Cairo University
Faculty of Computers and Artificial Intelligence
Department of Computer Sciences

Chemotherapy
Efficiency Detector

Final Documentation

Project Title

Breast cancer classification in treatment stages

Supervised by

Dr. Sabah Sayed

TA. Huda Ahmed

Implemented by

20178001	**Ibrahim Adel Mahmoud**
20188011	**Eman Mostafa Ahmed**
20188056	**Haidy Ali Khalil**

Graduation Project
Academic Year 2021-2022

Table of Contents

List of Figures

List of Tables

List of Abbreviations

CT ---------> Computed Tomography Scan

PCR -------> Polymerase Chain Reactiono

NONPCR -> non Polymerase Chain Reaction

PET ------- > Positron Emission Tomography

MR --------> Magnetic Resonance

t1 ----------> first

t2 ----------> second stage

t3 ----------> third stage

CNN -------> Convolutional Neural Network

GUI Design ---------> Graphic User Interface

Chapter 1: Introduction

Introduction

If we search about breast cancer, we can find a lot of projects detected the cancer if it malignant or benign. Or we find projects about chemotherapy, but we will not find project can detect if the chemotherapy is effective with patient or not.

FIGURE 1

In this project we will try to build a model that help to know the efficiency of drug.

The main application in our project is a website response input data from user as computed tomography (CT) images and send result to user as PCR or

NONPCR Which shows whether the patient has been completely cured or not

1.1 Motivation

The spread of breast cancer and its impact on a large number of women around the world is the main driver behind our project.

This project will provide assistance in the accurate diagnosis and selection of the appropriate treatment for each patient. These happens by noting the development of the patient's medical situation in the different stages of treatment and the effect of treatment on each stage.

In 2020, there were 2.3 million women diagnosed with breast cancer and 685 000 deaths globally. As of the end of 2020, there were 7.8 million women alive who were diagnosed with breast cancer in the past 5 years, making it the world's most prevalent cancer. There are more lost disability-adjusted life years (DALYs) by women to breast cancer globally than any other type of cancer. Breast cancer occurs in every country of the world in women at any age after puberty but with increasing rates in later life.

In 2022, there will be an estimated 287,850 new cases of invasive breast cancer diagnosed in women at the U.S; 2,710 cases diagnosed in men, and an additional 51,400 cases of ductal carcinoma in situ (DCIS) diagnosis in women.

The motivation of choosing QIN-Breast data is the mission of the QIN which is improving the role of quantitative imaging for clinical decision making in oncology by developing and validating data acquisition, analysis methods, and tools to tailor treatment for individual patients and predict or monitor the response to drug or radiation therapy.

Data

To build this classification model we need data that was documented the sessions of chemotherapy, so we use QIN-Breast data from The Cancer Imaging Archive (TCIA).
This data contains longitudinal PET/CT and quantitative MR images collected for the purpose of studying treatment assessment in breast cancer in the neoadjuvant setting, we filtered it and just took computed tomography (CT) images.

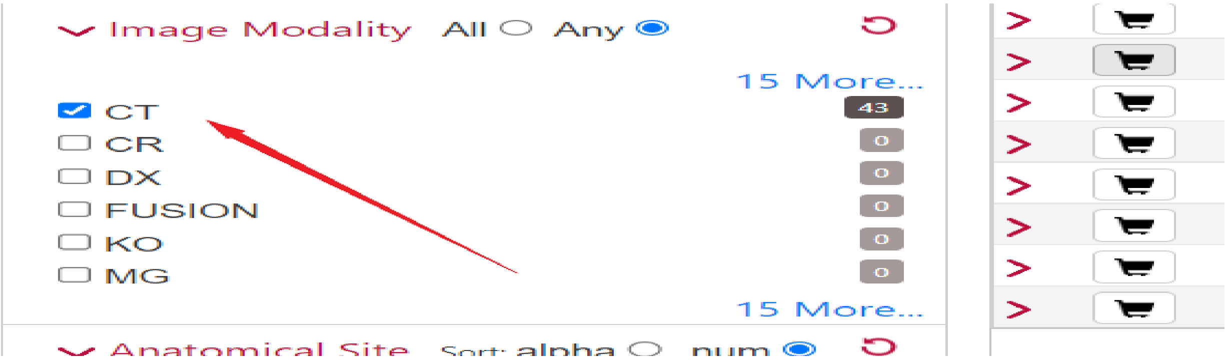


FIGURE 2

Images were acquired at three time points:

(t1) prior to the start of treatment, (t2) after the first cycle of treatment and (t3) either after the second cycle of treatment or at the completion of all treatments (prior to surgery).

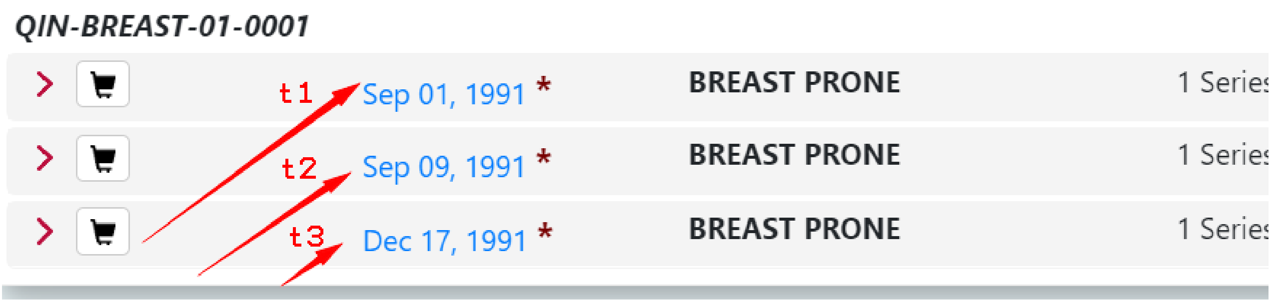


FIGURE 3

The PET/CT images were acquired with a support device built in-house to allow the patient to be in the prone position to facilitate registration with the MRI data. The value of this collection is to provide clinical imaging data for the development and evaluation of quantitative imaging methods for treatment assessment early in the course of therapy for breast cancer. Data is provided by Vanderbilt University, PI Dr. Thomas E. Yankeelov .

Standard-of-care supine images and research prone images were acquired at times t1 and t3, while only the prone images were acquired at t2.

In this data the number of participants is 68 patients, the number of studies is 214, the number of series is 530 and the number of images is 100835.

We filtered data by taking CT images only and choosing patients who completed the three stages of treatment.

After filter process the data contain the number of participants have three stages are 29 patients, and the number of images is 7221.

and the number of participants have two stages are 2 patients, and the number of images is 332.

When the best manual pictures selected manualy, 10 pictures were selected from each stage in the patients who went through the three treatment stages, as the 3 stages were one of them before treatment and two stages after taking it, while 15 pictures were taken from each stage in the patients who went through only two stages, as the two stages a stage before treatment and a stage after it.

We do some process on pictures in this data:

- We do crop on picture to identify the tumor more
- We found some pictures (PCR) and other (nonPCR)

What is the PCR: - Pathological complete response (PCR) is defined as the absence of residual invasive cancer on hematoxylin and eosin evaluation of the complete resected breast specimen and. all sampled regional lymph nodes following completion of neoadjuvant systemic therapy.

What is the PCR: -

We found that the PCR pictures indicate the patient's recovery from the disease, and the NONPCR pictures indicate that the patient did not recover from the disease.

FIGURE4

1.2 Problem definition

The main problem facing us in breast cancer treatment is our ignorance with the effect of chemotherapy in patient before starting it. By that we miss an opportunity for the patient to find alternative solutions before the disease spreads and his situation deteriorates.
And, if the tumor reappears, we will need an explanation of how the treatment affected the first time around, as well as the probability of it succeeding again.

1.3 Project Objective

The project use a CNN model which the data was filtered to CT images only and CNN model is perfect with data like this.

1.4 Gantt chart of project time plan

first 4 months

1. Search about project idea
2. Compare between different datasets
3. Get the data
4. Filter the data
5. Preprocessing
6. Write the documentation
7. Prepare the presentation

last 4 months

1. CNN model
2. Website
3. Documentation
4. Presentation

first 4 months

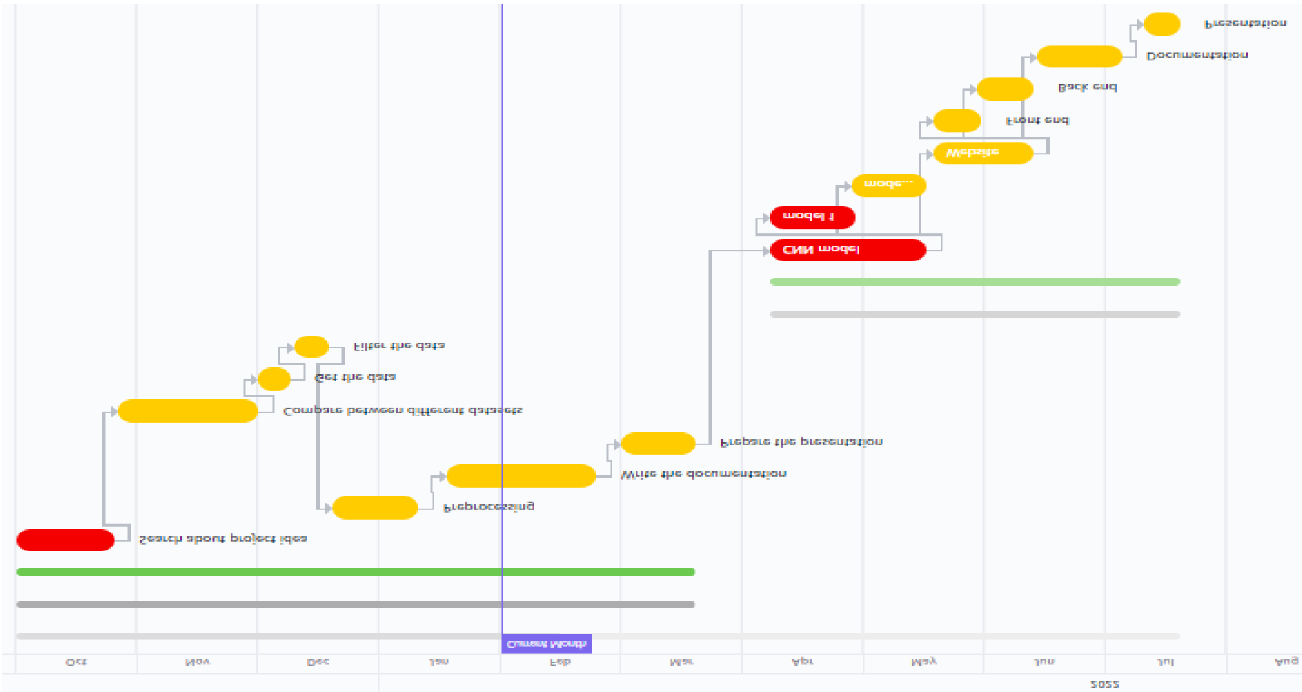


FIGURE 5

last 4 months

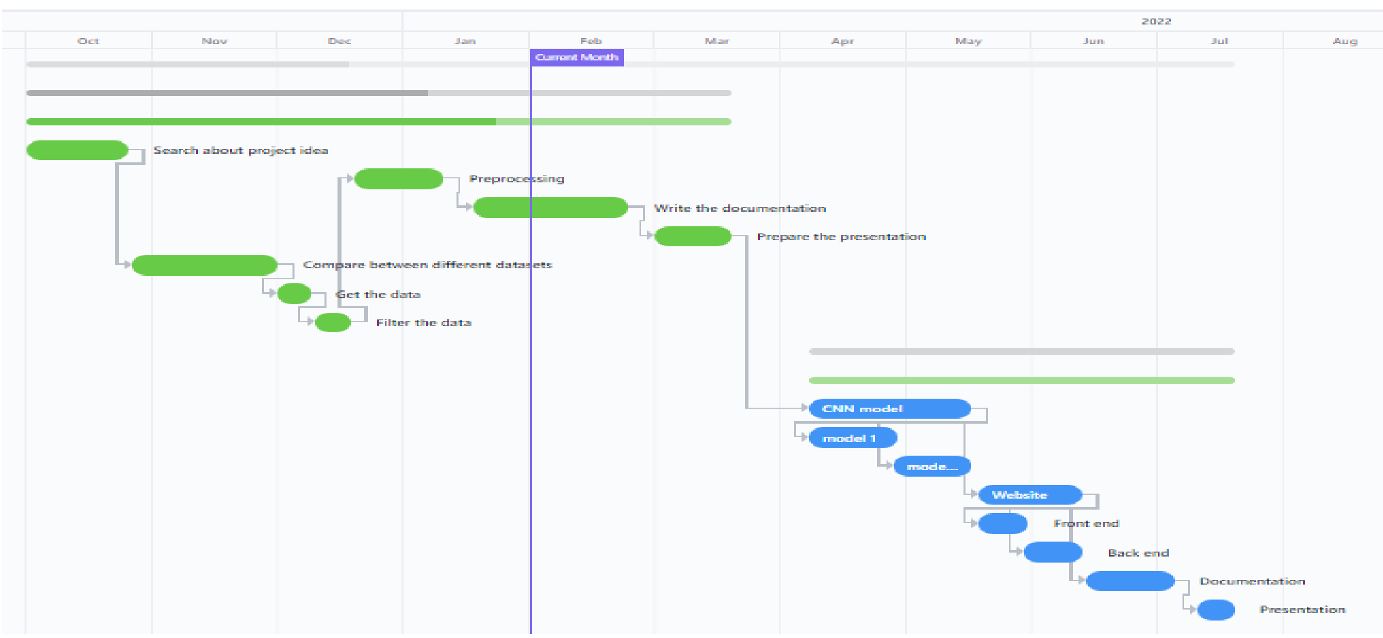


FIGURE 6

1.5 Project development methodology

Development methodology in our project separates into two main components:

- First one is front end which represent the user interface in our website.

  We use HTML and CSS to implement the web application pages.

  When user visit our website first page appears is the Signup page, in this page user can choose if he wants to create a new account or he already has an account and want to log in.

  If user clicked on register, then he will move to the next page which takes the user information and make him create a unique email, password and user id. When user click on submit, the last page will appear which takes user id, a unique patient id which user create, other information and path of the folder that contains the patient images. All these data will insert into database by clicking on submit button. User can get the result of his patients any time by write patient and user id then click show result button, the result will appear as PCR or nonPCR and the accuracy of detection.

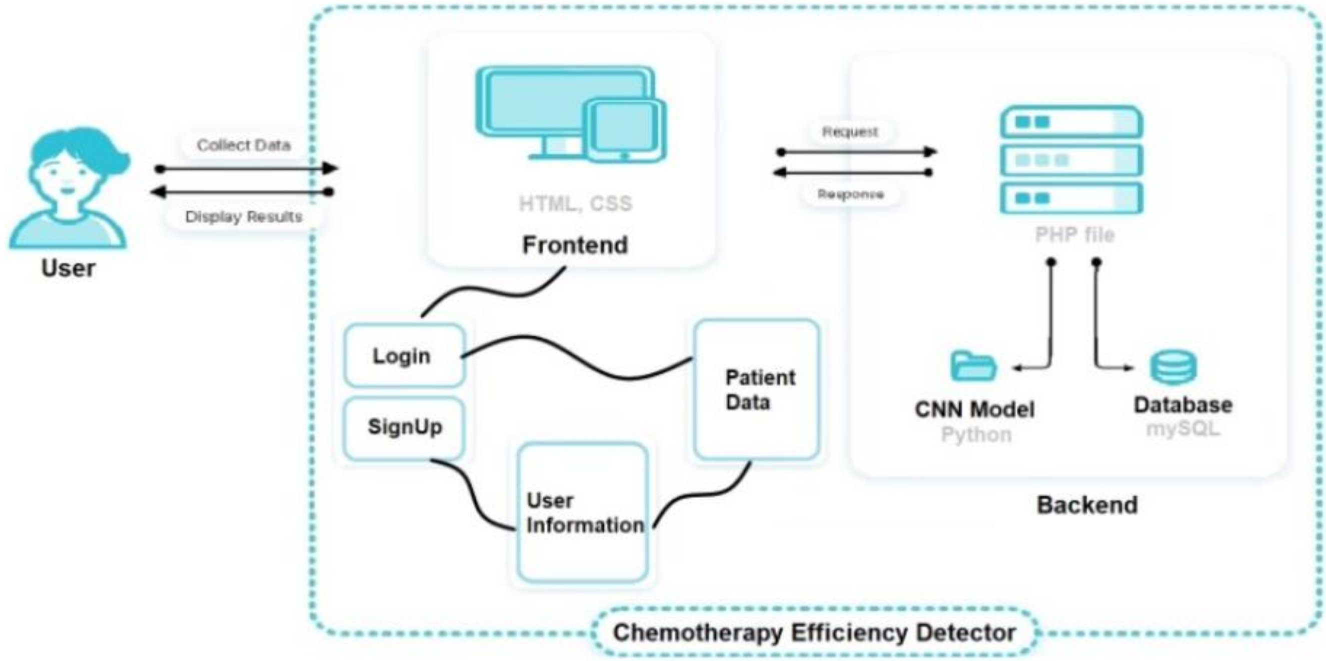
  But if user write his email and password and clicked on log in button, he will move to the last page directly.

- Second one is back end which represented in connecting all HTML files with database and CNN model.

  In user page all values that existed at input boxes will insert into user table at mySQL database by PHP. The same thing happened in the patient page.

  In PHP file of patient page, the path will send to python file which takes the path and call function predict to preprocessing the images before input to the CNN model. CNN model predicts the patient belonging to which class, then function predict handles the output and return it to PHP file.

  PHP file takes the output and insert it to the database.



1.6 The used tools in the project (SW and HW)

Software:

- Convolution Neural Network (CNN):

  In deep learning, a convolutional neural network (CNN/ConvNet) is a class of deep neural networks, most commonly applied to analyze visual imagery.

- Google Colaboratory.
- Python Programming Language.
- Python Libraries:
  - Numpy
  - Pandas
  - Pydicom
  - OpenCV
  - Matplotlib
  - Itertools
  - Sklearn
  - Keras
  - TensorFlow

    - Pickle
    - Imbalanced-learn
    - OS
    - SYS
- PHP Programming Language.
- HTML Standard Markup Language.
- CSS Style Sheet Language.
- SQL Programming Language.

Hardware:

Our intention is to build a Web Application, this means that our main focus is software, and we do not use any other hardware rather than a machine that can execute the program.

1.7. Report Organization

Chapter 1,2

We will be presenting the previous solutions and other related work, describe its methodology and criticize its disadvantages.
According to this analysis we figured out that we need to find a better solution. So, we started searching and studying different techniques to find the best
After months of studying and searching we managed to find some suitable data which gave us a good chance to start implementing our ideas.
This data contains longitudinal PET/CT and quantitative MR images collected for the purpose of studying treatment assessment in breast cancer in the neoadjuvant setting, we filtered it and just took computed tomography (CT) images.

Chapter 3

We determine the suitable functional requirements and non-functional requirements to our project.
Then we will illustrate the use case diagram to our project according to functional requirements of our project.

Chapter 4

We will present the project by designing System Component Diagram, System Class Diagrams, Sequence Diagrams and System GUI Design

Chapter 5

After take best sample from dataset where we did it twice, first sample has clearest 10 images from each stage in patient folder, second sample has clearest 5 images from each stage in patient folder.

We then preprocessing datasets and started to implement different models with different approaches, we compared and analyzed the results, nearly we tried a hundred of deep learning models.

A lot of problems faced us because the amount of our dataset. The biggest problem was overfitting, so we used augmented data and it gave us acceptable results.

One of the problems that the minority class is too small compared with majority class, we used balanced weights and it gave us a good result.

But the best result we got when we use SMOTE to equals the classes then we increase one of them by add a random sample to it, then we used balanced weights, that model gave us 83.26% accuracy.

We built a web application depend on this model as a back end. We built it using HTML, CSS and PHP. This web application has a database to save all information of user and patients result.

And the final results were very satisfiable to us.

Chapter 2: Related work

Project Name	Similarity	Difference
Prediction of pathological complete response to neoadjuvant chemotherapy in breast cancer using a deep learning (**DL**) method (PMID: 31944571)	This study established a machine learning technique to predict the impact of neoadjuvant therapy in tumor.	This project used MRI images data. Used deep learning (DL) method in classification. Data not divided into different stages of treatment.
Multi-input deep learning architecture for predicting breast tumor response to chemotherapy using quantitative MR images (PMID: 32556920)	This study established a machine learning technique to predict the impact of neoadjuvant therapy in tumor. Used CNN method in classification.	Used DCE-MR images data. Data not divided into different stages of treatment.

Table 1

Chapter 3: System Analysis

3.1. Project specification

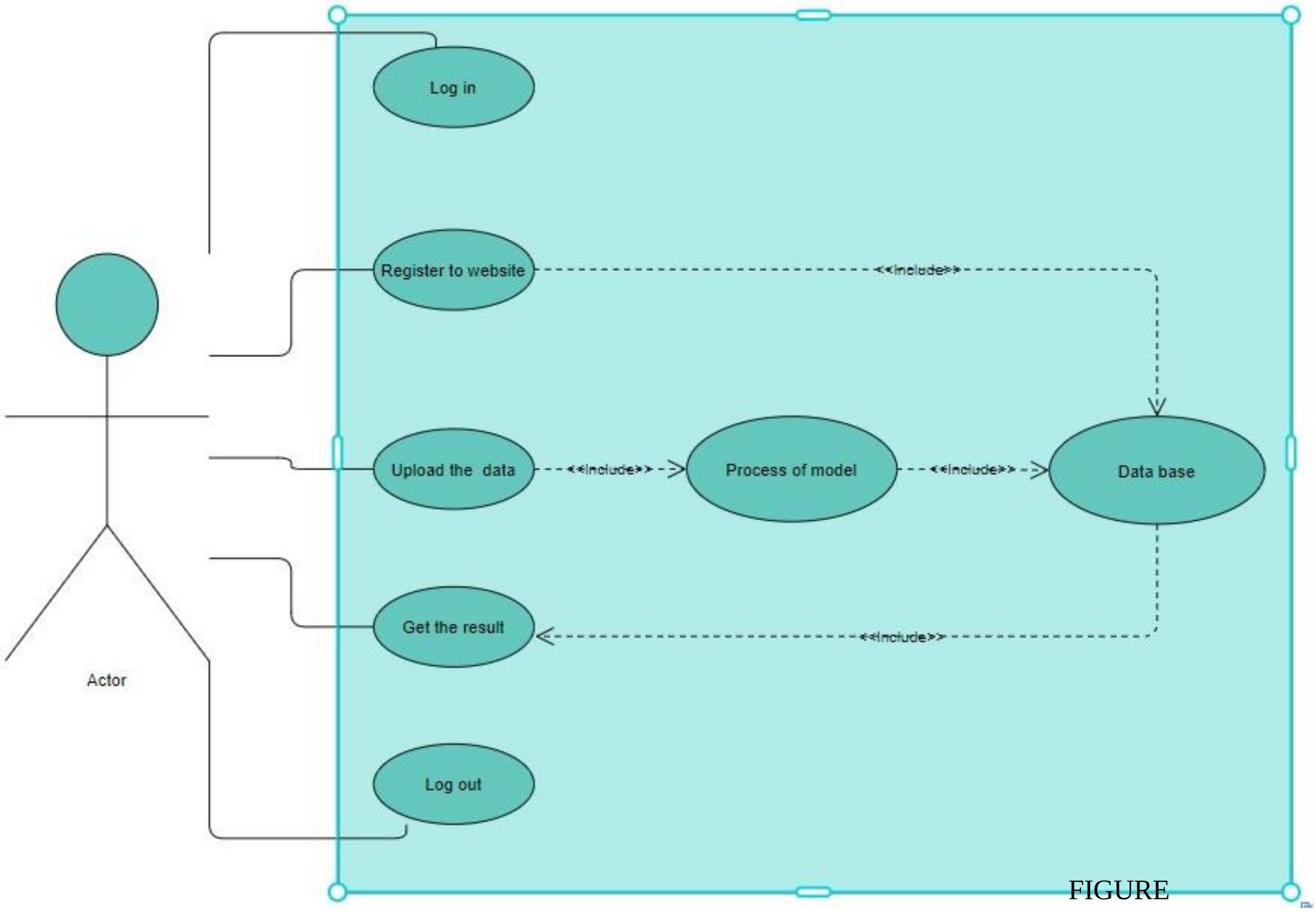
3.1.1. Functional requirement

- The website response input data from user as computed tomography (CT) images and send result to user as 0 or 1 which expresses if the chemotherapy is effective on patient or not.
- User can obtain the result of a previous patient directly by entering the ID of the patient only.

3.1.2. Non-functional requirement

- **usability**
  The site is easy to use and there is nothing ambiguous about it so that the user can register and enter the website and use it easily, and all labels are obvious and user can understand them easily.
- **Performance**
  Response time is as quick as possible.
- **Availability**
  The Application is available for anyone to use.
- **Scalability**
  The capability of the app handles a growing number of patients.

3.2. Use Case Diagram



FIGURE

Database

Database built to save user and patient information and to make user enable to get his patients result.

This database created by MySQL database, it contains to main tables:

- User table has 9 columns:

  FirstName, MiddleName, LastName, age, Gender and PhoneNumber columns are a metadata of user. Email, Password and UserID created by use to register and be able to use our web application. UserID column is a primary key of user table and it has a unique values.

- Patient table has 7 columns:

  PatientAge, StagesOfTreatment, StageData columns are the data of patient should user (doctor) enter. PatientID column is a primary key of user table and it has a unique values, this values should created by user. UserID is a foreign key from user table to make a relation between the two tables. Result and Accuracy columns have a values that returns from model.

Chapter 4: System Design

4.1. System Component Diagram

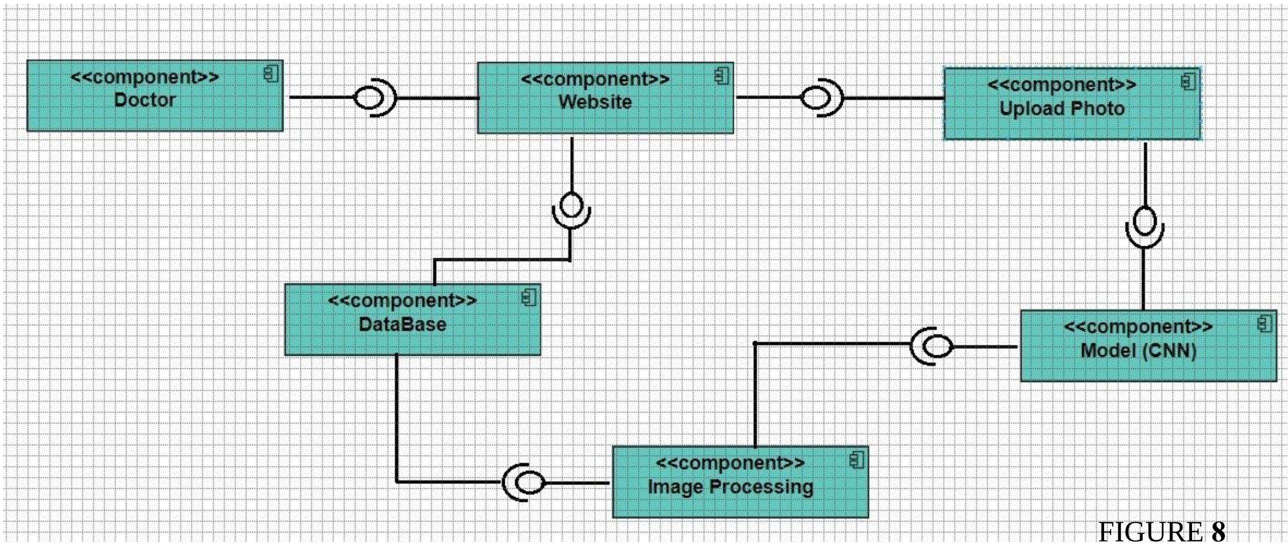


FIGURE 8

4.2. System Class Diagrams

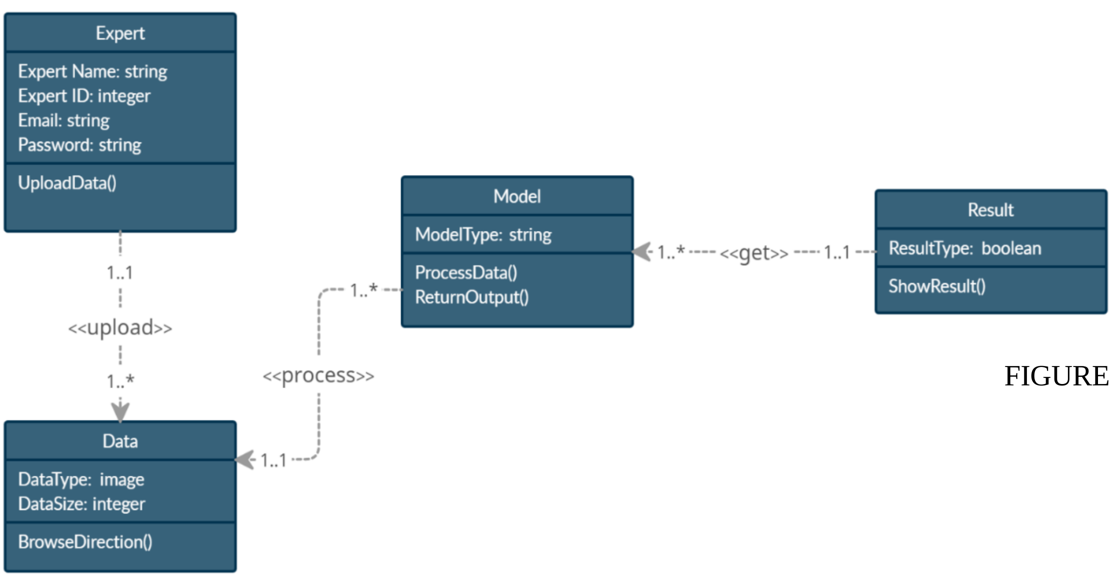


FIGURE 9

4.3. Sequence Diagrams

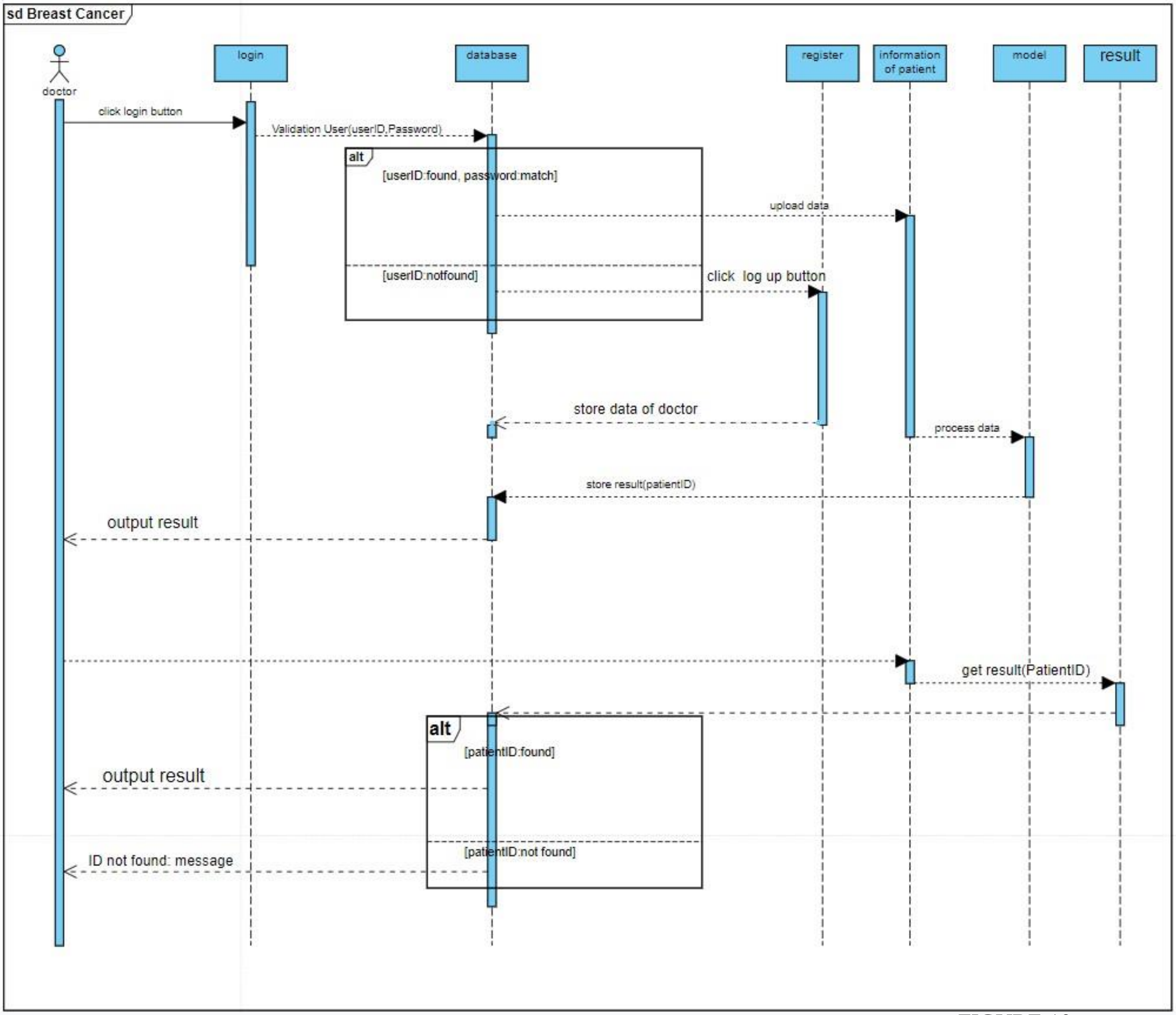


FIGURE 10

The above diagram illustrates that the doctor must log in to the website first, after the site validation the UserID and password doctor has two choices,

1. upload photos for a new patient then get the result.
2. get the result of old patient.

4.4. Project ERD

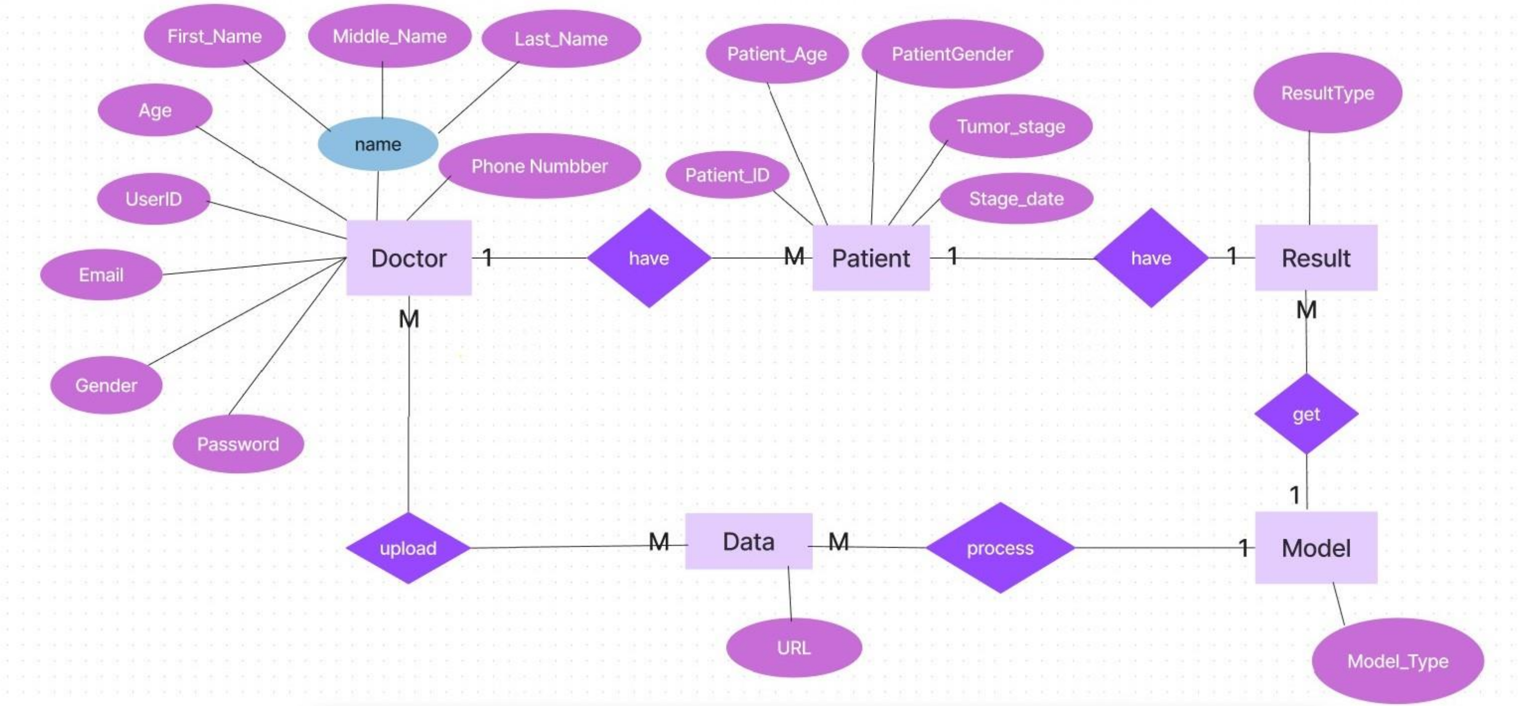


FIGURE 11

4.5. System GUI Design

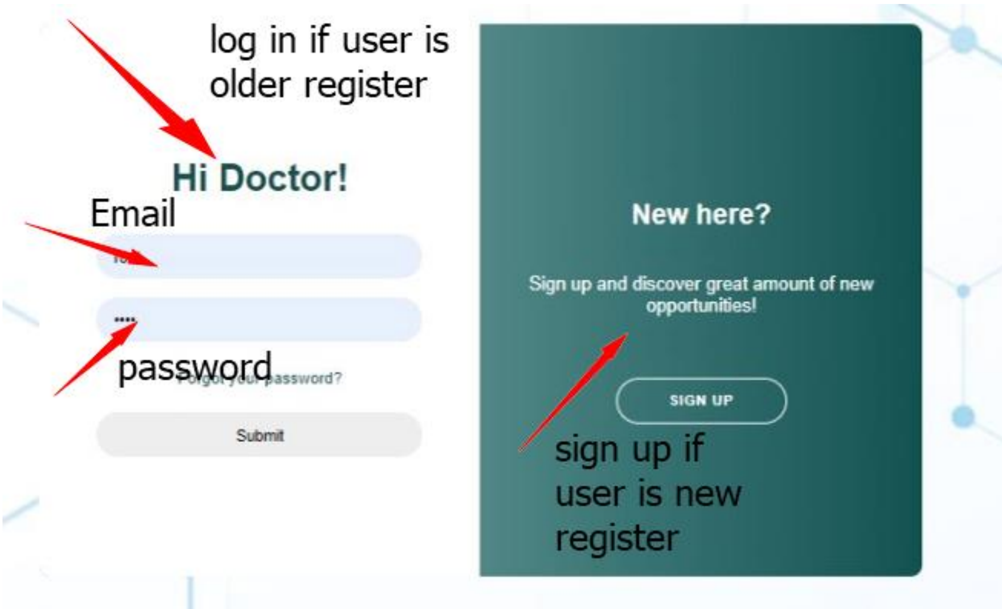
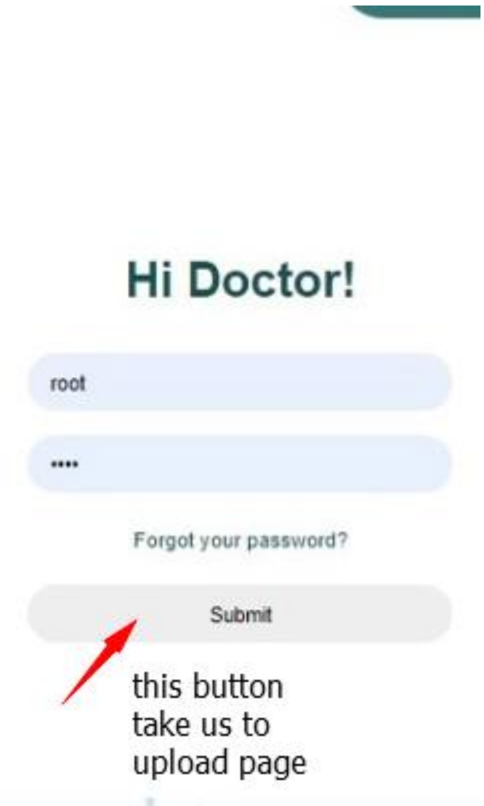


FIGURE 12


Hi Doctor!
root
Forgot your password?
Submit
this button
take us to
upload page

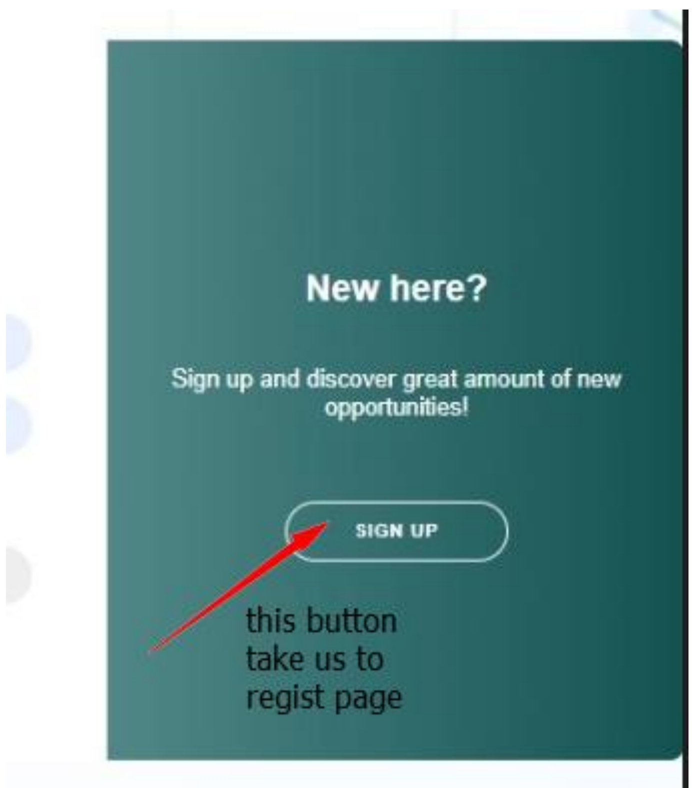

New here?
Sign up and discover great amount of new opportunities!
SIGN UP
this button
take us to
regist page

FIGURE 13

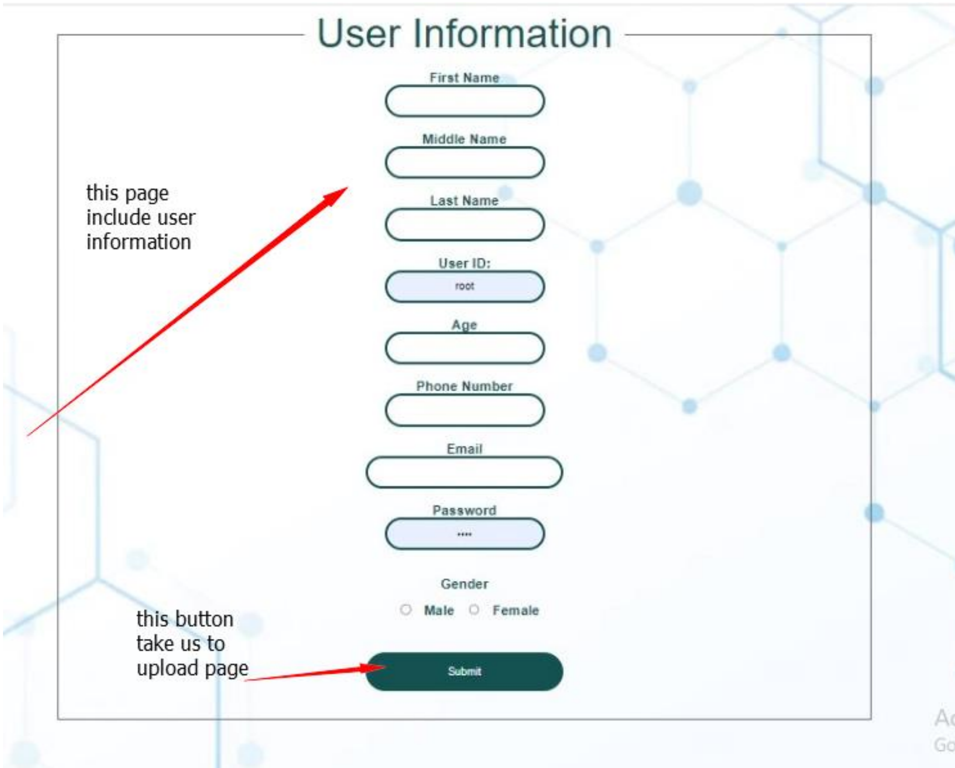


FIGURE 14

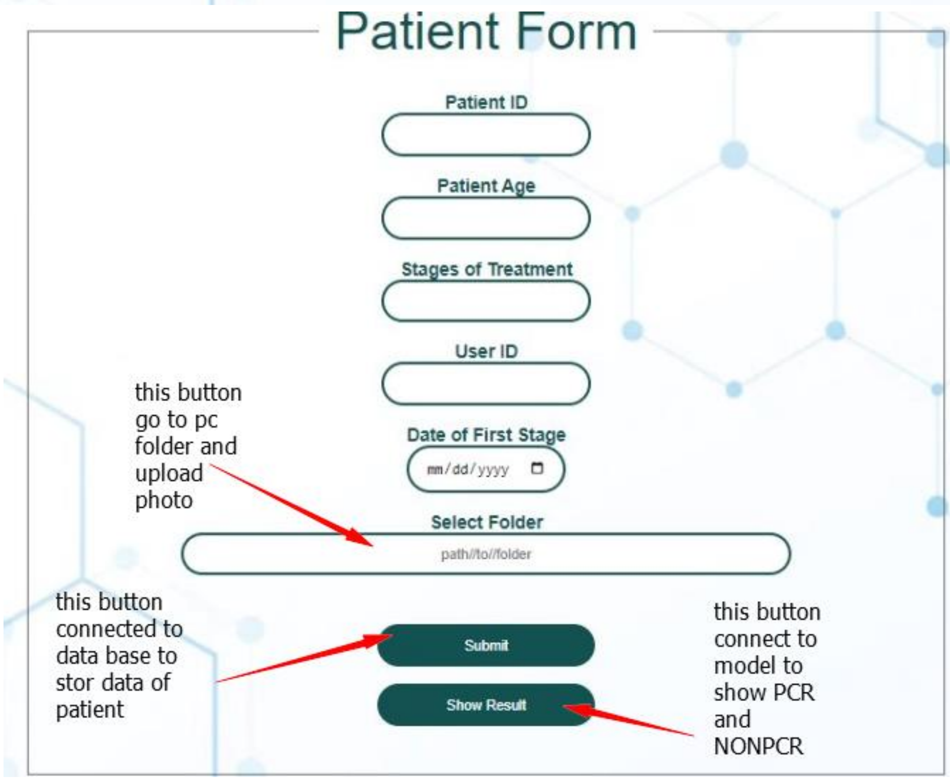


FIGURE 15

Chapter 5: Implementation and Testing

5.1. Overview

Dataset Overview

31 patients' CT images in different stages before and after chemotherapy are used in our project.
To ensure that we get the best accuracy, we took a sample of the clearest CT images which appears the tumor, we did it manually to:

- Dataset has 930 images, taking 30 images for each patient (10 from each stage).
- Dataset has 465images, taking 15 images for each patient (5 from each stage).

In all models' trials we will use both datasets and compare between accuracies.

5.2. Models Implementation and Testing

We classified CT images into two types, (i) positive CT (PCR) images where this image belongs to a fully recovered patient, and (ii) negative CT (nonPCR) images where this image belongs to a patient who still has a tumor. To enable image-based prediction, we adopted the next methods.

5.2.1. Traditional CNN Model

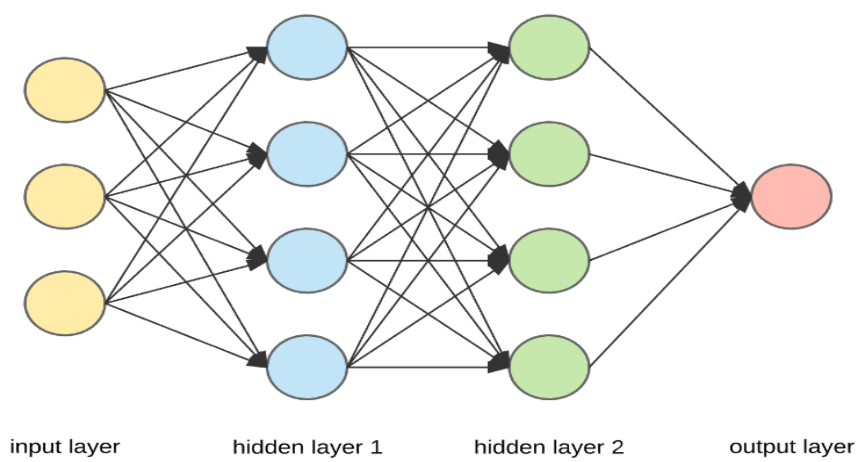


FIGURE 16

5.2.1.1. With 30 images for each patient

Trial 1

Trial 1 Architecture
Conv (64, 3*3, input_shape (200*200*1) ...) **MaxPooling (2*2)**
Conv (64, 3*3, …) **MaxPooling (2*2)**
Flatten () **Dense (64, activation='relu')**
Dense (1, activation='segmoid')

Number of Epochs	**10**
Train Data Size	**70%**
Test Data Size	**30%**
Training Accuracy	**74%**
Testing Accuracy	**74%**
Trial Evaluation	**74%**

Trial Details

Trail description:
A deep learning framework of 2-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 2 sets of dual convolutional and pooling layers (2 × 2), 2 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function categorical_crossentropy.
Note: This model is not good at training and testing.

Trial 2

Trial 2 Architecture
Conv (64, 3*3, input_shape (200*200*1) ...) **MaxPooling (2*2)**
Conv (64, 3*3, …) **MaxPooling (2*2)**
Flatten () **Dense (64, activation='relu')**
Dense (1, activation='segmoid')

Number of Epochs	**10**
Train Data Size	**60%**
Test Data Size	**40%**
Training Accuracy	**74%**
Testing Accuracy	**75%**
Trial Evaluation	**75%**

Trial Details

Trail description:
deep learning framework of 2-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 2 sets of dual convolutional and pooling layers (2 × 2), 2 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function categorical_crossentropy.
Note: This model is not good at training and testing.

Trial 3

Trial 3 Architecture
Conv (150, 3*3, input_shape (200*200*1) ...) Conv (120, 3*3, …) MaxPooling (2*2)
Conv (100, 3*3, …) Conv (64, 3*3, …) MaxPooling (2*2)
Conv (64, 3*3, …) MaxPooling (2*2)
Flatten () Dense (64, activation='relu')
Dense (32, activation='relu') Dropout (0.8)
Dense (1, activation=’segmoid’)

Number of Epochs	10
Train Data Size	60%
Test Data Size	40%
Training Accuracy	98%
Testing Accuracy	99%
Trial Evaluation	75%

Trial Details

Trail description:
A deep learning framework of 5-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 3 sets of dual convolutional and pooling layers (2 × 2), 3 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function binary_crossentropy.
Note: This model overfitting.

5.2.1.2. With 15 images for each patient

Trial 1

Trial 1 Architecture
Conv (64, 3*3, input_shape (200*200*1) ...) MaxPooling (2*2)
Conv (64, 3*3, …) MaxPooling (2*2)
Flatten () Dense (64, activation='relu')
Dropout (0.8) Dense (1, activation=’segmoid’)

Number of Epochs	10
Train Data Size	60%
Test Data Size	40%
Training Accuracy	99%
Testing Accuracy	94%
Trial Evaluation	74%

Trial Details

Table 2

Trail description:
A deep learning framework of 2-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 2 sets of dual convolutional and pooling layers (2 × 2), 2 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function binary_crossentropy.
Note: This model overfitting.

Trial 2

Trial 2 Architecture
Conv (120, 3*3, input_shape (200*200*1) ...) **Conv (100, 3*3, …)** **MaxPooling (2*2)**
Conv (100, 3*3, …) **Conv (64, 3*3, …)** **MaxPooling (2*2)**
Flatten () **Dense (64, activation='relu')**
Dropout (0.8) **Dense (1, activation='segmoid')**

Table 3

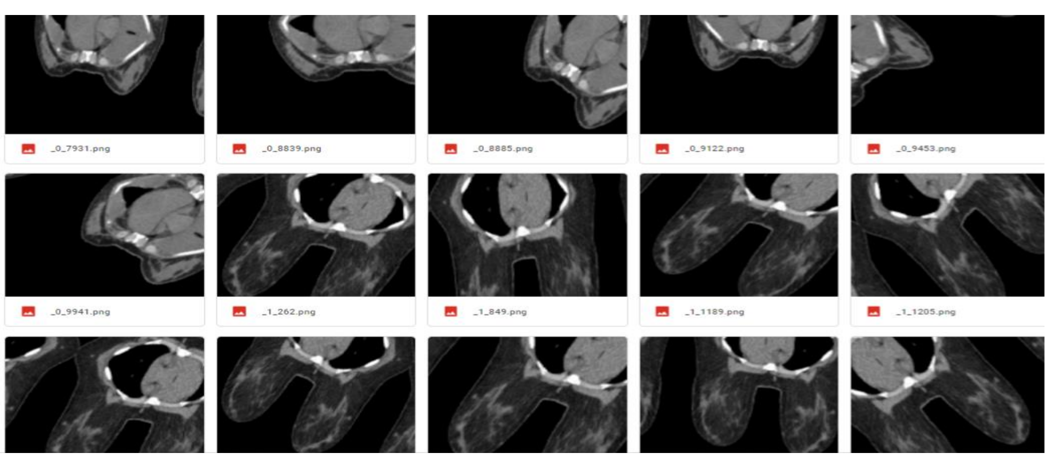
Number of Epochs	**10**
Train Data Size	**70%**
Test Data Size	**30%**
Training Accuracy	**98%**
Testing Accuracy	**98%**
Trial Evaluation	**75%**

Trial Details

Trail description:
A deep learning framework of 4-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 2 sets of dual convolutional and pooling layers (2 × 2), 2 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function binary_crossentropy.
Note: This model overfitting.

5.2.2. CNN Model with data augmentation

Data augmentation is a popular technique that is used to increase the generalizability of an overfitted data model.
It is the set of techniques that is used to increase the amount of data by adding modified copies of already existing data. Sometimes, it creates newly synthetic data from the existing data. FIGURE 17



5.2.2.1. With 30 images for each patient

Trial 1

Trial 1 Architecture
Conv (150, 3*3, input_shape (200*200*1) ...) Conv (120, 3*3, …) MaxPooling (2*2)
Conv (100, 3*3, …) Conv (64, 3*3, …) MaxPooling (2*2)
Conv (32, 3*3, …) MaxPooling (2*2)
Flatten () Dense (64, activation='relu')
Dense (32, activation='relu')
Dense (1, activation=’segmoid’)

Number of Epochs	20
Train Data Size	70%
Test Data Size	30%
Training Accuracy	74%
Testing Accuracy	74%

Trial Details

Table 4

Trail description:

A deep learning framework of 5-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 3 sets of dual convolutional and pooling layers (2 × 2), 3 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.

And we use loss function binary_crossentropy, ImageDataGenerator with: rotation_range = 45, width_shift_range = 0.2, height_shift_range = 0.2, shear_range = 0.2, zoom_range = 0.2, horizontal_flip = True, fill_mode = 'reflect' and batch_size = 16.

Note: This model is not good at training and testing.

Trial 2

Trial 2 Architecture
Conv (150, 3*3, input_shape (200*200*1) ...) **Conv (120, 3*3, …)** **MaxPooling (2*2)**
Conv (100, 3*3, …) **Conv (64, 3*3, …)** **MaxPooling (2*2)**
Conv (32, 3*3, …) **MaxPooling (2*2)**
Flatten () **Dense (64, activation='relu')**
Dense (32, activation='relu')
Dense (1, activation='segmoid')

Number of Epochs	**20**
Train Data Size	**60%**
Test Data Size	**40%**
Training Accuracy	**74%**
Testing Accuracy	**75%**

Trial Details

Table 5

Trail description:
A deep learning framework of 5-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 3 sets of dual convolutional and pooling layers (2 × 2), 3 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function binary_crossentropy, ImageDataGenerator with: rotation_range = 45, width_shift_range = 0.2, height_shift_range = 0.2, shear_range = 0.2, zoom_range = 0.2, horizontal_flip = True, fill_mode = 'reflect' and batch_size = 16.
Note: This model is not good at training and testing.

Trial 3

Trial 3 Architecture
Conv (120, 3*3, input_shape (200*200*1) ...) **Conv (100, 3*3, …)** **MaxPooling (3*3)**
Conv (64, 3*3, …) **Conv (32, 3*3, …)** **MaxPooling (3*3)**
Flatten () **Dense (64, activation='relu')**
Dense (32, activation='relu')
Dense (1, activation='segmoid')

Number of Epochs	**20**
Train Data Size	**80%**
Test Data Size	**20%**
Training Accuracy	**75%**
Testing Accuracy	**71%**

Trial Details

Table 6

Trail description:

A deep learning framework of 4-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 2 sets of dual convolutional and pooling layers (3 × 3), 3 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function binary_crossentropy, ImageDataGenerator with: rotation_range = 45, width_shift_range = 0.2, height_shift_range = 0.2, shear_range = 0.2, zoom_range = 0.2, horizontal_flip = True, fill_mode = 'constant' and batch_size = 16.
Note: This model is not good at training and testing.

Trial 4

Trial 4 Architecture
Conv (120, 3*3, input_shape (200*200*1) ...) Conv (100, 3*3, …) MaxPooling (3*3)
Conv (64, 3*3, …) Conv (32, 3*3, …) MaxPooling (3*3)
Flatten () Dense (64, activation='relu')
Dense (32, activation='relu')
Dense (1, activation=’segmoid’)

Number of Epochs	20
Train Data Size	80%
Test Data Size	20%
Training Accuracy	95%
Testing Accuracy	91%

Trial Details

Table 7

Trail description:

A deep learning framework of 4-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 2 sets of dual convolutional and pooling layers (3 × 3), 3 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function binary_crossentropy, ImageDataGenerator with: rotation_range = 40, width_shift_range = 0.02, height_shift_range = 0.02, shear_range = 0.02, zoom_range = 0.02, horizontal_flip = True, fill_mode = 'reflect' and batch_size = 10.
Note: This model overfitting.

Trial 5

Trial 5 Architecture
Conv (200, 3*3, input_shape (200*200*1) ...) Conv (150, 3*3, …) MaxPooling (3*3)
Conv (120, 3*3, …) Conv (100, 3*3, …) MaxPooling (3*3)
Conv (64, 3*3, …) Conv (32, 3*3, …) MaxPooling (3*3)
Flatten () Dense (100, activation='relu')
Dense (64, activation='relu')
Dense (32, activation='relu')
Dense (1, activation='segmoid')

Number of Epochs	**20**
Train Data Size	**80%**
Test Data Size	**20%**
Training Accuracy	**80%**
Testing Accuracy	**76%**

Trial Details

Table 8

Trail description:
A deep learning framework of 6-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 3 sets of dual convolutional and pooling layers (3 × 3), 4 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function binary_crossentropy, ImageDataGenerator with: rotation_range = 40, width_shift_range = 0.08, height_shift_range = 0.08, shear_range = 0.08, zoom_range = 0.08, horizontal_flip = True, fill_mode = 'reflect' and batch_size = 10.
Note: This model is good in training but not enough in testing.

5.2.2.2. With 15 images for each patient

Trial 1

Trial 1 Architecture
Conv (120, 3*3, input_shape (200*200*1) ...) **Conv (100, 3*3, …)** **MaxPooling (3*3)**
Conv (64, 3*3, …) **Conv (32, 3*3, …)** **MaxPooling (3*3)**
Flatten () **Dense (64, activation='relu')**
Dense (32, activation='relu')
Dense (1, activation='segmoid')

Number of Epochs	**20**
Train Data Size	**80%**
Test Data Size	**20%**
Training Accuracy	**75%**
Testing Accuracy	**74%**

Trial Details

Table 9

Trail description:
A deep learning framework of 4-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 2 sets of dual convolutional and pooling layers (3 × 3), 3 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function binary_crossentropy, ImageDataGenerator with: rotation_range = 45, width_shift_range = 0.2, height_shift_range = 0.2, shear_range = 0.2, zoom_range = 0.2, horizontal_flip = True, fill_mode = 'constant' and batch_size = 16.
Note: This model is not good at training and testing.

Trial 2

Trial 2 Architecture
Conv (200, 3*3, input_shape (200*200*1) ...) **Conv (150, 3*3, …)** **MaxPooling (3*3)**
Conv (120, 3*3, …) **Conv (100, 3*3, …)** **MaxPooling (3*3)**
Conv (64, 3*3, …) **Conv (32, 3*3, …)** **MaxPooling (3*3)**
Flatten () **Dense (100, activation='relu')**
Dense (64, activation='relu')
Dense (32, activation='relu')
Dense (1, activation='segmoid')

Number of Epochs	**20**
Train Data Size	**70%**
Test Data Size	**30%**
Training Accuracy	**76%**
Testing Accuracy	**81%**

Trail description:
A deep learning framework of 6-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 3 sets of dual convolutional and pooling layers (3 × 3), 4 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function binary_crossentropy,
ImageDataGenerator with:
rotation_range = 40, width_shift_range = 0.1, height_shift_range = 0.1, shear_range = 0.1, zoom_range = 0.1, horizontal_flip = True, fill_mode = 'constant' and batch_size = 16.

Table 10

Note: Not logic testing accuracy more than training.

Trial 3

Trial 3 Architecture
Conv (120, 3*3, input_shape (200*200*1) ...) Conv (100, 3*3, …) MaxPooling (3*3)
Conv (64, 3*3, …) Conv (32, 3*3, …) MaxPooling (3*3)
Flatten () Dense (64, activation='relu')
Dense (32, activation='relu')
Dense (1, activation=’segmoid’)

Number of Epochs	20
Train Data Size	60%
Test Data Size	40%
Training Accuracy	78%
Testing Accuracy	76%

Trial Details

Table 11

Trail description:

A deep learning framework of 4-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 2 sets of dual convolutional and pooling layers (3 × 3), 3 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function binary_crossentropy, ImageDataGenerator with: rotation_range = 40, width_shift_range = 0.1, height_shift_range = 0.1, shear_range = 0.1, zoom_range = 0.1, horizontal_flip = True, fill_mode = 'constant' and batch_size = 16.
Note: This model better than previous models but not enough.

Trial 4

Trial 4 Architecture
Conv (16, 3*3, input_shape (200*200*1) ...) Conv (16, 3*3, …) MaxPooling (2*2)
Conv (32, 3*3, …) Conv (32, 3*3, …) MaxPooling (2*2)
Conv (64, 3*3, …) Conv (64, 3*3, …) MaxPooling (2*2)
Flatten () Dense (64, activation='relu')
Dense (32, activation='relu')
Dense (1, activation='segmoid')

Table 12

Number of Epochs	**20**
Train Data Size	**60%**
Test Data Size	**40%**
Training Accuracy	**80%**
Testing Accuracy	**80%**

Trial Details

Trail description:
A deep learning framework of 6-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 3 sets of dual convolutional and pooling layers (2 × 2), 3 fully connected (dense) layers, Containing stride (1,1), Kernel_initializer(uniform), Padding (same), Activation function Relu/Segmoid and one output layer.
And we use loss function binary_crossentropy, ImageDataGenerator with: rotation_range = 40, width_shift_range = 0.08, height_shift_range = 0.08, shear_range = 0.08, zoom_range = 0.08, horizontal_flip = True, fill_mode = 'constant' and batch_size = 16.
Note: This model is good but could be better.

Trial 5

Trial 7 Architecture
Conv (16, 3*3, input_shape (200*200*1) ...) Conv (16, 3*3, …) MaxPooling (2*2)
Conv (32, 3*3, …) Conv (32, 3*3, …) MaxPooling (2*2)
Conv (64, 3*3, …) Conv (64, 3*3, …) MaxPooling (2*2)
Flatten () Dense (64, activation='relu')
Dense (32, activation='relu')
Dense (1, activation=’segmoid’)

Number of Epochs	20
Train Data Size	60%
Test Data Size	40%
Training Accuracy	83%
Testing Accuracy	79%

Trial Details

Table 13

Trail description:
A deep learning framework of 6-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 3 sets of dual convolutional and pooling layers (2 × 2), 3 fully connected (dense) layers, Containing stride (1,1), Kernel_initializer(uniform), Padding (same), Activation function Relu/Segmoid and one output layer.
And we use loss function binary_crossentropy, ImageDataGenerator with: rotation_range = 40, width_shift_range = 0.1, height_shift_range = 0.1, shear_range = 0.1, zoom_range = 0.1, horizontal_flip = True, fill_mode = 'constant' and batch_size = 16.
Note: This model better than last one in training accuracy but testing accuracy less than last model.

5.2.3. CNN Model with Cross-Validation

Cross-validation is a technique for evaluating a machine learning model and testing its performance. It helps to compare and select an appropriate model for the specific predictive modeling problem.
it tends to have a lower bias than other methods used to count the model's efficiency scores. It makes cross-validation a powerful tool for selecting the best model for the specific task.

Algorithm of playing cross validation:
1- Divide the dataset into two parts: one for training, other for testing.
2- Train the model on the training set.
3- Validate the model on the test set.
4- Repeat 1-3 steps a couple of times. This number depends on the CV method that you are using.

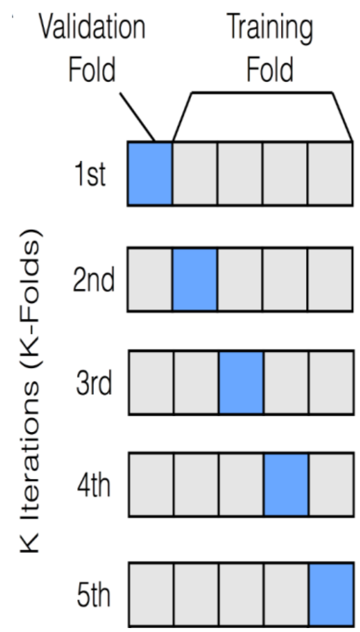


FIGURE 18

5.2.3. 1. With 30 images for each patient

Trial Architecture
Conv (200, 3*3, input_shape (200*200*1) ...) Conv (150, 3*3, …) MaxPooling (3*3)
Conv (120, 3*3, …) Conv (100, 3*3, …) MaxPooling (3*3)
Conv (64, 3*3, …) Conv (32, 3*3, …) MaxPooling (3*3)
Flatten () Dense (100, activation='relu')
Dense (64, activation='relu') Dropout (0.5)
Dense (32, activation='relu') Dropout (0.5)
Dense (1, activation='segmoid')

K	**5**
Number of Epochs	**10**
Training Accuracy	**97%, 97%, 99%, 100%, 90%**
Testing Accuracy	**94%, 97%, 99%, 100%, 90%**
Average Accuracy	**97%**
Trial Evaluation	**74%**

Trial Details

Trail description:
A deep learning framework of 6-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 3 sets of dual convolutional and pooling layers (3 × 3), 4 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function binary_crossentropy.

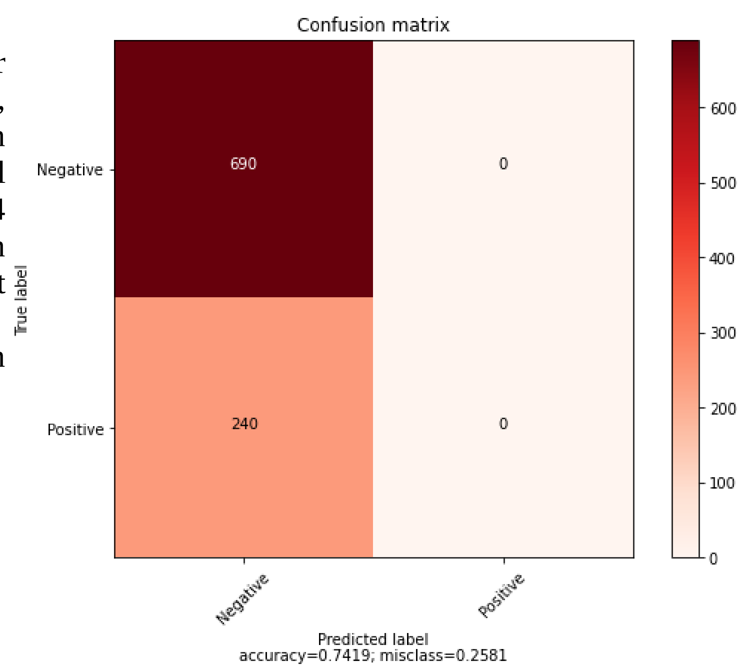


Table 14

Note: This model overfitting and not select any positive values.

5.2.3.2. With 15 images for each patient

Trial 1

Trial 1 Architecture
Conv (64, 3*3, input_shape (200*200*1) ...) **Conv (64, 3*3, …)** **MaxPooling (2*2)**
Conv (128, 3*3, …) **Conv (128, 3*3, …)** **MaxPooling (2*2)**
Conv (256, 3*3, …) **Conv (256, 3*3, …)** **Conv (256, 3*3, …)** **MaxPooling (2*2)**
Conv (512, 3*3, …) **Conv (512, 3*3, …)** **Conv (512, 3*3, …)** **MaxPooling (2*2)**
Conv (512, 3*3, …) **Conv (512, 3*3, …)** **Conv (512, 3*3, …)** **MaxPooling (2*2)** **GlobalMaxPooling2D ()**
Dense (1024, activation='relu')
Dense (1024, activation='relu') **Dropout (0.5)**
Dense (1, activation='segmoid')

K	**5**
Number of Epochs	**5**
Training Accuracy	**72%, 75%, 73%, 75%, 87%**
Testing Accuracy	**81%, 70%, 75%, 70%, 87%**
Average Accuracy	**77%**

Trail description:
A deep learning framework of 13-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 5 sets of dual convolutional and pooling layers (2 × 2), 3 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function binary_crossentropy.

Table 15

Note: This model not good in training and testing.

Trial 2

Trial 2 Architecture
Conv (64, 3*3, input_shape (200*200*1) ...) **Conv (64, 3*3, …)** **MaxPooling (2*2)**
Conv (128, 3*3, …) **Conv (128, 3*3, …)** **MaxPooling (2*2)**
Conv (256, 3*3, …) **Conv (256, 3*3, …)** **Conv (256, 3*3, …)** **MaxPooling (2*2)**
Conv (512, 3*3, …) **Conv (512, 3*3, …)** **Conv (512, 3*3, …)** **MaxPooling (2*2)**
Conv (512, 3*3, …) **Conv (512, 3*3, …)** **Conv (512, 3*3, …)** **MaxPooling (2*2)** **GlobalMaxPooling2D ()**
Dense (1024, activation='relu')
Dense (1024, activation='relu') **Dropout (0.5)**
Dense (1, activation='segmoid')

K	**5**
Number of Epochs	**10**
Training Accuracy	**72%, 75%, 74%, 75%, 80%**
Testing Accuracy	**81%, 70%, 75%, 71%, 76%**
Average Accuracy	**75%**

Trail description:
A deep learning framework of 13-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 5 sets of dual convolutional and pooling layers (2 × 2), 3 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function binary_crossentropy.

Table 16

Note: This model not good in training and testing.

Trial 3

Trial 3 Architecture
Conv (64, 3*3, input_shape (200*200*1) ...) Conv (64, 3*3, …) MaxPooling (3*3)
Conv (128, 3*3, …) Conv (128, 3*3, …) MaxPooling (3*3)
Conv (256, 3*3, …) Conv (256, 3*3, …) Conv (256, 3*3, …) MaxPooling (3*3)
Conv (512, 3*3, …) Conv (512, 3*3, …) Conv (512, 3*3, …) MaxPooling (3*3)
Conv (512, 3*3, …) Conv (512, 3*3, …) Conv (512, 3*3, …) MaxPooling (2*2) GlobalMaxPooling2D ()
Dense (1024, activation='relu')
Dense (1024, activation='relu') Dropout (0.5)
Dense (1, activation='segmoid')

K	5
Number of Epochs	10
Training Accuracy	72%, 84%, 94%, 97%, 98%
Testing Accuracy	81%, 78%, 90%, 91%, 95%
Average Accuracy	90%
Trial Evaluation	74%

Trial Details

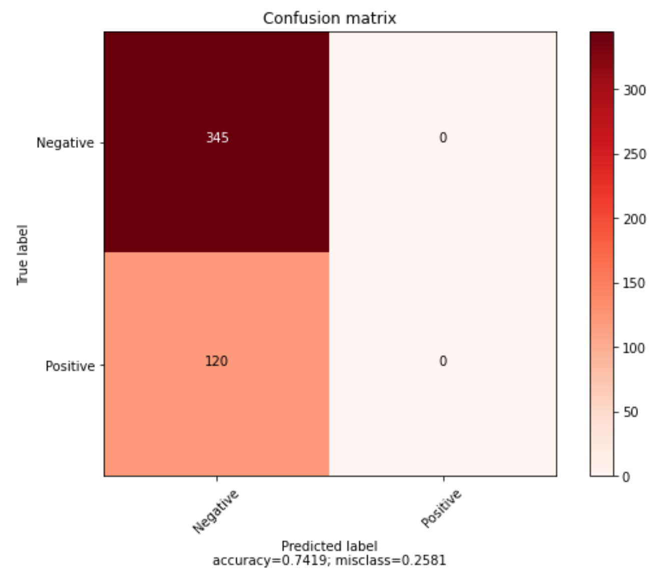


Table 17

Trail description: A deep learning framework of 13-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 5 sets of dual convolutional and pooling layers (3 × 3), 3 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.

And we use loss function binary_crossentropy.
Note: This model overfitting and not select any positive values.

5.2.4. CNN Model with categorical labels

After implementing a lot of models with different methods the biggest problem faced us is these models did not select from positive class.
Because of that we convert labels to categorical which means y (contain images labels) has positive and negative class values.
We implemented CNN models with data augmentation using the new y.

5.2.4.1. With 30 images for each patient

Trial Architecture
Conv (16, 3*3, input_shape (200*200*1) ...) **Conv (16, 3*3, …)** **MaxPooling (2*2)**
Conv (32, 3*3, …) **Conv (32, 3*3, …)** **MaxPooling (2*2)**
Conv (64, 3*3, …) **Conv (64, 3*3, …)** **MaxPooling (2*2)**
Flatten () **Dense (64, activation='relu')**
Dense (32, activation='relu')
Dense (2, activation='segmoid')

Table 18

Number of Epochs	**20**
Train Data Size	**80%**
Test Data Size	**20%**
Training Accuracy	**89%**
Testing Accuracy	**84%**
Trial Evaluation	**90%**

Trial Details

	precision	recall	f1-score	support
0	0.85	0.98	0.91	133
1	0.94	0.57	0.71	53
accuracy			0.87	186
macro avg	0.89	0.78	0.81	186
weighted avg	0.88	0.87	0.85	186

Trail description:
A deep learning framework of 6-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 3 sets of dual convolutional and pooling layers (2 × 2), 3 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function categorical_crossentropy,
ImageDataGenerator with:
rotation_range= 40, width_shift_range = 0.08, height_shift_range = 0.08, shear_range = 0.08, zoom_range = 0.08, horizontal_flip = True, fill_mode = 'reflect' and batch_size = 10.

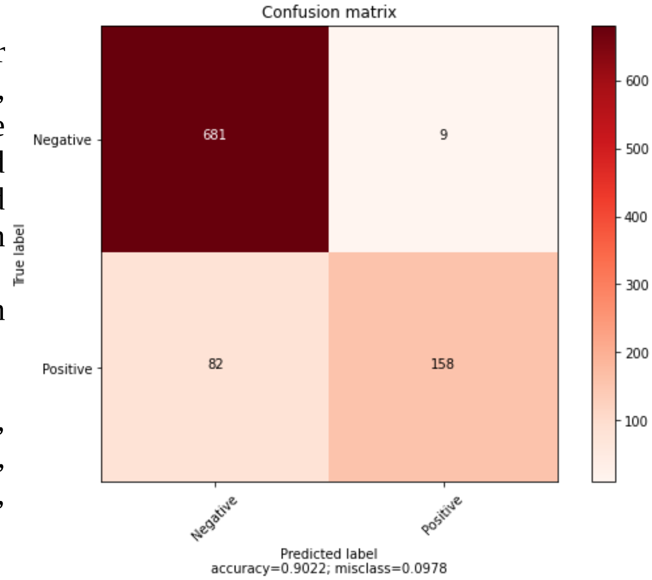


Table 19

Note: This model is good at training and testing but the f1-score for positive class is low, so it is not detect positive class efficiently.

5.2.4.2. With 15 images for each patient

Trial Architecture
Conv (16, 3*3, input_shape (200*200*1) ...) Conv (16, 3*3, …) MaxPooling (2*2)
Conv (32, 3*3, …) Conv (32, 3*3, …) MaxPooling (2*2)
Conv (64, 3*3, …) Conv (64, 3*3, …) MaxPooling (2*2)
Flatten () Dense (64, activation='relu')
Dense (32, activation='relu')
Dense (2, activation='segmoid')

Table 20

Number of Epochs	20
Train Data Size	70%
Test Data Size	30%
Training Accuracy	80%
Testing Accuracy	77%
Trial Evaluation	81%

Trial Details

```
              precision    recall  f1-score   support

           0       0.81      0.98      0.89       105
           1       0.85      0.31      0.46        35

    accuracy                           0.81       140
   macro avg       0.83      0.65      0.67       140
weighted avg       0.82      0.81      0.78       140
```

Trail description:
A deep learning framework of 6-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 3 sets of dual convolutional and pooling layers (2 × 2), 3 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function categorical_crossentropy,
ImageDataGenerator with:
rotation_range= 40, width_shift_range = 0.08, height_shift_range = 0.08, shear_range = 0.08, zoom_range = 0.08, horizontal_flip = True, fill_mode = 'reflect' and batch_size = 10.

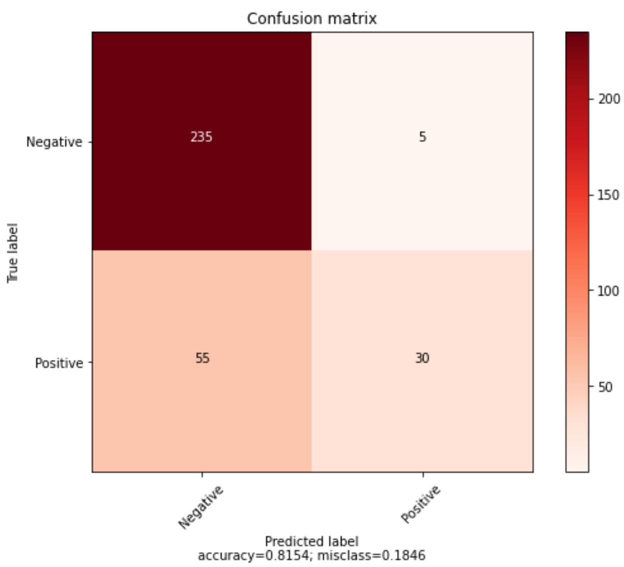


Table 21

Note: This model is good at training but not enough at testing and the f1-score for positive class is worse than last model.

5.2.5. CNN Model with balanced weights

Trying to make the f-score for both classes closer we used balanced weights.

5.2.5.1. With 30 images for each patient

Trial Architecture
Conv (16, 3*3, input_shape (200*200*1) Conv (16, 3*3, …) MaxPooling (2*2)
Conv (32, 3*3, …) Conv (32, 3*3, …) MaxPooling (2*2)
Conv (64, 3*3, …) Conv (64, 3*3, …) MaxPooling (2*2)
Flatten () Dense (64, activation='relu')
Dense (32, activation='relu')
Dense (2, activation='segmoid')

Number of Epochs	20
Train Data Size	80%
Test Data Size	20%
Training Accuracy	85%
Testing Accuracy	81%
Trial Evaluation	87%

Table 22

Trial Details

	precision	recall	f1-score	s
0	0.88	0.90	0.89	
1	0.74	0.70	0.72	
accuracy			0.84	
macro avg	0.81	0.80	0.81	
weighted avg	0.84	0.84	0.84	

Note: This model is good at training and testing but still a big difference in f1-score between two classes.

5.2.5.2. With 15 images for each patient

Trial Architecture
Conv (16, 3*3, input_shape (200*200*1) ...) Conv (16, 3*3, …) MaxPooling (2*2)
Conv (32, 3*3, …) Conv (32, 3*3, …) MaxPooling (2*2)
Conv (64, 3*3, …) Conv (64, 3*3, …) MaxPooling (2*2)
Flatten () Dense (64, activation='relu')
Dense (32, activation='relu')
Dense (2, activation='segmoid')

Table 23

Number of Epochs	**20**
Train Data Size	**70%**
Test Data Size	**30%**
Training Accuracy	**70%**
Testing Accuracy	**72%**
Trial Evaluation	**81%**

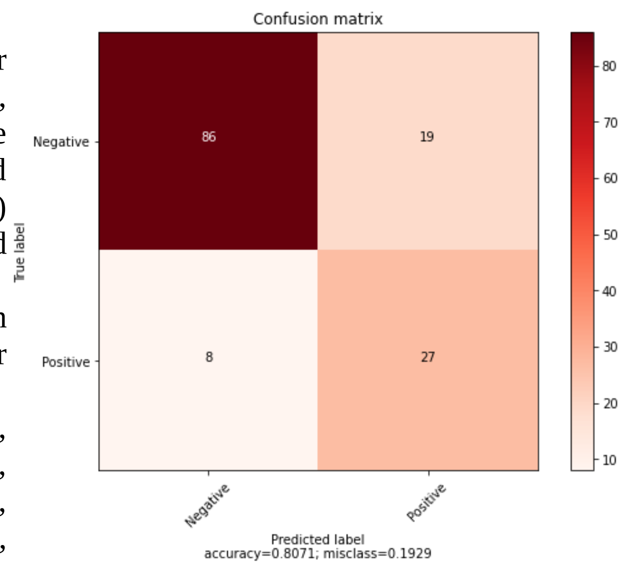
Trial Details

```
              precision    recall  f1-score   support

           0       0.91      0.82      0.86       105
           1       0.59      0.77      0.67        35

    accuracy                           0.81       140
   macro avg       0.75      0.80      0.77       140
weighted avg       0.83      0.81      0.81       140
```

Trail description:
A deep learning framework of 6-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 3 sets of dual convolutional and pooling layers (2 × 2), 3 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function categorical_crossentropy, ImageDataGenerator with:
rotation_range= 40, width_shift_range = 0.08, height_shift_range = 0.08, shear_range = 0.08, zoom_range = 0.08, horizontal_flip = True, fill_mode = 'reflect' and batch_size = 10, balanced_weight= 'balanced'.



Note: This model not good at training and testing and still a big difference in f1-score between two classes.

5.2.6. CNN Model with SMOTE

SMOTE is synthetic minority oversampling technique. It synthesizes new minority instances between existing minority instances. It generates the virtual training records by linear interpolation for the minority class. These synthetic training records are generated by randomly selecting one or more of the k-nearest neighbors for each example in the minority class. After the oversampling process, the data is reconstructed and several classification models can be applied for the processed data.

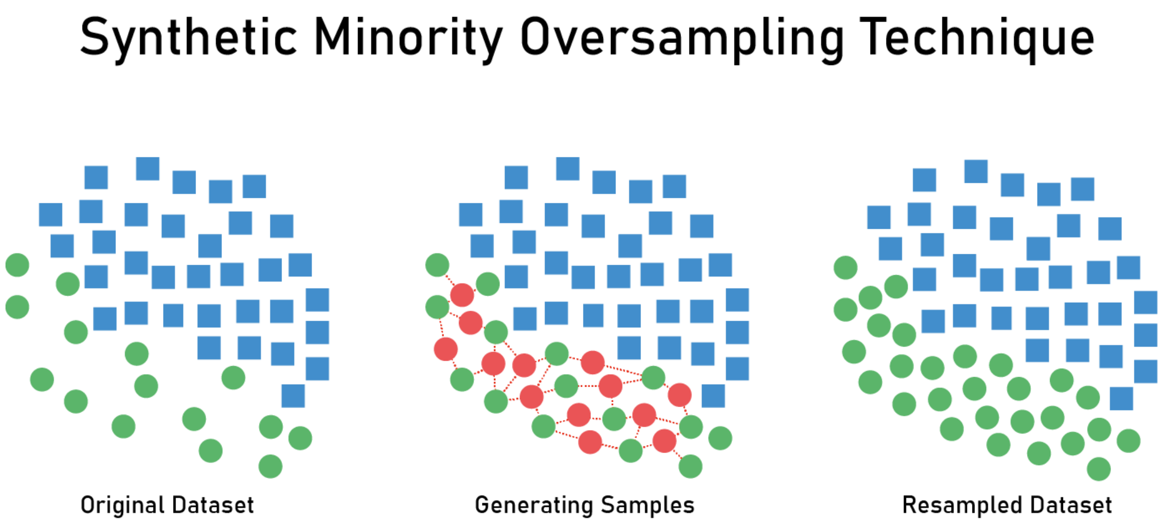


FIGURE 19

Trial Architecture
Conv (16, 3*3, input_shape (200*200*1) ...) **Conv (16, 3*3, …)** **MaxPooling (2*2)**
Conv (32, 3*3, …) **Conv (32, 3*3, …)** **MaxPooling (2*2)**
Conv (64, 3*3, …) **Conv (64, 3*3, …)** **MaxPooling (2*2)**
Flatten () **Dense (64, activation='relu')**
Dense (32, activation='relu')
Dense (2, activation='segmoid')

Table 24

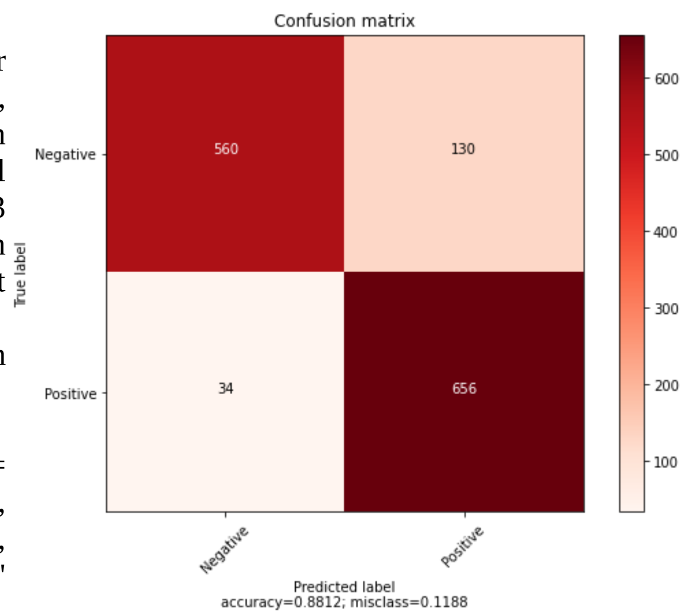
Number of Epochs	**20**
Train Data Size	**80%**
Test Data Size	**20%**
Training Accuracy	**87%**
Testing Accuracy	**89%**
Trial Evaluation	**88%**

Trial Details

```
              precision    recall  f1-score   support

           0       0.96      0.85      0.90       130
           1       0.88      0.97      0.92       146

    accuracy                           0.91       276
   macro avg       0.92      0.91      0.91       276
weighted avg       0.92      0.91      0.91       276
```



Trail description:
A deep learning framework of 6-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 3 sets of dual convolutional and pooling layers (2 × 2), 3 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function **categorical**_crossentropy,
ImageDataGenerator with:
rotation_range= 40, width_shift_range = 0.08, height_shift_range = 0.08, shear_range = 0.08, zoom_range = 0.08, horizontal_flip = True, fill_mode = 'reflect' and batch_size = 10.

Note: Not logic that the testing accuracy greater than training accuracy.

Using SMOTE by different way

SMOTE make minority class equals majority class, we think it is not right to make them equals. So, we will take a random sample from the dataset then add this sample to resynthesized data from SMOTE.
Then we will use a balanced weights from these new data.

Trial Architecture
Conv (16, 3*3, input_shape (200*200*1) ...) Conv (16, 3*3, …) MaxPooling (2*2)
Conv (32, 3*3, …) Conv (32, 3*3, …) MaxPooling (2*2)
Conv (64, 3*3, …) Conv (64, 3*3, …) MaxPooling (2*2)
Flatten () Dense (64, activation='relu')
Dense (32, activation='relu')
Dense (2, activation=’segmoid’)

Table 25

Number of Epochs	**20**
Train Data Size	**80%**
Test Data Size	**20%**
Training Accuracy	**83.159%**
Testing Accuracy	**81.944%**
Trial Evaluation	**83.26%**

Trial Details

```
              precision    recall  f1-score   support

           0       0.76      0.98      0.85       139
           1       0.97      0.70      0.82       149

    accuracy                           0.84       288
   macro avg       0.86      0.84      0.83       288
weighted avg       0.87      0.84      0.83       288
```

Trail description:
A deep learning framework of 6-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 3 sets of dual convolutional and pooling layers (2 × 2), 3 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer.
And we use loss function categorical_crossentropy, ImageDataGenerator with:
rotation_range= 40, width_shift_range = 0.08, height_shift_range = 0.08, shear_range = 0.08, zoom_range = 0.08, horizontal_flip = True, fill_mode = 'reflect' and batch_size = 10, balanced_weight= 'balanced'.

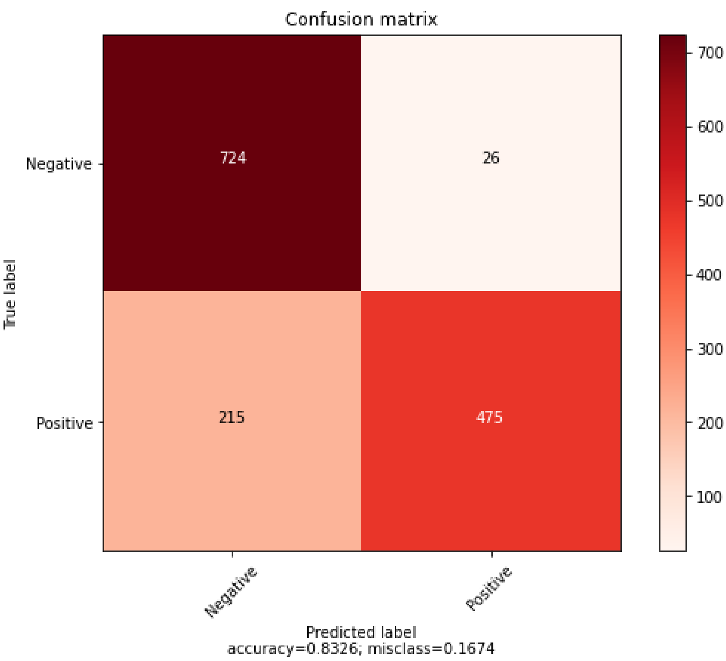


Table 26

Note : This model is best model we got.

5.2.7 System Test Cases

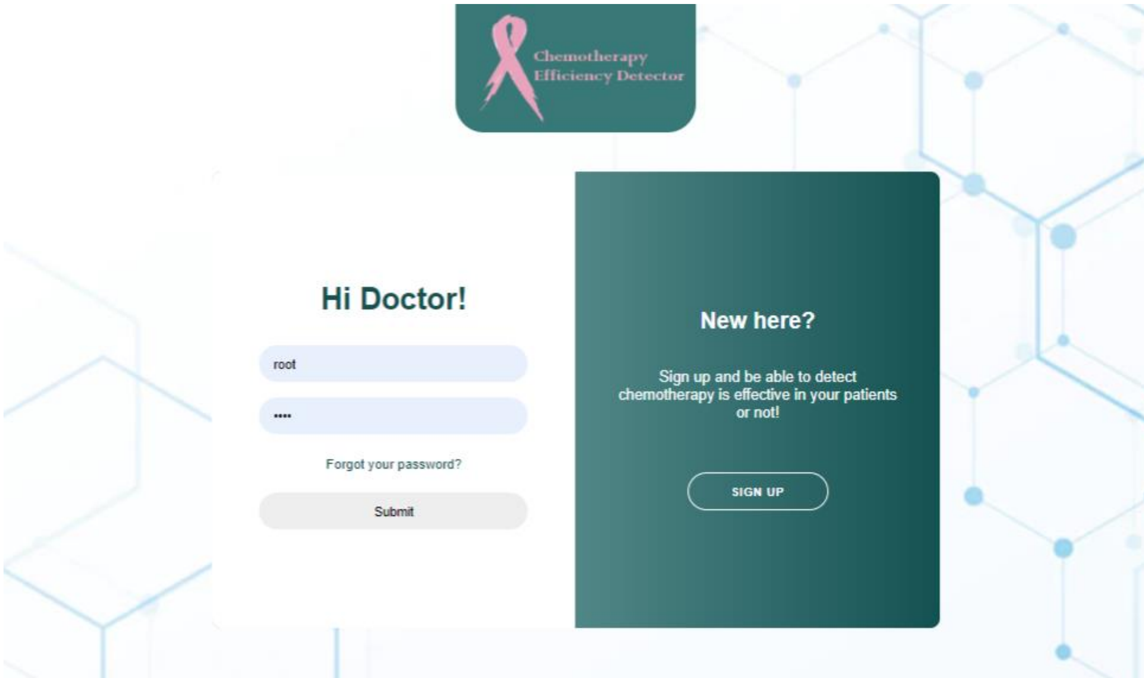


FIGURE 20

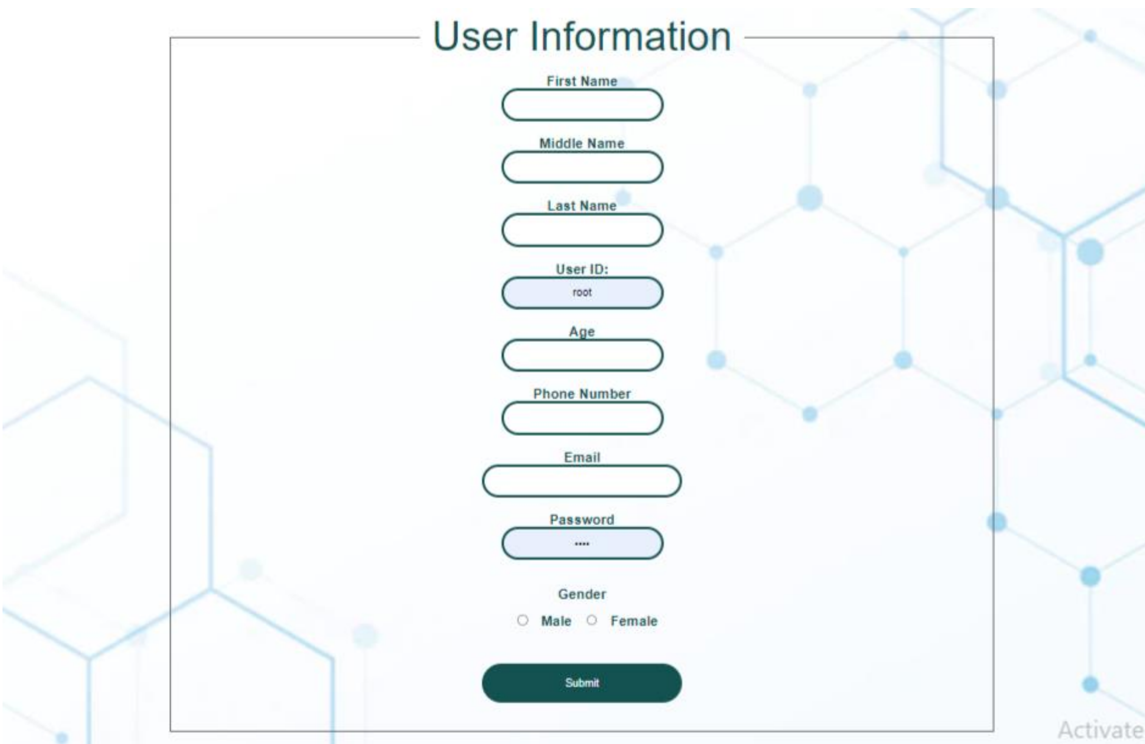


FIGURE 21

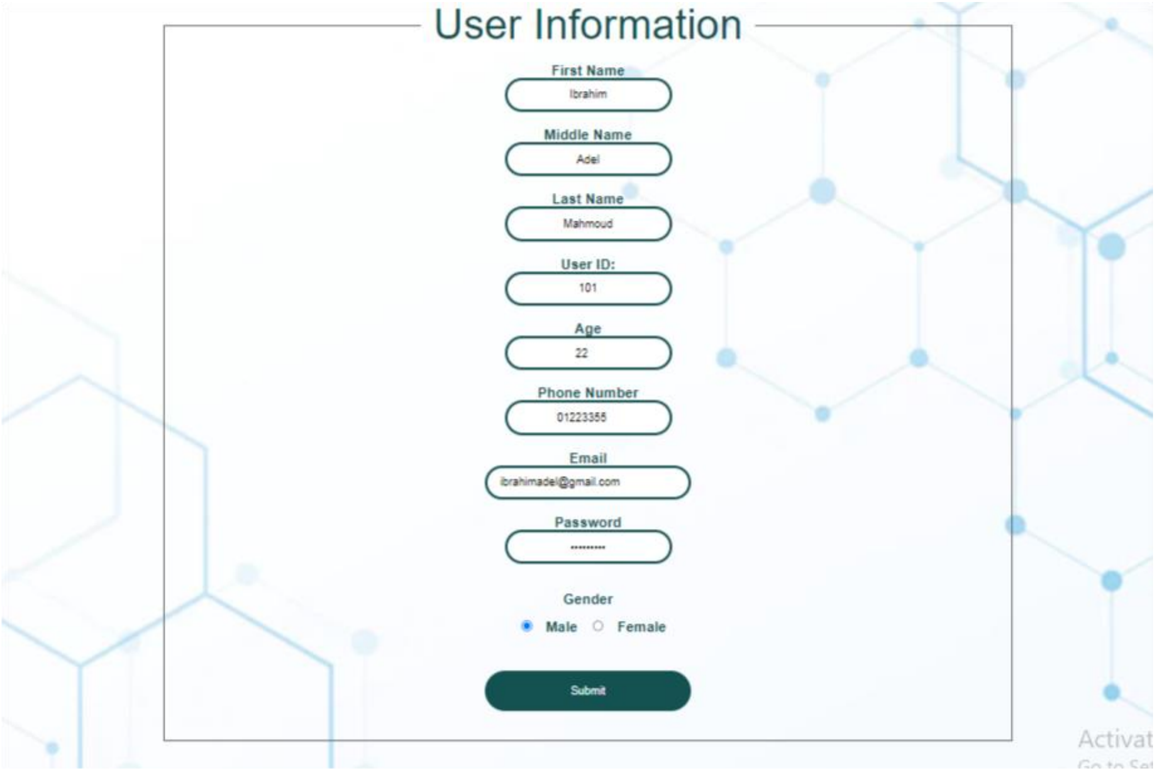


FIGURE 22

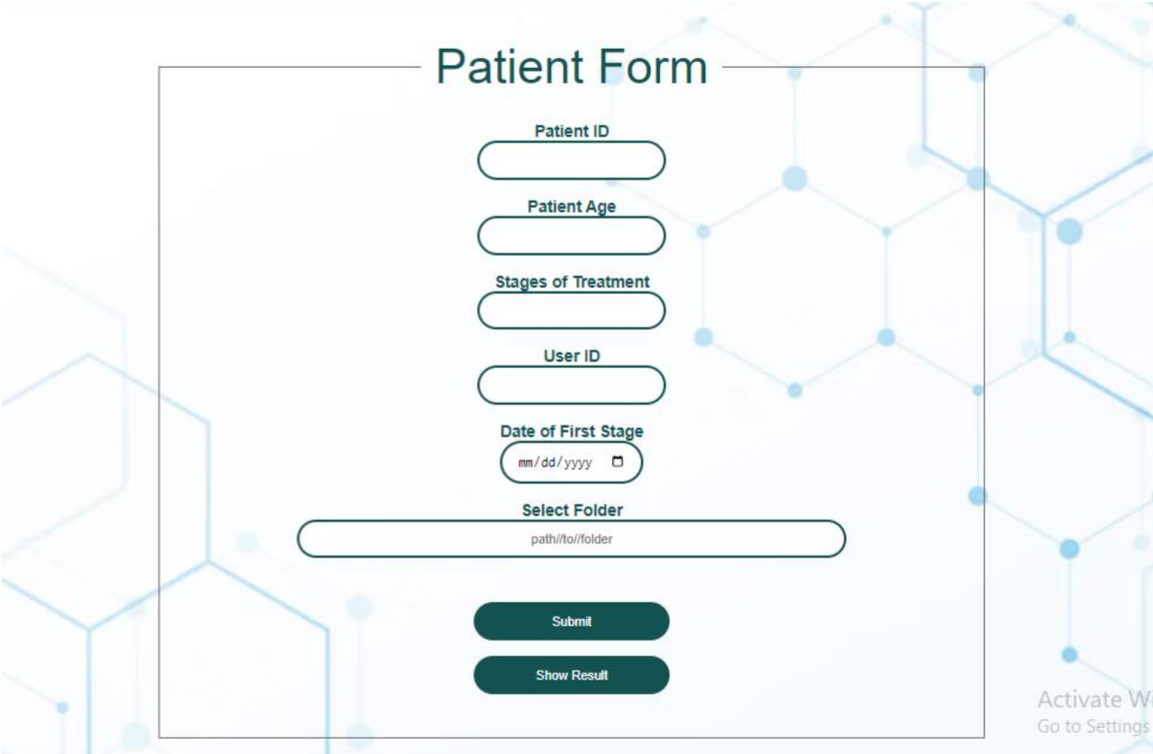


FIGURE 23

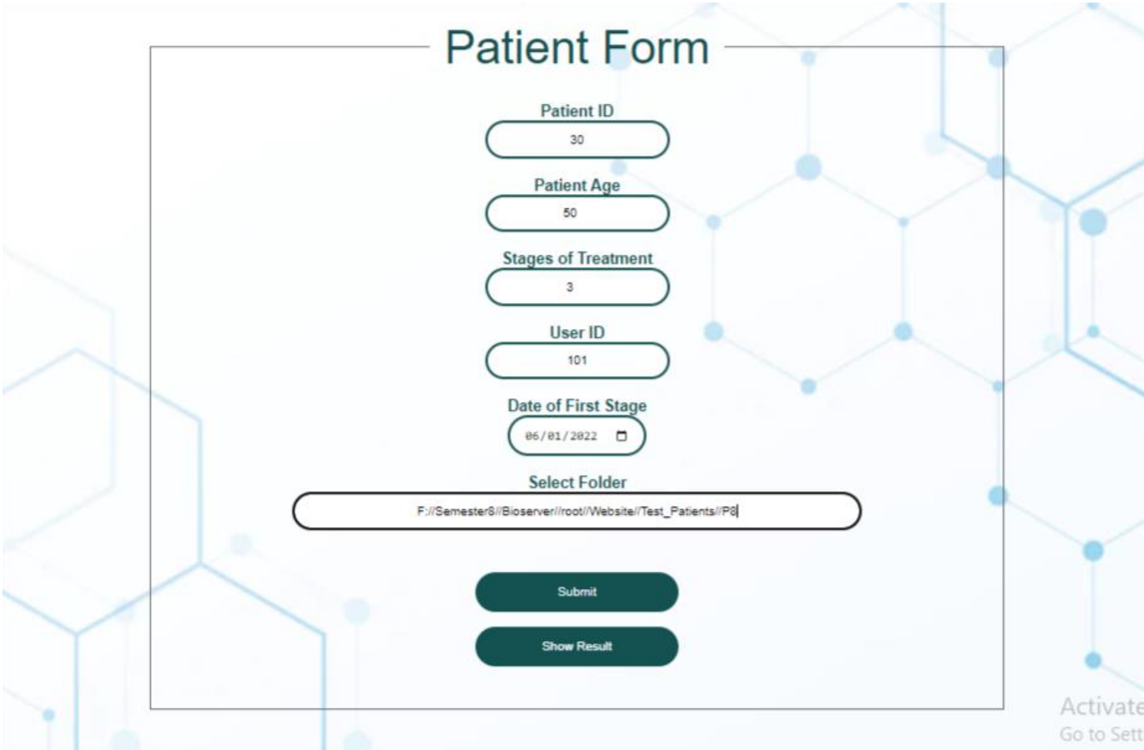


FIGURE 24

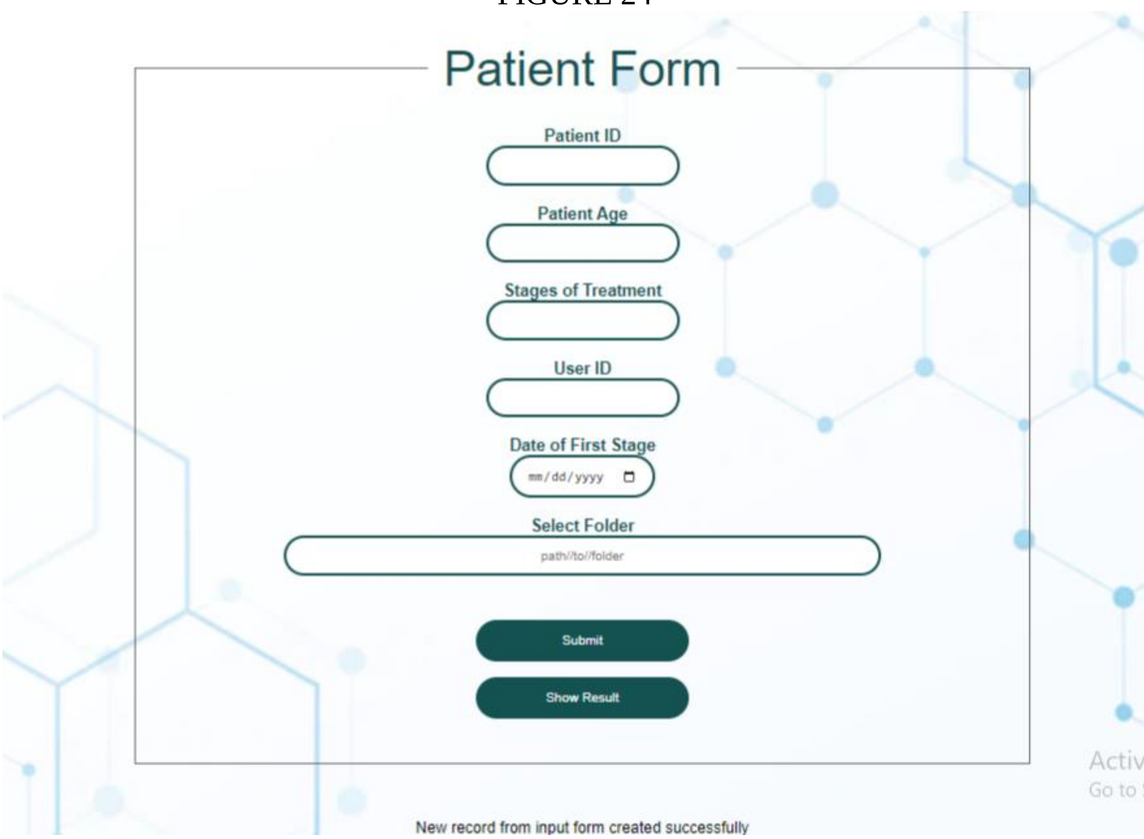


FIGURE 25

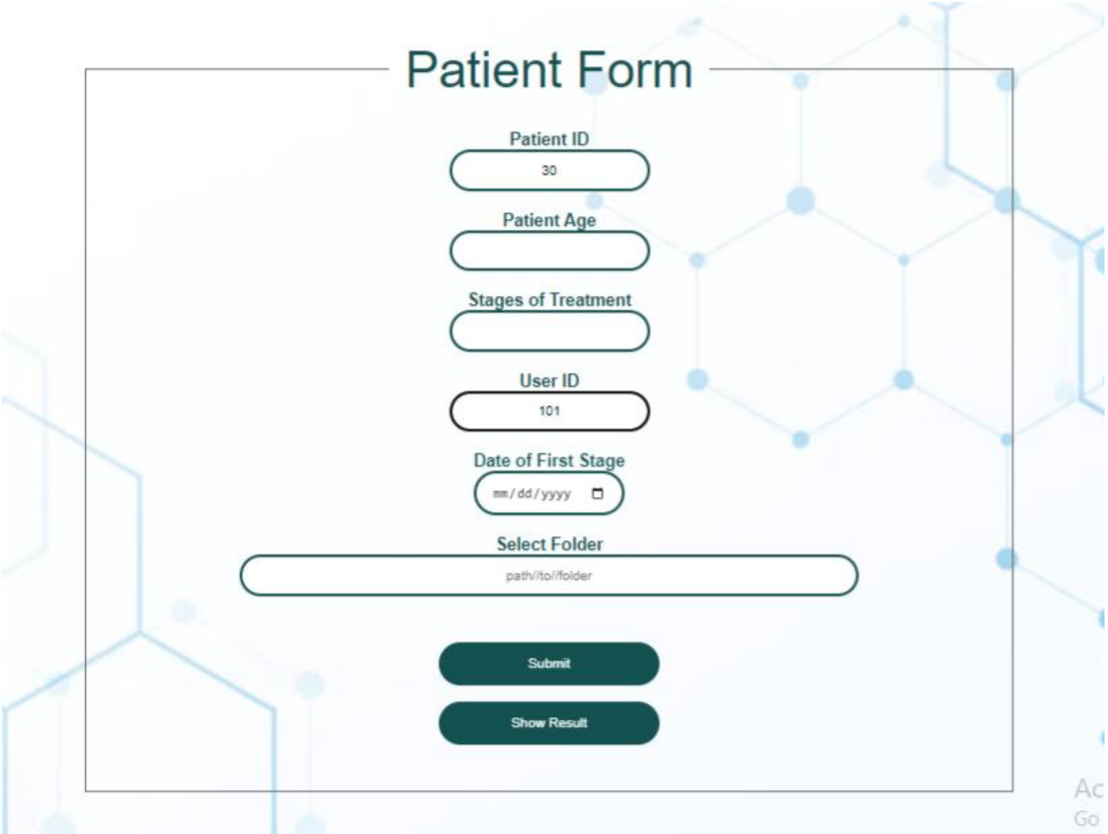


FIGURE 26

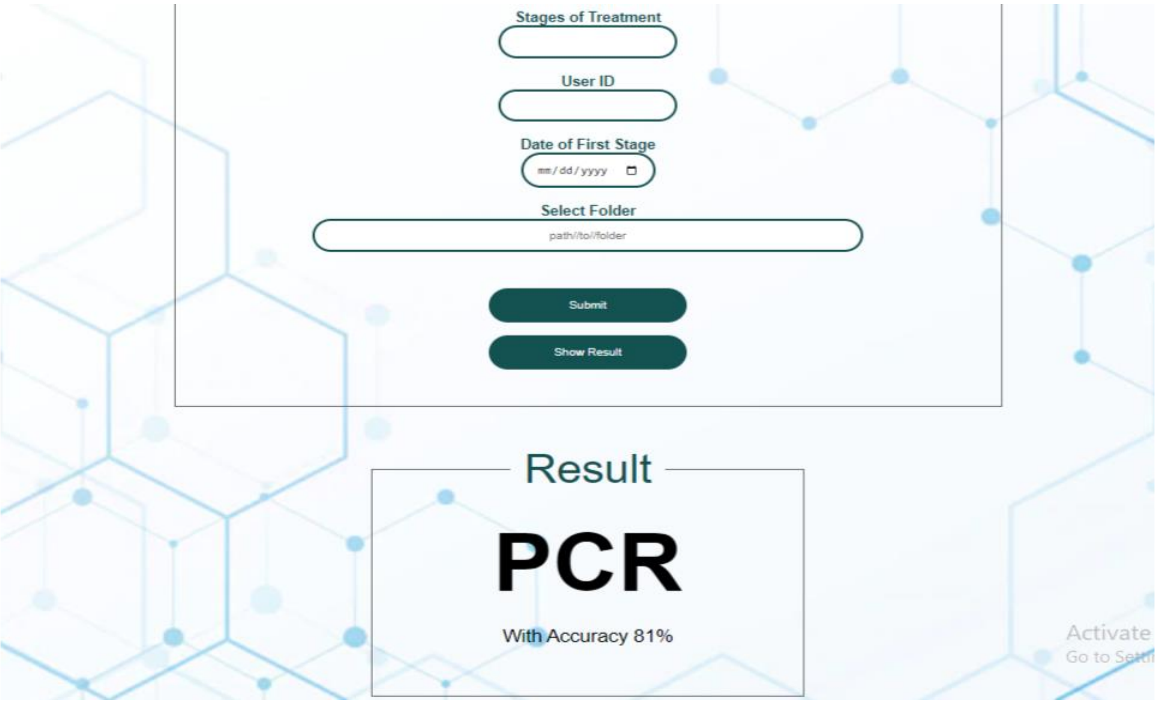


FIGURE 27

Poster

Chemotherapy Efficiency Detector

Haydey, Ibrahim, and Eman
Supervisor: Dr. sabah

Abstract

Breast cancer is one of the most dangerous diseases that women face, with the greatest mortality rate.

Cancer classifications are intended to provide an accurate diagnosis of disease and predict tumor behavior to facilitate oncologist decision-making.

Our project is a website working in cancer classification with different stages of treatment. And the aim of this project is to use image processing and machine learning techniques in analyze a CT scan. This analysis study the effect of chemotherapy on a patient. Whether positive or negative, to assess their progress and build a personalized treatment using data from patients diagnosed with breast cancer.

The website takes the data in the form of CT scans. The result will be whether chemotherapy is effective in this case or not.

Introduction

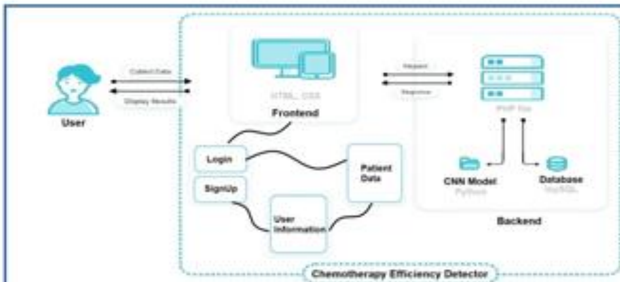
If we search about breast cancer, we can find a lot of projects detected the cancer if it malignant or benign. Or we find projects about chemotherapy, but we will not find project can detect if the chemotherapy is effective with patient or not.

In this project we will try to build a model that help to know the efficiency of drug.

The main application in our project is a website response input data from user as computed tomography CT images and send result to user as PCR or NONPCR Which shows whether the patient has been completely cured or not

Methods

In a deep learning technique, we use a framework of 6-layer convolutional neural networks (CNNs), Containing one input layer with input_shape (200*200*1), 3 sets of dual convolutional and pooling layers (2 × 2), 3 fully connected (dense) layers, Activation function Relu/Segmoid and one output layer and loss function categorical_crossentropy. And we use data augmentation method with rotation_range= 40, width_shift_range = 0.08, height_shift_range = 0.08, shear_range = 0.08, zoom_range = 0.08, horizontal_flip = True, fill_mode = 'reflect' and batch_size = 10. And last method is SMOTE from Imbalanced-learn library and use it with balanced weights. This model accuracy is 83.26%. In web application we use PHP programming language to implement the model python file and mySQL database. Whih was the best method for us to implement the model.



Primarily Design

Our project separates into two main components:

- First one is front end which represent the user interface in our website.

  We use HTML and CSS to implement the web application pages.

  When user visit our website first page appears is the Signup page, in this page user can choose if he wants to create a new account or he already has an account and want to log in.

  If user clicked on register, then he will move to the next page which takes the user information and make him create a unique email, password and user id. When user click on submit, the last page will appear which takes user id, a unique patient id which user create, other information and path of the folder that contains the patient images. All these data will insert into database by clicking on submit button. User can get the result of his patients any time by write patient and user id then click show result button, the result will appear as PCR or nonPCR and the accuracy of detection.

  But if user write his email and password and clicked on log in button, he will move to the last page directly.

- Second one is back end which represented in connecting all HTML files with database and CNN model.

  In user page all values that existed at input boxes will insert into user table at mySQL database by PHP. The same thing happened in the patient page.

  In PHP file of patient page, the path will send to python file which takes the path and call function predict to preprocessing the images before input to the CNN model. CNN model predicts the patient belonging to which class, then function predict handles the output and return it to PHP file.

  PHP file takes the output and insert it to the database.

Conclusion

After take best sample from dataset where we did it twice, first sample has clearest 10 images from each stage in patient folder, second sample has clearest 5 images from each stage in patient folder.

We then preprocessing datasets and started to implement different models with different approaches, we compared and analyzed the results, nearly we tried a hundred of deep learning models.

A lot of problems faced us because the amount of our dataset. The biggest problem was overfitting, so we used augmented data and it gave us acceptable results.

One of the problems that the minority class is too small compared with majority class, we used balanced weights and it gave us a good result.

But the best result we got when we use SMOTE to equals the classes then we increase one of them by add a random sample to it, then we used balanced weights, that model gave us 83.26% accuracy.

We built a web application depend on this model as a back end. We built it using HTML, CSS and PHP. This web application has a database to save all information of user and patients result.

And the final results were very satisfiable to us.

Contact with us:

ia91142@gmail.com
haydeyalikhalil@gmail.com
eman010038@gmail.com

Reference

Data:
https://wiki.cancerimagingarchive.net/display/Public/QIN-Breast

Related projects:

1. https://pubmed.ncbi.nlm.nih.gov/31944571/
2. https://pubmed.ncbi.nlm.nih.gov/32556920/

Statistics:

1. https://www.who.int/news-room/fact-sheets/detail/breast-cancer
2. https://www.stopbreastcancer.org/information-center/facts-figures/