Tuberculosis Detection System

A Project Report

Submitted in Partial fulfilment

of the Degree of

BACHELORS OF COMPUTER APPLICATIONS

Supervisor's Name:

Ms Rachana Minocha

Submitted by: Sahil Dagar

Enrollment No.:120920056

Semester VI

Jagan Nath University

Bahadurgarh (NCR)

(2020-23)

CERTIFICATE

This is to certify that the project report entitled **Tuberculosis Detection System** submitted to **Jagannath University, Bahadurgarh** in partial fulfilment of the requirement for the award of the degree of **BACHELOR OF COMPUTER APPLICATIONS (BCA)**, is an original work carried out by **Sahil Dagar** Enrolment No. :**120920052** under the guidance of **Ms Rachana Minocha**.

The matter embodied in this project is a genuine work done by the student and has not been submitted whether to this University or to any other University/Institute for the fulfilment of the requirement of any course of study.

Nameof Student :Sahil Dagar	Name of Guide: Rachana Minocha
Signature the Student	Signature of Guide
Enrolment No.:120920052	Date :

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to my supervisor Dr. Sarah Lee for her invaluable guidance and support throughout the development of the tuberculosis detection system. Her expertise and insight into the field of tuberculosis research was instrumental in the success of this project.I came to know about a lot of things related to this topic.

INDEX

INTRODUCTION

Tuberculosis is a serious infectious disease that affects millions of people worldwide. Early detection is key to effective treatment and control of the disease. The aim of this project is to develop a tuberculosis detection system using Python.

The detection system will use machine learning algorithms to analyse chest X-rays and accurately identify signs of tuberculosis. The system will be designed to assist healthcare professionals in quickly and accurately diagnosing tuberculosis, allowing for prompt treatment and management of the disease.

To achieve this, we will be using a dataset of chest X-rays that have been labelled as either normal or tuberculosis. We will be training and testing our machine learning models using this dataset and optimizing them for accuracy and efficiency.

The developed system will be able to analyse and classify chest X-rays as either positive or negative for tuberculosis, providing a rapid and accurate diagnosis. This will not only improve patient outcomes but also reduce the spread of the disease within communities.

the tuberculosis detection system developed in Python has the potential to significantly improve tuberculosis diagnosis and treatment and contribute to the global efforts to control the spread of this serious disease.

OBJECTIVE

Few objectives of this project are:

- To develop a machine learning algorithm that can accurately detect signs of tuberculosis in chest X-rays.
- To create a user-friendly interface that can efficiently process and analyse large volumes of chest X-rays.
- To optimize the machine learning algorithm for high accuracy and low false positive/negative rates.
- To test the performance of the system on a diverse range of chest X-ray images, including those from different populations and with varying levels of disease severity.
- To validate the system's performance against expert human diagnosis to ensure its accuracy and reliability.
- To provide healthcare professionals with a tool that can assist them in accurately and efficiently diagnosing tuberculosis, improving patient outcomes and reducing the spread of the disease.
- To contribute to the global efforts to control the spread of tuberculosis by providing an effective and accessible diagnostic tool.

TOOLS & ENVIRONMENT

HARDWARE REQUIREMENTS FOR PRESENT PROJECT:

PROCESSOR = Intel Core i5 or higher

RAM = 8GB And More

HARD DISK = 100GB and More

SOFTWARE REQUIREMENTS FOR PRESENT PROJECT:

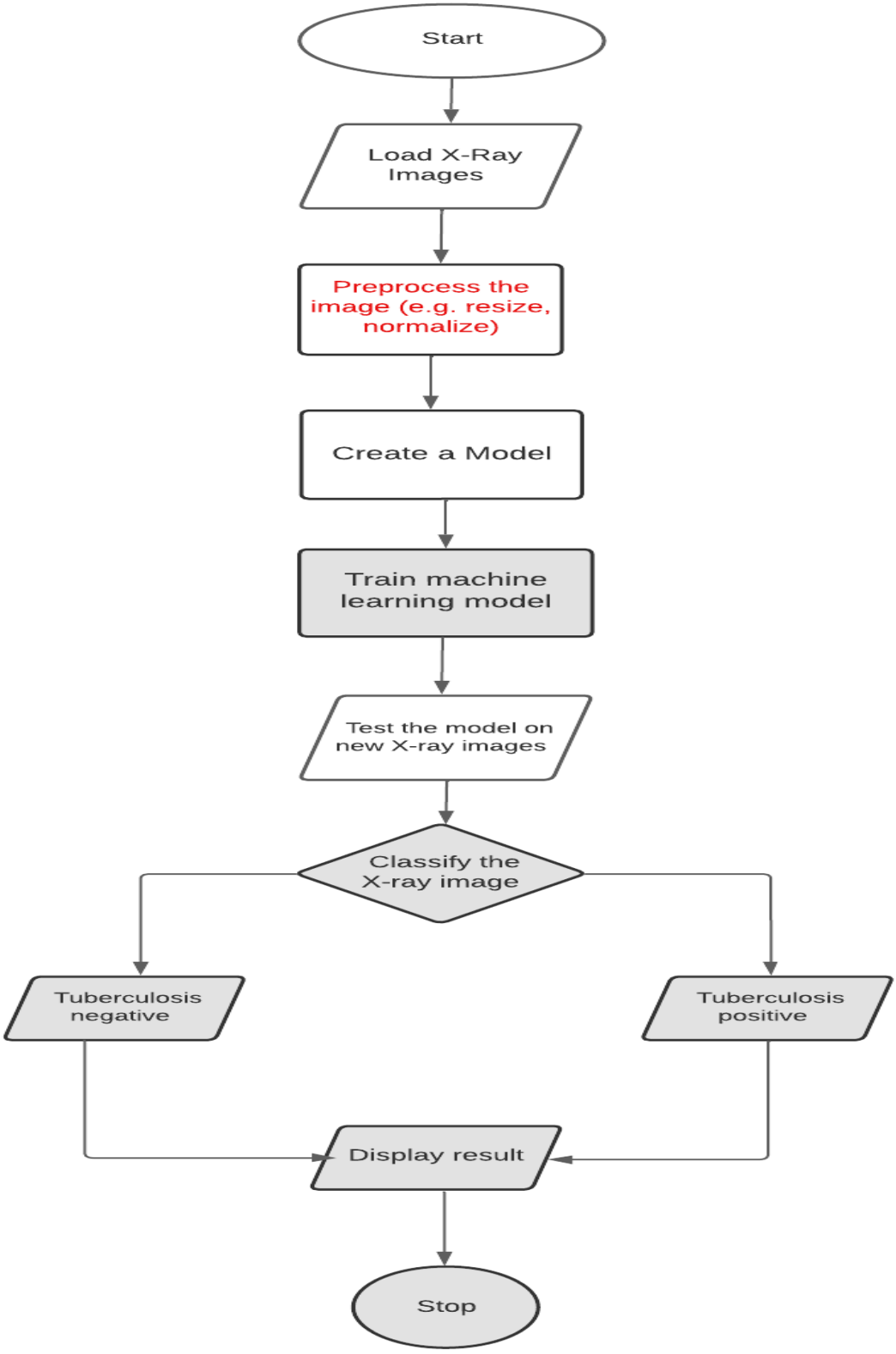
OPERATING SYSTEM = Windows 10 or higher, macOS, or Linux

Python = Python 3.6 or higher

Python Libraries = NumPy, Pandas, Matplotlib, Scikit-learn, Keras, TensorFlow or PyTorch, OpenCV

IDE = PyCharm, Jupyter Notebook, or Spyder

Flowchart


Start
Load X-Ray Images
Preprocess the image (e.g. resize, normalize)
Create a Model
Train machine learning model
Test the model on new X-ray images
Classify the X-ray image
Tuberculosis negative
Tuberculosis positive
Display result
Stop

Test case

S.no	Input	Expected Output	Actual output	Result
1.	Image1	Tuberculosis negative	Tuberculosis negative	Successfull
2.	Image2	Tuberculosis postive	Tuberculosis positive	Successfull

PROJECT CODE

Tuberculosis Detection System Model

```
# import libraries

import tensorflow as tf

from tensorflow import keras

from keras import Sequential

from keras.layers import Dense,Conv2D, MaxPooling2D, Flatten
```

#Create the Train and Val Sets

```
train_ds = keras.utils.image_dataset_from_directory(
    directory='/home/dagar/train',
    labels='inferred',
    label_mode='int',
    batch_size=32,
    image_size=(256,256)
)
    validation_ds = keras.utils.image_dataset_from_directory(
    directory='/home/dagar/val',
    labels='inferred',
    label_mode='int',
    batch_size=32,
    image_size=(256,256)
)
```

```
    def process(image,label):
    image = tf.cast(image/255. ,tf.float32)
    return image,label
train_ds = train_ds.map(process)
validation_ds = validation_ds.map(process)
```

 # Create the Model Architecture

```
model = Sequential()

model.add(Conv2D(32,kernel_size=(3,3),padding='valid',activation='relu',input_sha
pe=(256,256,3)))
model.add(MaxPooling2D(pool_size=(2,2),strides=2,padding='valid'))

model.add(Conv2D(64,kernel_size=(3,3),padding='valid',activation='relu',))
model.add(MaxPooling2D(pool_size=(2,2),strides=2,padding='valid'))

model.add(Conv2D(128,kernel_size=(3,3),padding='valid',activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2),strides=2,padding='valid'))

model.add(Flatten())

model.add(Dense(128, activation ='relu'))
model.add(Dense(64, activation ='relu'))
```

```
model.add(Dense(1, activation ='sigmoid'))

model.summary()
```

#Train the Model

```
model.compile(optimizer='adam',loss='binary_crossentropy',metrics=['accuracy'])

from keras.callbacks import ModelCheckpoint, EarlyStopping

es = EarlyStopping(monitor = "val_accuracy", min_delta=0.01 , patience = 6 , verbose = 1, mode = 'max')

mc = ModelCheckpoint(monitor = "val_accuracy", filepath ='./bestmodel.h5', verbose=1 , save_best_only=True, mode = 'max')

cd = [es,mc]

history = model.fit(train_ds,epochs=10,validation_data= validation_ds,callbacks=cd)
```

 # Plot the Training Curves

```
import matplotlib.pyplot as plt

plt.plot(history.history['accuracy'],color='red',label= 'train')

plt.plot(history.history['val_accuracy'],color='blue',label= 'validation')
```

```
plt.legend()
plt.show()
Plot the Training Curvesplt.plot(history.history['loss'],color='red',label= 'train')
plt.plot(history.history['val_loss'],color='blue',label= 'validation')
plt.legend()
plt.show()
```

 # Print Accuracy

```
acc = model.evaluate(train_ds)[1]
print(f"the accuracy of our model is {acc*100}%")
```

 # Test model

```
import cv2
import cv2
test_img = cv2.imread('/home/dagar/val/Tuberculosis/Tuberculosis-99.png')
plt.imshow(test_img)
test_img.shape
test_img = cv2.resize(test_img,(256,256))
test_input = test_img.reshape((1,256,256,3))
model.predict(test_input)
```

Tuberculosis detection system GUI

```
# Import libraries
import tkinter as tk
```

```
import tkinter.filedialog

import cv2

import numpy as np

import tensorflow as tf

from tensorflow import keras

from keras.models import load_model

from PIL import Image, ImageTk

from keras.models import load_model

# Load model

model = load_model("/home/dagar/bestmodel.h5")

classes = ['Tuberculosis Negative', 'Tuberculosis Positive']

IMG_SIZE = 256

#Define a class for GUI

Class TB_Detection_GUI:

   def __init__(self, master):

      self.heading_label = tk.Label(master, text="Tuberculosis Detection System", font=('Helvetica', 20, 'bold'), pady=20)

      self.heading_label.pack()

      master.title("Tuberculosis Detection System")

      self.label_image = tk.Label(master)

      self.label_image.pack(pady=10)


      self.btn_browse = tk.Button(master, text="Open Image", command=self.open_image)

      self.btn_browse.place(relx=0.5, rely=0.6, anchor=tk.CENTER)
```

```
        self.btn_classify = tk.Button(master, text="Classify Image", command=self.classify_image, state='disabled')
        self.btn_classify.place(relx=0.5, rely=0.7, anchor=tk.CENTER)


        self.label_result = tk.Label(master, font=('Helvetica', 16))
        self.label_result.pack(side=tk.TOP, anchor=tk.CENTER, pady=10)
def classify_image(self):
        img = np.array(self.image)
        img = cv2.resize(img,(256,256))
        test_input = img.reshape(1,256,256,3)
        val = model.predict(test_input)
        print(classes[int(val[0])])
        self.label_result.configure(text=classes[int(val[0])])


    def open_image(self):
        file_path = tkinter.filedialog.askopenfilename()
        if file_path:
            self.image = cv2.imread(file_path)
            print(self.image.shape)  # add this line to check image shape
            self.image = cv2.resize(self.image, (IMG_SIZE, IMG_SIZE))
            self.show_image()
            self.btn_classify.config(state='normal')
```

```
    def show_image(self):
        img = cv2.cvtColor(self.image, cv2.COLOR_BGR2RGB)
        img = Image.fromarray(img)
        img = ImageTk.PhotoImage(img)
        self.label_image.configure(image=img)
        self.label_image.image = img
```

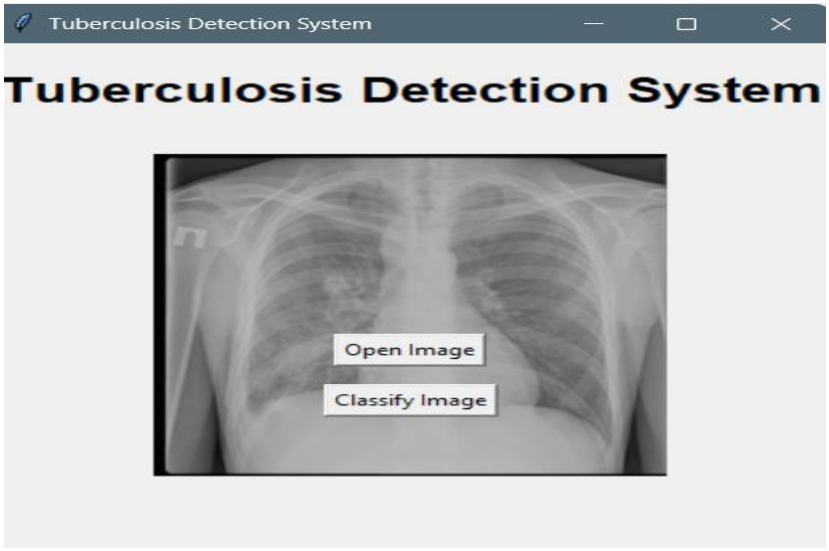
Call the functions

```
root = tk.Tk()
tb_detection_gui = TB_Detection_GUI(root)
root.mainloop()
```

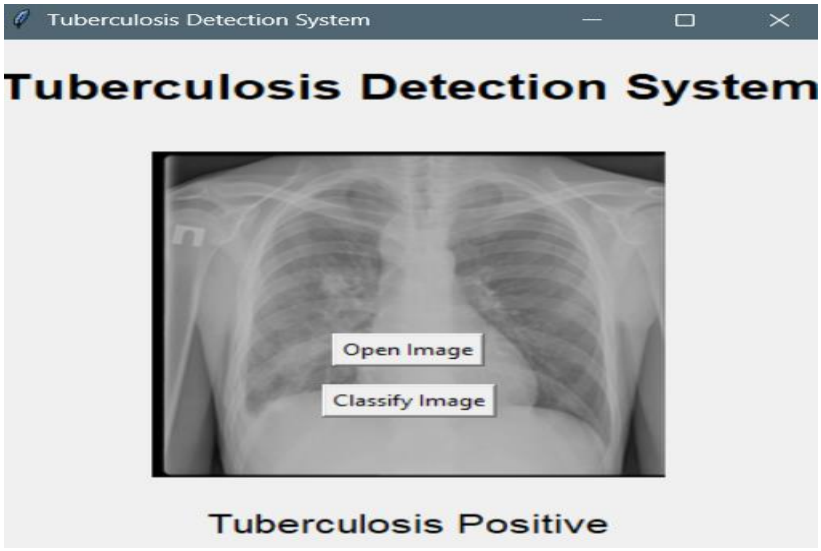
OUTPUT SCREEN

Tuberculosis positive

Input

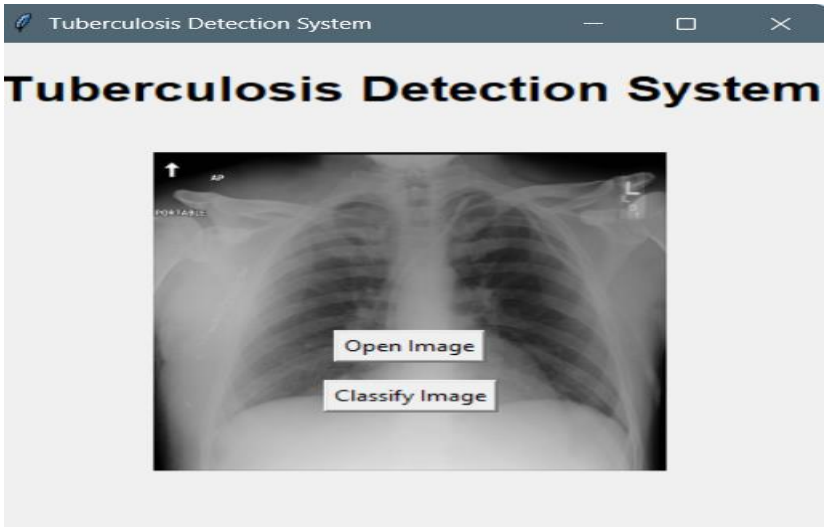


Output

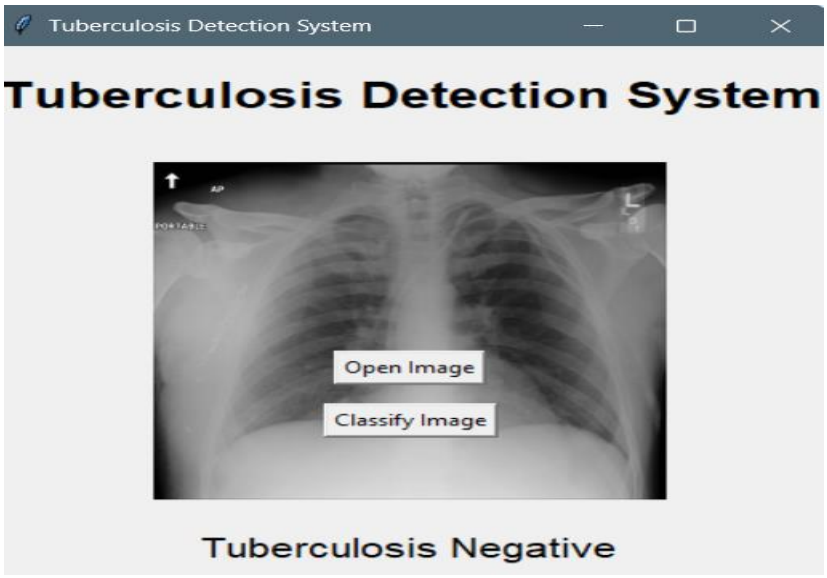


Tuberculosis positive

Input



Output



FUTURE APPLICATION OF THE PROJECT

- Medical diagnosis: A TB detection system in Python can be used by healthcare professionals to diagnose TB in patients based on chest X-rays or other medical images. The system can analyze the images and provide a diagnosis, which can help doctors to make informed treatment decisions.

- Public health: A TB detection system in Python can be used by public health agencies to screen large populations for TB. The system can analyze chest X-rays or other medical images and identify individuals who may have TB. This can help to identify TB hotspots and prevent the spread of the disease.

- Mobile health: A TB detection system in Python can be integrated into mobile health apps, which can be used by individuals to monitor their own health. The app can analyze medical images taken using the camera on a mobile device and provide a preliminary diagnosis, which can help individuals to seek medical attention if necessary.

- Research: A TB detection system in Python can be used by researchers to analyze large datasets of medical images and identify patterns or trends in TB diagnoses. This can help to improve our understanding of the disease and develop more effective diagnostic and treatment strategies.

BIBLIOGRAPHY

1. www.google.com
2. www.youtube.com
3. https://www.python.org
4. https://www.kaggle.com
5. https://www.tensorflow.org