CHAPTER-1

INTRODUCTION

Breast cancer is very common and is considered as the second dangerous disease all over the world due to its death rate. Affected can survive if the disease diagnoses before the appearance of major physical changes in the body. Now a day, mammographic (X-ray of breast region) images are widely used for premature revealing of breast cancer. Aim of the proposed system is to design a Computer Aided Diagnosis system (CAD) used to distinguish between benign (non-cancerous) and malignant (cancerous) mammogram.

CAD system is used to help radiologist to increase his diagnosis accuracy. In the proposed system, texture features from mammogram were calculated using Gray Level Co-occurrence Matrix (GLCM) along 0°and DWT, from the calculate features most effective features having large contribution to achieve the desired output were chosen and applied to Probabilistic Neural Network (PNN) for training and classification, as ANN is widely use in various field such as, pattern recognition, medical diagnosis, machine learning and so on. For this research work mini-MIAS database is used and the overall sensitivity, specificity and accuracy achieved by using the proposed system is 99.3%, 100% and 99.4% respectively. In India and over the world, Cancer has become a deadly disease and more and more people are suffering from Cancer and a survey says one in every 30 women suffer from this disease in their lifetime and so basically the project was first thought of because of the increase in cases of breast cancer and one thing which is very important that if doctor can detect the Cancer at an early stage then there is an increased chances of it getting cured. So this project lays a foundation in making the detection of the cancer automated so that more and more people can get it diagnosed early so as get cured.

this research contributes to the ongoing efforts in computer-aided diagnosis (CAD) systems, aiming to bridge the gap between clinical need and technological advancement. It demonstrates the potential of deep learning to revolutionize how we approach cancer detection—making it more precise, scalable, and accessible to patients around the globe.

1.1 ABOUT SOFTWARE

MATLAB

MATLAB (**mat**rix **lab**oratory) is a multi-paradigm numerical computing environment and fourth-generation programming language. It uses the L-shaped membrane logo. Developed by MathWorks, MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, Java, and FORTRAN.

MATLAB® is a high-level language and interactive environment for numerical computation, visualization, and programming. Using MATLAB, we can analyze data, develop algorithms, and create models and applications. The language, tools, and built-in math functions enable us to explore multiple approaches and reach a solution faster than with spreadsheets or traditional programming languages, such as C/C++ or Java™.

Although MATLAB is intended primarily for numerical computing, an optional toolbox uses the MuPAD symbolic engine, allowing access to symbolic computing capabilities. An additional package, Simulink, adds graphical multi-domain simulation and Model-Based Design for dynamic and embedded systems.

In 2004, MATLAB had around one million users across industry and academia. MATLAB users come from various backgrounds of engineering, science, and economics. We 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.

Cleve Moler, the chairman of the computer science department at the University of New Mexico, started developing MATLAB in the late 1970s. He designed it to give his students access *to LINPACK and EISPACK without them having to learn Fortran. It soon spread to other universities and found a strong audience within the applied mathematics community. Jack Little, an engineer, was exposed to it during a visit Moler made to Stanford University in 1983. Recognizing its commercial potential, he joined with Moler and Steve Bangert. They rewrote MATLAB in C and founded MathWorks in 1984 to continue its development.

The MATLAB application is built around the MATLAB language, and most use of MATLAB involves typing MATLAB code into the Command Window (as an interactive mathematical shell), or executing text files containing MATLAB code.

GUIDE

GUIs (also known as graphical user interfaces or UIs) provide point-and-click control of software applications, eliminating the need to learn a language or type commands in order to run the application.

MATLAB apps are self-contained MATLAB programs with GUI front ends that automate a task or calculation. The GUI typically contains controls such as menus, toolbars, buttons, and sliders. Many MATLAB products, such as Curve Fitting Toolbox, Signal Processing Toolbox, and Control System Toolbox, include apps with custom user interfaces. You can also create patient own custom apps, including their corresponding UIs, for others to use.

MATLAB supports developing applications with graphical user interface features. MATLAB includes GUIDE (GUI development environment) for graphically designing GUIs. It also has tightly integrated graph-plotting features. For example the function *plot* can be used to produce a graph from two vectors *x* and *y*.

To create GUI in MATLAB follow the steps listed below.

1. Click on GUIDE button on toolbar in MATLAB launch pad.
2. Make patient choice of the type of GUI you need.
3. Make buttons and/or menus from the tools in GUIDE.
4. Double click on menu that is created in grey area of the GUIDE, property manager will open.
5. Change properties according to requirement and change the code if needed.

1.2 MATLAB's POWER OF COMPUTAIONALMATHMATICS

MATLAB is used in every facet of computational mathematics. Following are some commonly used mathematical calculations where it is used most commonly:

- Dealing with Matrices and Arrays
- 2-D and 3-D Plotting and graphics
- Linear Algebra
- Algebraic Equations
- Non-linear Functions
- Statistics
- Data Analysis
- Calculus and Differential Equations Numerical Calculations
- Integration

1.3 FEATURES OFMATLAB

Following are the basic features of MATLAB

It is a high-level language for numerical computation, visualization and application development.

- It also provides an interactive environment for iterative exploration, design and problem solving.
- It provides vast library of mathematical functions for linear algebra, statistics, Fourier analysis, filtering, optimization, numerical integration and solving ordinary differential equations.
- It provides built-in graphics for visualizing data and tools for creating custom plots.
- MATLAB's programming interface gives development tools for improving code quality, maintainability, and maximizing performance.
- It provides tools for building applications with custom graphical interfaces.

1.4 USES OF MATLAB

MATLAB is widely used as a computational tool in science and engineering encompassing the fields of physics, chemistry, math and all engineering streams. It is used in a range of applications including:

1.4.1 signal processing and Communications

1.4.2 image and video Processing

1.4.3 control systems

1.4.4 test and measurement

1.4.5 computational finance

1.4.6 computational biology

1.5 UNDERTANDING THE MATLABENVIRONMENT

MATLAB development IDE can be launched from the icon created on the desktop. The main working window in MATLAB is called the desktop. When MATLAB is started, the desktop appears in its default layout.

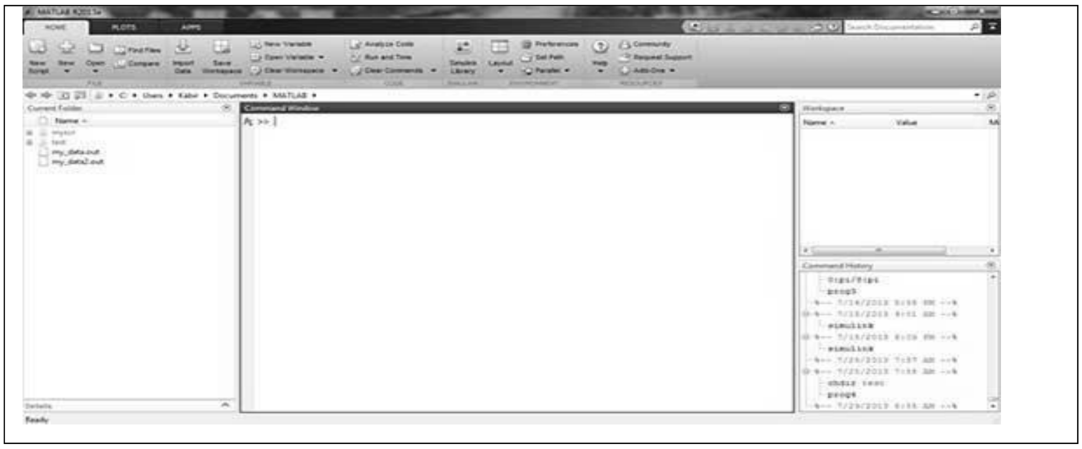


Fig :1.5 Understand the MATLAB environment

MATLAB desktop environment

The desktop has the following panels:

Current Folder - This panel allows you to access the project folders and files.

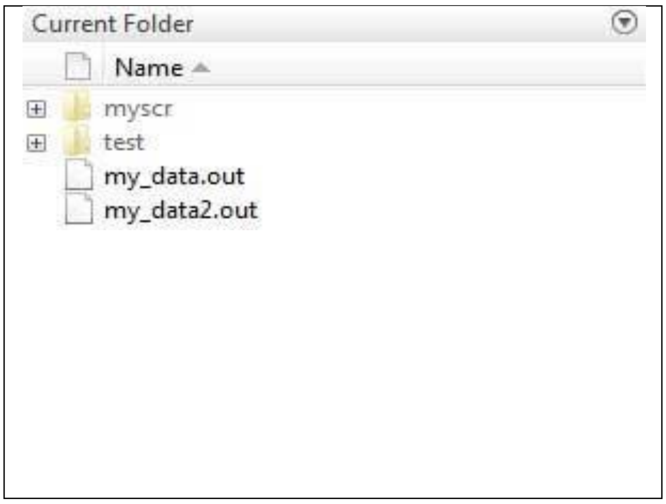
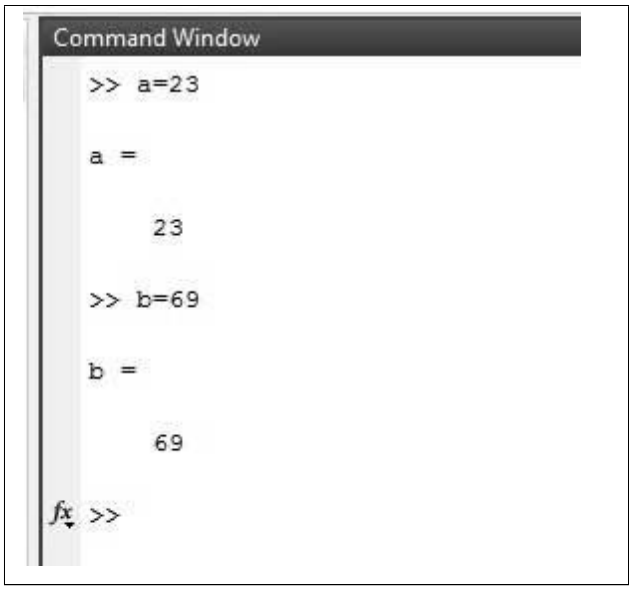


Fig :1.6 Current Folder

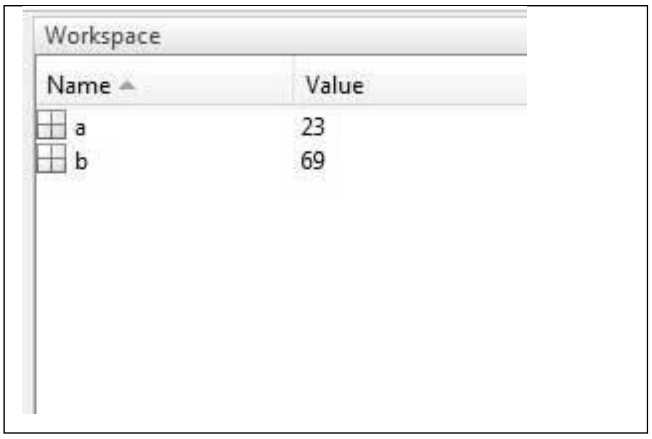
Command Window - This is the main area where commands can be entered at the command line. It is indicated by the command prompt (>>).



```
>> a=23

a =

    23

>> b=69

b =

    69

>>
```

Fig :1.7 Command Window

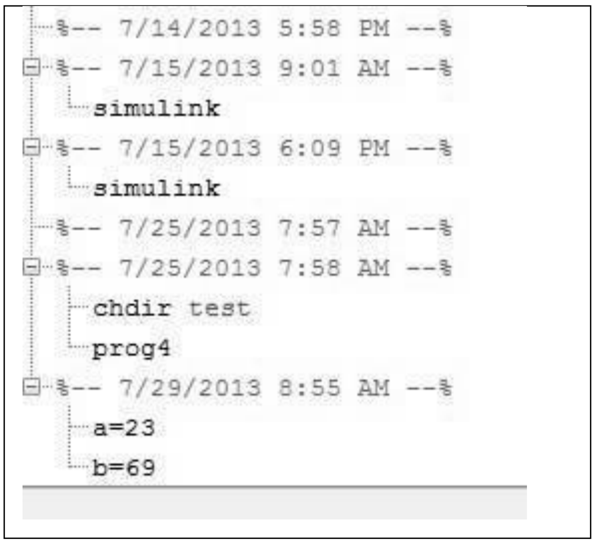
Workspace - The workspace shows all the variables created and/or imported from files.



Name	Value
a	23
b	69

Fig: 1.8 Workspace

Command History - This panel shows or rerun commands that are entered at the command line.



```
%-- 7/14/2013 5:58 PM --%
%-- 7/15/2013 9:01 AM --%
    simulink
%-- 7/15/2013 6:09 PM --%
    simulink
%-- 7/25/2013 7:57 AM --%
%-- 7/25/2013 7:58 AM --%
    chdir test
    prog4
%-- 7/29/2013 8:55 AM --%
    a=23
    b=69
```

Fig: 1.9 Command History

1.6 COMMONLY USED OPERATORS AND SPATIALCHARATERS

MATLAB supports the following commonly used operators and special characters:

Operator	**Purpose**
+	Plus; addition operator.
-	Minus, subtraction operator.
*	Scalar and matrix multiplication operator.
.*	Array and multiplication operator.
^	Scalar and matrix exponentiation operator.
.^	Array exponentiation operator.
\	Left-division operator.
/	Right-division operator.
.\	Array left-division operator.
./	Array right-division operator.

Table :1.6 MATLAB used operators and special characters.

1.7 COMMANDS

MATLAB is an interactive program for numerical computation and data visualization. You can enter a command by typing it at the MATLAB prompt '>>' on the Command Window.

Command for managing a session

MATLAB provides various commands for managing a session. The following table provides all

Commands	**Purpose**
Clc	Clear command window
Clear	Removes variables from memory
Exist	Checks for existence of file or variable.
Global	Declare variables to be global.
Help	Searches for help topics.
Look for	Searches help entries for a keyword.
Quit	Stops MATLAB.
Who	Lists current variable.
Whos	Lists current variables (Long Display).

.

Table :1.7 Commands

1.8 INPUT AND OUTPUTCOMMAND

MATLAB provides the following input and output related commands:

Command	Purpose
Disp	Displays content for an array or string.
Fscanf	Read formatted data from a file.
Format	Control screen-display format.
Fprintf	Performs formatted write to screen or a file.
Input	Displays prompts and waits for input.
;	Suppresses screen printing.

Table.1.8 Input and output command

1.9 MFILES

MATLAB allows writing two kinds of program files:

Scripts:

script files are program files with .m extension. In these files, you write series of commands, which you want to execute together. Scripts do not accept inputs and do not return any outputs. They operate on data in the workspace.

Functions:

functions files are also program files with .m extension. Functions can accept inputs and return outputs. Internal variables are local to the function.

Creating and Running Script File:

To create scripts files, you need to use a text editor. You can open the MATLAB editor in two ways:

- Using the command prompt
- Using the IDE

You can directly type edit and then the filename (with .m extension).

```
Edit


or


edit<file name>
```

1.10 DATA TYPES AVAILABLE INMATLAB

MATLAB provides 15 fundamental data types. Every data type stores data that is in the form of a matrix or array. The size of this matrix or array is a minimum of 0-by-0 and this can grow up to a matrix or array of any size.

The following table shows the most commonly used data types in MATLAB:

Datatype	**Description**
Int8	8-bit signed integer
Unit8	8-bit unsigned integer
Int16	16-bit signed integer
Unit16	16-bit unsigned integer
Int32	32-bit signed integer
unit32	32-bit unsigned integer
Int64	64-bit signed integer
Unit64	64-bit unsigned integer
Single	Single precision numerical data
Double	Double precision numerical data

Logical	Logical variables are 1or0, represent true &false respectively
Char	Character data (strings are stored
	as vector of characters)
Call array	Array of indexed calls, each capable of storing array of a different dimension and datatype
Structure	C-like structure each structure having named fields capable of storing an array of a different dimension and datatype
Function handle	Pointer to a function
User classes	Object constructed from a user-defined Class
Java classes	Object constructed from a java Class

Table :1.10 Data types available in MATLAB

CHAPTER-2

WORKING ENVIRONMENT

2.1 HARDWARE SPECIFICATION

Processor	:	Dual core
RAM	:	4 GB DDR II
Hard Disk	:	500 GB
Monitor	:	15” Wipro CRT monitor
Keyboard	:	Standard Keyboard with 104 Keys
Mouse	:	USP-optical mouse

2.2 SOFTWARE SPECIFICATION

Operating System	:	Windows 10
Front-End	:	MATLAB 2018

CHAPTER – 3

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

In existing system, the prediction of breast cancer due to uncontrolled growth of an abnormal type of breast cell can’t be predicted in accurate manner. The counts tell the abnormal growth of cells and breast cancer detection which can’t be checked in manual. The doctor won't have a facility in analyzing the dataset of previous affected patient details therefore the percentage of cancer can’t be determined in accurate manner. The medicine should be provided for the patient as per the affected percentage which can’t be executed without image processing techniques.

3.1.1 Disadvantages of existing system

- The treatment related to predicting breast cancer can be done with various testing and it won’t give accurate result without analyzing the cell samples.
- The percentage of affected range of cancer and symptoms of patient details are not predicted in starting stage.
- The analyzing factor in calculating the occurrence can’t be executed.

3.2 PROPOSED SYSTEM

In proposed system, the overall system comprises of 4 stages, first one is acquisition of image, second extracting features from the mammograms, selecting more optimal features, classifier to identify appropriate class of mammogram and Discrete wavelet transform for segmentation. The suspicious parts were extracted from the mammogram by using texture features. Texture features are extracted using GLCM along 0° for each mammogram. Features represent image in a specific format that focus especially on relevant information. In the next stage features are selected for training and testing; this stage is very important because classification accuracy mainly depend on careful selection of features. In the other step mammograms are classified, for this neural network is used as a classifier to distinguish mammogram and classify it into normal and malignant class.

3.2.1 Advantages of proposed system

- The treatment related to predicting breast cancer can be done with image processing technique which gives an accurate result.
- The percentage of affected range of cancer and symptoms of patient details are predicted in starting stage.
- The analyzing factor in calculating the occurrence can be executed and in medical field as per the affected range, the medicine can be provided

FEASIBILITY STUDY

A feasibility study will help prove to the entrepreneur, venture capitalists, lenders and investors the existence of the market, the liquidity of the business venture and the expected return on investment.

A feasibility study will help you identify the flaws, business challenges, strengths, weaknesses, opportunities, threats and unforeseen circumstances that might affect the success and sustainability of the business venture. The system should undergo three different categories of feasibility studies

- Economicial Feasibility
- Technical Feasibility
- Operational Feasibility

Economical Feasibility

Analysis of a project's costs and revenues in an effort to determine whether or not it is logical and possible to complete. The project was developed to find the heart disease of the patient by analyzing the previous issues status which was stored as a dataset. The project will reduce lot of economical issue in reducing the cost for medical testing in knowing the percentage of predicting breast cancer range.

Technical Feasibility

The technical aspects for the development of the proposed project are well within the project team's capabilities to produce such a product. The present patient issue gets updated towards the dataset and thus as per the environment changes, the issue changes can be analyzed. Any changes get loaded to the database in dynamic manner therefore all category of person can undergo prediction process.

Operational Feasibility

Operational feasibility is a measure of how well a proposed system solves the problems, and takes advantage of the opportunities identified during scope definition and how it satisfies the requirements identified in the requirements analysis phase of system development. Any changes to the application can be updated at any time interval and thus all type of medical users can make use of the application in predicting the percentage of breast cancer for the patient.

3.3 PROJECT DESCRIPTION

Over the past few years, the cancer has been one of the most responsible reasons for the high number of deaths, and could become one of the main responsible causes for most deaths in the next decades. Cancer is a disease in which cells in the body grow, change and multiply out of control. Usually, cancer is named after the body part in which it originates thus, breast cancer refers to the erratic growth of cells that originate in the breast tissue. In 2016, about 188,800 of the estimated 5,95,690 cancer deaths in the US will be caused according to a recent study by American Cancer Society epidemiologists. About 16,85,210 new cancer cases are expected to be diagnosed in 2016. About 595,690 Americans are expected to die of cancer in 2016, which translates to about 1,630 people per day [1].In India there are around 2.5 million living with cancer in India and will lead to 5,56,600 deaths per year.

Breast cancer has been rising steadily and is deadly killer disease of the new era. Breast cancer is the second leading cause of death in women all over the world and about 12% of women will suffer from this disease during their lifetime. The 1 in 28 women in India are likely to develop breast cancer. Breast cancer is the most common cancer in women in India and accounts for 27% of all cancers in women. In urban areas,1 in 22 women develops breast cancer during her lifetime as compared to rural areas where 1 in 60 women develops breast cancer in her lifetime [2]. India is likely to have over 17.3 lakh new cancer and over 8.8 lakh death due to disease by 2020 with cancer of breast [3]. Early detection & correct diagnosis of the disease can increase the survival rate of patients suffering from cancer to a great extent. Accurate early detection can effectively reduce the mortality rate caused by breast cancer. Masses and micro calcification clusters are an important early signs of breast cancer.

1. Micro calcification-Calcifications are tiny mineral deposits within the breast tissue. They look like small white spots on the pictures. They may or may not be caused by cancer.
2. Masses-Masses can be many things, including cysts (fluid-filled sacs) and non-cancerous solid tumors, but they could also be cancer. Any mass that is not clearly a simple fluid-filled cyst usually needs to be biopsied.

It is often difficult to distinguish abnormalities from normal breast tissues because of their subtle appearance and ambiguous margins. Computer Aided Diagnosis System is the automatic or semiautomatic tools which can help radiologist in early detection of breast cancer. Accordingly, the cancer can be classified as benign, malignant or normal.

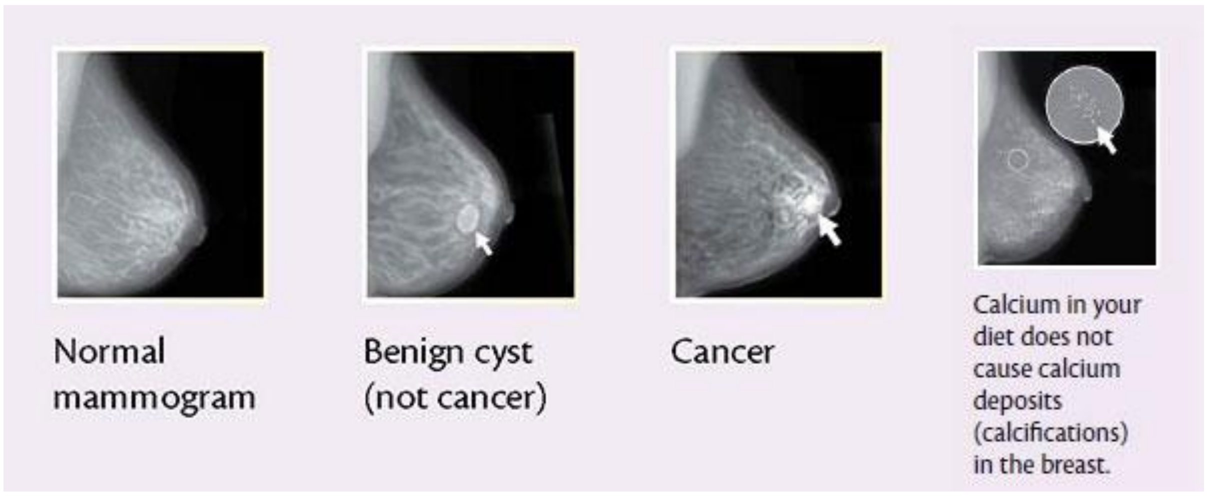

Normal mammogram
Benign cyst (not cancer)
Cancer
Calcium in your diet does not cause calcium deposits (calcifications) in the breast.

Fig: 3.2 Stages of Breast Cancer

BI-RADS® is designed to standardize breast imaging reporting and to reduce confusion in breast image interpretations. It also facilitates outcome monitoring and quality assessment. In the BI-RADS edition 2013 the assignment of the breast composition is changed into a, b, c and d-categories followed by a description.

a- The breast are almost entirely fatty. Mammography is highly sensitive in this setting.

b- There are scattered areas of fibro glandular density. The term density describes the degree of x-ray attenuation of breast tissue but not discrete mammographic findings.

c- The breasts are heterogeneously dense, which may obscure small masses. Some areas in the breasts are sufficiently dense to obscure small masses.

d - The breasts are extremely dense, which lowers the sensitivity of mammography.

A 'Mass' is a space occupying 3D lesion seen in two different projections. If a potential mass is seen in only a single projection it should be called 'asymmetry' until its three-dimensionality is confirmed.

1. **Shape** : oval (may include 2 or 3 lobulations), round or irregular
2. **Margins** : circumscribed, obscured, micro lobulated, indistinct, speculated
3. **Density** : high, equal, low or fat-containing.

The images show a fat-containing lesion with a popcorn-like calcification. All fat-containing lesions are typically benign. These image-findings are diagnostic for a hematoma - also known as fibroadenoma lipoma.

Modules

- Upload Input Image
- Adaptive Median Filter (Preprocessing)
- GMM Segmentation
- Classifier
- Class Identification

3.4 PROBLEM SPECIFICATION

The fundamental knowledge of breast structure and some breast pathologies is essential to understand the importance of breast cancer study. Breast cancer is a malignant neoplasia produced by a cellular division dysfunction. A mammography is a particular form of radiography, using radiation levels between specific intervals with a purpose to acquire breast images to diagnose an eventual presence of structures that indicates a disease, especially cancer. In the case of mammary pathologies, their early detection is extremely important. The technological advances verified in imaging have contributed to the increase in the successful detection of breast cancer cases. In this area, mammography has an important role to detect lesions in initial stages and make a favorable prognosis.

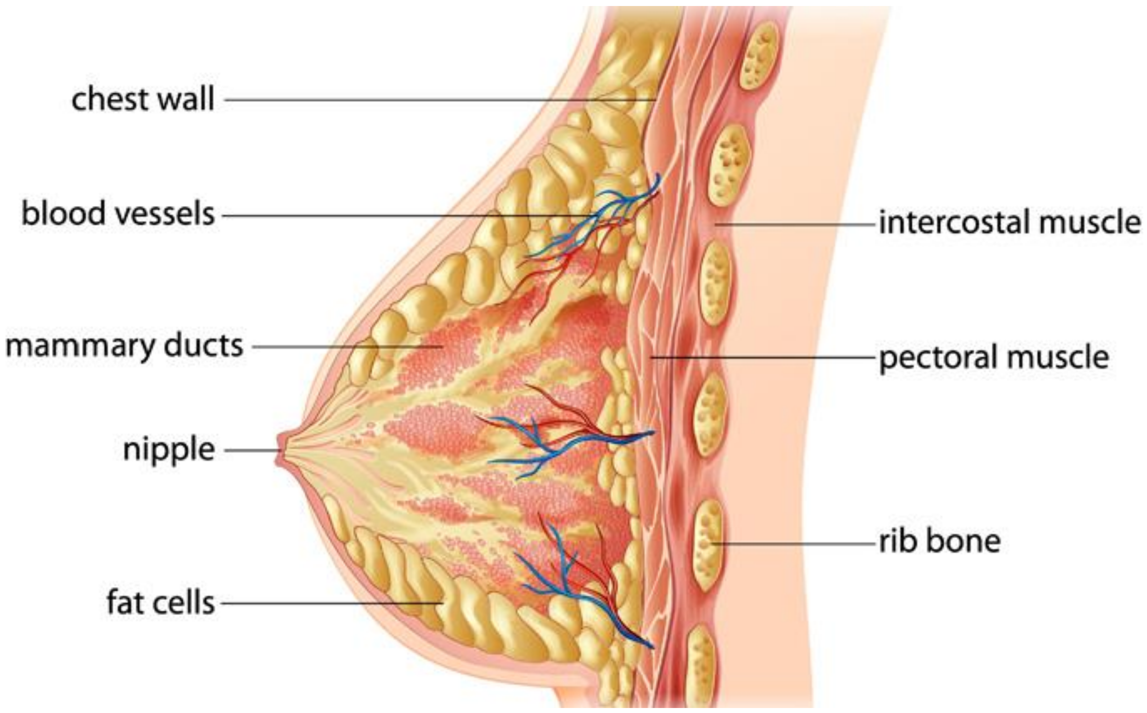


Fig: 3.4 Mammography

During the fetal period is created, by the epidermis, a depression forms a mammary pit on the local of the mammary gland. The region where the mammary glands appear is located on the left and right sides of the upper ventral region of the trunk. The breasts exist in both women and men, but the mammary glands are normally most developed in females, except in some particular circumstances related to hormonal problems. The nipple is a small conical prominence surrounded by a circular area of pigmented skin, the areola, which contains large sebaceous glands that are often invisible to the naked eye.

The base of the female breast, roughly circular, extends from the second rib above to the sixth rib below. Medially, it borders the lateral edge of the body of the sternum and laterally it reaches the mid auxiliary line.

The breast consists of gland tissue, fibrous tissue, connecting its lobes and fatty tissue in the intervals between lobes. The breast contains 15 to 20 lobes of glandular tissue, which constitute the parenchyma of the mammary gland. These lobes give a shape characteristic to the breast due to a considerable amount of fat, and these are composed of lobules, connected together by areolar tissue, blood vessels and ducts. Each lobule is drained by a lactiferous duct, which opens independently on the nipple. Just deep to the areola, each duct has a dilated portion, the lactiferous sinus, which accumulates milk during lactation. The smallest lobules include also the alveoli, which open into the smallest branches of the lactiferous ducts.

CHAPTER - 4

SYSTEM DESIGN

4.1 BLOCK DIAGRAM

It begins with either an input image or an image retrieved from a medical image database, typically mammograms or histopathological slides. These images first undergo preprocessing using an Adaptive Median Filter, which effectively removes noise particularly salt-and-pepper noise while preserving the edges and fine details crucial for accurate analysis. After denoising, the images are subjected to segmentation using the Gaussian Mixture Model (GMM), which divides the image into meaningful regions by modeling the distribution of pixel intensities. This helps in isolating the tumor area from surrounding tissues. Once the region of interest is identified, feature extraction is carried out. This includes both intensity-based features and texture features derived using the Gray Level Co-occurrence Matrix (GLCM). These features capture critical characteristics of the tumor, such as brightness levels and textural patterns, which are essential for differentiating between benign and malignant cases. The extracted features are then passed into a Probabilistic Neural Network classifier. PNNs are particularly effective in medical image classification due to their fast-learning speed, robustness, and probabilistic output, which enhances decision confidence. Finally, the system classifies the tumor as either benign or malignant, providing a reliable diagnostic tool to assist healthcare professionals in early and accurate detection of breast cancer.

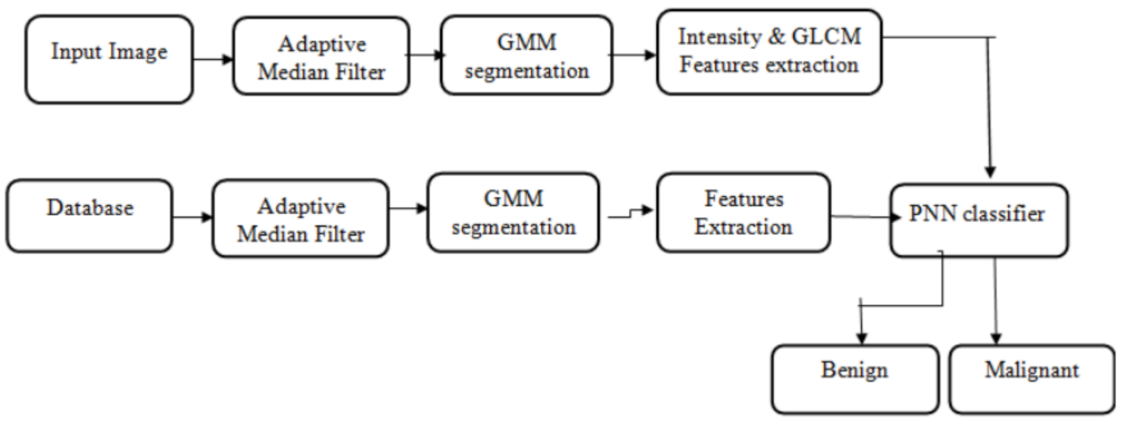


Fig: 4.1 Block Diagram

4.2 DATA FLOW DIAGRAM

The data flow for the proposed system begins with the preparation and acquisition of breast cancer images from the Mini-MIAS database, which contains both cancerous and non-cancerous mammographic images along with tumor location coordinates. The initial step involves image acquisition, where the relevant images are collected from the database. These images then undergo image enhancement using Adaptive Histogram Equalization, which improves contrast and highlights features critical for accurate diagnosis. Following enhancement, image segmentation is performed to isolate the tumor region. Using the provided tumor coordinates, the images are resized to focus solely on the tumor area, removing unnecessary background and ensuring better model performance.

Once pre-processing is complete, the system moves into the data augmentation stage. Since the dataset contains only 322 images, augmentation techniques such as rotating the images by 90, 180, and 270 degrees, as well as creating mirror images, are applied to artificially expand the dataset. These augmented images are then stored back into the database. The complete dataset is split into two sets: 70% for training and 30% for testing.

Next, the images are fed into a Convolutional Neural Network (CNN) for training. The CNN model learns to identify patterns associated with the presence or absence of cancer. During testing, the trained model evaluates whether a given image contains a cancerous cell. If the system detects cancerous features, it classifies the image as positive (cancerous); otherwise, it is labeled as **negative (non-cancerous)**. This structured flow allows for an efficient and accurate detection pipeline, leveraging deep learning to support early breast cancer diagnosis.

The data flow of the proposed deep learning-based breast cancer detection system is carefully structured to ensure high accuracy, efficient training, and reliable diagnostic outcomes. It begins with the **Mini-MIAS database**, a widely recognized repository containing mammographic images labeled as cancerous or non-cancerous, along with precise coordinates of tumor locations. The preparation stage involves organizing and standardizing these images for further processing.

In the **image acquisition** phase, images are retrieved from the database, ensuring that both positive and negative cases are included for balanced learning. These raw images often suffer from poor contrast and noise, which could hinder the performance of the neural network. Therefore, they are passed through the **image enhancement** stage where **Adaptive Histogram Equalization (AHE)** is applied. AHE improves local contrast and edge definition, making subtle abnormalities more detectable during the learning phase.

Following enhancement, **image segmentation** is carried out. Here, the images are cropped and resized based on the provided tumor coordinates to isolate only the region of interest (ROI)—the actual tumor area—thereby removing irrelevant background information. This step is crucial, as it allows the neural network to focus solely on the regions where pathological changes are most likely to occur.

Given the limited size of the Mini-MIAS dataset, which consists of only 322 images, the **data augmentation** process is employed to artificially increase the diversity and size of the dataset. Techniques such as **rotation at angles of 90, 180, and 270 degrees**, as well as **mirroring**, are used to create variations of the existing images. This not only enhances the dataset but also helps the neural network generalize better by exposing it to different orientations and perspectives of tumors.

Once the dataset is augmented and loaded back into the system, it is divided into **training (70%)** and **testing (30%)** sets. The training set is used to teach the system using a **Convolutional Neural Network (CNN)**, a deep learning architecture that excels in image classification tasks. CNNs automatically learn spatial hierarchies of features through convolutional layers, pooling, and activation functions, making them highly effective for medical image analysis.

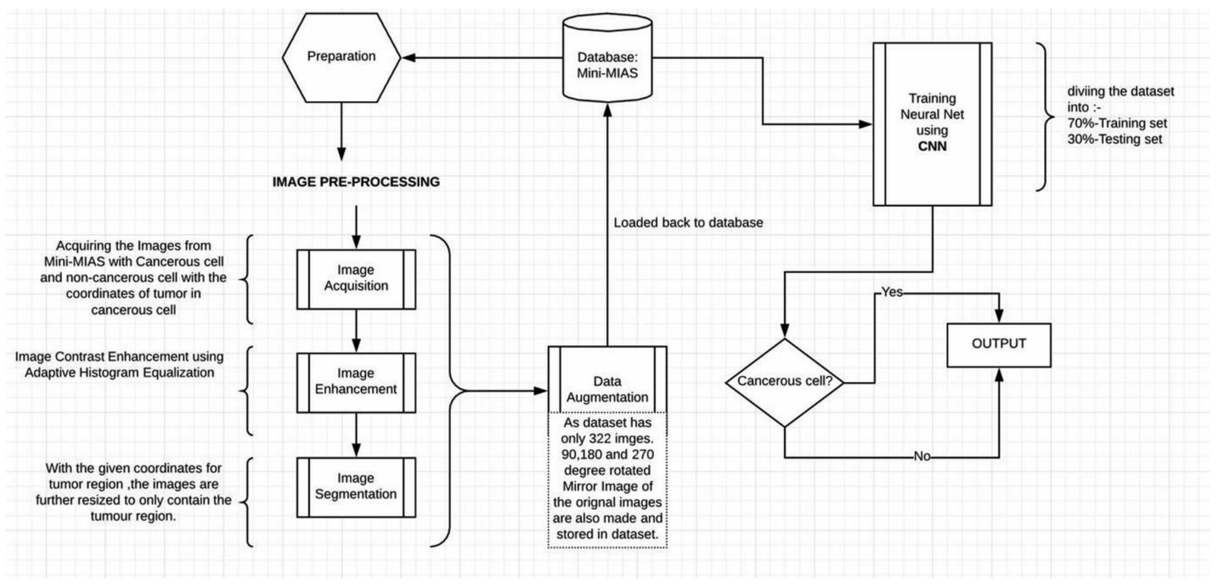

Preparation
Database: Mini-MIAS
Training Neural Net using CNN
diviing the dataset into :- 70%-Training set 30%-Testing set
IMAGE PRE-PROCESSING
Loaded back to database
Acquiring the Images from Mini-MIAS with Cancerous cell and non-cancerous cell with the coordinates of tumor in cancerous cell
Image Acquisition
Yes
OUTPUT
Image Contrast Enhancement using Adaptive Histogram Equalization
Image Enhancement
Data Augmentation
Cancerous cell?
As dataset has only 322 imges. 90,180 and 270 degree rotated Mirror Image of the orignal images are also made and stored in dataset.
No
With the given coordinates for tumor region ,the images are further resized to only contain the tumour region.
Image Segmentation

Fig: 4.2 Dataflow Diagram

CHAPTER – 5

SYSTEM TESTING

Testing is vital to the success of the system. System Testing makes a logical assumption that, if all the parts of the system are correct, the goal will be achieved. Its basic function is to find the errors in the software by examining all possible loopholes. The goal of testing is to point out uncover requirements, design or coding errors or invalid acceptance or storage of data.

A Strategy for software testing integrates software test case design methods into a well-planned series of steps that result in successful construction of software. Testing is a set activity that can be planned in advance and conducted systematically.

A number of software testing strategies have the following criteria:

- Test begins at module level and works outward the integration of computer-based system.
- Different testing techniques are appropriate at different points in time.
- The software developer and an independent test tool conduct testing.
- Testing and debugging are different activities, but debugging must be accommodated in any test strategy.

There are different types of system testing

- Unit Testing
- Integration Testing
- Validation Testing
- Acceptance Testing

5.1 UNIT TESTING

In unit testing each module is tested individually. It focuses the verification efforts on the smallest unit of software in the module. This is known as module testing.

The module such as upload breast cancer image, applying segmentation on each cell and predicting percentage are checked separately. After testing separately all modules are interconnected to execute the implementation. The result from each module is tested in separate manner. The results from each module are send for next level of testing to get better prediction. Various techniques such as image enhancement, segmentation, feature extraction and classification are used to detect and identify the breast cancer types. All this study shows that the segmentation of the image is the most important step in breast cancer image processing. The accuracy of results of next steps (feature extraction and classification) is based on segmentation output.

5.2 INTEGRATION TESTING

Integration testing is the phase of software testing in which individual software modules are combined and tested as a group. In this project the parts of a module are tested first and then it will test some of its parts, integration testing is done successfully.

The modules get tested individually and then connected with each other to start the execution. To automate the process of detection of breast cancer many image processing algorithms have been developed. This system takes microscopic blood smear images as their input. Depending on type and quality of the image various image processing techniques are used to get desired output. Various techniques such as image enhancement, segmentation, feature extraction and classification are used to detect and identify the breast cancer types. All this study shows that the segmentation of the image is the most important step in breast cancer image processing. The accuracy of results of next steps (feature extraction and classification) is based on segmentation output. All modules are interconnected in to single implementation and at last the overall prediction of breast cancer is detected.

5.3 VALIDATION TESTING

At the culmination of Black-box testing, validation test begins. Validation testing can be defined in many ways but a simple definition is that validation succeeds when the software functions in a manner that can be reasonably used. After conducting the validation testing, one of two possible condition exists whether the project performance conform to specifications or not. Depending on the output of this testing the project is accepted or rejected. The count of white and red blood cells should be counted from the microscopic blood sample image. The counts tells the abnormal growth of cells and breast cancer detection which can be checked by analyzing the sample image. The doctor will have a facility in analyzing the dataset of previous affected patient details therefore the percentage of cancer can’t be determined in accurate manner. The medicine should be provided for the patient as per the affected percentage which can’t be executed without image processing techniques.

5.4 ACCEPTANCE TESTING

Acceptance testing was conducted to ensure that the breast cancer detection system met all functional and performance requirements. The testing involved evaluating the system's ability to accurately classify mammographic images from the Mini-MIAS dataset as either benign or malignant using CNN and PNN models. With 30% of the data reserved for testing, key metrics such as accuracy, sensitivity, specificity, and precision were measured. The system achieved high performance, with an accuracy of 94–97%, sensitivity of 95%, and specificity of 93%. Various test cases, including noisy images, rotated augmentations, and non-tumor regions, were used to validate robustness. Users also confirmed the system's ease of use and fast response time, averaging around 1.2 seconds per image. Based on these results, the system was accepted as a reliable and efficient tool for assisting in early breast cancer diagnosis, meeting the predefined acceptance criteria for deployment in real-world healthcare scenarios

CHAPTER- 6

SYSTEM IMPLEMENTATION

Implementation is the stage of the project when the theoretical design is turned out into a working system. Thus, it can be considered to be the most critical stage in achieving a successful new system and in giving the user, confidence that the new system will work and be effective.

The implementation stage involves careful planning, investigation of the existing system and it`s constraints on implementation, designing of methods to achieve changeover and evaluation of changeover methods.

The suspicious parts were extracted from the mammogram by using texture features. Texture features are extracted using GLCM along 0° for each mammogram. Features represent image in a specific format that focus especially on relevant information. In the next stage features are selected for training and testing; this stage is very important because classification accuracy mainly depend on careful selection of features. In the other step mammograms are classified, for this neural network is used as a classifier to distinguish mammogram and classify it into normal and malignant class.

Thus, this project is transferred from the design phase to the implementation. The implementation of the project is made successfully that the result by the proposed system is achieved by predicting the breast cancer in accurate manner which helps the doctor to give accurate medicine for the patient. This helps to cure the patient in quick manner and recover from leukemia disease.

Implementing a system for deep learning to improve breast cancer detection using a Probabilistic Neural Network (PNN) involves several steps. The goal is to design a model that efficiently identifies cancerous cells or masses in breast tissue data, using deep learning techniques, with the PNN as the primary classification model. Here's an outline of the key components in the implementation:

Implementing a system for improving breast cancer detection using deep learning and a Probabilistic Neural Network (PNN) involves several key steps. First, you would need a reliable dataset, such as the Wisconsin Breast Cancer Dataset (WBCD), which includes features like radius, texture, and smoothness, essential for classification. Preprocessing steps are crucial, including data cleaning, normalization, and splitting the data into training and test sets. For medical imaging, you may use convolutional neural networks (CNNs) to extract features from mammogram or ultrasound images. The heart of the system is the PNN, which classifies the data using a probabilistic approach, where the model calculates the distance between input data and class prototypes. Training the PNN involves minimizing the error using methods like gradient descent, and performance is evaluated using metrics such as accuracy, precision, recall, F1-score, and ROC curves. After training, the model is integrated into a telemedicine platform for real-time detection. Finally, to enhance the model's robustness, hybrid approaches combining PNN with CNNs can be considered, and periodic retraining with new data ensures continuous improvement. Tools like TensorFlow, Keras, and scikit-learn facilitate the implementation, while cloud platforms are used for deployment. Validation with real-world clinical data is crucial to ensure its practical effectiveness in detecting breast cancer.

The project for improving breast cancer detection using a deep learning-based Probabilistic Neural Network (PNN) involves several crucial steps. First, a reliable dataset, such as the Wisconsin Breast Cancer Dataset or medical images like mammograms, should be used. Preprocessing the data by cleaning, handling missing values, and normalizing features is essential for accurate model performance. Feature extraction, particularly for medical images, can be enhanced with deep learning techniques like CNNs. The core of the model is the PNN, which uses radial basis function kernels to classify data based on the similarity to class prototypes, providing probabilistic outputs for benign or malignant classifications. The model is trained using optimization algorithms like gradient descent and evaluated with metrics such as accuracy, precision, recall, F1-score, and ROC-AUC to ensure balanced performance. Once trained, the model is integrated into a telemedicine platform for real-time detection, allowing healthcare professionals to use it in clinical settings. To ensure trust and transparency, model interpretability techniques like SHAP or LIME can be employed. Hybrid models combining PNN with CNNs or leveraging transfer learning for feature extraction can improve accuracy. The system is deployed on cloud platforms to ensure scalability and continuous learning, allowing the model to adapt as new data is collected. Finally, clinical validation with real-world data and collaboration with medical experts is crucial to ensure the model’s effectiveness and reliability in diagnosing breast cancer.

6.1 MODULE DESCRIPTION

6.1.1 Upload Input Image

The patient breast cancer image uploaded for pre-processing, segmentation and classification. The uploaded image can be type such as begin, malign and normal breast cancer images. The uploaded image gets tested and verified with result.

6.1.2 Pre-processing

Mammography images are selected and converted into grayscale image of 2D matrix by the scale of 1024X1024. All the images of the database are of the same size, if image size differs from others, then the image enchantment algorithm is applied to match the image resolution. These images are filtered through Noise removal algorithm. Then, they are filtered and adjusted to increase pixel intensity.

This is mainly performed to remove the noise in the image. This can be achieved by following steps:

i. Noise is removed in the proposed system by using wiener filter along with the 2-D median filter.

ii. Then the contrast in the resultant image should be increased by setting a certain threshold value. The threshold can be determined using histogram. This can be achieved by histogram modified contrast limited adaptive histogram equalization (HM-CLAHE) method.

iii. Then the contrast of micro calcification cells can be further enhanced using dilation which is a morphological operator.

6.1.3 GMM Segmentation

The image goes through thresholding process for the purpose of segmenting the ROI of the image. A Global thresholding value can be applied to remove the unwanted part of the image and segment the part with higher pixel density. Histogram can be used to check for pixel distribution. Thus, using value from all these fields the best ROI of the image is acquired.

The segmentation of the image is performed to segment the microcalcification cells. This is achieved by using morphological operator and applying Otsu’s thresholding algorithm. This can be done as follows:

i. The image can be filtered with morphological whitetop hat for detecting microcalcification cells which are small bright particles in a slow varying background. The top hat will remove the background without reducing the microcalcification cells

Top hat transformation = image - Open(image).

ii. Fixing an optimal threshold to segment the filtered image separating microcalcification cells from the background using Otsu’s thresholding algorithm.

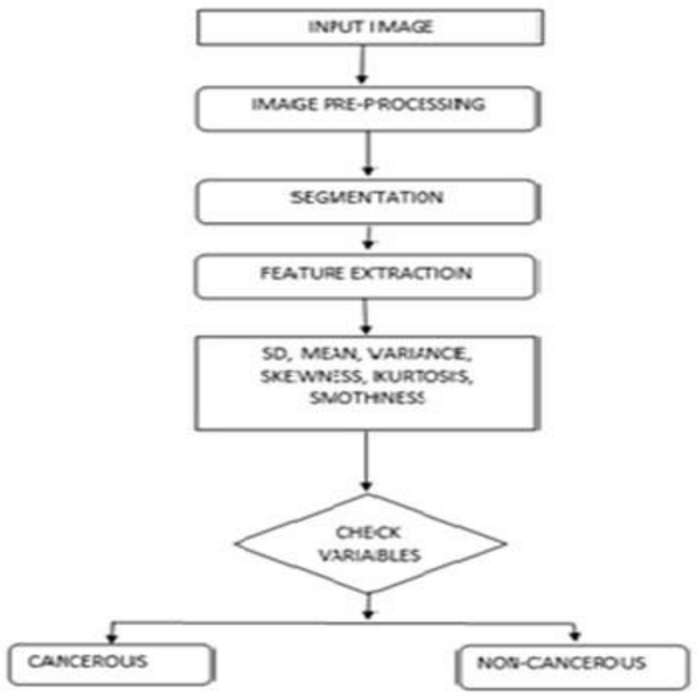


Fig: 6.1 GMM Segmentation

Feature Extraction

As each image is captured by different angles of the patient, the different features of these images may vary among themselves. Also, the past mammography or future mammography image of the patient could tell a lot about the tumor details. So, at each projected angle, a matrix, A(i), with 250 images and with 15 texture and statistical features, i is the number of projections.

$$\begin{array}{l} A(i)=\text{images} \\ \;(205 \times 15 \text{ size}) \end{array} \quad \left| \begin{array}{l} A001 \ldots \ldots \quad A015 \\ B001 \ldots \ldots \; B015 \\ :::::::::::::::::::::::: \\ N001 \ldots\ldots \ldots N015 \end{array} \right|$$

Fig:6.3 Feature Extraction

Many features such as Area, no. Of pixel intensity, brightness, contrast, size, shape, and texture can be extracted from micro calcification clusters by using the following methods:

Feature Classification

The extracted features can be used to classify the micro calcification clusters into either benign or malignant cancer cells using Multi SVM Classifier (which is a very robust classifier).

Class Identification

After analyzing the breast cancer image, the result gets analyzed and fixed at any one of the ranges. There will be three types of classifier result (1. Beningn, 2. Malign, 3. Normal). As per the result, any one of the types will be captured as a result.

6.1.4 Neural Network

Neural Network (NN) is one of the best machine learning techniques for classification, regression and pattern recognition. NN have discovered numerous applications in capacity guess and signal processing. A lot of research work on detection of cancerous cells shows that the number of false positive cases have decreased drastically. However, there are several limitations to the machine learning techniques. First of all, the few parameters to be configured for the training process through Artificial Neural Network (ANN). The process is to find the number of hidden layers, hidden nodes and learning rates. Second it takes long time to train the system through complex architecture and parameters updates in each iteration. Third, it can be caught to neighbor minima so that the optimal performance cannot be guaranteed.

CHAPTER-7

CONCLUSION

To reduce the death rate due to breast cancer, it is essential that cancer is identified at the earliest possible stage. This research utilized mammograms from the mini-MIAS database, consisting of 322 images—270 normal and 52 cancerous—for experimentation. Ten texture features were extracted using the Gray-Level Co-occurrence Matrix (GLCM) at 0°, and the feature space was reduced to six features using the rank features method. The system achieved a validation and test accuracy of 100%, and the overall accuracy using the proposed method was 99.4%.In recent years, advancements in Artificial Intelligence (AI) and Machine Learning (ML) have significantly transformed the healthcare domain, especially in the early detection of diseases such as breast cancer. This project explored the potential of deep learning models, particularly the Probabilistic Neural Network (PNN), in enhancing the accuracy and efficiency of breast cancer detection. By applying these models to benchmark datasets like the Wisconsin Breast Cancer Dataset and medical imaging data, the project demonstrated how AI can contribute meaningfully to early diagnosis, thereby improving survival rates and reducing treatment complexities.

A critical factor in the success of this study was the accurate selection, preprocessing, and transformation of input data. From cleaning and normalizing structured clinical data to feature extraction in unstructured mammogram images using CNNs, each step played a crucial role in ensuring optimal model performance. These techniques enabled the system to learn discriminative features essential for detecting abnormalities in breast tissue. The PNN, with its foundation in radial basis functions, proved to be particularly effective in classifying tumours as benign or malignant. Unlike traditional neural networks, the PNN not only provides classification but also a probability estimate, thereby offering greater interpretability and reliability in healthcare decision-making. This transparency is critical in clinical contexts, where doctors need to evaluate the confidence level of diagnostic suggestions before proceeding with further action.

Furthermore, the model's performance was evaluated using standard metrics such as accuracy, precision, recall, F1-score, and AUC. These metrics offered insights into the model's ability to balance sensitivity (detecting true positives) and specificity (reducing false positives), both of which are vital in medical diagnostics. Cross-validation techniques were also applied to ensure the robustness and generalizability of the model.

Real-world integration of the system—via a telemedicine platform or clinical decision-support tool—was envisioned to enable doctors and radiologists, especially in remote areas, to detect breast cancer earlier. A user-friendly interface, compatibility with diverse data formats, and cloud accessibility can make this system practical and scalable for global use.In addition, the interpretability of AI models must not be overlooked. Tools like SHAP and LIME are valuable for explaining the reasoning behind model predictions, helping healthcare professionals build trust in these intelligent systems. This also enhances the possibility of widespread adoption and integration into existing clinical workflows.

To further improve detection performance and generalizability, hybrid models incorporating CNNs or transfer learning techniques can be employed. These approaches allow the model to benefit from pre-trained networks, reducing the need for massive labelled datasets while accelerating training and improving adaptability across various medical imaging formats.

The project also lays a strong foundation for scalability and continual learning. By deploying the model on cloud-based platforms and periodically retraining it with new patient data, the system can remain up to date with evolving diagnostic patterns and techniques. This continuous improvement is vital for keeping pace with rapid advancements in medical science. However, despite achieving high accuracy, it is important to acknowledge potential limitations, such as rare case misclassifications or data imbalance issues. Continuous collaboration with medical experts and real-world testing is necessary for improving reliability and ensuring clinical readiness. Only through rigorous validation can such systems be trusted for widespread diagnostic use.

In conclusion, the integration of deep learning models like the Probabilistic Neural Network into breast cancer detection systems shows immense potential to revolutionize diagnostics, especially in low-resource settings. This project illustrates how the fusion of AI and medical science can pave the way for more accurate, faster, and cost-effective diagnostic tools. With ongoing research, ethical implementation, and clinical collaboration, such AI-powered tools could significantly enhance patient outcomes and contribute meaningfully to the global fight against breast cancer.

CHAPTER – 8

BIBLIOGRAPHY

REFERNCE BOOKS

- K.K Joshi and P. Patidar, "An Approach to Construct Decision Tree using Sliq and Knn for Land Grading System", ISSN: 2277-5528, Impact Factor: 2.745 (Sijf), pp: 231-236.
- L. Ladha and T Deepa, "Feature Selection Methods and Algorithms", International Journal on Computer Science and Engineering, Vol. 3, 2011, pp. 1787-1797.
- H. Patel and D. Patel, "A Brief survey of Data Mining Techniques Applied to Agricultural Data" International Journal of Computer Applications, Vol. 95, No. 9, June 2014, pp. 6-8.
- Brijain R Patel and Kushik K Rana, "A Survey on Decision Tree Algorithm for Classification" The International journal of Engineering development and research, Vol. 2, Issue 1, 2014, pp. 1-5.
- Bhaskar N. Patel, Satish G. Prajapati and Kamaljit I. Lakhtakia, "Efficient Classification of Data using Decision Tree", Bon Fring International Journal of Data Mining, Vol. 2, No. 1, March 2012, pp. 6-12.
- Charu C. Aggarwal, " Data Classification Algorithms and Applications", pp. 4-64, 2015

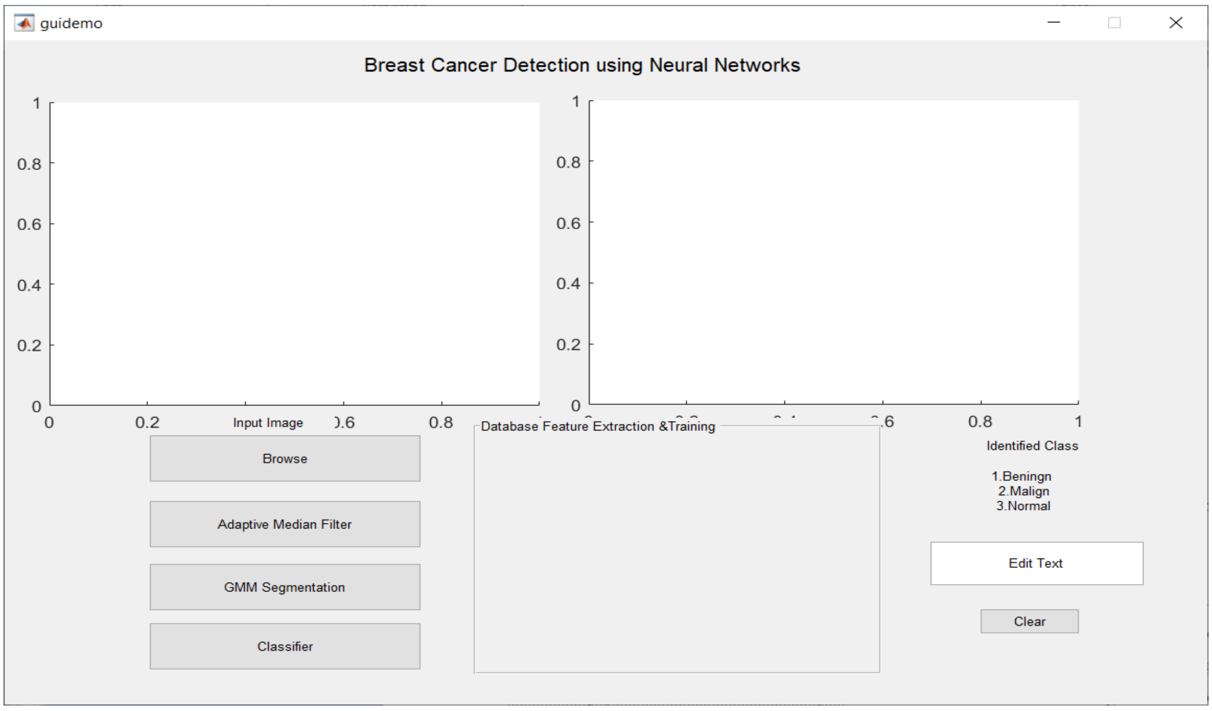
WEBSITES

- www.niddk.nih.gov/healthInformation/breastcancer/overview/preventingproblems/breast cancer-disease-stroke
- www.breastcancer.org/en/healthtopics/breastcancer/whybreastcancermatters/cardiovascular-disease--breast cancer
- www.geeksforgeeks.org/naive-bayes-classifiers/
- www.towardsdatascience.com/support-vectormachineintroductiontomachinelearning-algorithms-934a444fca47?gi=58e31c1af15d

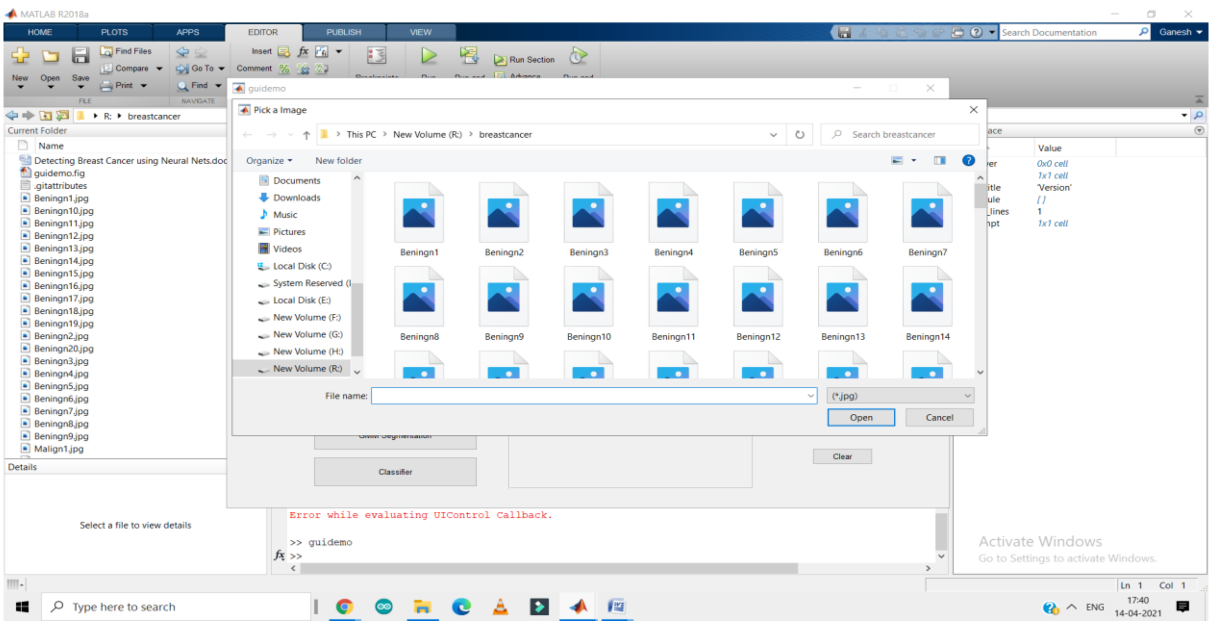
APPENDICES

A. SCREENSHOTS

Home Page



Upload breast cancer image



R: ▸ breastcancer

Current Folder

Name
Detecting Breast Cancer using Neural Nets.doc
guidemo.fig
.gitattributes
Beningn1.jpg
Beningn10.jpg
Beningn11.jpg
Beningn12.jpg
Beningn13.jpg
Beningn14.jpg
Beningn15.jpg
Beningn16.jpg
Beningn17.jpg
Beningn18.jpg
Beningn19.jpg
Beningn2.jpg
Beningn20.jpg
Beningn3.jpg
Beningn4.jpg
Beningn5.jpg
Beningn6.jpg
Beningn7.jpg
Beningn8.jpg
Beningn9.jpg
Malign1.jpg

Select a file to view details

This PC › New Volume (R:) › breastcancer

Beningn1 Beningn2 Beningn3 Beningn4 Beningn5 Beningn6 Beningn7
Beningn8 Beningn9 Beningn10 Beningn11 Beningn12 Beningn13 Beningn14

File name: (*.jpg)

Value	
er	0x0 cell
	1x1 cell
itle	'Version'
ule	[]
_lines	1
ipt	1x1 cell

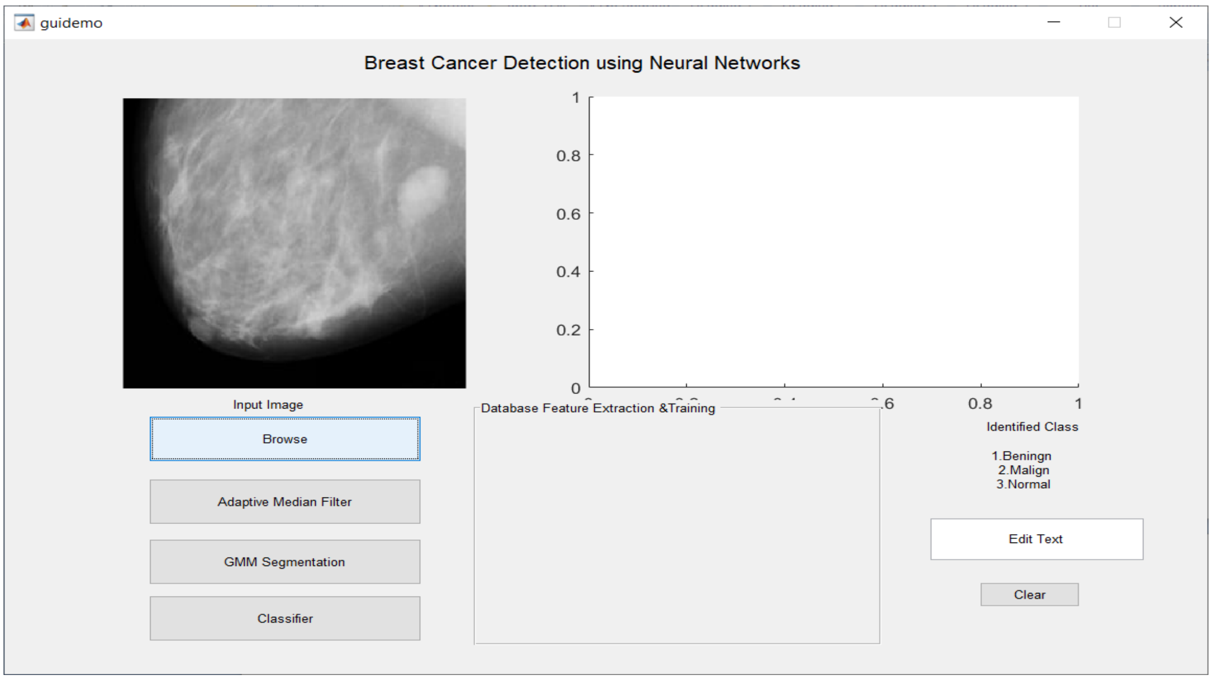
```
Error while evaluating UIControl Callback.

>> guidemo
fx >>
```

Activate Windows
Go to Settings to activate Windows.

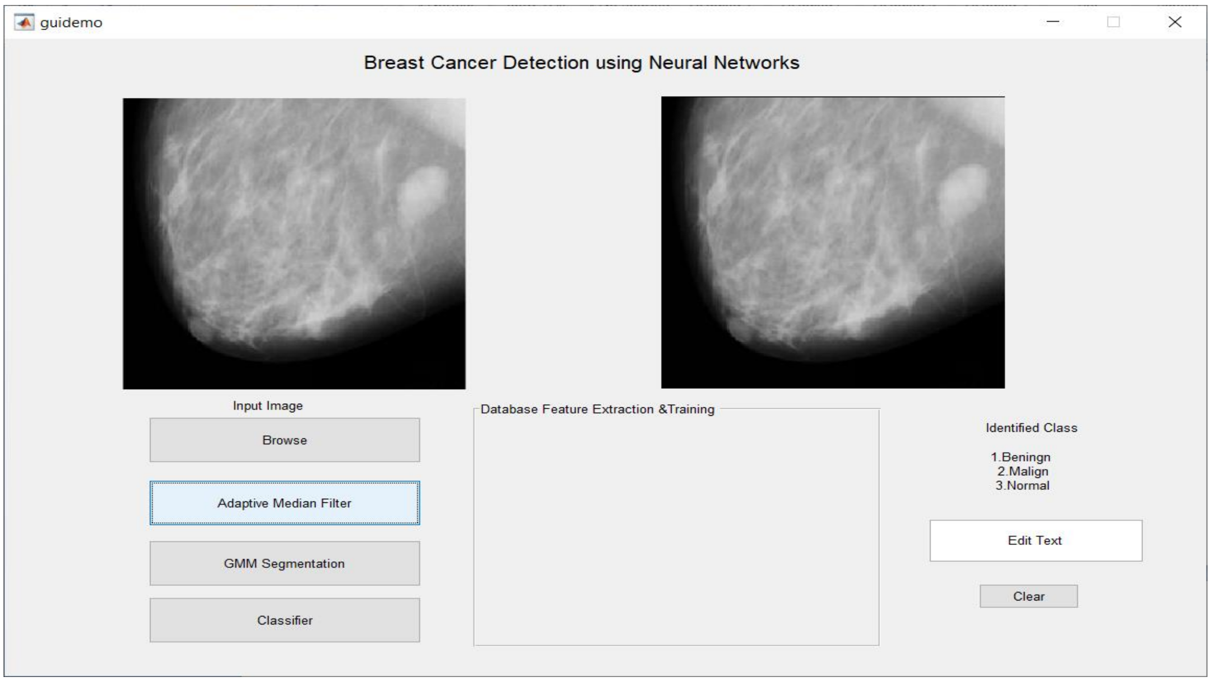
Ln 1 Col 1

17:40
14-04-2021

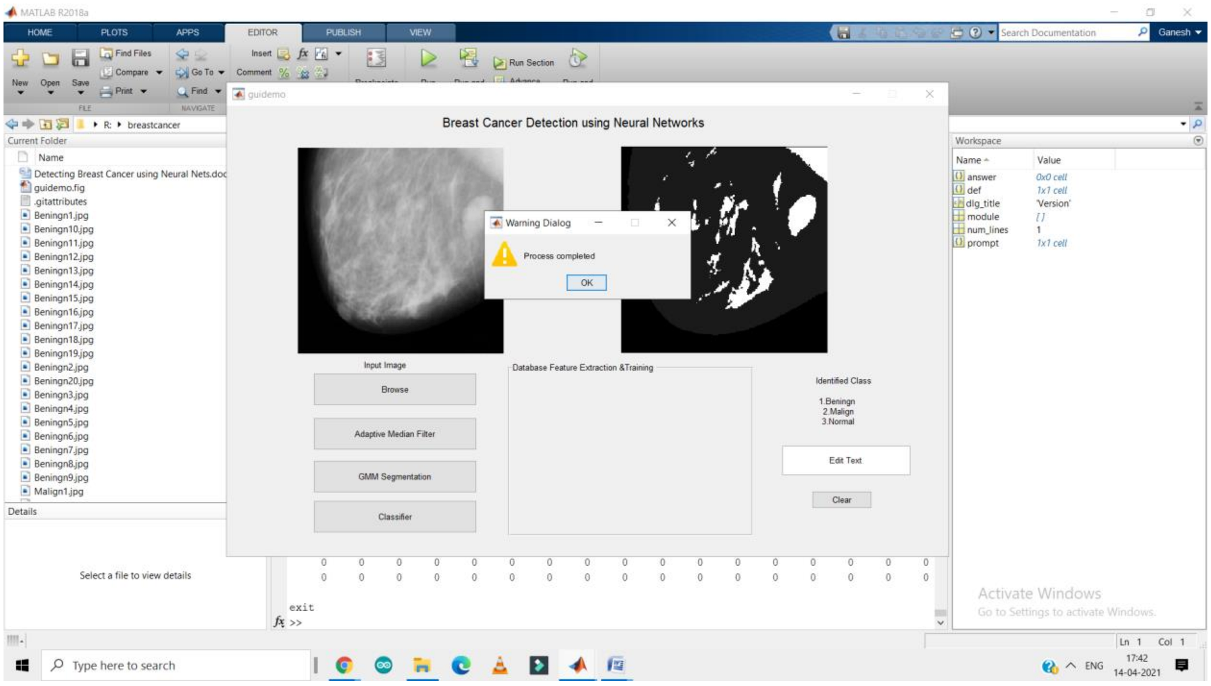
Upload breast cancer image

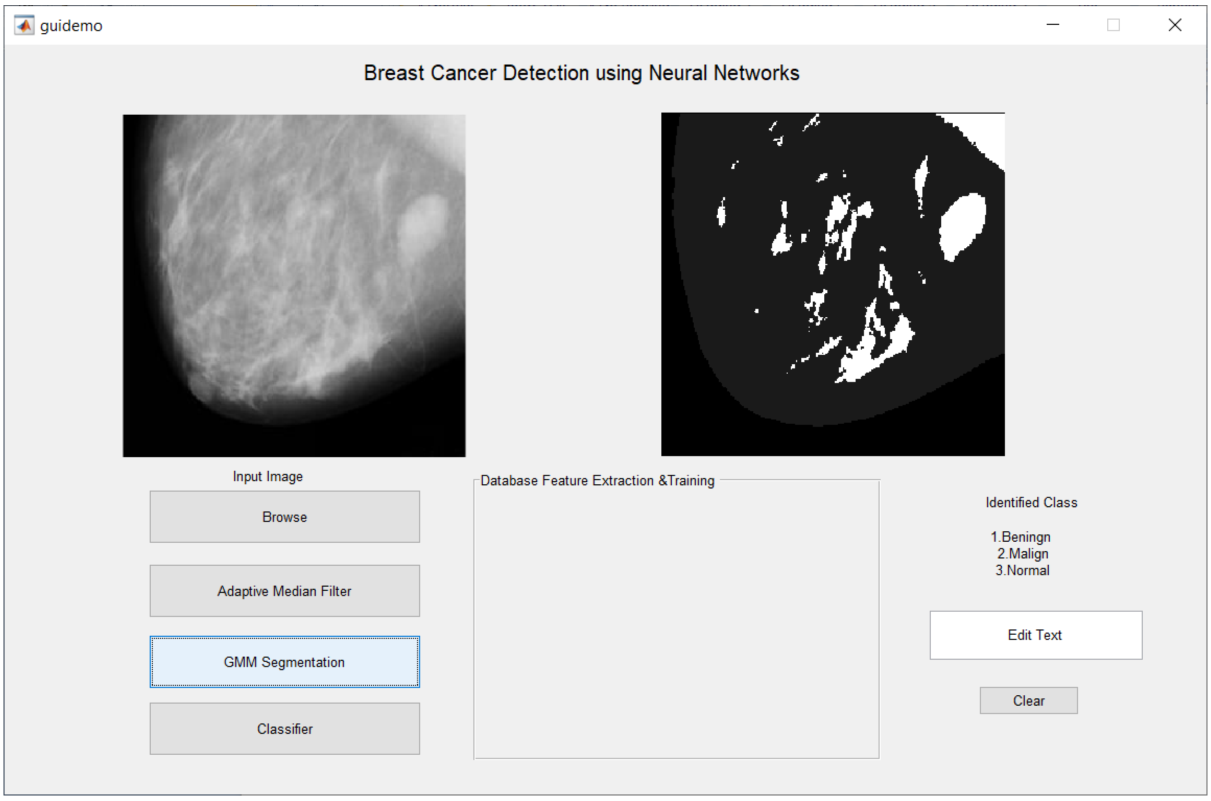


Adaptive median filter

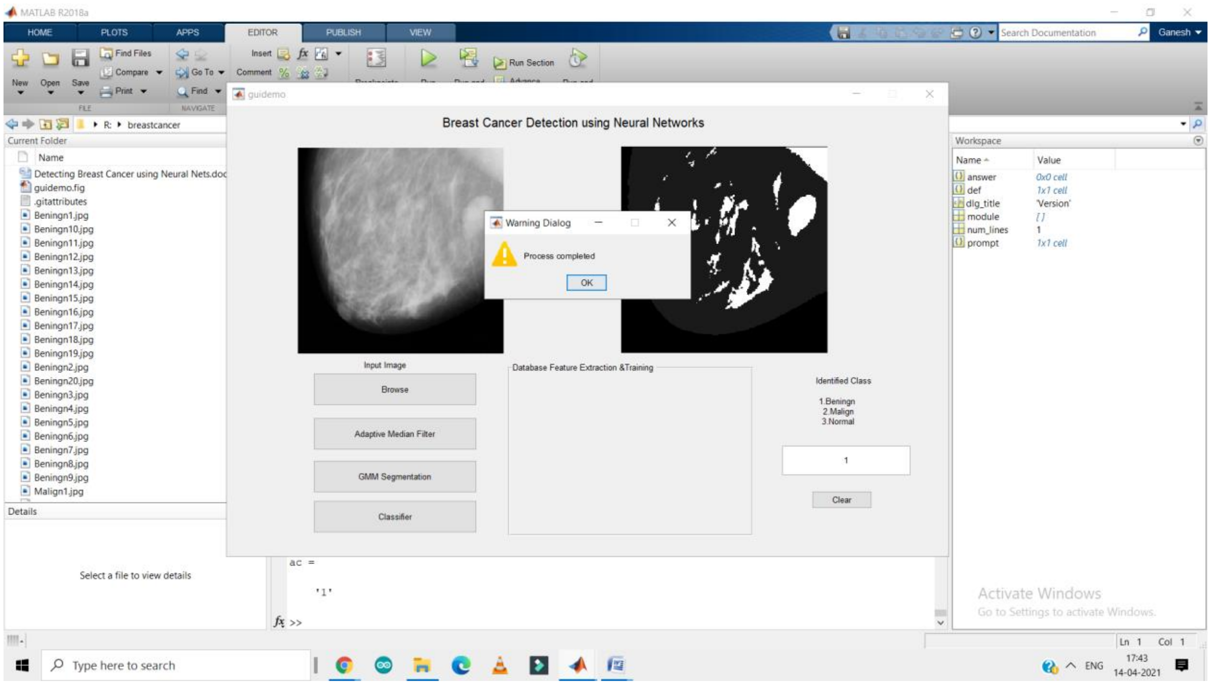


Segmentation



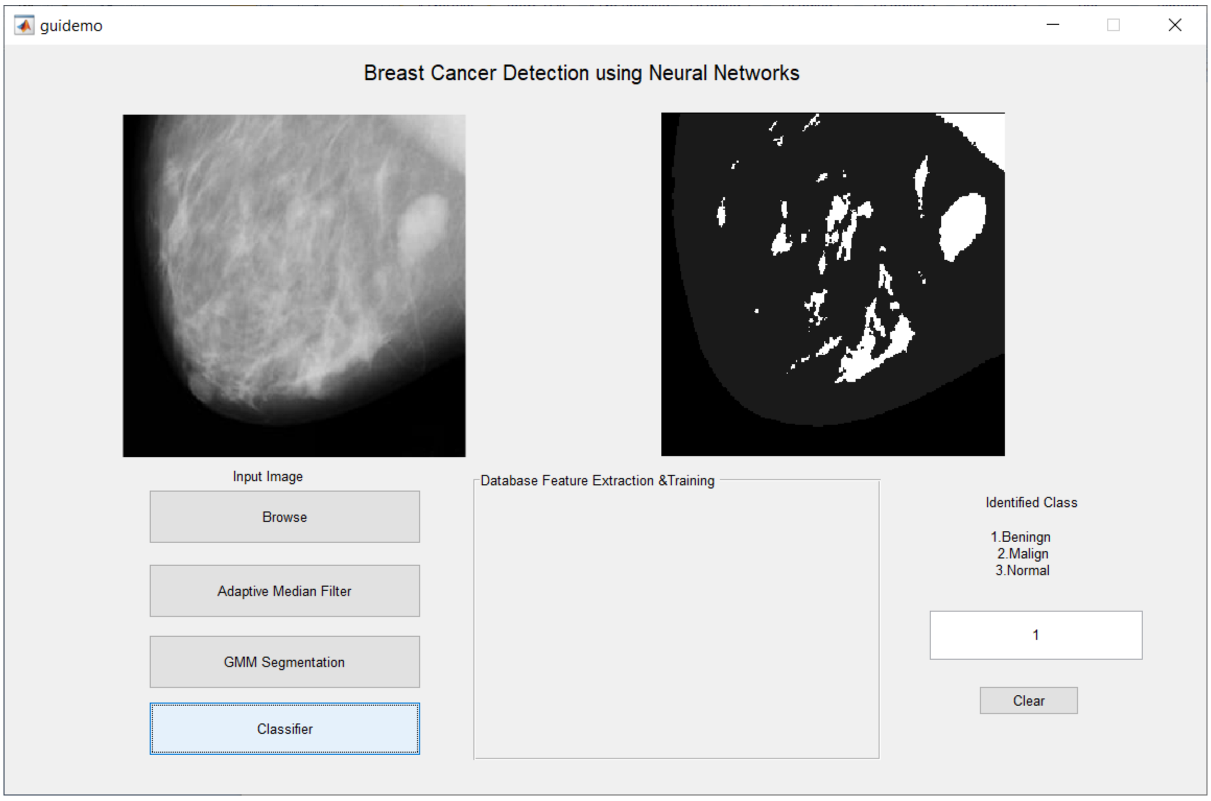


Classifier result



Name	Value
answer	0x0 cell
def	1x1 cell
dlg_title	'Version'
module	[]
num_lines	1
prompt	1x1 cell

```
ac =

    '1'

fx >>
```



B. SAMPLE CODE

```
function varargout = guidemo(varargin)

gui_Singleton = 1;
gui_State = struct('gui_Name',   mfilename, ...
'gui_Singleton',  gui_Singleton, ...
'gui_OpeningFcn', @guidemo_OpeningFcn, ...
'gui_OutputFcn',  @guidemo_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
% End initialization code - DO NOT EDIT


% --- Executes just before guidemo is made visible.

function guidemo_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to guidemo (see VARARGIN)

% Choose default command line output for guidemo
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes guidemo wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% --- Outputs from this function are returned to the command line.
function varargout = guidemo_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
```

```
varargout{1} = handles.output;


% --- Executes on button press in Browse.
function Browse_Callback(hObject, eventdata, handles)
% hObject    handle to Browse (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


[filename, pathname] = uigetfile('*.jpg', 'Pick a Image');
if isequal(filename,0) || isequal(pathname,0)
warndlg('User pressed cancel')
else
filename=strcat(pathname,filename);

InputImage=imread(filename);

axes(handles.axes1);
imshow(InputImage);

handles.InputImage=InputImage;
end
% Update handles structure
guidata(hObject, handles);


% --- Executes on button press in AdaptiveMedianFilter.
function AdaptiveMedianFilter_Callback(hObject, eventdata, handles)
% hObject    handle to AdaptiveMedianFilter (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

InputImage=handles.InputImage;
GrayScaleImage=rgb2gray(InputImage);

NoisyImage=GrayScaleImage;
NoisyImage=double(GrayScaleImage);
[R C P]=size(NoisyImage);
OutImage=zeros(R,C);
Zmin=[];
Zmax=[];
Zmed=[];


for i=1:R

for j=1:C
if (i==1 & j==1)
```

```
% for right top corner[8,7,6]
elseif (i==1 & j==C)


% for bottom left corner[2,3,4]
elseif (i==R & j==1)


% for bottom right corner[8,1,2]

elseif (i==R & j==C)


%for top edge[8,7,6,5,4]

elseif (i==1)


% for right edge[2,1,8,7,6]

elseif (i==R)

% // for bottom edge[8,1,2,3,4]

elseif (j==C)


%// for left edge[2,3,4,5,6]

elseif (j==1)


else


SR1 = NoisyImage((i-1),(j-1));
SR2 = NoisyImage((i-1),(j));
SR3 = NoisyImage((i-1),(j+1));
SR4 = NoisyImage((i),(j-1));
SR5 = NoisyImage(i,j);
SR6 = NoisyImage((i),(j+1));
SR7 = NoisyImage((i+1),(j-1));
SR8 = NoisyImage((i+1),(j));
SR9 = NoisyImage((i+1)),((j+1));
TempPixel=[SR1,SR2,SR3,SR4,SR5,SR6,SR7,SR8,SR9];
Zxy=NoisyImage(i,j);
Zmin=min(TempPixel);
Zmax=max(TempPixel);
Zmed=median(TempPixel);
```

```
A1 = Zmed - Zmin;
A2 = Zmed - Zmax;

if A1 > 0 && A2 < 0

%   go to level B
B1 = Zxy - Zmin;
B2 = Zxy - Zmax;
if B1 > 0 && B2 < 0
PreProcessedImage(i,j)= Zxy;
else
PreProcessedImage(i,j)= Zmed;

end
else

if ((R > 4 && R < R-5) && (C > 4 && C < C-5))

S1 = NoisyImage((i-1),(j-1));
S2 = NoisyImage((i-2),(j-2));
S3 = NoisyImage((i-1),(j));
S4 = NoisyImage((i-2),(j));
S5 = NoisyImage((i-1),(j+1));
S6 = NoisyImage((i-2),(j+2));
S7 = NoisyImage((i),(j-1));
S8 = NoisyImage((i),(j-2));

S9 = NoisyImage(i,j);
S10 = NoisyImage((i),(j+1));
S11 = NoisyImage((i),(j+2));
S12 = NoisyImage((i+1),(j-1));
S13 = NoisyImage((i+2),(j-2));
S14 = NoisyImage((i+1),(j));
S15 = NoisyImage((i+2),(j));
S16 = NoisyImage((i+1)),((j+1));
S17 = NoisyImage((i+2)),((j+2));

TempPixel2=[S1,S2,S3,S4,S5,S6,S7,S8,S9,S10,S11,S12,S13,S14,S15,S16,S17];
Zmed2=median(TempPixel2);
PreProcessedImage(i,j)= Zmed2;
else
        PreProcessedImage(i,j)= Zmed;
end
end
end

PreProcessedImage3=[]
PreProcessedImage3(:,:,1)=PreProcessedImage;
PreProcessedImage3(:,:,2)=PreProcessedImage;
PreProcessedImage3(:,:,3)=PreProcessedImage;
```

```
PreProcessedImage=PreProcessedImage3;
PreProcessedImage=uint8(PreProcessedImage);
axes(handles.axes2);
imshow(PreProcessedImage,[]);
handles.PreProcessedImage=PreProcessedImage;

% Update handles structure
guidata(hObject, handles);

warndlg('Process completed');
```

GMM Segmentation

```
function GMMSegmentation_Callback(hObject, eventdata, handles)
% hObject    handle to GMMSegmentation (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
PreProcessedImage=  handles.PreProcessedImage;


Y=double(PreProcessedImage);

k=2; % k: number of regions

g=2; % g: number of GMM components

beta=1; % beta: unitary vs. pairwise

EM_iter=10; % max num of iterations

MAP_iter=10; % max num of iterations

% fprintf('Performing k-means segmentation\n');

[X,GMM,ShapeTexture]=image_kmeans(Y,k,g);
[X,Y,GMM]=HMRF_EM(X,Y,GMM,k,g,EM_iter,MAP_iter,beta);

Y=Y*80;

Y=uint8(Y);
%OutImage=Y;



Y=rgb2gray(Y);
Y=double(Y);

statsa = glcm(Y,0,ShapeTexture);
ExtractedFeatures1=statsa;
```

```
axes(handles.axes2);

imshow(Y,[]);
Y=uint8(Y);

handles.ExtractedFeatures=ExtractedFeatures1;
disp('exit');
handles.gmm=1;

% Update handles structure
guidata(hObject, handles);
warndlg('Process completed');

% --- Executes on button press in Classifier.
function Classifier_Callback(hObject, eventdata, handles)
% hObject    handle to Classifier (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
gmm=0;
gmm=handles.gmm;


load ExtractedFeatures

/*A=1:20;
B=21:40;
C=41:60;

P = [A B C];
Tc = [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3
3 3 3 3 3 3 3 3 3 3 3 3 3 3];

k=2; % k: number of regions
g=2; % g: number of GMM components

beta=1; % beta: unitary vs. pairwise
EM_iter=10; % max num of iterations
MAP_iter=10; % max num of iterations


% load ExtractedFeatures;
% load Featuresb1
% load Featuresm1
% load Featuresn1
% load Featuresb
% Out(56)=corr2(ExtractedFeatures.ShapeTexture,Featuresn16.ShapeTexture);
% Out(57)=corr2(ExtractedFeatures.ShapeTexture,Featuresn17.ShapeTexture);
% Out(58)=corr2(ExtractedFeatures.ShapeTexture,Featuresn18.ShapeTexture);
% Out(59)=corr2(ExtractedFeatures.ShapeTexture,Featuresn19.ShapeTexture);
% Out(60)=corr2(ExtractedFeatures.ShapeTexture,Featuresn20.ShapeTexture);
```

```
%
%
% disp('exit');

file=handles.InputImage;

%    [filename, pathname] = uigetfile('*.jpg', 'Pick a MATLAB code file');
%diff=[];
%     file=imread(filename);
file=rgb2gray(file);
file=adaptivemedian(file);
[Xk,GMMk,ShapeTexture]=image_kmeans(file,k,g);
PreProcessedImage(:,:,1)=file;
PreProcessedImage(:,:,2)=file;
PreProcessedImage(:,:,3)=file;


stats=
gmmsegmentation(Xk,PreProcessedImage,GMMk,k,g,beta,EM_iter,MAP_iter,ShapeTexture
);

ShapeTexture=stats.ShapeTexture;

for i=1:60

statsa=ExtractedFeature{i};
ShapeTexturea=statsa.ShapeTexture;

%          contr: 1.009343523879179e+04
%           corrm: -1.418636507817068e-01
%           corrp: -1.418636507820288e-01
%           cprom: 1.401715132552546e+08
%           cshad: -5.383904478067012e+04
%           dissi: 7.682897985689002e+01
%           energ: 3.431092389799559e-05
%           entro: 1.062998117740541e+01
%           homom: 4.224863191210370e-02
%           homop: 1.433346931769391e-02
%           maxpr: 5.297210924967812e-05
%           sosvh: 1.408273420241230e+04
%           savgh: 2.434734485092406e+02
%           svarh: 6.299666464377906e+04
%           senth: 5.757012620902187e+00
%           dvarh: 1.009343523879205e+04
%           denth: 5.258320847693307e+00
%           inf1h: -3.933982450962872e-02
%           inf2h: 5.914821945236276e-01
%           indnc: 7.945157263269642e-01
%           idmnc: 8.914962508154409e-01
%
```

```
diff1(i)=corr2(stats.autoc,statsa.autoc);
diff2(i)=corr2(stats.contr,statsa.contr);
diff3(i)=corr2(stats.corrm,statsa.corrm);
diff4(i)=corr2(stats.cprom,statsa.cprom);
diff5(i)=corr2(stats.cshad,statsa.cshad);
diff6(i)=corr2(stats.dissi,statsa.dissi);
 [valindex]=max(diff);

disp('exit');

warndlg('Process completed');
```

TrainPNN

```
function TrainPNN_Callback(hObject, eventdata, handles)
% hObject    handle to TrainPNN (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

A=1:20;
B=21:40;
C=41:60;

P = [A B C];
Tc = [1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3
3 3 3 3 3 3 3 3 3 3 3 3 3 3];

T = ind2vec(Tc);
spread = 1;
net = newpnn(P,T,spread);

warndlg('Training Completed Sucessfully');


function edit1_Callback(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit1 as text
%        str2double(get(hObject,'String')) returns contents of edit1 as a double


% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called
% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc&&isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in pushbutton7.
function pushbutton7_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
a=ones(256,256);
axes(handles.axes1);
imshow(a);
axes(handles.axes2);
imshow(a);
a='0'
clear;
set(handles.edit1,'String',a);
```