensure reliability and stability: System testing helps ensure that the software system is reliable, stable, and performs correctly under different load conditions.
5. Validate security and compatibility: System testing validates that the system is secure and compatible with other systems and platforms.
6. Meet user expectations: System testing helps ensure that the software system meets the user's requirements and expectations and delivers the desired user experience.

6.5 Summary

This chapter gave you an immense idea about the software testing and various testing methods that are used in our software and reasons to choose some of the particular testing method.

7. SCREENSHOTS

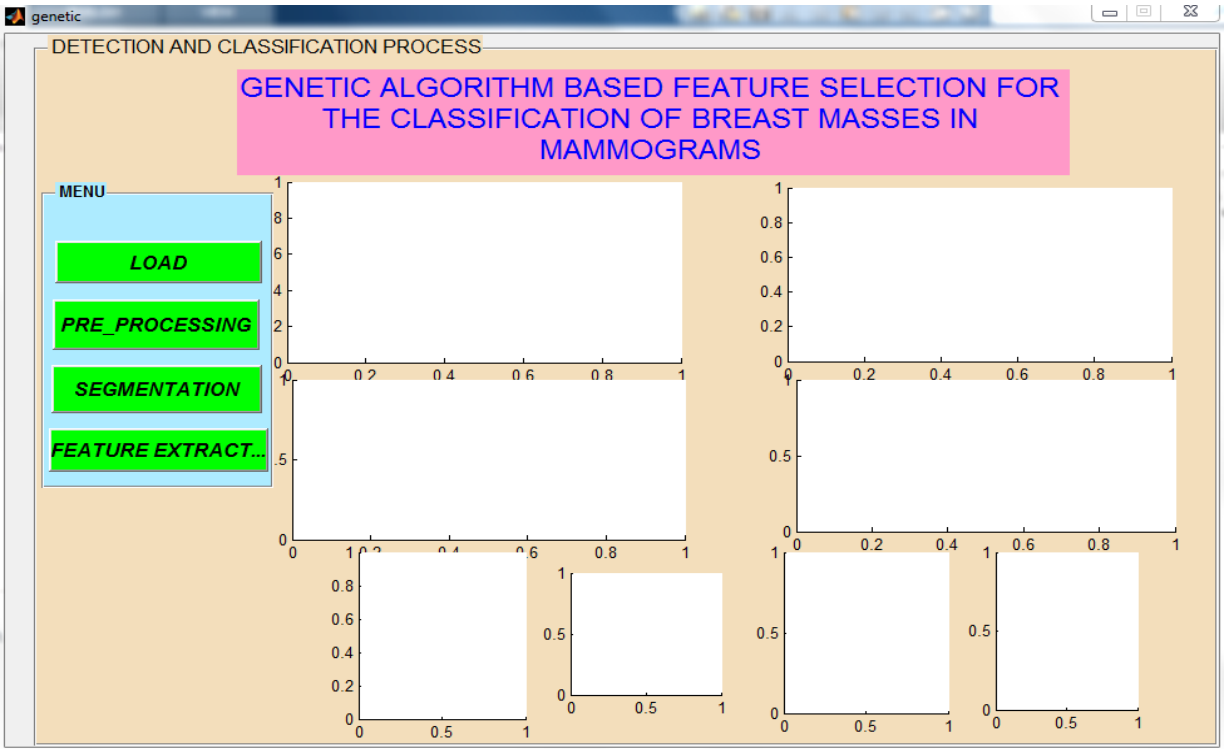


Figure 7.1: GUI DESIGN

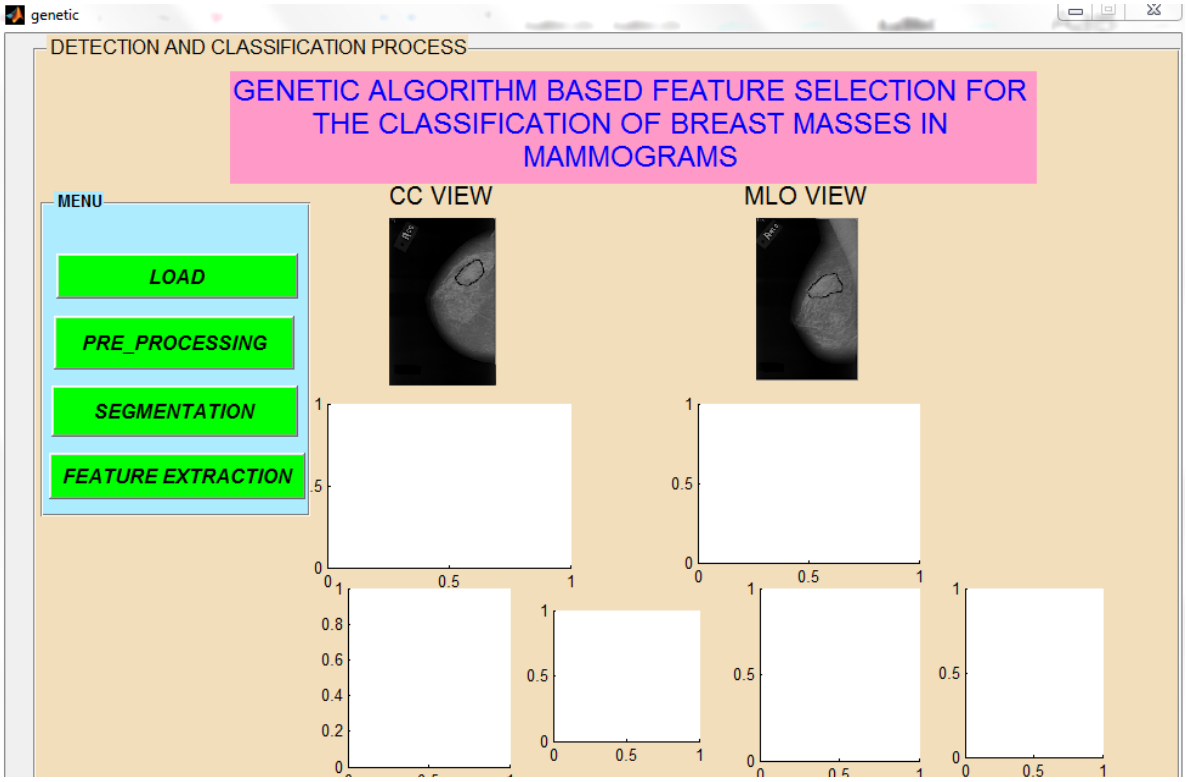


Figure 7.2: INPUT IMAGE

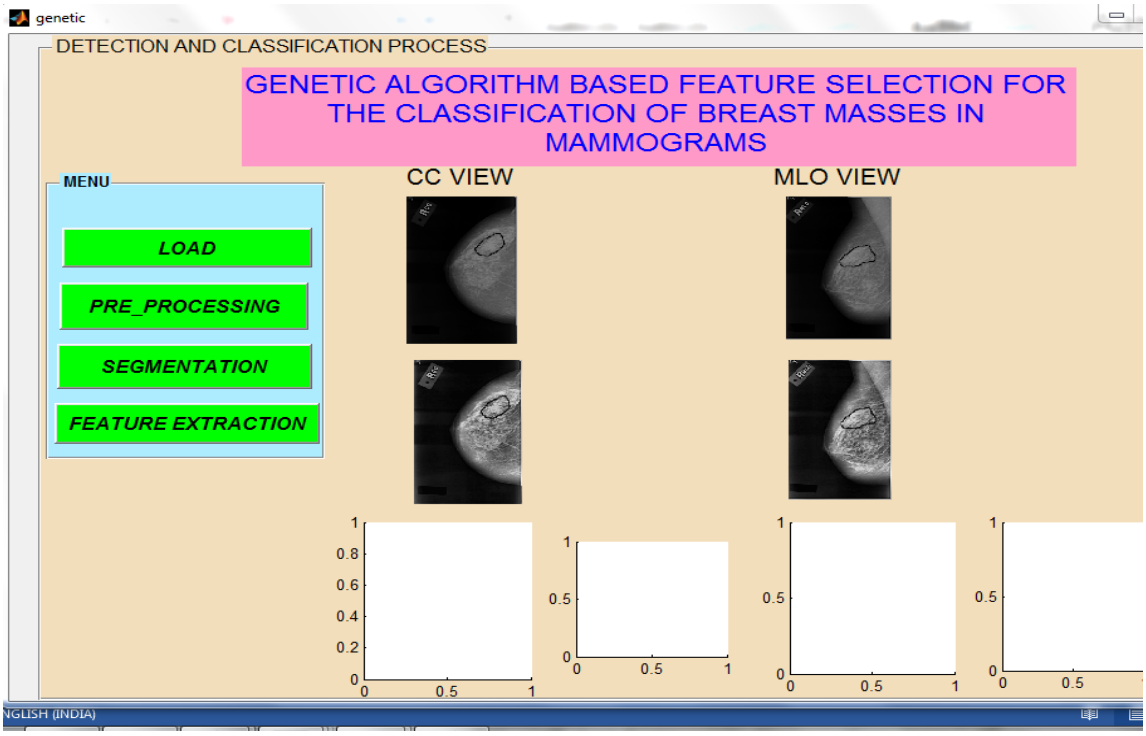


Figure 7.3: PRE-PROCESSED IMAGE

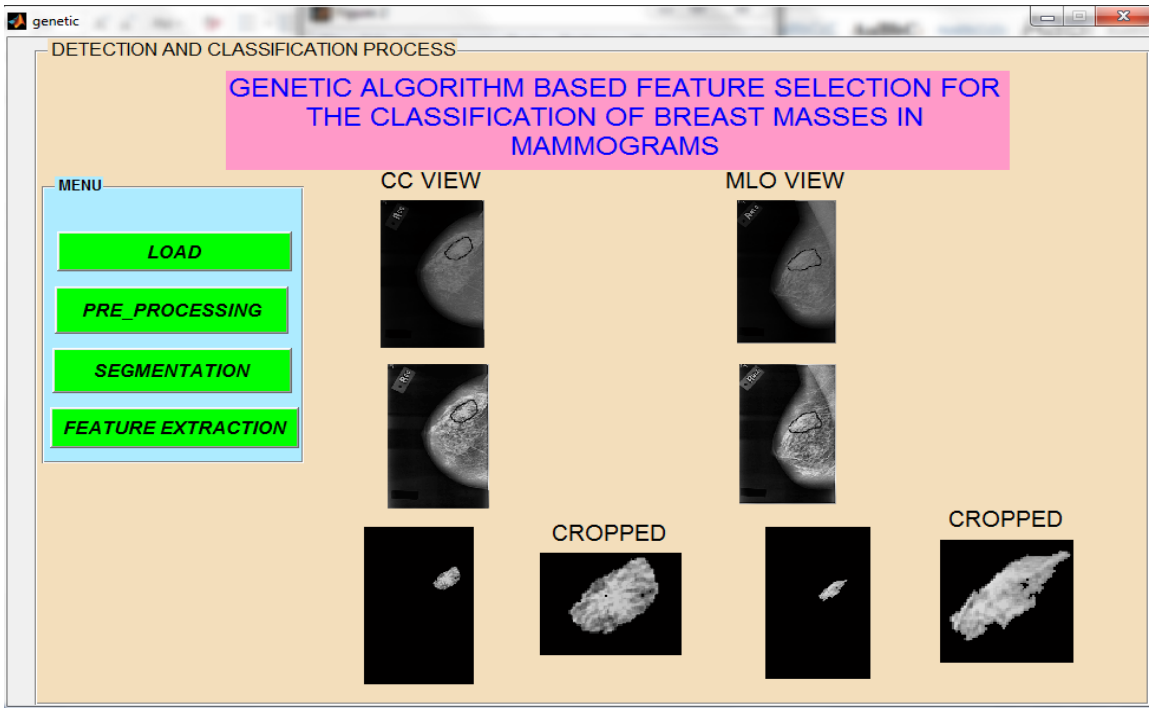
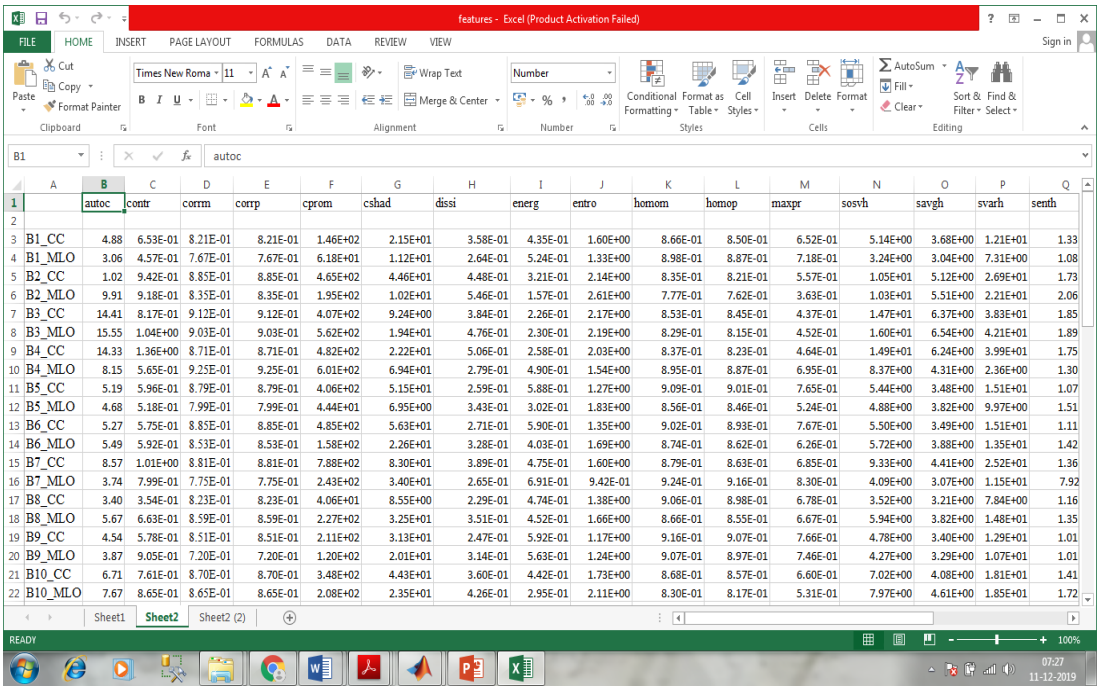


Figure 7.4: SEGMENTED IMAGE



	autoc	contr	corrm	corrp	cprom	cshad	dissi	energ	entro	homom	homop	maxpr	sosvh	savgh	svarh	senth
B1_CC	4.88	6.53E-01	8.21E-01	8.21E-01	1.46E+02	2.15E+01	3.58E-01	4.35E-01	1.60E+00	8.66E-01	8.50E-01	6.52E-01	5.14E+00	3.68E+00	1.21E+01	1.33
B1_MLO	3.06	4.57E-01	7.67E-01	7.67E-01	6.18E+01	1.12E+01	2.64E-01	5.24E-01	1.33E+00	8.98E-01	8.87E-01	7.18E-01	3.24E+00	3.04E+00	7.31E+00	1.08
B2_CC	1.02	9.42E-01	8.85E-01	8.85E-01	4.65E+02	4.46E+01	4.48E-01	3.21E-01	2.14E+00	8.35E-01	8.21E-01	5.57E-01	1.05E+01	5.12E+00	2.69E+01	1.73
B2_MLO	9.91	9.18E-01	8.35E-01	8.35E-01	1.95E+02	1.02E+01	5.46E-01	1.57E-01	2.61E+00	7.77E-01	7.62E-01	3.63E-01	1.03E+01	5.51E+00	2.21E+01	2.06
B3_CC	14.41	8.17E-01	9.12E-01	9.12E-01	4.07E+02	9.24E+00	3.84E-01	2.26E-01	2.17E+00	8.53E-01	8.45E-01	4.37E-01	1.47E+01	6.37E+00	3.83E+01	1.85
B3_MLO	15.55	1.04E+00	9.03E-01	9.03E-01	5.62E+02	1.94E+01	4.76E-01	2.30E-01	2.19E+00	8.29E-01	8.15E-01	4.52E-01	1.60E+01	6.54E+00	4.21E+01	1.89
B4_CC	14.33	1.36E+00	8.71E-01	8.71E-01	4.82E+02	2.22E+01	5.06E-01	2.58E-01	2.03E+00	8.37E-01	8.23E-01	4.64E-01	1.49E+01	6.24E+00	3.99E+01	1.75
B4_MLO	8.15	5.65E-01	9.25E-01	9.25E-01	6.01E+02	6.94E+01	2.79E-01	4.90E-01	1.54E+00	8.95E-01	8.87E-01	6.95E-01	8.37E+00	4.31E+00	2.36E+00	1.30
B5_CC	5.19	5.96E-01	8.79E-01	8.79E-01	4.06E+02	5.15E+01	2.59E-01	5.88E-01	1.27E+00	9.09E-01	9.01E-01	7.65E-01	5.44E+00	3.48E+00	1.51E+01	1.07
B5_MLO	4.68	5.18E-01	7.99E-01	7.99E-01	4.44E+01	6.95E+00	3.43E-01	3.02E-01	1.83E+00	8.56E-01	8.46E-01	5.24E-01	4.88E+00	3.82E+00	9.97E+00	1.51
B6_CC	5.27	5.75E-01	8.85E-01	8.85E-01	4.85E+02	5.63E+01	2.71E-01	5.90E-01	1.35E+00	9.02E-01	8.93E-01	7.67E-01	5.50E+00	3.49E+00	1.51E+01	1.11
B6_MLO	5.49	5.92E-01	8.53E-01	8.53E-01	1.58E+02	2.26E+01	3.28E-01	4.03E-01	1.69E+00	8.74E-01	8.62E-01	6.26E-01	5.72E+00	3.88E+00	1.35E+01	1.42
B7_CC	8.57	1.01E+00	8.81E-01	8.81E-01	7.88E+02	8.30E+01	3.89E-01	4.75E-01	1.60E+00	8.79E-01	8.63E-01	6.85E-01	9.33E+00	4.41E+00	2.52E+01	1.36
B7_MLO	3.74	7.99E-01	7.75E-01	7.75E-01	2.43E+02	3.40E+01	2.65E-01	6.91E-01	9.42E-01	9.24E-01	9.16E-01	8.30E-01	4.09E+00	3.07E+00	1.15E+01	7.92
B8_CC	3.40	3.54E-01	8.23E-01	8.23E-01	4.06E+01	8.55E+00	2.29E-01	4.74E-01	1.38E+00	9.06E-01	8.98E-01	6.78E-01	3.52E+00	3.21E+00	7.84E+00	1.16
B8_MLO	5.67	6.63E-01	8.59E-01	8.59E-01	2.27E+02	3.25E+01	3.51E-01	4.52E-01	1.66E+00	8.66E-01	8.55E-01	6.67E-01	5.94E+00	3.82E+00	1.48E+01	1.35
B9_CC	4.54	5.78E-01	8.51E-01	8.51E-01	2.11E+02	3.13E+01	2.47E-01	5.92E-01	1.17E+00	9.16E-01	9.07E-01	7.66E-01	4.78E+00	3.40E+00	1.29E+01	1.01
B9_MLO	3.87	9.05E-01	7.20E-01	7.20E-01	1.20E+02	2.01E+01	3.14E-01	5.63E-01	1.24E+00	9.07E-01	8.97E-01	7.46E-01	4.27E+00	3.29E+00	1.07E+01	1.01
B10_CC	6.71	7.61E-01	8.70E-01	8.70E-01	3.48E+02	4.43E+01	3.60E-01	4.42E-01	1.73E+00	8.68E-01	8.57E-01	6.60E-01	7.02E+00	4.08E+00	1.81E+01	1.41
B10_MLO	7.67	8.65E-01	8.65E-01	8.65E-01	2.08E+02	2.35E+01	4.26E-01	2.95E-01	2.11E+00	8.30E-01	8.17E-01	5.31E-01	7.97E+00	4.61E+00	1.85E+01	1.72

Figure 7.5: EXTRACTED FEATURES

Table 7.1. OVERALL ACCURACY

	Classifier		
Feature selector	**KNN**	**multiSVM**	**NB**
GA	92.07	90.68	86.95
t-Test	88.81	87.64	84.38

Table 7.2. ACCURACY PER CLASS

Feature Selector	**Classifier**	**Benign**	**Malignant**	**Normal**
Genetic Algorithm	**kNN**	93.24	95.21	94.25
	multiSVM	87.84	90.42	90.8
	Naïve Bayes	91.22	92.81	91.95
t-Test	**kNN**	91.22	92.81	93.1
	multiSVM	85.81	88.02	88
	Naïve Bayes	89.86	91.02	90.8

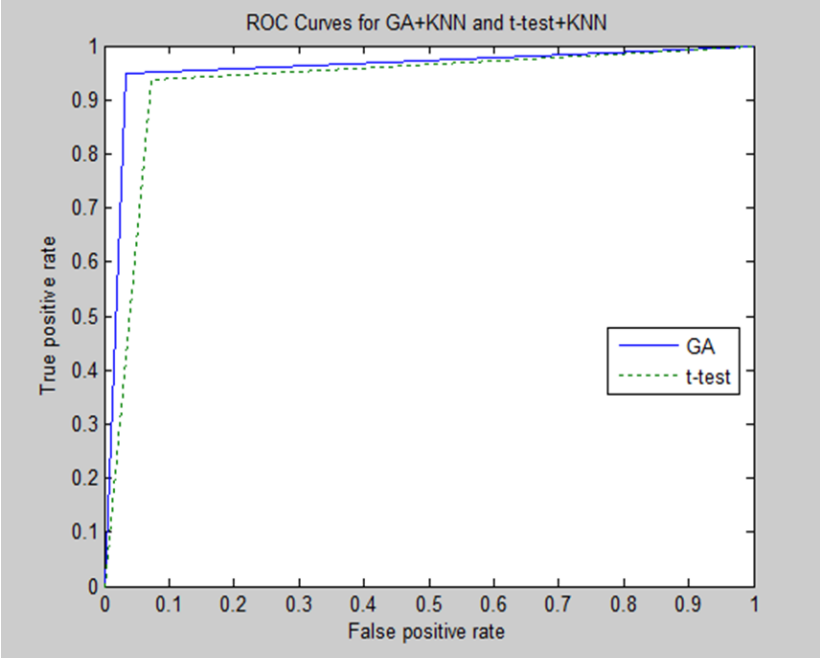


Figure 7.6 ROC CURVE FOR KNN CLASSIFIER

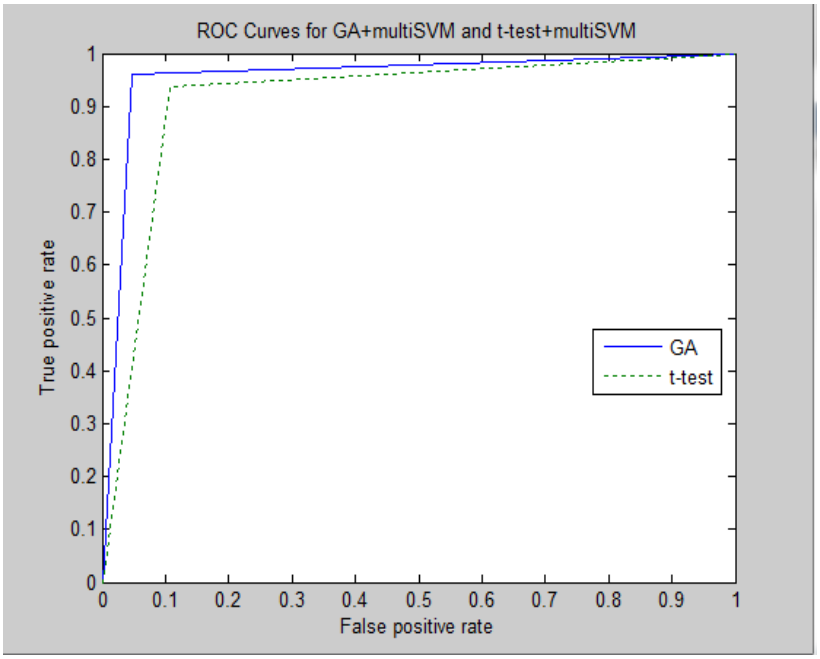


Figure 7.7 ROC CURVE FOR MULTISVM CLASSIFIER

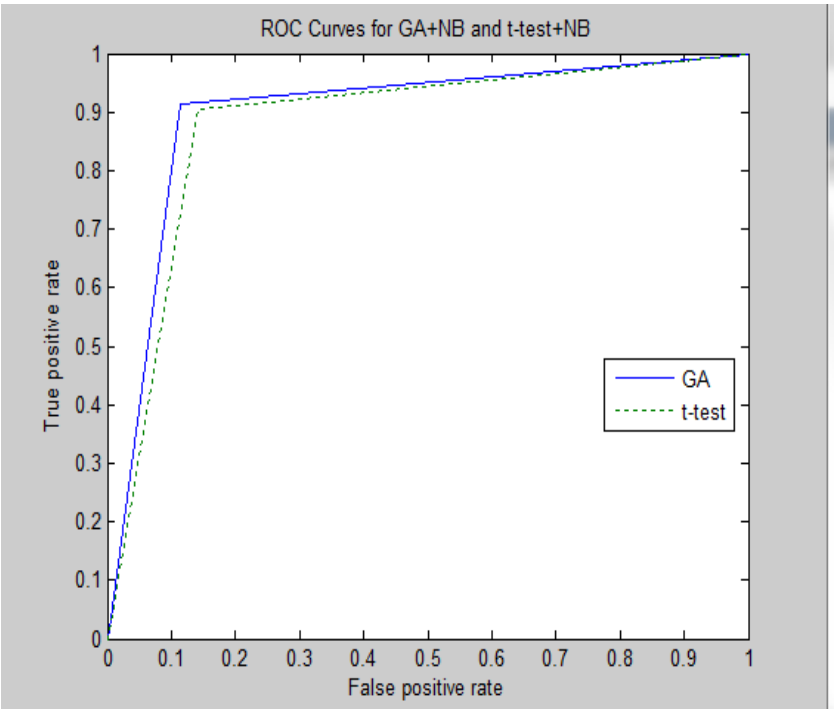


Figure 7.8 ROC CURVE FOR NAIVE BAYES CLASSIFIER

8. CONCLUSION

8.1 Conclusion

CHAPTER 8

In this work, a 3-class classification system was proposed to classify breast masses in mammogram images into benign, malignant, and normal. 979 images were collected from the DDSM database with 296 benign images, 334 malignant images, 349 normal images. Adaptive Histogram Equalization technique was used for pre-processing the collected images. The Region of Interest was segmented from the pre-processed image by using the region growing algorithm. From the segmented image, the GLCM features were extracted for classifying the images. The optimal features were selected by using a genetic algorithm and t-test. To evaluate the performance, three classifiers were used and the results were compared. These are, multiSVM, kNN, and Naïve Bayes classifiers. Six combinations were evaluated. GA+multiSVM, GA+kNN, GA+Naïve Bayes, t-test+multiSVM, t-test+kNN, and t-test+Naïve Bayes. From the experimentation, it was concluded that the combination of the GA and kNN algorithm produced promising results.

8.2 Future Enhancement

In the future, the algorithm can be extended to work with more than 5000 images. An automatic segmentation algorithm can also be developed. The feature extraction techniques can be developed with the focus on BIRADS results. Another direction in the future, will be the investigation of non-linear feature reduction techniques. The classification algorithms can be extended to work with other abnormalities like micro-calcifications, bilateral asymmetry, and architectural distortion in mammogram images.

9. APPENDIX

9.1 MATLAB CHAPTER 9

MATLAB-based technique provides easy debugging with extensive data analysis and visualization, easy implementation and algorithmic-testing without recompilation. Besides, MATLAB's computational codes can be enhanced and exploited to process and create simulations of both still and video images. Moreover, MATLAB codes are much concise compared to C++, thus making it easier for perusing and troubleshooting. MATLAB can handle errors prior to execution by proposing various ways to make the codes faster. The proposed technique enables advanced image processing operations such as image cropping/resizing, image denoising, blur removal, and image sharpening

Recent technological advances have triggered a growing research interest in image processing techniques. Generally, image processing techniques are rapidly increasing among technologies, and form an important research area in Engineering and Computing disciplines. This technique involves operations on an image for the purpose of enhancing the quality of the image or extracting some useful and desirable information from the image. It describes a type of signal processing operation on an image in which the image (e.g. photo or video frame) is an input signal with output characteristics of the original/input image.

Image processing has been described as a mathematical operation of a computer on a two-dimensional photo and a compendium of signal processing operations on an image that transform it into a digital form for the purpose of quality enhancement and extraction of useful information from it. The above definition mathematically presents an image as a function of two real variables in the x and y coordinates which can be digitally processed with the brightness of the image represented by the amplitude of the function. The image data is digitally processed after being sampled and converted into a matrix of numbers which are quantized and digitally represented. Accordingly, captured image data must undergo some critical phases and include image pre-processing, image enhancement and display, and extraction of useful information from the image data. Incidentally, image processing technique has become more relevant in our daily lives as it has great impact and applications in technical fields exploiting different types of electronic devices including computers, digital cameras, and mobile phones. Image data can take the form of grayscale image, index image, true color image, and binary image. A grayscale image depicts one whose pixel value is a single sample that indicates only the amount of light. This type of image consists essentially of shades of gray with variations ranging from black showing the weakest intensity to white showing the strongest intensity.

Although digital images can be produced by exploiting different image processing applications running in several physical devices (such as ultrasonic, x-ray devices, cameras, and electron microscopes), MATLAB-based image processing can be exploited to give the desired quality suitable for a two-dimensional image.

- MATLAB is a high-level programming language with desirable features that are fundamental to both the computing and engineering disciplines with algorithms that have potentials for a wide range of applications.
- Owing to the rapid growth of technology, many scientists and engineers currently have ubiquitous access to multimedia software and hardware devices running image processing applications including 3D visualization. However, majority of these are limited in applications such that they require the support of other applications for image processing operations.
- Thus, this study is aimed at bringing to the fore the potentials of some of the novel features inherent in MATLAB as a standalone application tool for signal processing operations including digital image processing.

This work presents an image processing technique using MATLAB-based analytics. Compared to the traditional and state-of-the-art image processing techniques and applications, MATLAB gives several advantages. For instance, MATLAB provides easy implementation and algorithmic testing without recompilation as well as easy debugging with extensive data analysis and visualization. In addition, MATLAB's computational codes can easily be enhanced and exploited to process and create simulations of both still and video images. Compared to c++, MATLAB's codes are much concise thus enabling easy perusing and troubleshooting operations. Perhaps, the most fascinating feature of MATLAB-based image processing technique is its ability to handle errors prior to execution by proposing various ways to make the codes faster. Besides, MATLAB provides easy development of computational codes, provides a large database of built-in algorithms. Moreover, MATLAB performs mathematical and numerical manipulation for the implementation of image processing techniques, and is also useful for successful documentation step-by-step image processing operations. Furthermore, maintenance of numerical precision is possible through the enhancement process of image in MATLAB. Image scrutinization and testing are also possible because MATLAB provides a quicker access to the image processing source-codes in its toolbox. This paper studies an updated performance evaluation of MATLAB-based image processing applications using the latest MATLAB toolbox. The present study is meant to give readers and researchers a broader perspective of the latest image processing applications running in MATLAB. An empirical method of image processing with the 2D-DCT is also provided.

Figure 9.1 is the block diagram of an image processing technique. Basically, three steps are involved in image processing which are stated as shown :

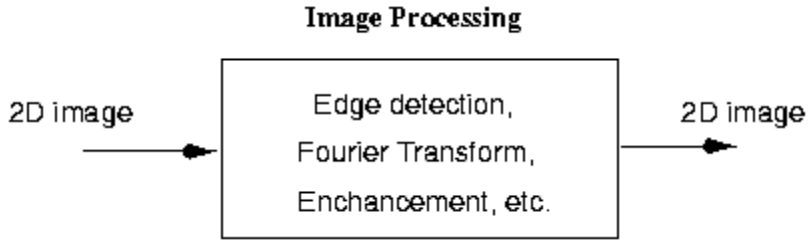


Figure 9.1: The block diagram of a 2D image processing technique

- Image acquisition. This involves importation of the image using the image acquisition tools such as optical scanner or digital photography
- Image analysis and manipulation
- Outputting the result of the image analysis

10. REFERENCE

[1] ZEXIAN HUANG ,DAQI CHEN, “A Breast Cancer Diagnosis Method Based on VIM Feature Selection and Hierarchical Clustering Random Forest Algorithm”, IEEE Access VOLUME 10, 2022

[2] Sweta Bhise , Simran Bepari , Shrutika Gadekar, “Breast Cancer Detection using Machine Learning Techniques”, International Journal of Engineering Research & Technology (IJERT) ISSN: 2278-0181 Vol. 10 Issue 07, July-2021

[3] About Breast Cancer 2019. <https://www.cancer.org/cancer/breast-cancer/about/what-is-breast-cancer.html> [10 May 2019]

[4] A discussion of conventional mammography 2019. Available from: <https://breast-cancer.ca/mammopics> [2 July 2019].

[5] Aditya A. Shastri, Deepti Tamrakar, Kapil Ahuja. “Density-wise two stage mammogram classification using texture exploiting descriptors”. Expert systems with applications, Volume 99, pp.71–82, 2018.

[6] Abdel-Nasser, M., Rashwan, H. A., Puig, D., & Moreno, A. “Analysis of tissue abnormality and breast density in mammographic images using a uniform local directional pattern”. Expert Systems with Applications, Volume 42, Issue 24, pp.9499–9511, 2018.

[7] Oliver, A., Tortajada, M., Llado, X., Freixenet, J., Ganau, S., Tortajada, L.,Vilagran, M., Sents, M., & Mart, R. “Breast density analysis using an automatic density segmentation algorithm”. Journal of Digital Imaging, pp.1-9, 2018.

[8] Rouhi, R., Jafari, M., Kasaei, S. and Keshavarzian, P. “Benign and malignant breast tumors classification based on region growing and CNN segmentation”. Expert Systems with Applications, Volume 42, Issue 3, pp.990–1002, 2017.

[9] Beura, S. Majhi, B, & Dash, R. “Mammogram classification using two dimensional discrete wavelet transform and gray-level co-occurrence matrix for detection of breast cancer”. Neurocomputing, Volume 154, pp. 1–14, 2018.

[10] Krishna Chaitanya Tatikonda, Chandra Mohan Bhuma, Srinivas Kumar Samayamantula. “The analysis of digital mammograms using HOG and GLCM features”. IEEE Trans. 9th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 2018.

[11] Yi-Chong Zeng. “Mammogram Density Classification using Double Support Vector Machines”. IEEE Trans. 7th Global Conference on Consumer Electronics (GCCE), 2018.

[12] Routray I, Rath N P. “Textural feature based classification of mammogram images using ANN”, Proceedings of 9th International Conference on Computing. Communication and Networking Technologies (ICCCNT). IEEE. 1–6, 2018.

[13] Amara Nedra, Muhammad Shoaib. “Detection and Classification of the Breast Abnormalities in Digital Mammograms via Linear Support Vector Machine”. IEEE 4th Middle East Conference on Biomedical Engineering (MECBME), 2018.

[14] Basma A. Mohamed and Nancy M. “Automatic Classification of Masses from Digital Mammograms”. IEEE Trans. 35th National Radio Science conference (NRSC), 2018.

[15] Al-Najdawi, N., Biltawi, M., and Tedmori, S., “Mammogram image visual enhancement, Mass Segmentation and Classification”, Applied Soft Computing, Volume 35, pp.175-185, 2015.