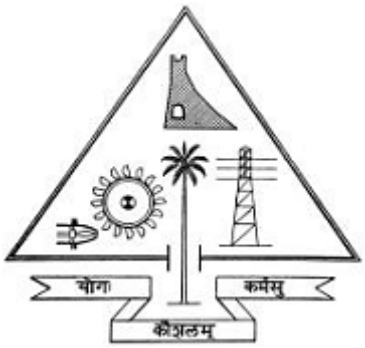
BREAST CANCER DIAGNOSIS USING DEEP LEARNING

Mini project submitted in partial fulfillment of the requirements for the award of the degree of ***Master of Computer Applications*** *of the* ***APJ Abdul Kalam Technological University***

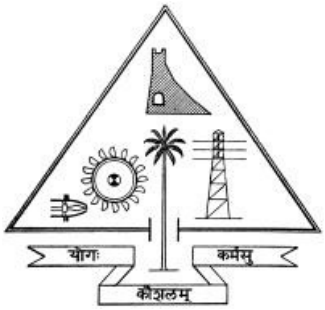
submitted by

ANJU S **(TCR20MCA-2010)**



DEPARTMENT OF COMPUTER APPLICATIONS
GOVERNMENT ENGINEERING COLLEGE
THRISSUR - 680009

DEPARTMENT OF COMPUTER APPLICATIONS
GOVERNMENT ENGINEERING COLLEGE, THRISSUR
THRISSUR, KERALA STATE, PIN 680009



CERTIFICATE

This is to certify that the mini project titled ***"BREAST CANCER DIAGNOSIS USING DEEP LEARNING"*** *is a bonafide work done by*

ANJU S ***(TCR20MCA-2010)***

under my supervision and guidance, and is submitted in February 2022 in partial fulfillment of the requirements for the award of the Degree of Master of Computer Applications from APJ Abdul Kalam Technological University(KTU).

Project Coordinator	**Project Guide**	**Head of the Dept.**
Dr. Sminesh C N	Hussain Ahmed	Dr. Sminesh C N
Asst.Professor	Asst.Professor	Assoc. Professor

Place : THRISSUR
Date :

DECLARATION

I hereby declare that the mini project named, **Breast Cancer Diagnosis Using Deep Learning**, is my own work and that, to the best of my knowledge and belief, it contains no material previously published another person nor material which has been accepted for the award of any other degree or course of the university or any other institute of higher learning, except where due acknowledgement and reference has been made in the text.

Signature

ANJU S **(TCR20MCA-2010)**

ACKNOWLEDGEMENT

We would like to thank Computer Application Department of GEC Thrissur, for giving us this opportunity to pursue this project and successfully complete it.

I'm highly indebted to our guide, **Asst.Prof. Hussain Ahmed** for his guidance and constant supervision as well as for providing necessary information regarding the project and also for his support in completing the project.

I would like to express our heart-felt gratitude to project coordinator, **Dr. Sminesh C N** , for her committed guidance, valuable suggestions, constructive criticisms and precious time that she invested throughout the work.

I also express special thanks to the Head of the Computer Applications Department, **Dr. Sminesh C N**, for his keen support and consistent encouragement in our academic activities.

I sincerely thank all other faculties of MCA department for guiding through the processes involved in the project .

ABSTRACT

In the present days one of the most meaning diseases found on women is the Breast Cancer. The prime target must be curing the Cancer. Before that we should first detect the cancer then only the next process of diagnosing takes place. The detecting of cancer at early stages which decrease the odds for survival. But it becomes worst when it is not detected as early as possible. The later stages will even lead to death. This paper is focused on developing a model that will clinically help the physicians to diagnose the Breast Cancer at early stages. Diagnosis mainly occurs when there is a tumour growth is observed. But before the tumour growth the change in texture may happen in the area near the breast where tumour may be grown. In this paper, that changes in structural parameter of the mammography images are detected with the help of Image Processing techniques along with Convolutional Neural Network in regards with Deep Learning.

CONTENTS

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Cancer is one of the deadliest diseases around the world. Among all types of cancers, women are mostly affected by breast cancer which can turn to fatal if it is not diagnosed earlier. Currently the first procedure of diagnosing breast cancer is by taking mammography images. After that radiologist and physician clinically detect the mammography image as malign or benign only if a tumour is found on those mammography image regions. But this is not the accurate may of diagnosing breast cancer because tumours can be observed only at the end of early stages and stage 1. So, in such a condition there is a huge chance of spreading cancerous cell to various part of our body. Thus, always detection of cancerous cell in stage 0 or starting of early stages is preferable and is more prone for curing. Also diagnosing at earlier stages will also reduce the struggle faced by the patient.

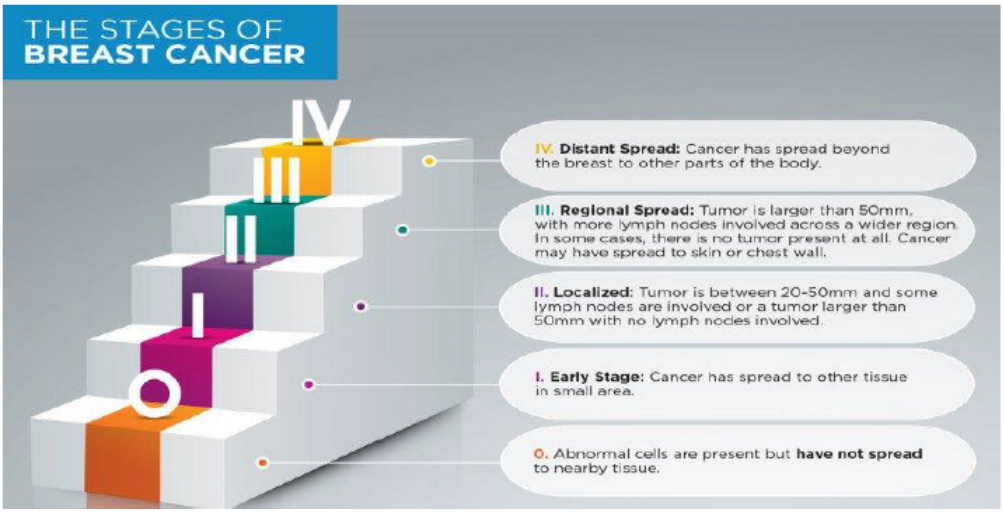


Fig. 1.1: Different stages of Breast Cancer

The Figure 1.1 shows the different stages of breast cancer. In first two stage, It is the initial stage of cancer formation. Basic tissue texture change and cancerous cell formation occurs. There won't be much changes to health condition and patient cant recognize any symptoms or pain. From stage 2 it starts growing and it will start easily spreading to nearby cells thus forming a cluster of cancerous cell and patient start getting various changes in health condition and symptoms. It clearly shows at the end of stage 1 the cancerous cell will start spreading and difficult to treat and this will also increase the number of chemotherapy and radiation counts thus making the patient week. So, it is always better to diagnose earlier.

1.1 Background

Breast Cancer diagnosis using Deep learning helps the Doctors to identify the tumour in its early stages. So that adequate treatment can be provided to the affected patient. In this system we use Deep Learning method. Mammography images are given as input to train the machine to detect whether it stage of Breast Cancer is Benin or malign. Tkinter is used as an user interface to predict the output.

1.2 Motivation

In the earlier times the Medical Team could only identify the tumour when it is visible to human eye or at the final stages. So it will give a mere chance of survival of the patient. this proposed system will assist the medical team to provide a support to detect the cancer at its early stages. This will increase the chances of survival of the patient.

1.3 Objective

The objectives of the project Breast Cancer Diagnosis Using Deep Learning is as follows :

1. To develop a deep learning model which diagnose breast cancer in its earlier stages
2. To give support to physician to notify the structural changes in breast tissue before tumor growth is observed.
3. Breast cancer diagnosis model with transparent and clear manual prediction.
4. Image classification using Convoluted neural network.

1.4 Contribution

The major contributions in this project are:

1. Designed and Developed improved model for detect the Breast Cancer
2. Improved the accuracy greater than 90 percentage

CHAPTER 2
EXISTING SYSTEM

In Existing System black and white Xray image are used as input and this input is used as an older method of diagnosing medium.Comparing to colour images black and white images are very difficult to capture the texture difference in the early stages. So, it will be very difficult to diagnose in the early stages. Late diagnosing always increase the risk rate and reduces the treatment chances. In the working system of Ahmet Kadir et al. (2019), it is proposed as a web-based application to detect breast cancer detection platform which need an input image and classifies as benign and malignant image. They use convoluted neural network using keras and shiny packages in R programming to create user interface. The drawbacks of this proposed model is they are using black and white Xray image as input. Tariq et al. they used most of the supervised and unsupervised algorithms with black and white images as input.

CHAPTER 3
ENVIRONMENTAL STUDY

3.1 System Configuration

System configuration describe the hardware and software requirement of the system for development

3.1.1 Hardware Requirements

- Memory : 4 GB of RAM
- Processor : Intel Core i3 or equivalent CPU
- Speed : 2.4 GHz
- Proper Internet Connection

3.1.2 Software Requirements

- Operating system : Windows
- Front End : Python
- IDE Used : Jupyter Notebook
- Libraries : NumPy, Tensorflow, Keras

3.2 Software Specification

3.2.1 Python

Python was developed by Guido van Rossum in the early 1990s. Python is an interpreted, high-level and general-purpose programming language.Python's

design philosophy emphasizes code readability with its notable use of significant white space. Its language constructs and object-oriented approach aim to help programmers write clear, logical code for small and large-scale projects.

3.2.2 Tensorflow

TensorFlow is an open-source framework developed by Google researchers to use machine learning, deep learning, and other analytical and statistical work elements. Like similar platforms, it was designed to simplify the process of designing and implementing advanced analytic programs for users such as data scientists. And typical programmers and forecasters. TensorFlow handles data sets that are placed as graphical interfaces, and the edges connecting the nodes to the graph can represent vectors or multidimensional matrices, forming what is known as tensors. We imported keras from tensorflow. Tensorflow provides libraries required for classification, data splitting (validation and training).h5py library used to convert array to HDF5 binary data format . We created validation and training files using h5py library in h5 format.

3.2.3 Keras

Keras is a small library of neural networks written mostly in Python and capable of running over TensorFlow . It is built with the focus on enabling fast test. It puts a front and centre user experience. Keras follows the best ways to reduce the load of understanding: provides consistent and simple APIs, reduces the number of actions by the user required for standard use cases, and provides clear and effective feedback about user error. In particular, the neural layers, cost functions, optimizers, implementation schemes, implementation functions and general programs are all standalone modules that you can integrate to create new models. We used keras for implementing Neural network architecture.

3.2.4 NumPy

NumPy is a library for Python. It adds support for large matrices and multi-dimensional arrays, along with a large collection of high-level mathematical functions to operate on these arrays. NumPy in Python gives functionality comparable to MATLAB since they are both interpreted. They allow the user to write fast programs as long as most operations work on arrays or matrices instead of scalars. Trained model divides the given video into specified number of frames.A frames contains attributes such as width, height andRGBcolor code. Dataset trained with Corrosion detection stored as array values. And Validation set and training set is also taken as array for training with lstm . We used numpy to fit these with the model.

CHAPTER 4
SYSTEM DESIGN

4.1 Workflow

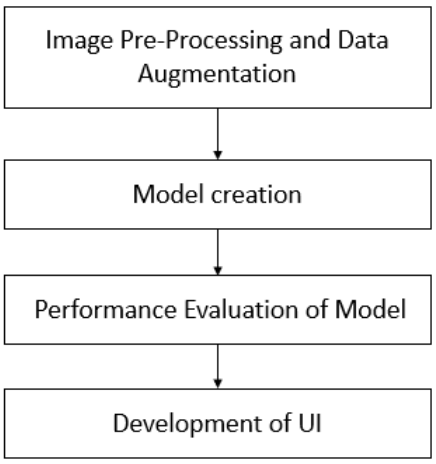


Fig. 4.1: Workflow Diagram

The implementation of our project will have 4 steps:

1. Data Collection and Image Processing
2. Create a Neural Network Model
3. Evaluate the model
4. UI Integration and Development

The Workflow diagram of the system is given in Figure 4.1

4.2 Methodology

4.2.1 Introduction

4.2.2 CNN

Convolutional Neural Network is one of the main categories to do image classification and image recognition in neural networks. The main operations in convolutional networks are:

4.2.2.1 Convolutional layer

convolution layer transforms the input image in order to extract features from it. In this transformation, the image is convolved with a kernel (or filter). A kernel is a small matrix, with its height and width smaller than the image to be convolved. It is also known as a convolution matrix or convolution mask.

4.2.2.2 Pooling Layer

Pooling layer plays an important role in pre-processing of an image. Pooling layer reduces the number of parameters when the images are too large. Pooling is "downscaling" of the image obtained from the previous layers.

4.2.2.3 Dense Layer

A dense layer speaks to a network vector duplication (expecting your group size is 1). The qualities in the network are the trainable boundaries which get refreshed during backpropagation. A dense layer in this way is utilized to change the elements of your vector. Numerically, it applies a pivot, scaling, interpretation change to your vector.

4.2.2.4 Fully Connected Layer

In a fully-connected layer, neurons are fully connected with all activations in the past layer, whose structure is same as ordinary neural systems. The main contrast between fully connected layers and convolutional layers is,

convolutional layers are associated with just a nearby area of the info. Fully connected layer takes the info volume and yields a N-dimensional vector.

4.2.2.5 Activation Function

The activation function is the non direct change that we do over the information signal. This changed yield is then sent to the following layer of neurons as input.In this undertaking utilizes two principle actuation functions.

4.2.2.6 ReLU

Rectified Linear unit(ReLU) transform functions only activates a node if the input is above a certain quantity. While the data is below zero, the output is zero, but when the input rises above a certain threshold. It has a linear relationship with the dependent variable.

4.2.2.7 Sigmoid Function

The sigmoid function is good for representing a probability. A weighted sum of inputs is passed through an activation function and this output serves as an input to the next layer. When the activation function for a neuron is a sigmoid function it is a guarantee that the output of this unit will always be between 0 and 1.

4.3 Data Flow Diagram

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated. DFDs can also be used for the visualization of data processing (Structured Design). A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of process or information about whether processes will operate in sequence or in parallel.

Fig. 4.2: Level 0 - Data-Flow Diagram

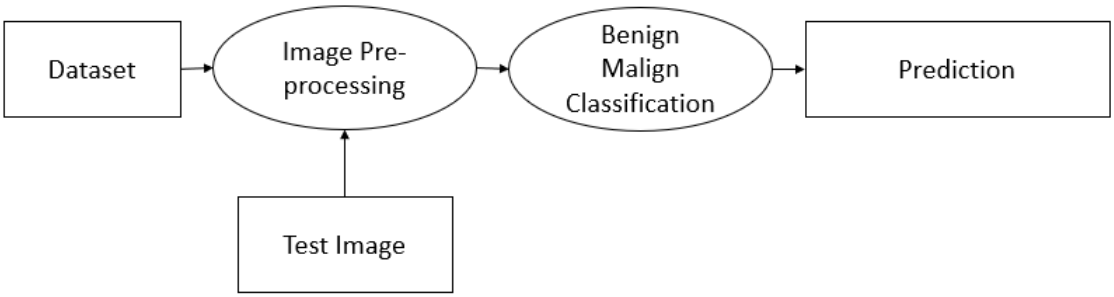


Fig. 4.3: Level 1 - Data-Flow Diagram

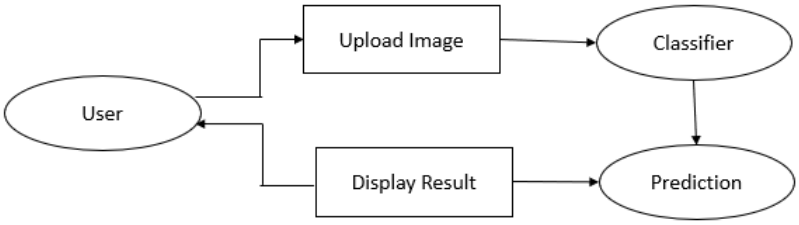


Fig. 4.4: Level 2 - Data-Flow Diagram

4.4 Implementation

Breast Cancer detection using deep learning is used in this project.The dataset has mammography images.This model has 2 main classes, that is malignant images and benignant images. This model train 8000 mammography images and test 1600 mammography images. Deep learning is used for this, more specifically saying Convolutional Neural Network (CNN) is used for build a model . CNN can extract features without losing much characteristics of an input. CNN has are 3 layers for convolutional neural network such as input layer, hidden layer, output layer. Input and output layers are basically front and back visible end but the middle layer determine the end products. In middle hidden layer it mainly consists of pixel conversion, activation function which is followed by adding filters, pooling and normalisation layer. In this system create a sequential CNN model to detect Breast Cancer.The Workflow diagram of the CNN model is given in Figure 4.3. The Hyper-parameters used in this CNN model are.

Parameters	Values/Attribute
Training sample data	8000
Validation sample data	1600
Batch size	32
Epochs	30
Class mode	Binary
Activation function	ReLu and Sigmoid
Normalization	Batch normalization
Pooling	Max pooling
Optimizer	Adam
Loss function	Binary cross entropy

Fig. 4.5: Hyper-parameters of proposed model

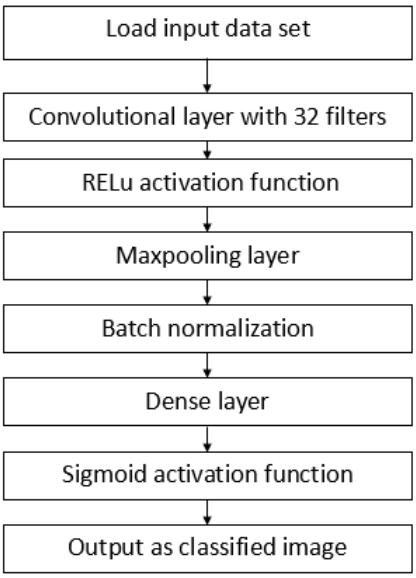


Fig. 4.6: workflow diagram of CNN model

The proposed model gives 93 percentage accuracy when train the mammography images at 30 epochs. Then save the model . Then predict the sample mammography images which is Benign or Malign. By using that saved model, Generate User interface using tkinter. User Interface contains one input field for collecting the input image. And one Button for posting the collected image into the model and for prediction. A box in the user interface show the result image. And the result status ie: Benign or Malign shown in a message box.

CHAPTER 5

RESULTS AND SCREENSHOT

5.1 Source code screenshots

```
import keras
from keras.models import Sequential
from keras.layers import  BatchNormalization
from keras.preprocessing import image
import tensorflow as tf
from keras.preprocessing.image import ImageDataGenerator
from keras.callbacks import History
from keras.layers.core import Dense, Dropout, Activation, Flatten
from keras.layers.convolutional import Conv2D, MaxPooling2D, ZeroPadding2D
from keras import backend as K
```

Fig. 5.1: Source code sample screenshot.1

```
img_width, img_height = 150, 150
train_data_dir = 'C:/Users/Anju Satheesh/Dropbox/My PC (LAPTOP-N0RD666U)/Desktop/mini_project/masterwork/train'
validation_data_dir = 'C:/Users/Anju Satheesh/Dropbox/My PC (LAPTOP-N0RD666U)/Desktop/mini_project/masterwork/test'
nb_train_samples = 8000
nb_validation_samples = 1600
epochs = 30
batch_size = 32
```

Fig. 5.2: Source code sample screenshot.2

```
if K.image_data_format() == 'channels_first':

    input_shape = (3, img_width, img_height)
else:
    input_shape = (img_width, img_height, 3)


train_datagen = ImageDataGenerator(
rescale=1. / 255,
shear_range=0.2,
zoom_range=0.2,
horizontal_flip=True)
```

Fig. 5.3: Source code sample screenshot.3

```
test_datagen =ImageDataGenerator (rescale=1. / 255)

train_generator = train_datagen.flow_from_directory(
train_data_dir,
target_size=(img_width, img_height),
batch_size=batch_size,
class_mode ='binary')

validation_generator = test_datagen.flow_from_directory(
    validation_data_dir,
    target_size=(img_width, img_height),
    batch_size=batch_size,
    class_mode='binary'
)
```

Fig. 5.4: Source code sample screenshot.4

```
model1 = Sequential()
model1.add(Conv2D(32, (3, 3), input_shape = [150,150,3]))
model1.add(Activation('relu'))
model1.add(BatchNormalization())
model1.add(MaxPooling2D(pool_size = (2, 2)))

model1.summary ()

model1.add(Conv2D(64, (3, 3)))
model1.add(Activation('relu'))
model1.add(MaxPooling2D(pool_size = (2, 2)))

model1.add(Conv2D(128, (3, 3)))
model1.add(Activation('relu'))
model1.add(MaxPooling2D(pool_size = (2, 2)))
```

Fig. 5.5: Source code sample screenshot.5

```
model1.add(Flatten())
model1.add(Dense(128))
model1.add(Activation('relu'))
model1.add(Dropout(0.5))
model1.add (Dense(1))
model1.add(Activation('sigmoid'))

model1.summary()
```

Fig. 5.6: Source code sample screenshot.6

```
model1.compile(loss ='binary_crossentropy',optimizer ='Adam',metrics = ['accuracy'])
```

```
history=model1.fit_generator(
train_generator,
steps_per_epoch=nb_train_samples // batch_size,
epochs =epochs,
validation_data= validation_generator,
validation_steps =nb_validation_samples // batch_size)
```

Fig. 5.7: Source code sample screenshot.7

```
import matplotlib.pyplot as plt
plt.plot(history.history['accuracy'])
plt.plot(history.history['val_accuracy'])
plt.title('model accuracy')
plt.ylabel('accuracy')
plt.xlabel('epoch')
plt.legend(['train', 'Validation'], loc='upper left')
plt.show()
```

Fig. 5.8: Source code sample screenshot.8

```
import tkinter as tk
from PIL import ImageTk, Image
from tkinter import filedialog
import keras
from keras.preprocessing import image
import numpy as np
from tkinter import messagebox


def load_img():
    global img, image_data
    for img_display in frame.winfo_children():
        img_display.destroy()

    image_data = filedialog.askopenfilename(initialdir="", title="Choose an image",filetype =[("image","*.png")])
    #filetypes=(("all files", "*.*"), ("png files", "*.png")))

    basewidth = 150 # Processing image for displaying
    img = Image.open(image_data)
    wpercent = (basewidth / float(img.size[0]))
    hsize = int((float(img.size[1]) * float(wpercent)))
    img = img.resize((basewidth, hsize), Image.ANTIALIAS)
    img = ImageTk.PhotoImage(img)
    file_name = image_data.split('/')
    panel = tk.Label(frame, text= str(file_name[len(file_name)-1])).pack()
    panel_image = tk.Label(frame, image=img).pack()
```

Fig. 5.9: Source code sample screenshot.9

```
def classify():
    original = Image.open(image_data)
    original = original.resize((150, 150), Image.ANTIALIAS)
    model = keras.models.load_model('C:/Users/Anju Satheesh/Dropbox/My PC (LAPTOP-N0RD666U)/Desktop/mini_project/cnn1.mo
    image_test = image.img_to_array(original)

    image_test = image. img_to_array(image_test)
    image_test = np.expand_dims( image_test, axis =0 )

    result = model.predict(image_test)

    print(result)

    if (result == 0):
        #pred = "benign"
        messagebox.showinfo("YOUR RESULT", "BENIGN")
    else:
        #pred = "malign"
        messagebox.showinfo("YOUR RESULT", "MALIGN")
    label = pred
    table = tk.Label(frame, text="Result:").pack()
    result = tk.Label(frame,text= str(label).upper()).pack()
```

Fig. 5.10: Source code sample screenshot.10

```
root = tk.Tk()
root.title('Cancer')
#root.iconbitmap('class.ico')
root.resizable(False, False)

tit = tk.Label(root, text="BREAST CANCER DIAGNOSIS", padx=25, pady=6, font=("", 12)).pack()

canvas = tk.Canvas(root, height=600, width=800)
canvas.pack()

frame = tk.Frame(root, bg='white')
frame.place(relwidth=0.6, relheight=0.5, relx=0.2, rely=0.25)

chose_image = tk.Button(root, text='Choose Image',
                        padx=35, pady=10,
                        fg="white", bg="grey", command=load_img)
chose_image.pack(side=tk.LEFT)

class_image = tk.Button(root, text='Submit',
                        padx=35, pady=10,
                        fg="white", bg="grey", command=classify)
class_image.pack(side=tk.RIGHT)


root.mainloop()
```

Fig. 5.11: Source code sample screenshot.11

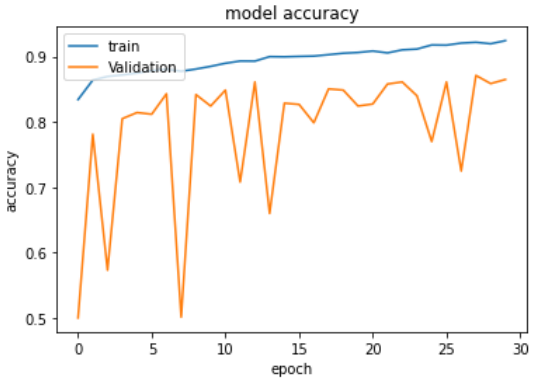


Fig. 5.12: Model accuracy

5.2 Results

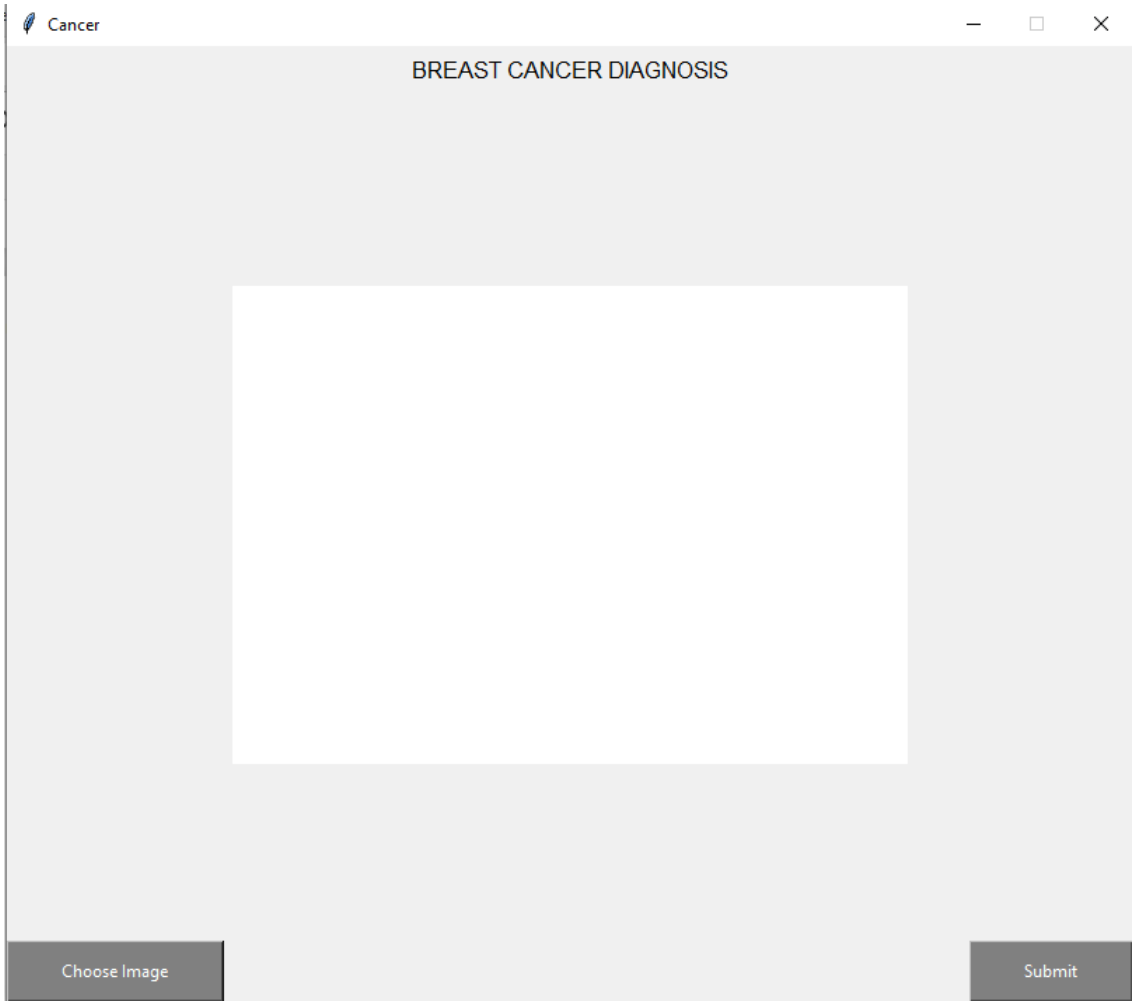


Fig. 5.13: User Interface

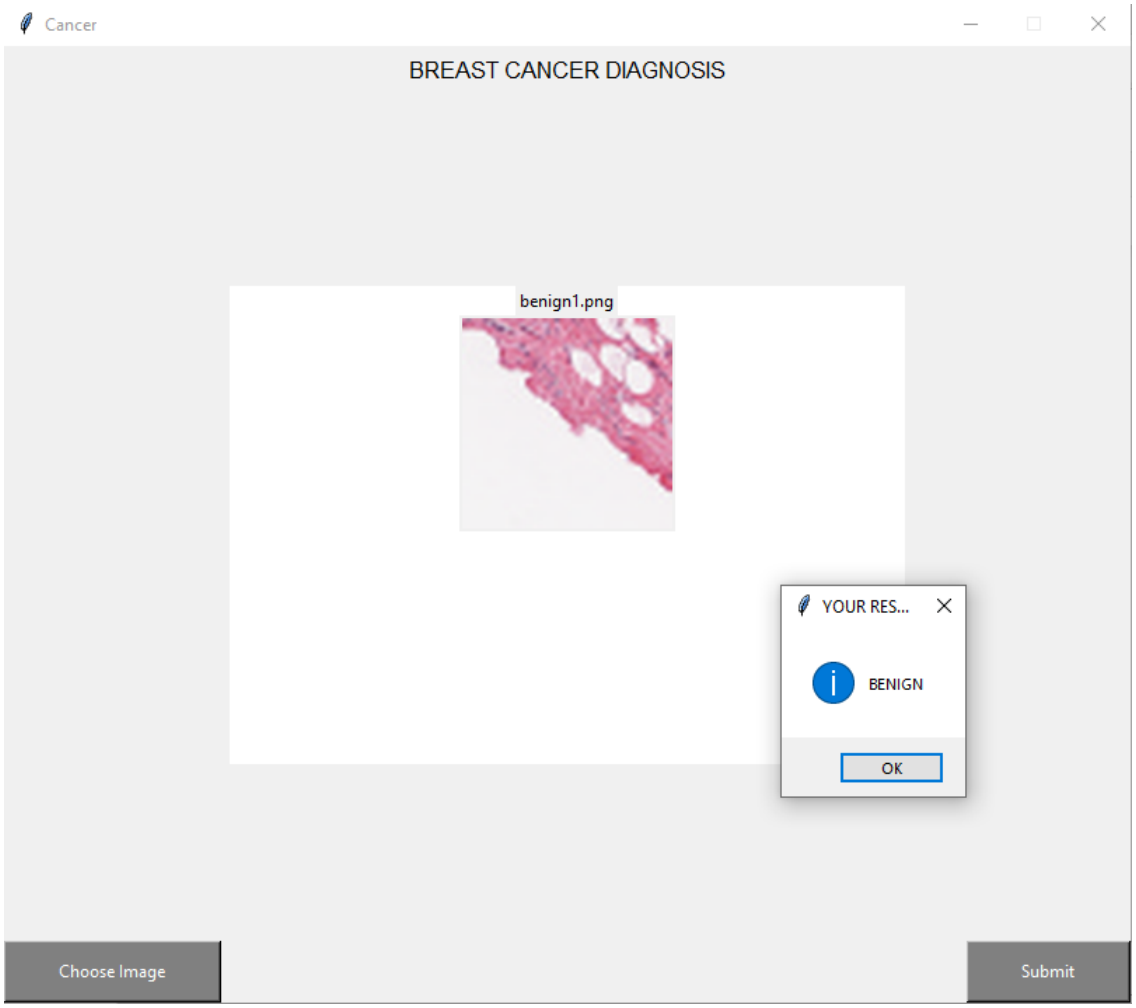


Fig. 5.14: Output.1

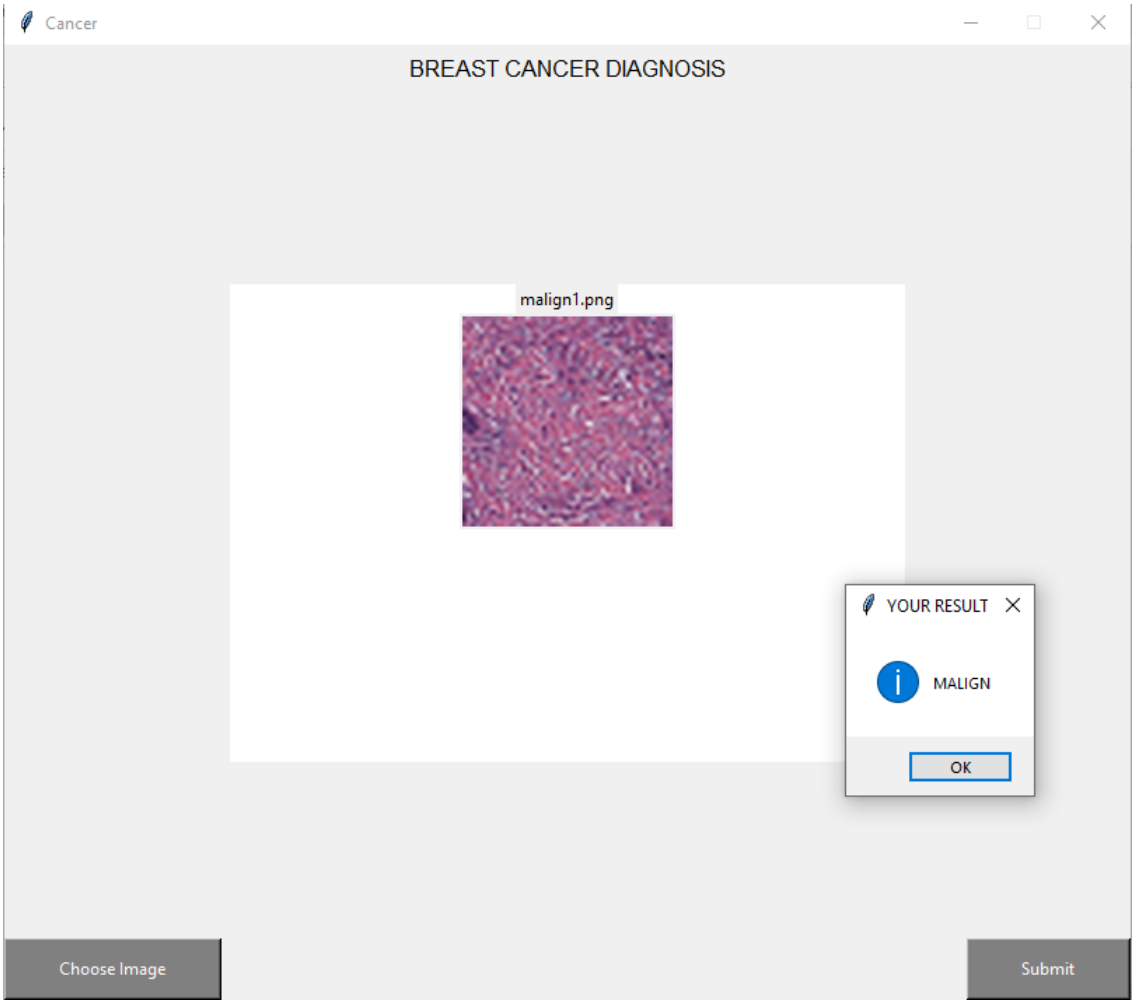


Fig. 5.15: Output.2

CHAPTER 6
CONCLUSION

One of the major criteria for the proper treatment of Breast Cancer is to diagnose within the early stages. It is impossible to predict the Breast Cancer with human eye in early stages before the tumour or until there is a structural change occurs in the affected path. So for better convenience machine aided support helps to notice the changes in texture of the tissues. This model provides assistant to the clinical team to recognize the changes in the mammography images in the early stage. So, with the help of this deep learning model of Breast Cancer Diagnosing using convolutional neural network can provide a partial assistance to the medical team.

CHAPTER 7
REFERENCES

[1] Tobias Christian Cahoon, Melanie A Sutton, James C Bezdek, *"Breast cancer detection using image processing technique"*, IEEE trans. On study of computerized disease diagnostic, pp.91885-91903 July 25 2019.

[2] Ahmet Kadir Arslan, Şeyma Yaşar, Cemil Çolak,*" Breast cancer classification using a constructed convolutional neural network on the basis of the histopathological images by an interactive web-based interface"*, IEEE trans on image classification, sep2019.

[3] Mahamood ,Tariq. , Li, jianqlang., Yan, Pei .,Faheem,Akthar. , Azhar Imran and Khalil, Ur Rehman. (2016) . *"A Brief Survey on Breast Cancer Diagnostic with Deep Learning Schemes Using Multi-Image Modalities."* IEEE trans, volume 6.

[4] Karthikeyan Ganesan, U. Rajendra Acharya, Chua Kuang Chua, Lim Choo Min, K. Thomas Abraham, and Kwan-Hoong Ng: *"Computer-Aided Breast Cancer Detection Using Mammograms"*, IEEE trans. On Biomedical engineering vol6 2013.

[5] Prannoy Giri , K Saravanakumar: *"Breast cancer detection using image processing techniques"*, oriental journal on computer science and technology,volume10, no2,pgs391-399, April 2017.