Prediction of Heart Disease Using Deep Learning

A project report submitted in partial fulfillment of the requirement for degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE & ENGINEERING

By

Mulabagal Suryateja (R170184)

Under the guidance of

Ms C. Suneetha

Asst.Prof. In Department of Computer Science & Engineering

AP IIIT, RGUKT-RK Valley,

Vempalli, Kadapa (Dist), Andhra Pradesh-516330,India.

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

(A.P.Government Act 18 of 2008)

RGUKT-RK Valley

Vempalli, Kadapa, Andhrapradesh-516330.

CERTIFICATE OF PROJECT COMPLETION

This is to certify that I have examined the thesis entitled submitted by Mulabagal Suryateja (R170184) under our guidance and supervision for the partial fulfillment for the degree of Bachelor of Technology in computer Science and Engineering during the academic session September 2022 – April 2023 at RGUKT-RKVALLEY.

Project Guide

Ms C. Suneetha,

Asst.Prof. in Dept of CSE,

RGUKT-RK Valley.

Head of the Department

Mr. N. Satyanandaram,

Lecturer in Dept of CSE,

RGUKT-RK Valley.

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

(A.P.Government Act 18 of 2008)

RGUKT-RK Valley

Vempalli, Kadapa, Andhrapradesh-516330.

DECLARATION

I, Mulabagal Suryateja (R170184) hereby declare thatthe project report entitled "Prediction of Heart Disease using Deep Learning" done under guidance of Ms C. Suneetha is submitted in partial fulfillment for the degree of Bachelor of Technology in Computer Science and Engineering during the academic session September 2022 – April 2023 at RGUKT-RK Valley. I also declare that this project is a result of ourown effort and has not been copied or imitated from any source. Citations from any websites are mentioned in the references. To the best of my knowledge, the results embodied in this dissertation work have not been submitted to any university or institute for the award of any degree or diploma.

Mulabagal Suryateja(R170184)

ACKNOWLEDGEMENT

I would like to express my deep sense of gratitude & respect to all those people behind the screen who guided, inspired and helped us crown all our efforts with success. I wish to express our gratitude to Ms. C. Suneetha for her valuable guidance at all stages of study, advice, constructive suggestions, supportive attitude and continuous encouragement, without which it would not be possible to complete this project.

We would also like to extend our deepest gratitude & reverence to the Director of RGUKT, RK Valley **Prof. K. Sandyarani** and HOD of Computer Science and Engineering **Mr. N. Satyanandaram** for their constant support and encouragement.

Last but not least I express my gratitude to my parents for their constant source of encouragement and inspiration for me to keep my morals high.

Table of Contents

Abstract

Cardiovascular diseases CVD are among the most common serious illnesses affecting human health. CVDs may be prevented or mitigated by early diagnosis, and this may reduce mortality rates. Identifying risk factors using deep learning models is a promising approach. We would like to propose a model that incorporates different methods to achieve effective prediction of heart disease. For our proposed model to be successful, Pre-processing and Data Transformation methods to create accurate information for the training model. We have used a heart disease image dataset. The results are shown separately to provide comparisons. Based on the result analysis, we can conclude that our proposed model produced the highest accuracy while using RFBM and Relief feature selection methods.

Problem Statement

The Main objective of this project is detected whether a patient have any chance to get a heart stroke or not and to know this, we used classification techniques. implementation of machine learning algorithms is bit complex to build due to the lack of information about the data visualization and Relief feature selection. Mathematical calculations are used in existing system for model building this may take the lot of time and complexity. To overcome all this, we use machine learning packages available in the scikit-learn library.

Introduction

Cardiovascular disease is a major global health issue with a high mortality rate. Machine learning and artificial intelligence have shown promising results in predicting heart disease by leveraging large datasets and identifying relevant risk factors. However, challenges such as inconsistent and redundant clinical datasets, feature selection, and the appropriate application of deep learning algorithms still exist. Researchers have used various techniques such as ensemble approaches, individual learning algorithms, and feature selection algorithms to improve the accuracy of prediction models. The accuracy of these models has been reported to range from 70% to 97.47% in different studies. These models have the potential to aid in early diagnosis and prevention of severe heart attacks, thereby reducing the burden of cardiovascular disease on healthcare systems and improving patient outcomes.

Existing System

This model emphasizes an existing method that which is designed using the some of the algorithms of machine learning. Here the process is performed using the deep learning, which is one of the transfer learning methods, but this could not get the high accuracy.

Drawbacks of Existing Method:

- High complexity.
- Time consuming.
- Low accuracy

Proposed System

The proposed method we are performing the classification of either the Plant Leaf Disease identification using Convolution Neural Network (CNN) of deep learning along with the MobileNet. As image analysis-based approaches for Heart Disease detection. Hence, proper classification is important for the Leaf disease that which will be possible by using our proposed method. Block diagram of proposed method is shown below.

Advantages:

- Highest accuracy
- Reduces time complexity

Tools & Technologies Used:

Front-end:

- HTML
- CSS
- JavaScript

Backend:

- Python
- Django

Algorithms:

- Convolution Neural Network (CNN)
- MobileNet

Tools

- Visual Studio Code Editor

Algorithms:

1.Convolutional Neural Network

Step1: convolutional operation

The first building block in our plan of attack is convolution operation. In this step, we will touch on feature detectors, which basically serve as the neural network's filters. We will also discuss feature maps, learning the parameters of such maps, how patterns are detected, the layers of detection, and how the findings are mapped output

The convolution Operation

0	1	0	0	0	0	0
0	1	0	0	0	1	0
0	0	0	0	0	0	0
0	0	0	1	0	0	0
0	1	0	0	0	1	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0

0	0	1
1	0	0
0	1	1

Step (1b): ReLU Layer

The second part of this step will involve the Rectified Linear Unit or Relook. We will cover Relook layers and explore how linearity functions in the context of Convolutional Neural Networks. Not necessary for understanding CNNs, but there's no harm in a quick lesson to improve your skills.

Step 2: Conv2D

Keras Conv2D is 2D Convolution Layer; this layer creates a convolution kernel that is wind with layers input which helps produce a tensor of outputs. Kernel: In image processing kernel is a convolution matrix or masks which can be used for blurring, sharpening, embossing, edge detection, and more by doing a convolution between a kernel and an image.

Step 3: Flattening

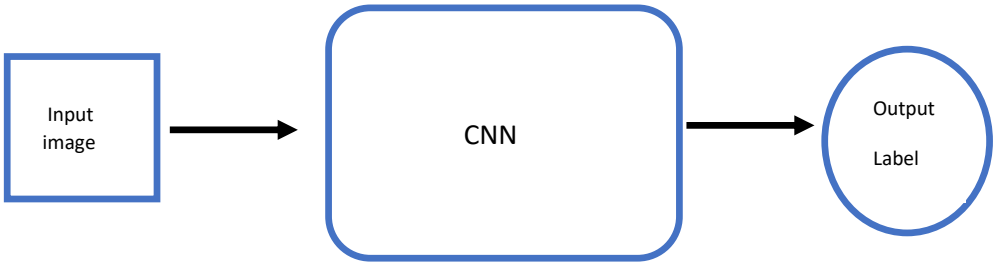
This will be a brief breakdown of the flattening process and how we move from pooled to flattened layers when working with Convolutional Neural Networks.

Step 4: Full Connection

In this part, everything that we covered throughout the section will be merged together. By learning this, you'll get to envision a fuller picture of how Convolutional Neural Networks operate and how the "neurons" that are finally produced learn the classification of images.

Convolutional neural network (CNN):

A convolutional neural network consists of an input layer, hidden layers and an output layer. In any feed-forward neural network, any middle layers are called hidden because their inputs and outputs are masked by the activation function and final convolution.

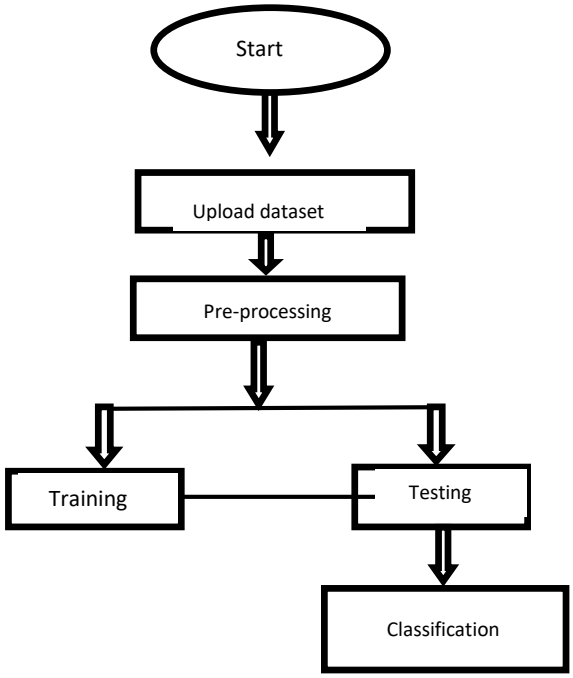


Mobile Net:

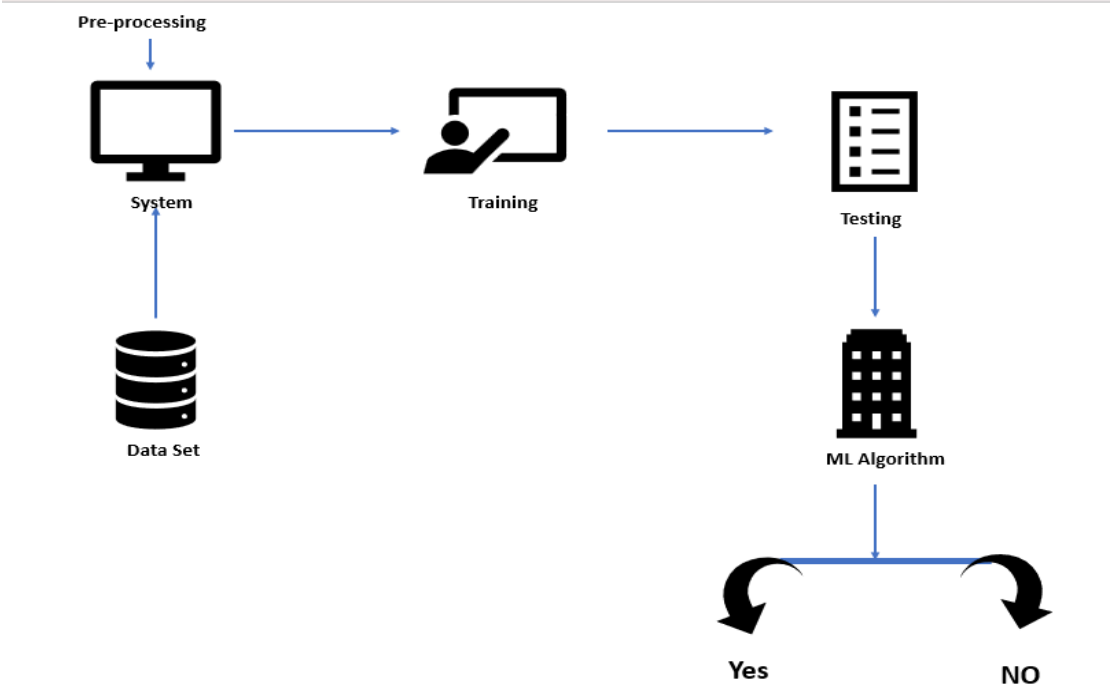
As the name applied, the MobileNet model is designed to be used in mobile applications, and it is Tensor Flow’s first mobile computer vision model.

MobileNet uses depthwise **separable convolutions.** It significantly **reduces the number of parameters** when compared to the network with regular convolutions with the same depth in the nets. This results in lightweight deep neural networks.

Dataflow Diagram



Architecture:



SYSTEM DESIGN:

Input Design:

In an information system, input is the raw data that is processed to produce output. During the input design, the developers must consider the input devices such as PC, MICR, OMR, etc.

- It should serve specific purpose effectively such as storing, recording, and retrieving the information.
- All these objectives are obtained using the knowledge of basic design principles regarding −
    - What are the inputs needed for the system?
    - How end users respond to different elements of forms and screens.

Objectives for Input Design:

The objectives of input design are −

- To design data entry and input procedures
- To design source documents for data capture or devise other data capture methods

Output Design:

The design of output is the most important task of any system. During output design, developers identify the type of outputs needed, and consider the necessary output controls and prototype report layouts.

Objectives of Output Design:

The objectives of input design are:

- To develop output design that serves the intended purpose and eliminates the production of unwanted output.
- To form the output in appropriate format and direct it to the right person.

SOFTWARE REQUIREMENTS

Hardware:

Operating system	: Windows 7 or 7+
RAM	: 8 GB
Hard disc or SSD	: More than 256 GB
Processor	: Intel 3rd generation or high or Ryzen with 8 GB Ram

Software:

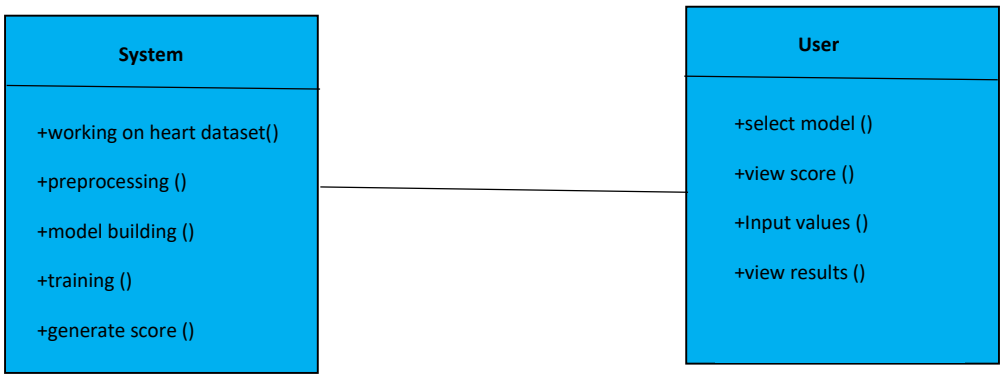
Software's	: Python 3.6 or high version
IDE	: VS code
Framework	: Django

UML DIAGRAMS

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

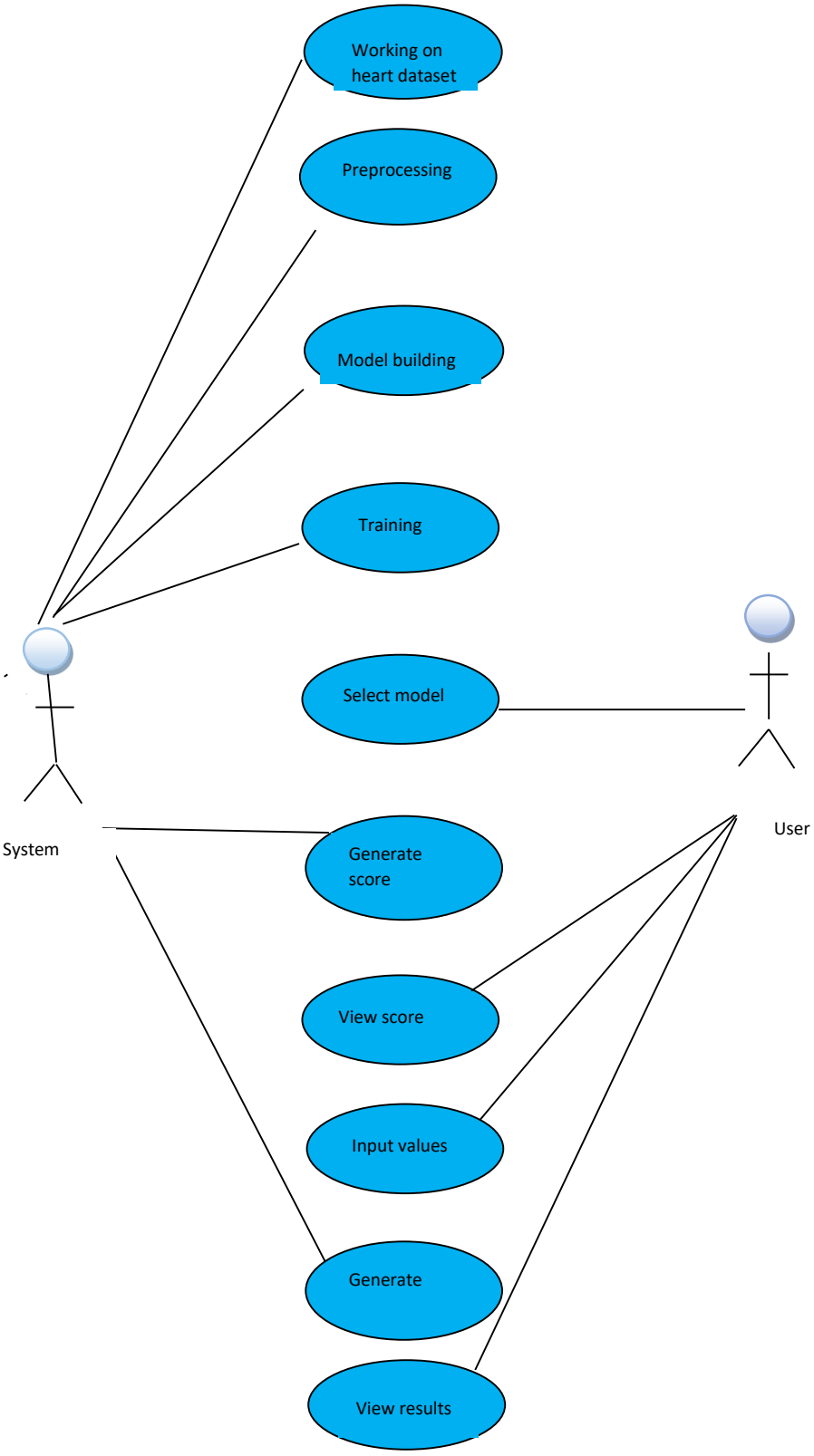
Class Diagram:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information



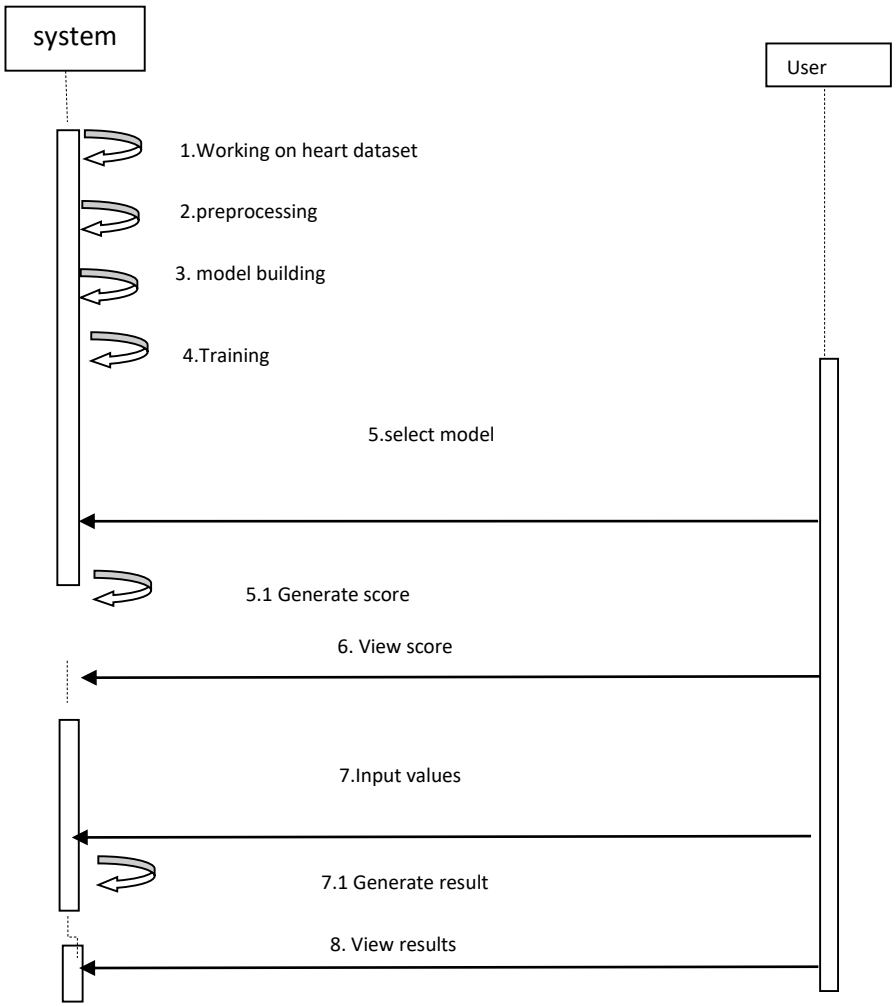
USE CASE DIAGRAM

- A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis.



SEQUENCE DIAGRAM

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order.



MODULES

1. **User**:

1.1 View Home page:

Here user view the home page of cardiovascular diseases web appellation.

1.2 View About page:

In the about page, users can learn more about cardiovascular diseases and risk and symptoms of the particular disease.

1.3 Model Building

To create a model that predicts disease with better accuracy, this module will help user.

1.4 Input Values:

The user must provide input values for the certain fields in order to get results.

1.5 View Results:

User view's the generated results from the model.

1.6 View score:

Here user have ability to view the score in %

2.System:

1.7 Working on heart dataset:

System checks for data whether it is available or not and load the data in CSV

1.8 Pre-processing:

Data need to be pre-processed according the models it helps to increase the accuracy of the model and better information about the data.

1.9 Training the data:

After pre-processing the data will split into two part as train and test data before training with the given algorithms.

1.10 Generated Score:

Here user view the score in %

Source Code

Frontend:

```
from django.shortcuts import render,redirect
import pandas as pd
import numpy as np
import tensorflow as tf
from tensorflow.keras.preprocessing import image
from tensorflow.keras.models import load_model
from django.http import HttpResponse
import os
from .models import Image


Classes=['Normal', 'Sick']

def index(request):
    return render(request,'index.html')


def result(request):
    if request.method=='POST':
        m=int(request.POST['alg'])
        file=request.FILES['file']
        fn=Image(image=file)
        fn.save()
        path=os.path.join('webapp/static/images/',file.name)
        acc=pd.read_csv("webapp/Accuracy.csv")

        if m==1:
            new_model=load_model("webapp\CNN.h5")
            test_image=image.load_img(path,target_size=(256,256))
            test_image=image.img_to_array(test_image)
            test_image/=255
            a=acc.iloc[m -1,1]
        else:
            new_model=load_model("webapp\Mobilenet.h5")
            test_image=image.load_img(path,target_size=(256,256))
            test_image=image.img_to_array(test_image)
            test_image/=255
            a=acc.iloc[m-1,1]

        test_image=np.expand_dims(test_image,axis=0)
        result=new_model.predict(test_image)
        pred=Classes[np.argmax(result)]
        print(pred)

        if pred=="Normal":
            msg="No Treatment Requried"
        else:
            msg='Patient needs the Treatment'

        return render(request,'result.html',{'text':pred,'msg':msg,'path':'static/images/'+file.name,'a':round(a*100,3)})
    return render(request,'result.html')
```

Backend:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import cv2
import os, glob
import tensorflow as tf
from tqdm import tqdm
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import train_test_split
from keras.utils.np_utils import to_categorical
from keras.models import Model, Sequential, load_model
from keras.layers import Dense, Dropout, Flatten, Conv2D, MaxPool2D, BatchNormalization, GlobalAveragePooling2D,Activation,Input
from tensorflow.keras.optimizers import Adam
from keras.preprocessing.image import ImageDataGenerator
```

[1] Python

```
img_height,img_width=256,256
batch_size=10
```

[2] Python

```
data_dir="Data"
Classes=[]
for file in os.listdir(data_dir):
    Classes+=[file]
print(Classes)
```

[3] Python

```
['Normal', 'Sick']
```

```
train_datagen = ImageDataGenerator(rescale=1./255,validation_split=0.3)
train_generator = train_datagen.flow_from_directory(data_dir,
                                                    target_size=(img_height,img_width),
                                                    batch_size=batch_size,
                                                    class_mode='categorical',
                                                    subset='training')
test_generator = train_datagen.flow_from_directory(data_dir,
                                                   target_size=(img_height,img_width),
                                                   batch_size=batch_size,
                                                   class_mode='categorical',
                                                   subset='validation')
```

[4] Python

```
Found 2217 images belonging to 2 classes.
Found 949 images belonging to 2 classes.
```

```
model = Sequential()
model.add(Conv2D(filters=32, kernel_size=(3,3), padding='Same', activation='relu', input_shape=(img_height,img_width, 3)))
model.add(MaxPool2D(pool_size=(2,2)))

model.add(Conv2D(filters=64, kernel_size=(3,3), padding='Same', activation='relu'))
model.add(MaxPool2D(pool_size=(2,2), strides=(2,2)))

model.add(Conv2D(filters=96, kernel_size=(3,3), padding='Same', activation='relu'))
model.add(MaxPool2D(pool_size=(2,2), strides=(2,2)))

model.add(Conv2D(filters=96, kernel_size=(3,3), padding='Same', activation='relu'))
model.add(MaxPool2D(pool_size=(2,2), strides=(2,2)))

model.add(Flatten())
model.add(Dense(512))
model.add(Activation('relu'))
model.add(Dense(2, activation="softmax"))
```

```
model.summary()
```

[5] Python

```
Output exceeds the size limit. Open the full output data in a text editor
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 256, 256, 32)      896

 max_pooling2d (MaxPooling2D  (None, 128, 128, 32)     0
 )

 conv2d_1 (Conv2D)           (None, 128, 128, 64)      18496

 max_pooling2d_1 (MaxPooling  (None, 64, 64, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 64, 64, 96)        55392

 max_pooling2d_2 (MaxPooling  (None, 32, 32, 96)       0
 2D)

 conv2d_3 (Conv2D)           (None, 32, 32, 96)        83040

 max_pooling2d_3 (MaxPooling  (None, 16, 16, 96)       0
 2D)

 flatten (Flatten)           (None, 24576)             0
...
Total params: 12,742,274
Trainable params: 12,742,274
Non-trainable params: 0
```

```
fig, ax = plt.subplots(2,1)
ax[0].plot(hist.history['accuracy'], color='b', label="Training accuracy")
ax[0].plot(hist.history['val_accuracy'], color='r',label="Validation accuracy")
legend = ax[0].legend(loc='best', shadow=True)

ax[1].plot(hist.history['loss'], color='b', label="Training loss")
ax[1].plot(hist.history['val_loss'], color='r',label="validation loss",axes =ax[1])
legend = ax[1].legend(loc='best', shadow=True)
```

[7] Python

```
model.evaluate(test_generator)
model.save("CNN.h5")
```

[8] Python

```
95/95 [==============================] - 13s 140ms/step - loss: 0.1917 - accuracy: 0.9747
```

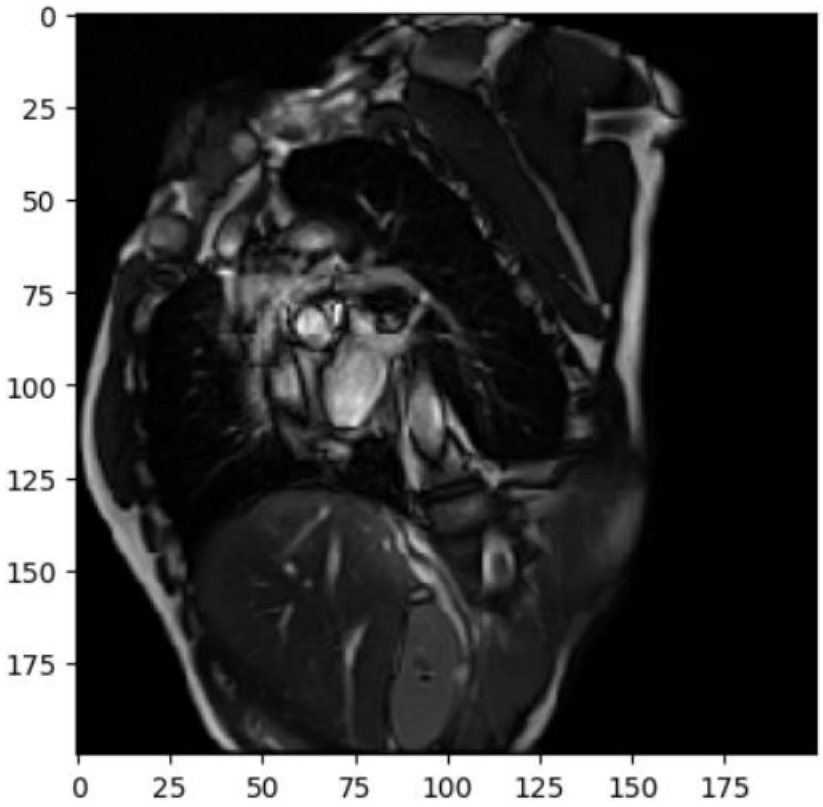
```
base_model = tf.keras.applications.MobileNet(input_shape=(img_height,img_width, 3), include_top=False,
                                              weights='imagenet')
model1 = Sequential()
model1.add(base_model)
model1.add(GlobalAveragePooling2D())
model1.add(Dense(64, activation='relu'))
model1.add(BatchNormalization())
model1.add(Dropout(0.2))
model1.add(Dense(2, activation='softmax'))
model1.summary()
```

[9] Python

```python
from skimage import io
from tensorflow.keras.preprocessing import image

img = image.load_img(r'Data\Sick\IM00001 (10).jpg', grayscale=False, target_size=(256,256))
show_img=image.load_img(r'Data\Sick\IM00001 (10).jpg', grayscale=False, target_size=(200, 200))
Classes = Classes
x = image.img_to_array(img)
x = np.expand_dims(x, axis = 0)

x /= 255

custom = model1.predict(x)
print(custom[0])


plt.imshow(show_img)
plt.show()

a=custom[0]
ind=np.argmax(a)

print('Prediction:',Classes[ind])
```

Python

```
[2.9071571e-06 9.9999714e-01]
```



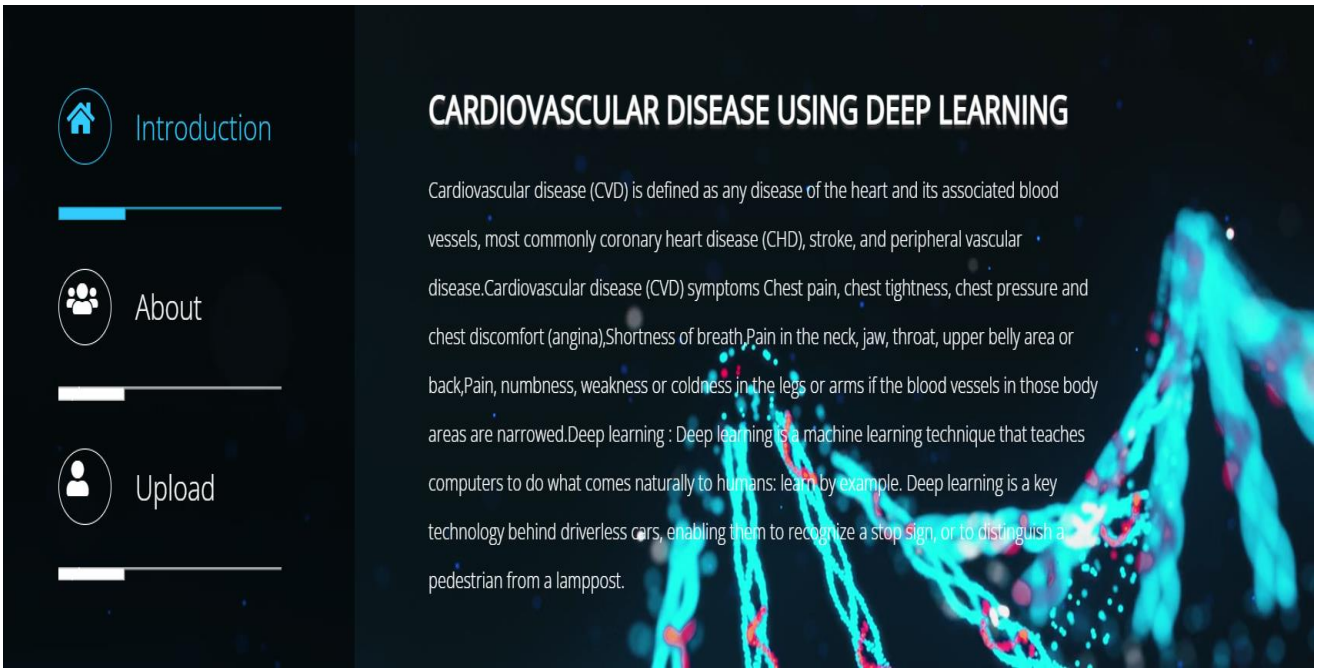

```
Prediction: Sick
```

Results

Output screenshot with Description

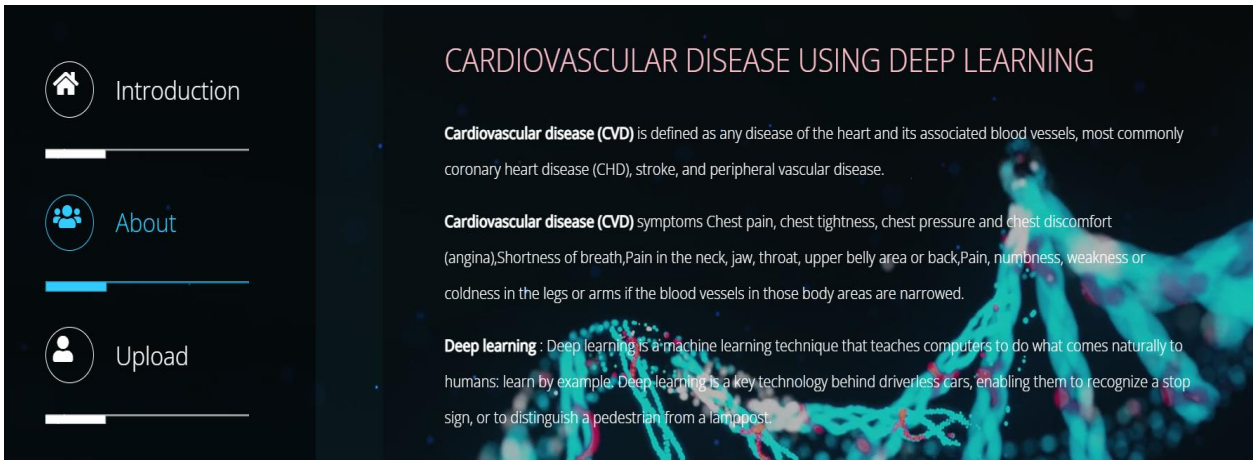
Homepage

Here user view the home page of cardiovascular diseases web appellation.



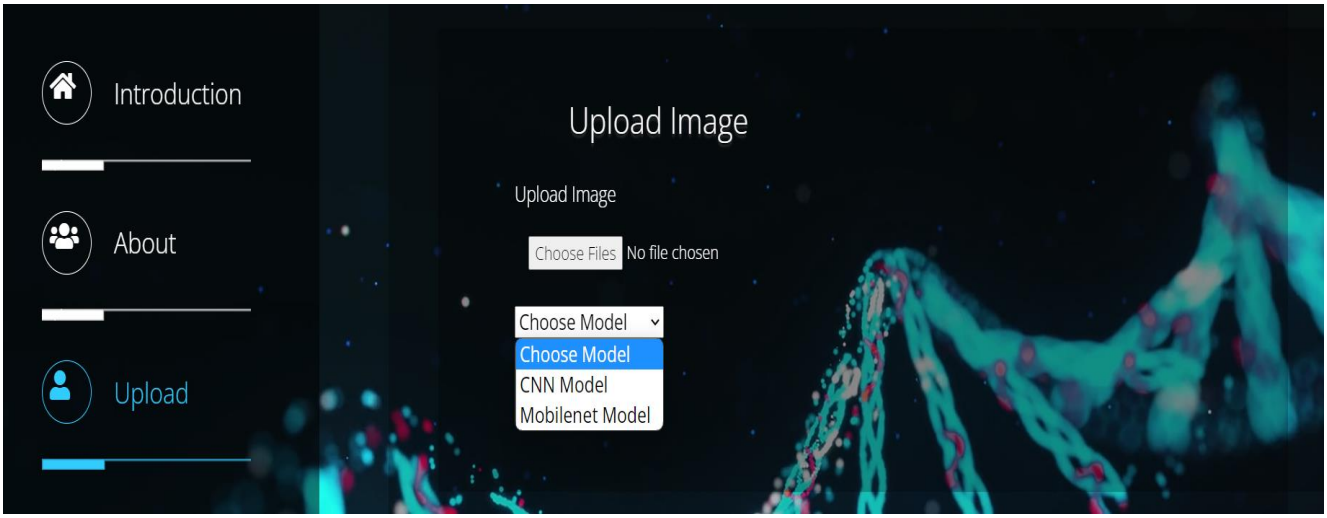
About Page

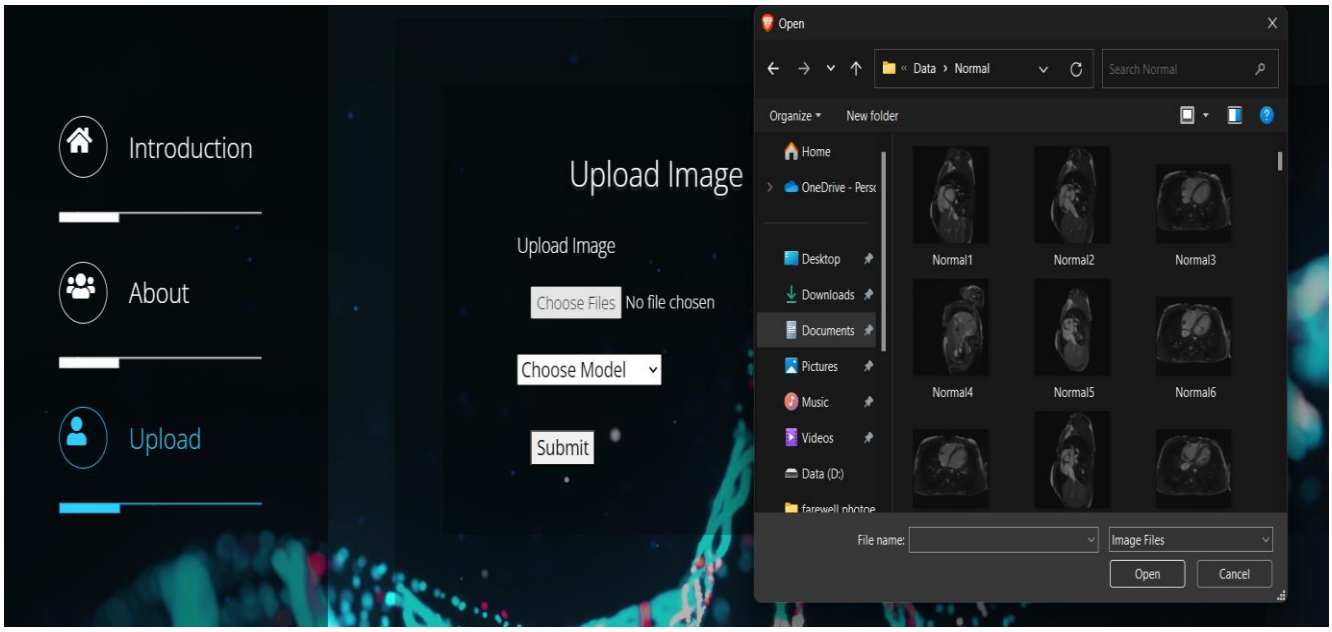
In the about page, users can learn more about cardiovascular diseases and risk and symptoms of the particular disease



Upload Page

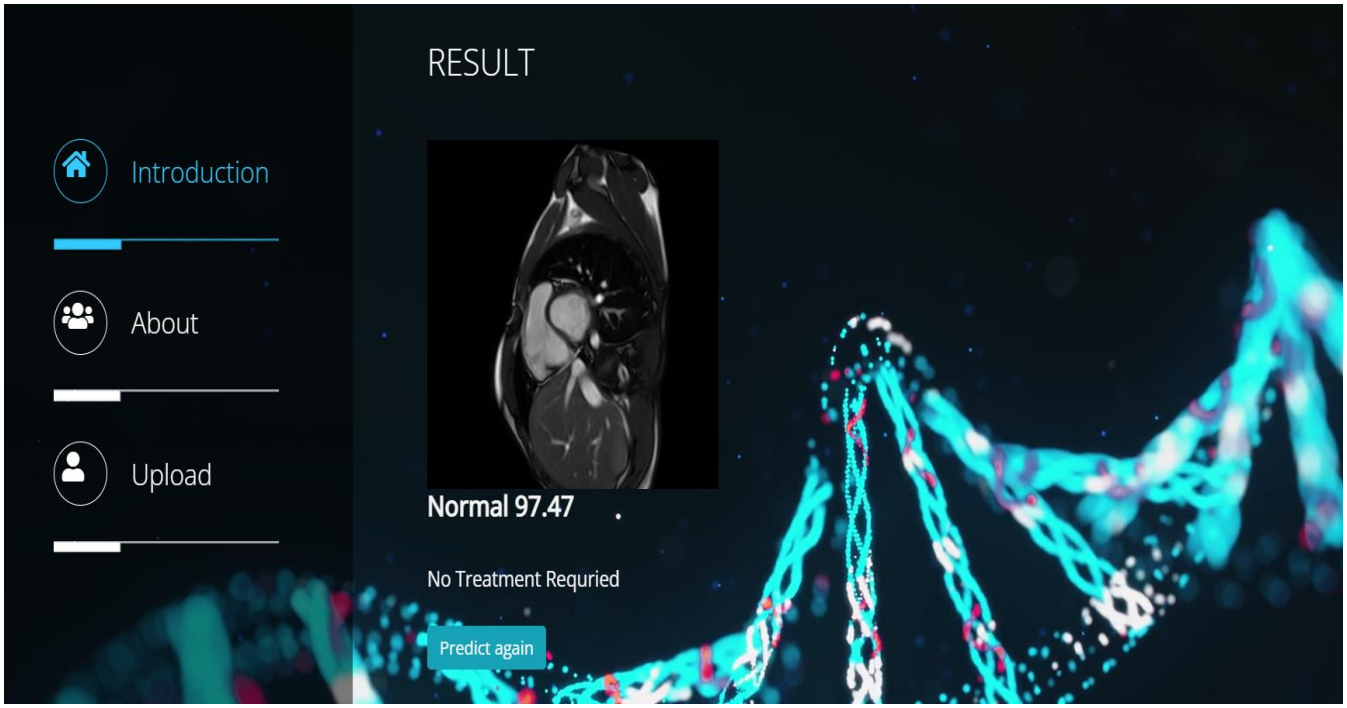
In this page, user can upload the image to predict the disease.

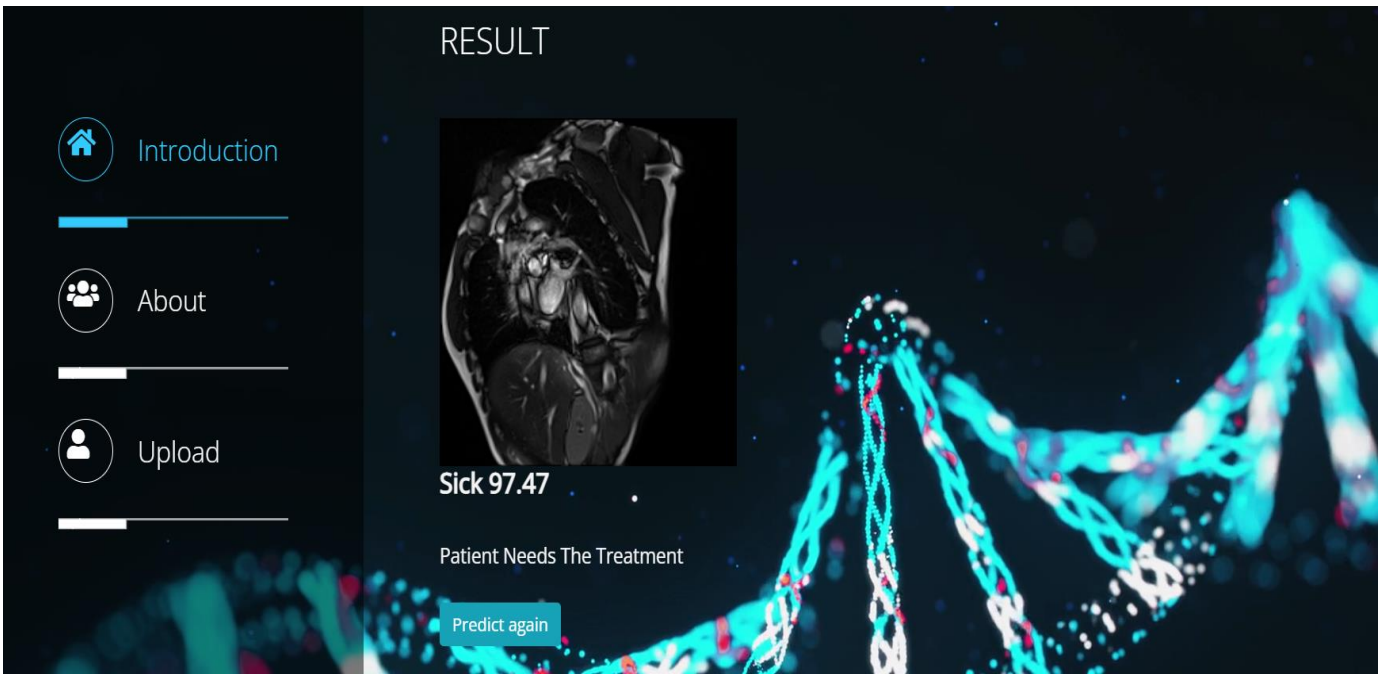




Result Page

In this page, the result will be shown. It shows whether the patients require the treatment or not.





Conclusion

This study takes similar route, but with an improved and novel method and with a larger dataset for training the model. This research demonstrates that the Relief feature selection algorithm can provide a tightly correlated feature set which then can be used with several machine learning algorithms. The study has also identified that RFBM works particularly well with the high impact features and produces an accuracy, substantially higher than related work. RFBM achieved a best accuracy with 13 features. Cardiovascular disease is used to determine whether or not a patient is at risk of having a heart attack.