VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama, Belgaum-590018

A PROJECT REPORT **(15CSP85)** ON

"PREDICTION AND PROGNOSIS OF BREAST CANCER"

Submitted in Partial fulfillment of the Requirements for the Degree of

Bachelor of Engineering in Computer Science & Engineering

By

AMRITA NARENDRANATH (1CR15CS021)

PEDDAMEKALA HITHEESHA (1CR16CS115)

NAVYA C (1CR16CS101)

MURARI GREESHMA (1CR16CS099)

Under the Guidance of,

DR. P N SINGH

Professor, Dept. of CSE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR INSTITUTE OF TECHNOLOGY

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI, BANGALORE-560037

CMR INSTITUTE OF TECHNOLOGY

#132, AECS LAYOUT, IT PARK ROAD, KUNDALAHALLI, BANGALORE-560037

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

Certified that the project work entitled **"PREDICTION AND PROGNOSIS OF BREAST CANCER"** carried out by **Ms. AMRITA NARENDRANATH**, USN **1CR15CS021**, **Ms. NAVYA C**, USN **1CR16CS101**, **Ms. PEDDAMEKALA HITHEESHA**, USN **1CR16CS115, Ms. MURARI GREESHMA**, USN **1CR16CS099,** bonafide students of CMR Institute of Technology, in partial fulfillment for the award of **Bachelor of Engineering** in Computer Science and Engineering of the Visveswaraiah Technological University, Belgaum during the year 2019-2020. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the Report deposited in the departmental library.

The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

Dr. P N Singh	**Dr. Prem Kumar Ramesh**	**Dr. Sanjay Jain**
Professor	**Professor & Head**	**Principal**
Dept. of CSE, CMRIT	**Dept. of CSE, CMRIT**	**CMRIT**

External Viva

Name of the examiners	Signature with date
1.	___________________
2.	___________________

DECLARATION

We, the students of Computer Science and Engineering, CMR Institute of Technology, Bangalore declare that the work entitled " **PREDICTION AND PROGNOSIS OF BREAST CANCER** " has been successfully completed under the guidance of Prof. Dr. P N Singh, Computer Science and Engineering Department, CMR Institute of technology, Bangalore. This dissertation work is submitted in partial fulfillment of the requirements for the award of Degree of Bachelor of Engineering in Computer Science and Engineering during the academic year 2019 - 2020. Further the matter embodied in the project report has not been submitted previously by anybody for the award of any degree or diploma to any university.

Place: Bengaluru

Date:

Team members:

AMRITA NARENDRANATH (1CR15CS021) ___________________

PEDDAMEKALA HITHEESHA (1CR16CS115) ___________________

NAVYA C (1CR16CS101) ___________________

MURARI GREESHMA (1CR16CS099) ___________________

ABSTRACT

Cancer has been characterized as a heterogeneous disease consisting of many different subtypes. The early diagnosis and prognosis of a cancer type have become a necessity in cancer research, as it can facilitate the subsequent clinical management of patients.

The importance of classifying cancer patients into high or low risk groups has led many research teams, from the biomedical and bioinformatics field, to study the applications of machine learning methods.

Developing automated malignant Breast Cancer (BC) detection system applied on patient's imagery can help dealing with this problem more efficiently, making diagnosis more scalable and less prone to errors. Not less importantly, such kind of research can be extended to other types of cancer, making even more impact to help saving lives. Results on BC recognition show that Convolution Neural Network (CNN) can achieve higher recognition rates than hand-crafted feature descriptors.

The aim of this thesis is to compare and evaluate among the various classifiers to see which classifier is best suited to predict breast cancer at very initial stage.

ACKNOWLEDGEMENT

We take this opportunity to express our sincere gratitude and respect to **CMR Institute of Technology, Bengaluru** for providing us a platform to pursue our studies and carry out our final year project

We have a great pleasure in expressing our deep sense of gratitude to **Dr. Sanjay Jain,** Principal, CMRIT, Bangalore, for his constant encouragement.

We would like to thank **Dr. Prem Kumar Ramesh,** Professor and Head, Department of Computer Science and Engineering, CMRIT, Bangalore, who has been a constant support and encouragement throughout the course of this project.

We consider it a privilege and honor to express our sincere gratitude to our guide **Dr. P N Singh,** Professor, Department of Computer Science and Engineering, for the valuable guidance throughout the tenure of this review.

We also extend my thanks to all the faculty of Computer Science and Engineering who directly or indirectly encouraged me.

Finally, we would like to thank our parents and friends for all their moral support they have given us during the completion of this work.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

BC	**Breast Cancer**
CNN	**Convolution Neural Network**
CT	**Computer Topography**
DeCAF	**Deep Convolution Activation Feature**
FNAC	**Fine-needle Aspiration Cytology**
LVQ	**Learning Vector Quantization**
NLTK	**Natural Language Toolkit**
PNN	**Probabilistic Neural Network**
SVM	**Support Vector Machine**

CHAPTER 1

INTRODUCTION

Among the several existing types of cancer, breast cancer (BC) presents two very concerning characteristics:

- It is the most common cancer among women worldwide.
- It presents a very high mortality rate when compared to other types of cancer.

The number of victims of this deadly cancer is a constant reminder that new approaches capable of improving patient survival are desperately needed.

1.1 Relevance of the Project

In the existing systems the prediction was only whether the person has cancer or not but there was less chances of identifying the stage and chances of survival. All the existing systems are implemented using supervised learning.

In proposed system semi supervised learning is used to learn.

1.2 Problem Statement

Breast cancer is the top cancer in women both in the developed and developing world. Early detection of cancer generally increases the chances for successful treatment. Automatic classification of histopathological images in a research topic that can make Breast Cancer diagnosis faster and less prone to errors.

1.3 Objective

- To diagnose cancer at early stage to reduce mortality rate from cancer
- To identify the types of benign and malignant breast cancer
- To suggest strategies on individual risk profiles

- To assess the use of imaging for the measurement of extent of disease and the monitoring of treatment protocols
- To shift the current state of diagnosing breast cancer from its late stage to the earlier stages where the disease is most curable and survival rate becomes more and treatment cost becomes lower

1.4 Scope of the Project

This project can be used by doctors which enables them to give faster results than the conventional way of testing and scanning.

1.5 Breast Cancer Symptoms

- The first noticeable symptom of breast cancer is typically a lump that feels different from the rest of the breast tissue. More than 80% of breast cancer cases are discovered when the woman feels a lump.
- Thickening different from the other breast tissue, one breast becoming larger or lower, a nipple changing position or shape or becoming inverted, skin puckering or dimpling, a rash on or around a nipple, discharge from nipple/s, constant pain in part of the breast or armpit and swelling beneath the armpit or around the collarbone.
- Another symptom complex of breast cancer is Paget's disease of the breast. This syndrome presents as skin changes resembling eczema; such as redness, discoloration or mild flaking of the nipple skin. Approximately half the women diagnosed with Paget's disease of the breast also have a lump in the breast.

1.6 Methodology

Use of multiple feature vector framework to evaluate this feature set in different scenarios by combining classification results from sub images and /or from combining different feature sets.

Evaluate the performance of DeCAF features when a patch-based-method is used. still DeCAF features from different layers of the pre-trained CNN.

The automatic image processing for cancer diagnosis has been explored as a topic of research for more than fort years but is still challenging due to the complexity of the images to analyse.

In order to obtain distinguishable characteristics structure specific features are needed and the images could contain both benign and malignant regions. Therefore, it is challenging to learn discriminative features from the previous images hence we need segmentation.

Using different machine learning models such as neural networks and support vector machines their report accuracy rates ranging from 76% to 94% on a dataset of 92 images.

In this a deep net is trained by feeding it a input and letting it compute layer by layer to generate the final output for comparison with correct answer. After computing error at this output the error flows backward through the net by back propagation.

1.7 Classification

Breast cancers are classified by several grading systems. Each of these influences the prognosis and can affect treatment response. Description of a breast cancer optimally includes all these factors.

- **Histopathology**

  Breast cancer is usually classified primarily by its histological appearance. Most breast cancers are derived from the epithelium lining the ducts or lobules, and these cancers are classified as ductal or lobular carcinoma. Carcinoma in situ is growth of low-grade cancerous or precancerous cells within a tissue compartment such as the mammary duct without invasion of the surrounding tissue.

- **Grade**

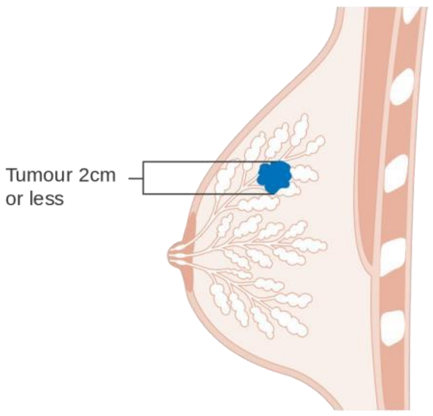
  Grading compares the appearance of the breast cancer cells to the appearance of normal breast tissue. Normal cells in an organ like the breast become differentiated, meaning that they take on specific shapes and forms that reflect their function as part of that organ. Cancerous cells lose that differentiation. In cancer, the cells that would normally line up in an orderly way to make up the milk ducts become disorganized. Cell division becomes uncontrolled. Cell nuclei become less uniform. Pathologists describe cells as well differentiated (low grade), moderately differentiated (intermediate grade), and poorly differentiated (high grade) as the cells progressively lose the features seen in normal breast cells. Poorly differentiated cancers (the ones whose tissue is least like normal breast tissue) have a worse prognosis.

- **Stage**

  Breast cancer staging using the TNM system is based on the size of the tumour (T), whether or not the tumour has spread to the lymph nodes (N) in the armpits, and whether the tumour has metastasized (M) (i.e. spread to a more distant part of the body). Larger size, nodal spread, and metastasis have a larger stage number and a worse prognosis.

  The main stages are:

  - **Stage 0:** is a pre-cancerous or marker condition, either ductal carcinoma in situ (DCIS) or lobular carcinoma in situ (LCIS).
  - **Stages 1–3:** are within the breast or regional lymph nodes as shown in Fig 1.1, 1.2, 1.3.
  - **Stage 4:**is 'metastatic' cancer that has a less favourable prognosis since it has spread beyond the breast and regional lymph nodes as shown in Fig 1.4.



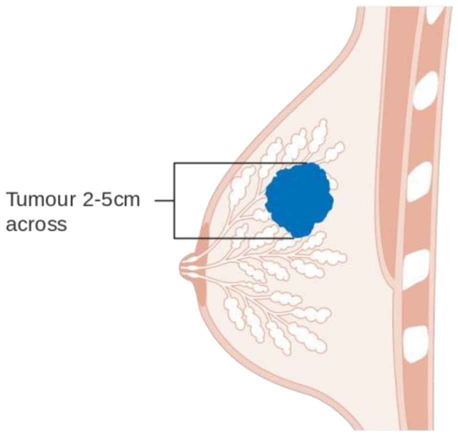
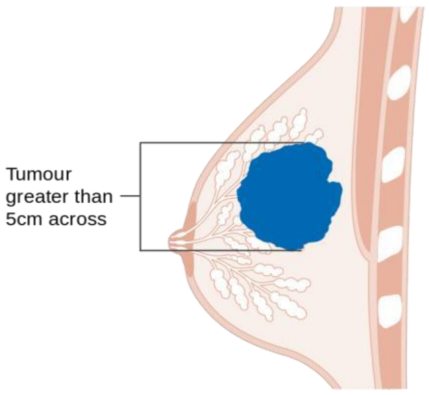


Fig 1.1 Stage T1

Fig 1.2 Stage T2



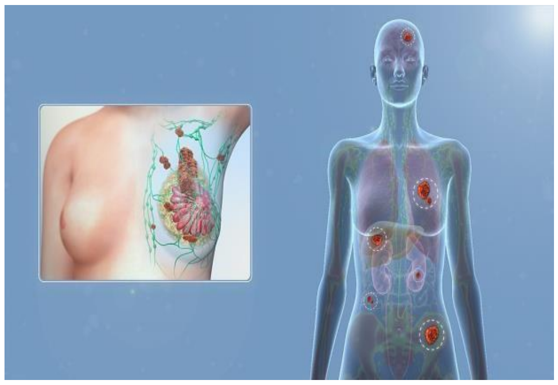


Fig 1.3 Stage T3 **Fig 1.4 Metastatic or Stage**

1.8 Existing System

In all the existing systems the prediction was only whether the person has cancer or not but there were no chances of identifying the stage and chances of survival. All the existing systems are implemented using supervised learning.

Among the several existing types of cancer, breast cancer (BC) presents two concerning characteristics

- It is the most common cancer among women worldwide.
- It presents a very high mortality rate when compared to other types of cancer.

Since histopathological analysis remains the most widely used method for BC diagnosis and most of the diagnosis continues being done by pathologists applying visual inspection of histological samples under the microscope, automatic classification of histopathological images is a research topic that can make BC diagnosis faster and less prone to errors

1.9 Proposed System

The investigation of the use of DeCAF features for breast cancer recognition using the dataset. The large size of dataset has given us the opportunity to compare, on the same dataset, CNN trained on natural images, which often is not possible with medical image datasets since they are too small.

From the results we can observe that these features are a viable alternative for a faster creation of image recognition systems using visual feature descriptors. Compared with a CNN trained from scratch, DeCAF features present comparable recognition rates.

In proposed system semi supervised learning is used to build and stages of cancer and chances of survival are also predicted.

1.10 Chapter Wise Summary

Here is a chapter wise summary of this report.

- **Chapter 2: Literature Survey**

  We reviewed 5 research papers and compared them to find the best approach for going about our project
- **Chapter 4: System Requirements Specification**

  This chapter contains all the hardware and software requirements
- **Chapter 5: System Analysis and Design**

  In this chapter we described both the approaches for preparing our model i.e. the Naive Bayes approach and the SVM approach
- **Chapter 6: Implementation**

  This chapter contains how the detailed implementation and phases.
- **Chapter 7: Results and Discussion**

  This chapter contains the results we observed and obtained from the project\
- **Chapter 8: Testing**

  This chapter contains the various testing methods used to test the project.
- **Chapter 9: Conclusion**

  This chapter contains the various conclusions drawn from our work

CHAPTER 2

LITERATURE SURVEY

There are various papers published on prediction of breast cancer using various methods. We have reviewed five papers and have drawn the following conclusions.

2.1Deep Features for Breast Cancer Histopathological Image Classification

Author: Fabio A. Spanhol, Paulo R. Cavalin, Luiz S. Oliveira, Caroline Petitjean, and Laurent Heutte

The idea of DeCAF features consists of extracting features from an image and using them as input for a classifier, as any other feature set. Nevertheless, DeCAF are based on representation learning, where the parameters of a neural network are learned in a way that raw data, i.e., the pixels of the images, can be converted to a high-level representation. The main difference between DeCAF features and the current standard of using CNNs is that a previously trained CNN is simply reused as feature extractor, the output of which is fed into another classifier, trained on problemspecific data.[1]

2.2 Remote Computer-Aided Breast Cancer Detection and Diagnosis System Based on Cytological Images

Author: Yasmeen Mourice George, Hala Helmy Zayed, Mohamed Ismail Roushdy, and Bassant Mohamed Elbagoury

They have developed a computer-aided diagnosis system for breast FNAC cancer. For the feature extraction, they used ten shape-based features and two textural features. The values obtained for these features yield a good differentiation between benign and malignant cells. For the classification phase, they have performed four different classification models, namely, multilayer perceptron, PNN, LVQ, and SVM. The classification results were obtained using tenfold cross validation.[2]

2.3 Breast Cancer Histopathological Image Classification using Convolutional Neural Network

Author: Fabio Alexandre Spanhol, Luiz S Oliveira, Caroline Petitjean, Laurent Heutte

They proposed a method based on the extraction of image patches for training the CNN and the combination of these patches for final classification. This method aims to follow the high-resolution Histopathological images from BreaKHis as input to existing CNN, avoiding adaptions of the model that can lead to a more complex and computationally costly architecture. The CNN performance is better than compared to previously reported results obtained by other Machine Learning models trained with hand-crafted textual descriptors. Finally, they also investigated the combination of different CNNs using simple fusion rules, achieving some improvement in recognition rates.[3]

2.4 Breast cancer diagnosis from biopsy images with highly reliable random subspace classifier ensembles

The classification system is built as a serial fusion of two different Random Subspace classifier ensembles with rejection options to enhance the classification reliability. The first ensemble consists of a set of Support Vector Machine classifiers that converts the original K-class classification problem into a number of K 2-class problems. The second ensemble consists of a Multi-Layer Perceptron ensemble, that focuses on the rejected samples from the first ensemble. For both of the ensembles, the reject option is implemented by relating the consensus degree from majority voting to a confidence measure, and abstaining to classify ambiguous samples if the consensus degree is lower than some threshold. We also investigated the effectiveness of a feature description approach by combining Local Binary Pattern (LBP) texture analysis, statistics derived using the Gray Level Co-occurrence Matrix (GLCM) and the Curvelet Transform. While the LBP analysis efficiently describes local texture properties and the GLCM reflects global texture statistics, the Curvelet Transform is particularly appropriate for the representation of piece-wise smooth images with rich edge information. The combined feature description thus provide a comprehensive biopsy image characterization by taking advantages of their complementary strengths. Using a benchmark microscopic biopsy image dataset, obtained from the Israel

Institute of Technology, a high classification accuracy of 99.25 % was obtained (with a rejection rate of 1.94 %) using the proposed system. [4]

2.5 Deep Learning for Magnification Independent Breast Cancer Histopathology Image Classification

Author: NeslihanBayramoglu, JuhoKannala, JanneHeikkil¨a

They proposed to classify breast cancer histopathology images independent of their magnifications using convolutional neural networks (CNNs). We propose two different architectures; single task CNN is used to predict malignancy and multi-task CNN is used to predict both malignancy and image magnification level simultaneously.

Evaluations and comparisons with previous results are carried out on BreaKHis dataset. Experimental results show that our magnification independent CNN approach improved the performance of magnification specific model. Our results in this limited set of training data are comparable with previous state-of-the-art results obtained by hand-crafted features. However, unlike previous methods, our approach has potential to directly benefit from additional training data, and such additional data could be captured with same or different magnification levels than previous data.[5]

CHAPTER 3

SYSTEM REQUIREMENTS SPECIFICATION

The chapter describes about the requirements. It specifies the hardware and software requirements that are in order to run the application properly. The Software Requirement Specification (SRS) is explained in detail, which includes the overview of functional and non-functional requirements.

An SRS document describes the functional and behavioural requirements of the software under development. SRS is a fundamental document which forms the foundation of the software development process. It is the complete description of the behaviour of a system to be developed it is not only the list of requirements of a system but also has a description of its major features. Requirement Analysis in system engineering and software engineering encompasses those tasks that go into determining the need or conditions to meet for a new or altered product.

The SRS functions as a blueprint for completing a project. It is important to note that an SRS contains functional and non- functional requirements.

Thus, the goal of preparing the SRS document is to

- To firm foundation for the design phase
- Support system testing facilities
- Support project management and control
- Controlling the Evolution of system

3.1 Functional Requirement

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

- Input test case must not have compilation and runtime errors.
- The application must not stop working when kept running for even a long time.
- The application should generate the output for a given input testcase.
- The application should generate the output for a given input test case

3.2 Non-Functional Requirement

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.

Given below are the non-functional requirements:

- Product requirements
- Organizational requirements
- Basic operational requirements
-

3.3 Hardware Requirements

- Atleast 128 MB of RAM. Large Datasets might require more RAM than this minimum.
- P2 instances of AWS: 4 core processor, 60GB RAM, Nvidia Tesla K80 Graphics card, 20 GB Hard disk space
- Atleast 75 MB of disk space must be available. Again, larger datasets will require more hard drive space than this

3.4 Software Requirements

- Atleast 128 MB of RAM. Large Datasets might require more RAM than this minimum.
- Language: Python 3.x, Libraries: Pandas, Tensorflow, NLTK, Seaborn ,
- Additional software: CUDA 8.0 for GPU support, Linux GPU driver
- Front end: Tkinter
- Database: SqLite
- Documentation: MS Word

CHAPTER 4

SYSTEM ANALYSIS AND DESIGN

4.1 System Architecture:

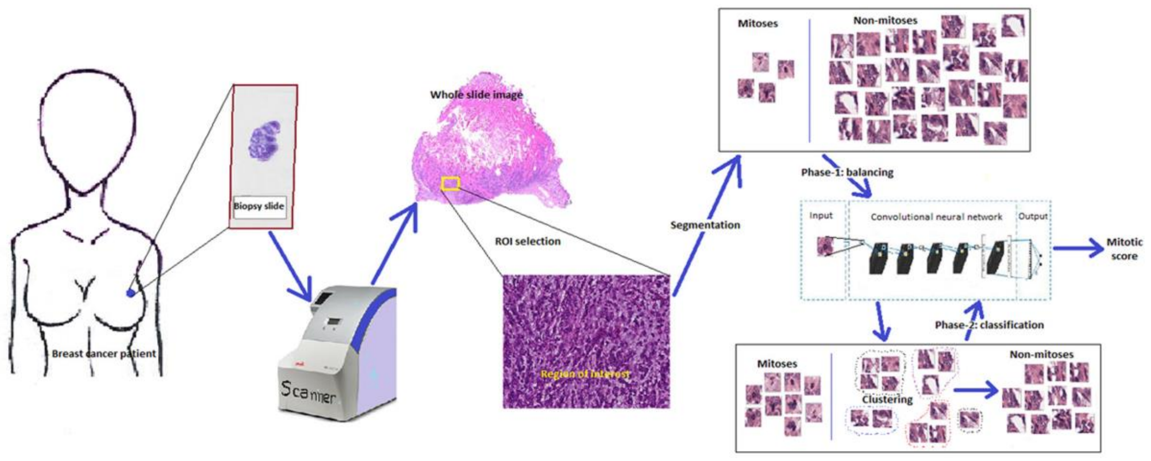


Fig 4.1 System Architecture

In this the image of the breast will be scanned to find out the lump and then the scanned image is then segmented into mitoses and non mitoses and then it is classified as malignant or benign cancer.

4.2 Flow Chart

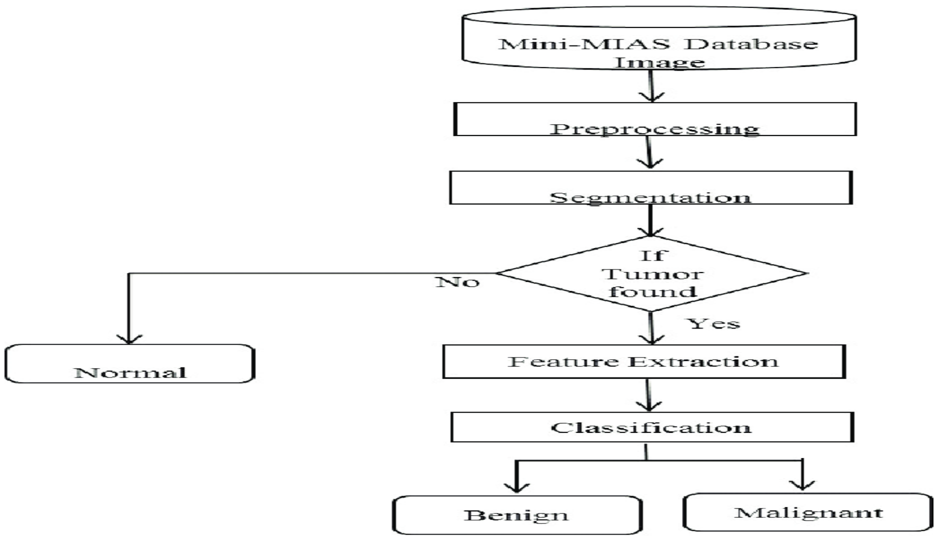


Fig 4.2 Flow Chart

Image from the database will be taken and it will be pre-processed and segmented. If the tumor is found then feature extraction takes place and then it is classified as benign or malignant cancer.

4.3 Data Flow of Use Case:

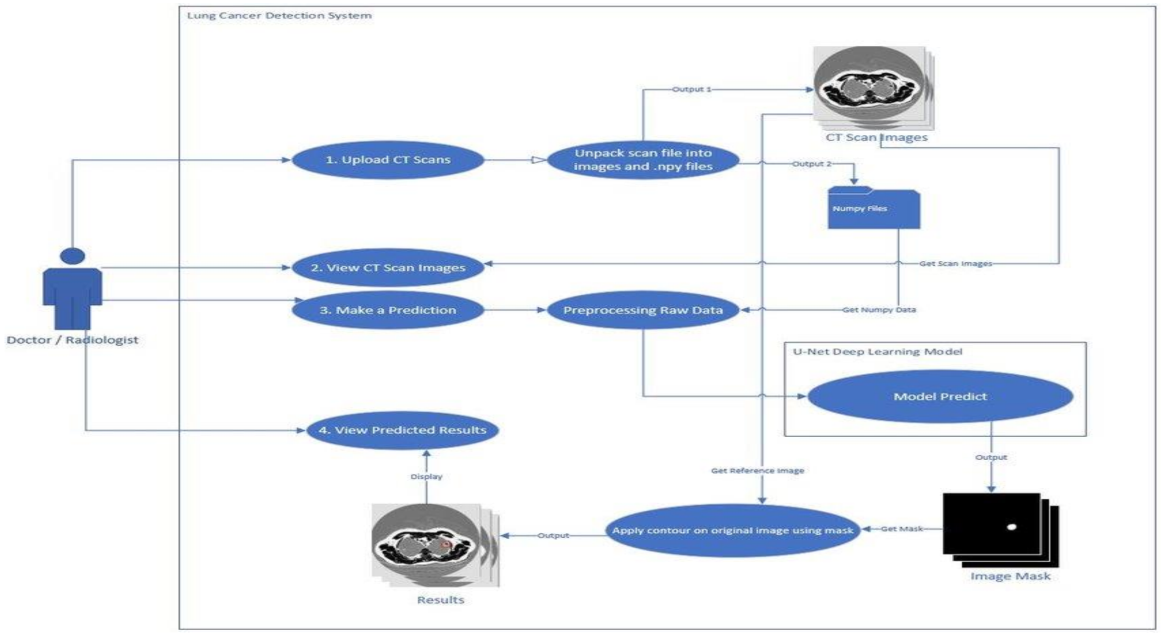


Fig 4.3 Data Flow of Use Case

In this doctor will upload CT scan and then view the CT scan images and then he will make predictions and also view the prediction results and then the image is processed and results will be predicted by masking the outer region of lump.

4.4 Data Flow of Sequence Diagram

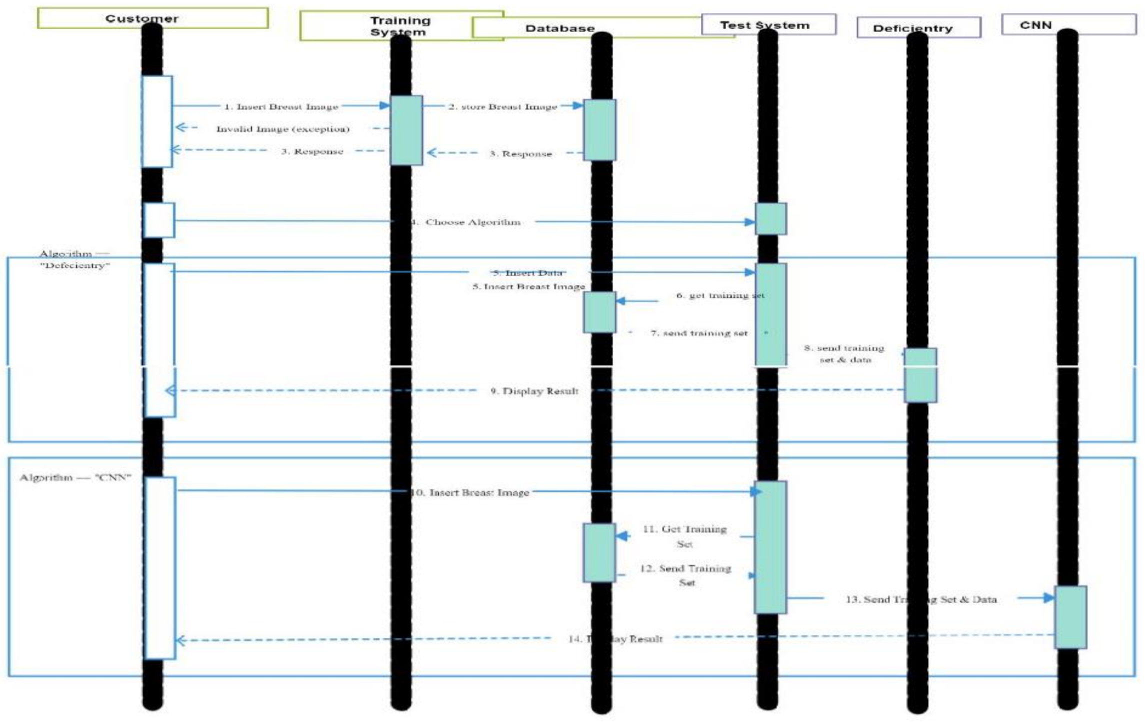


Fig 4.4 Data Flow of Sequence Diagram

In sequence diagram the images of the patient is scanned and uploaded in a database and then the training system is used to train the model and after training the model it is tested and the algorithms like CNN and Random forest are used to predict the accuracy.

CHAPTER 5

IMPLEMENTATION

The algorithms used are Neural network classifier and Random forest. Neural Networks Classifiers consists of units (**neurons**), arranged in layers, which convert an input vector into some output. Each unit takes an input, applies a (often nonlinear) function to it and then passes the output on to the next layer. CNN's are the most mature form of deep neural networks to produce the most accurate i.e. better than human results in computer vision. CNN's are made of layers of Convolutions created by scanning every pixel of images in a dataset. As the data gets approximated layer by layer, CNN's start recognizing the patterns and thereby recognizing the objects in the images

Convolutional Neural Networks have the following layers:

• Convolutional Layer

It is the input layer. The convolutional layer is the core building block of a CNN. This is where most of the user-specified parameters are in the network. The most important parameters are the number of kernels and the size of the kernels. A convolutional layer should have the following:

- Convolutional kernels defined by a width and height
- Number of input channels and output channels
- Depth of the Convolution filter must be equal to the number channels of the input feature map.

Convolutional layers convolve the input and pass its result to the next layer which is similar to the response of a neuron in the visual cortex to a specific stimulus. Each convolutional neuron processes data only for its respective receptive field.

• ReLU Layer

ReLU is the most commonly used activation function in neural networks, mainly in CNN. It applies the non-saturating activation function. It eliminates negative values from an activation map by making them zero. It increases the nonlinear properties of the decision function and of the overall network. ReLU is quick to evaluate. The main purpose of this layer is to introduce nonlinearity to a system that basically has just been computing linear operations during the previous conv layers. ReLU layers work far better because the network is able to train a lot faster (because of the computational efficiency) without making a significant difference to the accuracy.

- **Pooling Layer**

Convolutional networks include local or global pooling layers. Pooling layers help reduce the dimensions of the data by combining the outputs of neuron clusters at one layer into a single neuron in the next layer. Local pooling combines small clusters. Global pooling acts on all the neurons of the convolutional layer. *Max pooling* uses the maximum value from each of a cluster of neurons at the prior layer. *Average pooling* uses the average value from each of a cluster of neurons at the prior layer. Several non-linear functions are used to implement pooling among which *max pooling* is the most common. The pooling layer helps to reduce the spatial size of the representation, to reduce the number of parameters, memory footprint and amount of computation in the network, and therefore to also control overfitting. It is common to insert a pooling layer between successive convolutional layers in CNN.

- **Fully Connected Layer**

Fully connected layers connect every neuron in one layer to every neuron in another layer. After several convolutional and max pooling layers, the high-level reasoning is done via fully connected layers. Neurons in a fully connected layer have connections to all activations in the previous layer, as in regular (non-convolutional)artificial neural networks.

Random Forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction (See Fig 4.1).

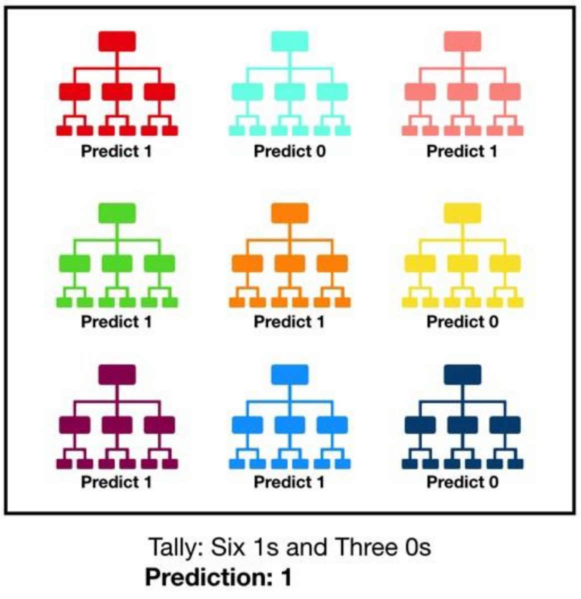


Fig 5.1: Random Forest

Compared to the classical Decision Tree explained above, every tree in a Random Forest is a weak tree because he did not have access to the complete data during the build process. But all together, they build an ensemble of "decision makers" where the collective result can be created by majority voting in the case of classification or averaging in the case of regression (see figure 4.2). This compensates for potential errors of single trees in the forest so that the model is less likely to produce results further away from the real values.

5.1 Algorithm (Bootstrap Aggregation)

1. Grow K different decision trees

- *i. Pick a random subset of the training examples (with return)*
- *ii. Pick d << D random attributes to split the data*
- *iii. Each tree is grown to the largest extent possible and there is no pruning.*

2. Given a new data point χ

- *i. Classify χ using each of the trees $T1 \ldots TK$*

ii. *Predict new data by aggregating the predictions of the tree trees (i.e., majority votes for classification, average for regression).*

An important strength of Random Forests is that they are able to perform still well in the case of missing data. According to their construction principle, not every tree is using the same features. If there is any missing value for a feature during the application there usually are enough trees remaining that do not use this feature to produce accurate predictions.

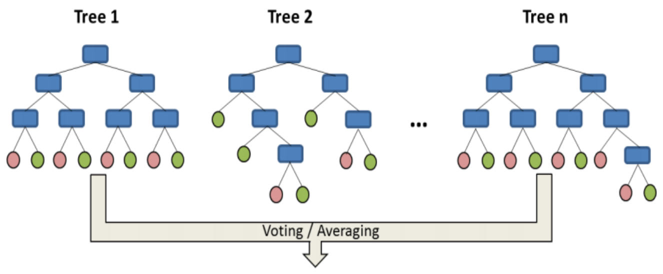


Fig 5.2 Random Forest as Tree Ensemble

5.2 Code

```
import pandas as pd

import numpy as np

import tensorflow
```

Load npz file containing image arrays

```
import pickle

x = pickle.load(open("images_data.pkl", "rb"))

y = pickle.load(open("class_label.pkl", "rb"))

x.shape

from sklearn.model_selection import train_test_split
```

First split the data in two sets, 90% for training, 10% for Val/Test

```
X_train, X_val, y_train, y_val = train_test_split(x,y, test_size=0.1, random_state=1,
stratify=y)
```

Import the VGG16 network architecture

```
from keras.applications import VGG16;

IMG_WIDTH = 50

IMG_HEIGHT = 50

IMG_DEPTH = 3

BATCH_SIZE = 16
```

Instantiate the model with the pre-trained weights (no top)

```
conv_base = VGG16(weights='./vgg16_weights_tf_dim_ordering_tf_kernels_notop.h5',

include_top=False,

input_shape=(IMG_HEIGHT, IMG_WIDTH, IMG_DEPTH))
```

Show the architecture

```
conv_base.summary()
```

Extract features

```
train_features = conv_base.predict(np.array(X_train), batch_size=BATCH_SIZE,
verbose=1)

val_features = conv_base.predict(np.array(X_val), batch_size=BATCH_SIZE, verbose=1)
```

Save extracted features

```
np.savez("train_features", train_features, y_train)

np.savez("val_features", val_features, y_val)
```

Current shape of features

```
print(train_features.shape, "\n",  val_features.shape)
```

Setting labels as numerical values for CNN

```
labs = list(set(y))

print('Length of labs: ', len(labs))

convbin = lambda x:[1 if i == x else 0 for i in range(len(labs))]

num_labs_train = np.asarray([convbin(labs.index(i)) for i in y_train])

num_labs_val = np.asarray([convbin(labs.index(i)) for i in y_val])

print(num_labs_train[0:10])

print(y_train[0:10])

print(num_labs_train.shape)

print(num_labs_val.shape)
```

Classification using Random Forest on extracted features

```
from sklearn.ensemble import RandomForestClassifier

tr = train_features.reshape(train_features.shape[0], -1)

clf = RandomForestClassifier(max_depth=5, n_estimators=5)



print(tr.shape)

clf.fit(tr, y_train)

val = val_features.reshape(val_features.shape[0], -1)

clf.score(val, y_val)
```

Classification using neural network

```
import tensorflow as tf
```

```
labs = list(set(y))

alt_num_labs_train = np.asarray([labs.index(i) for i in y_train])

alt_num_labs_val = np.asarray([labs.index(i) for i in y_val])

model = tf.keras.models.Sequential([

tf.keras.layers.Flatten(),

tf.keras.layers.Dense(512, activation=tf.nn.relu),

tf.keras.layers.Dropout(0.2),

tf.keras.layers.Dense(2, activation=tf.nn.softmax)])

adm = tf.keras.optimizers.Adam(lr=0.001, beta_1=0.9, beta_2=0.999, decay=0.9,
amsgrad=False)

model.compile(optimizer=adm,

          loss='sparse_categorical_crossentropy',

          metrics=['accuracy'])

model.fit(train_features, alt_num_labs_train, epochs=15)

model.evaluate(val_features, alt_num_labs_val)
```

Flatten extracted features

```
tr_flat = np.reshape(train_features, (train_features.shape[0], 1*1*512))

tst_flat = np.reshape(val_features, (val_features.shape[0], 1*1*512))

from keras import models

from keras import layers

from keras import optimizers

from keras import callbacks
```

```
import tensorflow as tf
```

Define the densely connected classifier

```
NB_TRAIN_SAMPLES = tr_flat.shape[0]

NB_VALIDATION_SAMPLES = tst_flat.shape[0]

NB_EPOCHS = 50

model = models.Sequential()

model.add(layers.Dense(512, activation='relu', input_dim=(1*1*512)))

model.add(layers.Dense(2, activation=tf.nn.softmax))

model.compile(

    loss='binary_crossentropy',

    optimizer=optimizers.RMSprop(lr=2e-4),

    metrics=['acc'])

reduce_learning = tf.keras.callbacks.ReduceLROnPlateau(

    monitor='val_loss',

    factor=0.2,

    patience=2,

    verbose=1,

    mode='auto',

    epsilon=0.0001,

    cooldown=2,

min_lr=0)

eary_stopping = tf.keras.callbacks.EarlyStopping(
    monitor='val_loss',

min_delta=0,

    patience=7,

    verbose=1,

    mode='auto')

callbacks = [reduce_learning, eary_stopping]
```

Train the model

```
history = model.fit(

tr_flat,

num_labs_train,

    epochs=NB_EPOCHS,

validation_data=(tst_flat, num_labs_val),

callbacks=callbacks

)

import matplotlib.pyplot as plt

%matplotlib inline

acc = history.history['acc']

val_acc = history.history['val_acc']

loss = history.history['loss']

val_loss = history.history['val_loss']

epochs = range(1, len(acc) + 1)

plt.title('Training and validation accuracy')
plt.plot(epochs, acc, 'red', label='Training acc')

plt.plot(epochs, val_acc, 'blue', label='Validation acc')

plt.legend()

plt.figure()

plt.title('Training and validation loss')

plt.plot(epochs, loss, 'red', label='Training loss')

plt.plot(epochs, val_loss, 'blue', label='Validation loss')

plt.legend()

plt.show()

labs_pred = model.predict(tr_flat[:10])

pred = [np.where(i == max(i))[0][0] for i in labs_pred]

pred

y_train[:10]
```

Classification on a specific image

```
path = r'./8863_idx5_x1151_y1101_class1.png'

import cv2



image = cv2.imread(path)

image = image.reshape(1, image.shape[0], image.shape[1], image.shape[2])

_sp_features = conv_base.predict(np.array(image), batch_size=BATCH_SIZE,
verbose=1)

class_lab = ["Benign", "Malignant"]
```

```
sp_features = _sp_features.reshape(-1, 512)

lab = model.predict(sp_features)

pred = [np.where(i == max(i))[0][0] for i in lab]

print("Diagnosis (predicted):", class_lab[pred[0]])

import matplotlib.pyplot as plt

import numpy as np

import pandas as pd

from sklearn import preprocessing

from sklearn.naive_bayes import GaussianNB

import seaborn as sns

import pandas as pd

import numpy as np

import pickle

x = pickle.load(open("images_data.pkl", "rb"))

y = pickle.load(open("class_label.pkl", "rb"))

from sklearn.model_selection import train_test_split
```

First split the data in two sets, 90% for training, 10% for Val/Test)

```
X_train, X_val, y_train, y_val = train_test_split(x,y, test_size=0.1, random_state=1, stratify=y)

from PIL import ImageTk

from PIL import Image as PILIMAGE

import requests
import json

from datetime import datetime

import sqlite3

conn = sqlite3.connect('breast_cancer')

cur = conn.cursor()

from tkinter import *

from tkinter import messagebox

from tkinter.filedialog import askopenfilename
```

Creating tables to enter the data of the doctors and patients

```
try:

cur.execute('''CREATE TABLE Doctor (

    name varchar(20) DEFAULT NULL,

     email varchar(50) DEFAULT NULL,

    password varchar(20) DEFAULT NULL,

    gender varchar(10) DEFAULT NULL,

    age int(11) DEFAULT NULL

  )''')

cur.execute('''CREATE TABLE Patient_details (

    name varchar(20) DEFAULT NULL,

    email varchar(50) DEFAULT NULL,

    gender varchar(10) DEFAULT NULL,

    age int(11) DEFAULT NULL,
pragnacy varchar(10) DEFAULT NULL

   )''')

except:

   pass

def sign_up():

    conn = sqlite3.connect('breast_cancer')

    cur = conn.cursor()

    root2 = Tk()

    root2.title('Breast Cancer')

    root2.geometry('850x650')

    root2.configure(background="white")

var = StringVar()

    label=Label(root2,textvariable= var,font=('arial',20,'bold'),bd=20,background="white")

var.set('Breast Cancer detection Using deep learing')

label.grid(row=0,columnspan=6,column=1)


L1=Label(root2,text="Name",bd=8,background="white",height=1,padx=16,pady=16,font
=('arial',16,'bold'),width=10,)

    L1.grid(row = 1,column=0)

    E1  = Entry(root2)

    E1.grid(row=1,column=5)
```

```
L2=Label(root2,text="Email",bd=8,background="white",height=1,padx=16,pady=16,font
=('arial',16,'bold'),width=10,)

   L2.grid(row = 2,column=0)

   E2  = Entry(root2)

   E2.grid(row=2,column=5)


L3=Label(root2,text="Password",bd=8,background="white",height=1,padx=16,pady=16,
font=('arial',16,'bold'),width=10,)

   L3.grid(row = 3,column=0)

   E3  = Entry(root2,show='*')

   E3.grid(row=3,column=5)


L4=Label(root2,text="Gender",bd=8,background="white",height=1,padx=16,pady=16,fo
nt=('arial',16,'bold'),width=10,)

   L4.grid(row = 4,column=0)

   E4  = Entry(root2)

   E4.grid(row=4,column=5)


L5=Label(root2,text="Age",bd=8,background="white",height=1,padx=16,pady=16,font=
('arial',16,'bold'),width=10,)

   L5.grid(row = 5,column=0)

   E5  = Entry(root2)

   E5.grid(row=5,column=5)
```

```
 def insert_1():

cur.execute("insertinto Doctor(name,email,password,gender,age) values ('%s','%s','%s','%s','%s')" % (E1.get(), E2.get(), E3.get(), E4.get(), E5.get()))

conn.commit()

    print('data inserted')


B1=Button(root2,text="Submit",height=1,padx=16,pady=16,bd=8,font=('arial',16,'bold'), width=10,bg="white",command = insert_1)

 B1.grid(row=6,column=0)

root = Tk()

root.title('Breast cancer')

root.geometry('850x650')

root.configure(background="white")

var = StringVar()

label = Label( root, textvariable = var,font=('arial',20,'bold'),bd=20,background="white")

var.set('Breast cancer')

label.grid(row=0,columnspan=6,column=1)

conn = sqlite3.connect('breast_cancer')

cur = conn.cursor()

var = StringVar()

label = Label( root, textvariable = var,font=('arial',20,'bold'),bd=20,background="white")

var.set('Breast Cancer detection Using deep learing')

label.grid(row=0,columnspan=6,column=1)
```

```
L1=Label(root,text="Email",bd=8,background="white",height=1,padx=16,pady=16,font=('arial',16,'bold'),width=10,)

L1.grid(row = 1,column=0)

E1  = Entry(root)

E1.grid(row=1,column=5)

L2=Label(root,text="Password",bd=8,background="white",height=1,padx=16,pady=16,font=('arial',16,'bold'),width=10,)

L2.grid(row = 2,column=0)

E2  = Entry(root,show='*')

E2.grid(row=2,column=5)

def auth():

global conv_base

cur.execute('select * from Doctor where email = "{}" and password = "{}"'.format(E1.get(),E2.get()))

    c = len(cur.fetchall())

    if c > 0:

        root1= Tk()

        root1.title('Crop and Yield Prediction Using Data Mining')

        root1.geometry('750x750')

        #root1.configure(background="#82b74b")

lbl = Label(root1, text="")

lbl.place(x=250,y=20)

lbl.config(font=("Courier 40 italic bold "),background="white")
```

```
def OpenFile_train():

     name                                                                                    =
askopenfilename(initialdir="C:/Users/Batman/Documents/Programming/tkinter/",

                         filetypes =(("png file", "*.png"),("All Files","*.*")),

                         title = "Choose a file.")

      try:

         with open(name,'r') as UseFile:

           print(name)

           """root11 = Tk()

           label = Label( root11, text = name , fg='red' )

label.grid(row=0,columnspan=6,column=1)"""

           image = cv2.imread(name)

           image = image.reshape(1, image.shape[0], image.shape[1], image.shape[2])

           _sp_features = conv_base.predict(np.array(image), batch_size=BATCH_SIZE,
verbose=1)

class_lab = ["Benign", "Malignant"]

sp_features = _sp_features.reshape(-1, 512)

           lab = model.predict(sp_features)

pred = [np.where(i == max(i))[0][0] for i in lab]

           print("Diagnosis (predicted): {}".format( class_lab[pred[0]]))

           root11 = Tk()

           root11.geometry('850x650')
```

```
                root11.configure(background="white")

                label = Label( root11, text ="Diagnosis (predicted): {}".format(
class_lab[pred[0]]),  fg='red' )

label.grid(row=0,columnspan=6,column=1)

        except FileNotFoundError:

                root11 = Tk()

                label = Label( root11, text = "No file exists" , fg='red' )

label.grid(row=0,columnspan=6,column=1)

                print("No file exists")

        def pteint_details():

                conn = sqlite3.connect('breast_cancer')

                cur = conn.cursor()

cur.execute("select * from Patient_details")

                s = cur.fetchall()

                print(s)

                root11 = Tk()

                label = Text( root11 )

label.grid(row=0,column=1)

label.insert('1.0',s)

        def pteint():

            conn = sqlite3.connect('breast_cancer')

            cur = conn.cursor()
        root2 = Tk()

        root2.title('pateint Cancer')

        root2.geometry('850x650')

        root2.configure(background="white")

var = StringVar()


label=Label(root2,textvariable=var,font=('arial',20,'bold'),bd=20,background="white")

var.set('Breast Cancer detection Using deep learing')

label.grid(row=0,columnspan=6,column=1)


L1=Label(root2,text="Name",bd=8,background="white",height=1,padx=16,pady=16,font
=('arial',16,'bold'),width=10,)

        L1.grid(row = 1,column=0)

        E1  = Entry(root2)

        E1.grid(row=1,column=5)


L2=Label(root2,text="Email",bd=8,background="white",height=1,padx=16,pady=16,font
=('arial',16,'bold'),width=10,)

        L2.grid(row = 2,column=0)

        E2  = Entry(root2)

        E2.grid(row=2,column=5)


L3=Label(root2,text="Gender",bd=8,background="white",height=1,padx=16,pady=16,fo
nt=('arial',16,'bold'),width=10,)
```

```
        L3.grid(row = 3,column=0)

        E3  = Entry(root2)

        E3.grid(row=3,column=5)


L4=Label(root2,text="Age",bd=8,background="white",height=1,padx=16,pady=16,font=
('arial',16,'bold'),width=10,)

        L4.grid(row = 4,column=0)

        E4  = Entry(root2)

        E4.grid(row=4,column=5)


L5=Label(root2,text="Pragnacy",bd=8,background="white",height=1,padx=16,pady=16,f
ont=('arial',16,'bold'),width=10,)

        L5.grid(row = 5,column=0)

        E5  = Entry(root2)

        E5.grid(row=5,column=5)

 def insert_1():

cur.execute("insert into Patient_details (name,email,gender,age,pragnacy) values
('%s','%s','%s','%s','%s')" % (E1.get(), E2.get(), E3.get(), E4.get(), E5.get()))

conn.commit()

           print('data inserted')


B1=Button(root2,text="Submit",height=1,padx=16,pady=16,bd=8,font=('arial',16,'bold'),
width=10,bg="white",command = insert_1)

        B1.grid(row=6,column=0)
```

```
B=Button(root1,text="BrowseImage",height=1,padx=16,pady=16,bd=20,font=('arial',16,'
bold'),width=10,command=OpenFile_train)#,bg="#d9bf77")

B.place(x=50,y=250)

 B1=Button(root1,text="Entry
Pateints",height=1,padx=16,pady=16,bd=20,font=('arial',16,'bold'),width=10,command=p
teint)#,bg="#d9bf77")

        B1.place(x=350,y=250)



        B2=Button(root1,text="Pateints
Details",height=1,padx=16,pady=16,bd=20,font=('arial',16,'bold'),width=10,command=pt
eint_details)#,bg="#d9bf77")

        B2.place(x=650,y=250)

    else:

        root11 = Tk()

        label = Label( root11, text = 'Invalid Username and password' , fg='red' )

label.grid(row=0,columnspan=6,column=1)



bg_image = PhotoImage(file ="agri1.ppm")

x = Label (image = bg_image)

x.grid(row = 0, column = 0)
```

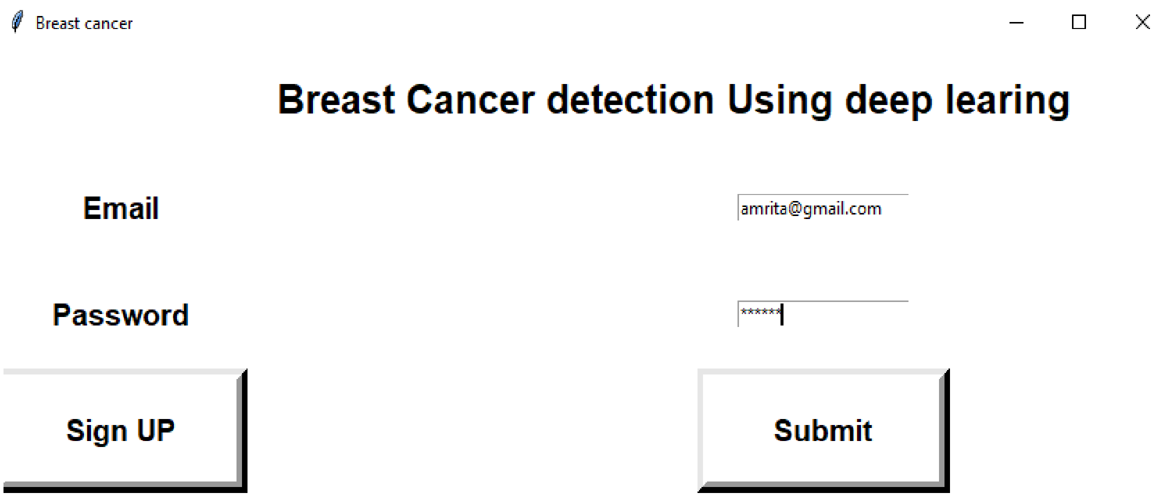
```
B1=Button(root,text="Submit",height=1,padx=16,pady=16,bd=8,font=('arial',16,'bold'),width=10,bg="white",command = auth)

B1.grid(row=3,column=5)


B2=Button(root,text="SignUP",height=1,padx=16,pady=16,bd=8,font=('arial',16,'bold'),width=10,bg="white",command = sign_up)

B2.grid(row=3,column=0)

root.mainloop()
```

CHAPTER 6

RESULTS AND DISCUSSION

6.1 Screenshots

This is the first page that gets displayed when the code is run. The doctor can sign in here. If doctor is using it for the first time then he/she can sign up. To sign in, doctor can give email id, password and click submit so that doctor will be navigated to another page.



Email amrita@gmail.com

Password ******

Fig 6.1 Login page

The page where user can sign up. The doctor should provide the details. Such as name, email id, password, gender, age and then submit. Once the doctor is registered, next time just by giving email id and password he can login into the website.

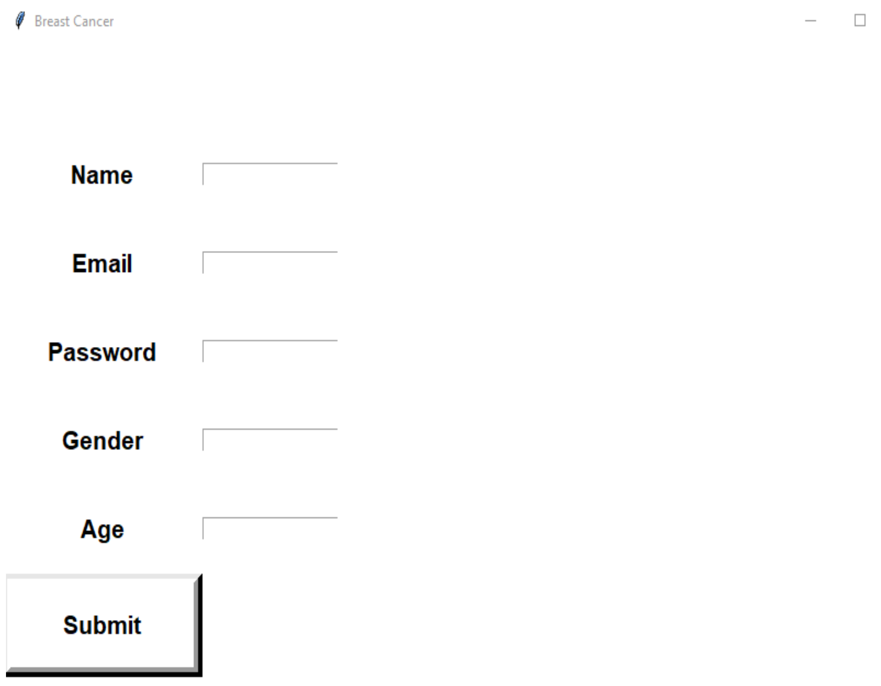


Fig 6.2 Sign up page

Once the user logs in this is the page that gets displayed. Browse image, Entry patients, Patients details are the three blocks here.

Browse image is where the image should be given and it will be predicted to benign or malignant.

Entry Patient is where details of the patients will be registered.

Patient details shows the details of all the patients registered.

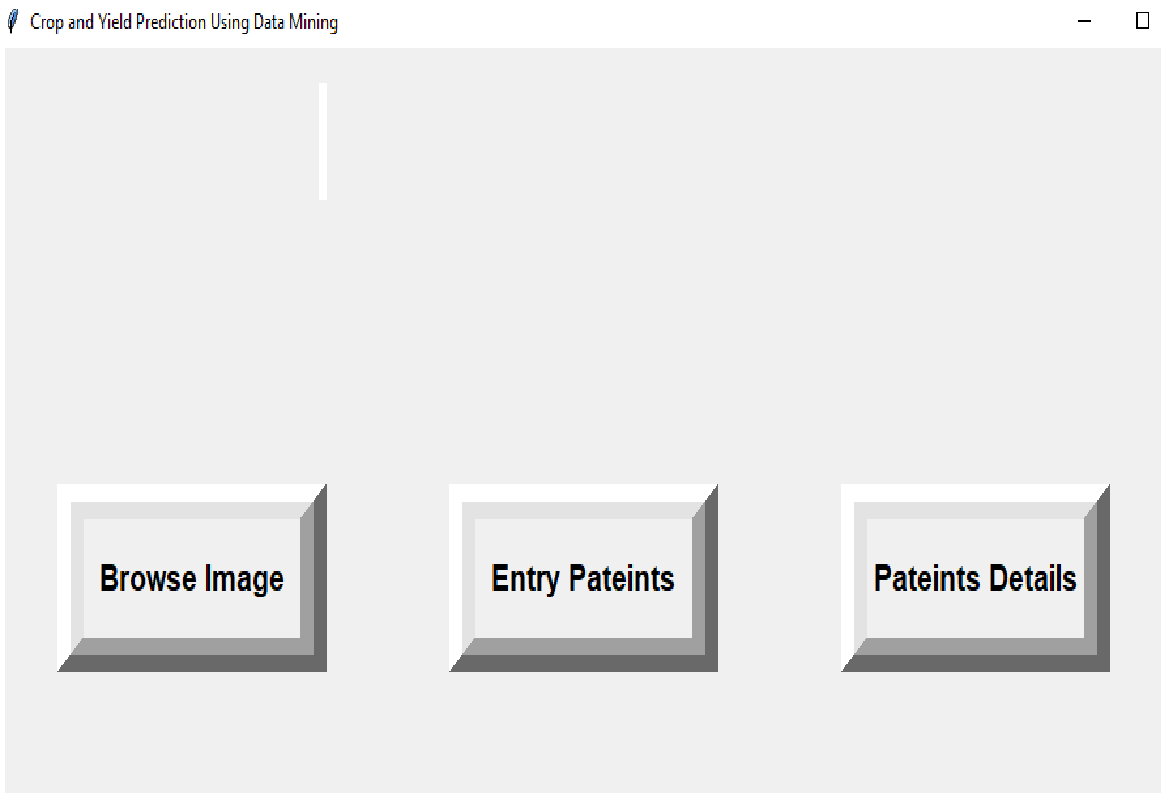


Fig 6.3 Page after login

If Browse Image in the above figure is selected then this page gets displayed. Anyone image can be selected. After selecting the image, click on open. Some Backend operations takes place and image will be predicted if it is benign or malignant.

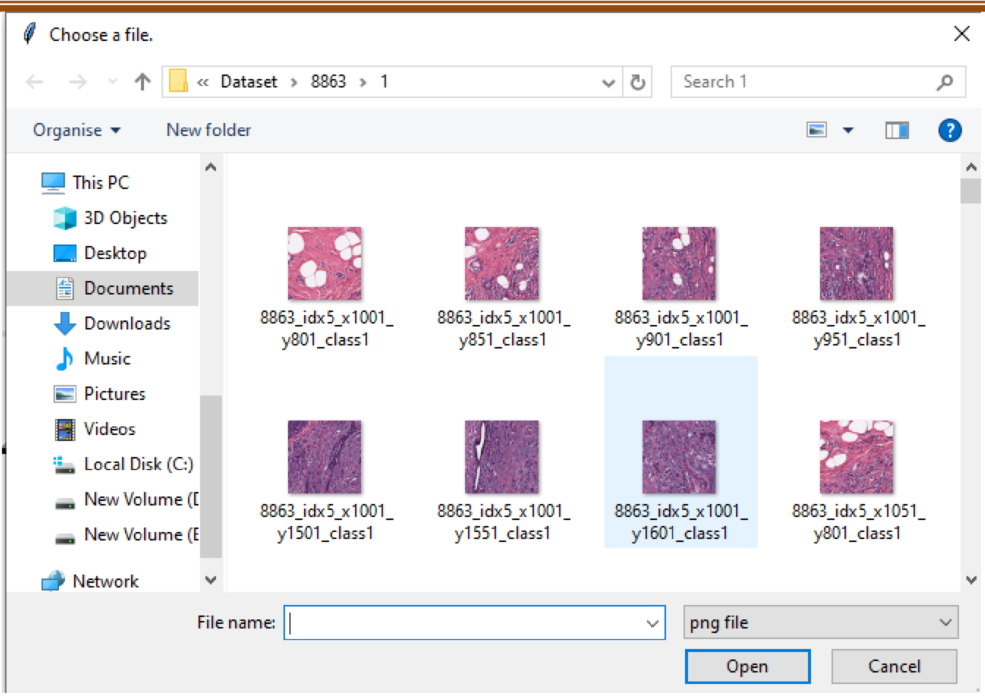

Choose a file.
« Dataset › 8863 › 1
Search 1
Organise
New folder
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Local Disk (C:)
New Volume (D
New Volume (E
Network
8863_idx5_x1001_y801_class1
8863_idx5_x1001_y851_class1
8863_idx5_x1001_y901_class1
8863_idx5_x1001_y951_class1
8863_idx5_x1001_y1501_class1
8863_idx5_x1001_y1551_class1
8863_idx5_x1001_y1601_class1
8863_idx5_x1051_y801_class1
File name:
png file
Open
Cancel

Fig 6.4 To select the image

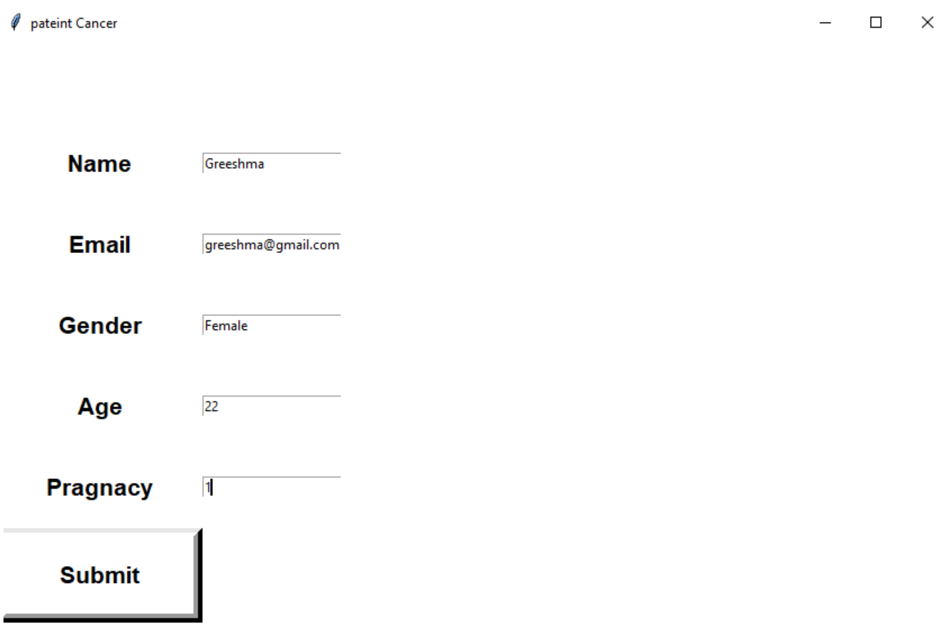
After browsing the image, the result is predicted. It shows that the result predicted as Benign.

Diagnosis (predicted): Benign

Fig 6.5 Result page

If Entry Patient is selected this is the page that gets displayed. It asks for the details of the patient such as name, email, gender, age, pregnancy.

After entering the details click on submit and it will get stored.

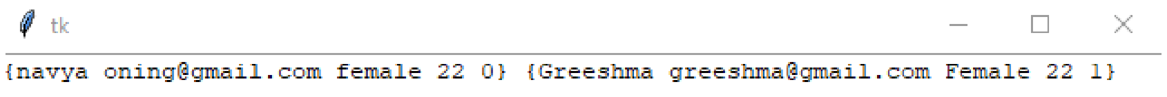


Name	Greeshma
Email	greeshma@gmail.com
Gender	Female
Age	22
Pragnacy	1

Fig 6.6 Patient registration

If patient details block is selected then details of all the patients that are registered will be displayed.

The details include name, email, gender, age, pregnancy of the patient.



```
{navya oning@gmail.com female 22 0} {Greeshma greeshma@gmail.com Female 22 1}
```

Fig 6.7 Patient details

CHAPTER 7

TESTING

Software testing defines as an activity to check whether the actual results match the expected results and to ensure that the system is defect free.

7.1 Functional vs. Non-functional Testing

The goal of utilizing numerous testing methodologies in your development process is to make sure your software can successfully operate in multiple environments and across different platforms. These can typically be broken down between functional and non-functional testing. Functional testing involves testing the application against the business requirements. It incorporates all test types designed to guarantee each part of a piece of software behaves as expected by using uses cases provided by the design team or business analyst. These testing methods are usually conducted in order and include:

- Unit testing
- Integration testing
- System testing
- Acceptance testing

Non-functional testing methods incorporate all test types focused on the operational aspects of a piece of software. These include:

- Performance testing
- Security testing
- Usability testing
- Compatibility testing

7.2 Unit Testing

Unit testing is the first level of testing and is often performed by the developers themselves. It is the process of ensuring individual components of a piece of software at the code level are functional and work as they were designed to. Developers in a testdriven environment will typically write and run the tests prior to the software or feature being passed over to the test team. Unit testing can be conducted manually, but automating the process will speed up delivery cycles and expand test coverage. Unit testing will also make debugging easier because finding issues earlier means they take less time to fix than if they were discovered later in the testing process. TestLeft is a tool that allows advanced testers and developers to shift left with the fastest test automation tool embedded in any IDE.

7.3 Integration Testing

After each unit is thoroughly tested, it is integrated with other units to create modules or components that are designed to perform specific tasks or activities. These are then tested as group through integration testing to ensure whole segments of an application behave as expected (i.e, the interactions between units are seamless). These tests are often framed by user scenarios, such as logging into an application or opening files. Integrated tests can be conducted by either developers or independent testers and are usually comprised of a combination of automated functional and manual tests.

7.4 System Testing

System testing is a black box testing method used to evaluate the completed and integrated system, as a whole, to ensure it meets specified requirements. The functionality of the software is tested from end-to-end and is typically conducted by a separate testing team than the development team before the product is pushed into production.

7.5 Acceptance Testing

Acceptance testing is the last phase of functional testing and is used to assess whether or not the final piece of software is ready for delivery. It involves ensuring that the product is in compliance with all of the original business criteria and that it meets the end user’s needs. This requires the product be tested both internally and externally, meaning you’ll need to get it into the hands of your end users for beta testing along with those of your QA team. Beta testing is key to getting real feedback from potential customers and can address any final usability concerns.

7.6 Performance Testing

Performance testing is a non-functional testing technique used to determine how an application will behave under various conditions. The goal is to test its responsiveness and stability in real user situations. Performance testing can be broken down into four types:

- **Load testing** is the process of putting increasing amounts of simulated demand on your software, application, or website to verify whether or not it can handle what it's designed to handle.
- **Stress testing** takes this a step further and is used to gauge how your software will respond at or beyond its peak load. The goal of stress testing is to overload the application on purpose until it breaks by applying both realistic and unrealistic load scenarios. With stress testing, you'll be able to find the failure point of your piece of software.
- **Endurance testing,** also known as soak testing, is used to analyze the behavior of an application under a specific amount of simulated load over longer amounts of time. The goal is to understand how your system will behave under sustained use, making it a longer process than load or stress testing (which are designed to end after a few hours). A critical piece of endurance testing is that it helps uncover memory leaks.
- **Spike testing** is a type of load test used to determine how your software will respond to substantially larger bursts of concurrent user or system activity over varying amounts of time. Ideally, this will help you understand what will happen when the load is suddenly and drastically increased.

7.7 Security Testing

With the rise of cloud-based testing platforms and cyber attacks, there is a growing concern and need for the security of data being used and stored in software. Security testing is a non-functional software testing technique used to determine if the information and data in a system is protected. The goal is to purposefully find loopholes and security risks in the system that could result in unauthorized access to or the loss of information by probing the application for weaknesses. There are multiple types of this testing method, each of which aimed at verifying six basic principles of security:

1. Integrity

2. Confidentiality
3. Authentication
4. Authorization
5. Availability
6. Non-repudiation

7.8 Usability Testing

Usability testing is a testing method that measures an application's ease-of-use from the end-user perspective and is often performed during the system or acceptance testing stages. The goal is to determine whether or not the visible design and aesthetics of an application meet the intended workflow for various processes, such as logging into an application. Usability testing is a great way for teams to review separate functions, or the system as a whole, is intuitive to use.

7.9 Compatibility Testing

Compatibility testing is used to gauge how an application or piece of software will work in different environments. It is used to check that your product is compatible with multiple operating systems, platforms, browsers, or resolution configurations. The goal is to ensure that your software's functionality is consistently supported across any environment you expect your end users to be using.

7.10 Testing With TestComplete

TestComplete is our robust automated GUI testing tool that excels in compatibility and integration testing. It helps QA teams create and run tests across desktop, mobile, and web applications – enabling testing professionals to speed up delivery cycles and improve software quality. Testcomplete comes with built-in support for various test environments, integrations to performance testing tools, as well as support for developer friendly SCMs, allowing you to seamlessness integrate it into your development t process. Using TestComplete will enable you to build a robust testing framework that utilizes the broad spectrum of available software testing methodologies.

CHAPTER 8

CONCLUSION AND FUTURE SCOPE

8.1 Conclusion

In this work we presented an investigation of the use of DeCAF features for breast cancer recognition using the BreakHis dataset. The large set of the dataset has given us the opportunity to compare, on the same dataset, CNN trained from scratch with features repurposed from another CNN trained on natural images , which often is not possible with medical image dataset .Since they are too small.

8.2 Contribution

CNN trained on ImageNet enable detection of nodules in medical images. As future work, one direction is to improve the recognition accuracy of DeCAF features and to predict the chances of a person getting cancer in future based on his habits and previous medical history.

8.3 Future Scope

The project gives barrier-free access to the literature of research. It increases convenience, reach and retrieval power. This puts rich and poor on an equal footing for these key resources.

The research will keep up to date with latest advances in the field of cancer. It will reach the readers directly by email for free of cost.

REFERENCES

[1] Fabio A. Spanhol, Paulo R. Cavalin, Luiz S. Oliveira, Caroline Petitjean, and Laurent Heutte, "Deep Features for Breast Cancer Histopathological Image Classification ", *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Banff, AB, 2017, pp. 1868-1873, doi: 10.1109/SMC.2017.8122889.

[2] Yasmeen Mourice George, Hala Helmy Zayed, Mohamed Ismail Roushdy, and Bassant Mohamed Elbagoury "Remote Computer-Aided Breast Cancer Detection and Diagnosis System Based on Cytological Images", in *IEEE Systems Journal*, vol. 8, no. 3, pp. 949-964, Sept. 2014, doi: 10.1109/JSYST.2013.2279415.

[3] F. A. Spanhol, L. S. Oliveira, C. Petitjean and L. Heutte, "Breast cancer histopathological image classification using Convolutional Neural Networks," *2016 International Joint Conference on Neural Networks (IJCNN),* Vancouver, BC, 2016, pp. 2560-2567, doi: 10.1109/IJCNN.2016.7727519.

[4] Zhang, Y., Zhang, B., Coenen, F. *et al.* Breast cancer diagnosis from biopsy images with highly reliable random subspace classifier ensembles. *Machine Vision and Applications* **24,** 1405–1420 (2013). https://doi.org/10.1007/s00138-012-0459-8

[5] N. Bayramoglu, J. Kannala and J. Heikkilä, "Deep learning for magnification independent breast cancer histopathology image classification," *2016 23rd International Conference on Pattern Recognition (ICPR)*, Cancun, 2016, pp. 2440-2445, doi: 10.1109/ICPR.2016.7900002.

Text Books:

[1] Tom M.Mitchell Machine learning india edition 2013, McGraw hill education

[2] Zen K,Zang CY, a novel class of biomarks to diagnose and monitor human breast cancer.