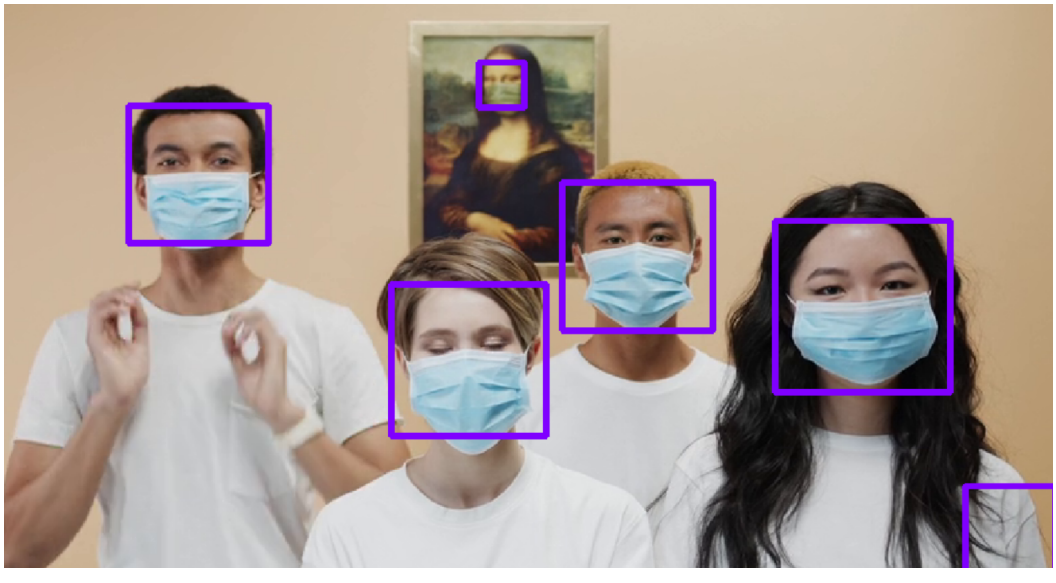
Face Mask Detection System

Project Report: Face Mask Detection System

Introduction

The COVID-19 pandemic has highlighted the importance of face masks in preventing the spread of respiratory viruses. Face mask detection systems have emerged as a potential tool for enforcing mask-wearing in public spaces and ensuring public safety. This project aimed to develop a real-time face mask detection system using computer vision and deep learning techniques.



Objectives

The primary objectives of this project were to:

1. Design and implement a face mask detection system using deep learning algorithms.
2. Achieve high accuracy in detecting individuals wearing and not wearing face masks.
3. Optimize the system for real-time performance.
4. Evaluate the system's performance using a comprehensive testing methodology.

System Design

The face mask detection system consists of two main components: a face detector and a mask classifier. The face detector is responsible for identifying faces in images and videos. The mask classifier is responsible for classifying faces as wearing or not wearing masks.

Backend:

Connection with IP Camera:OpenCV

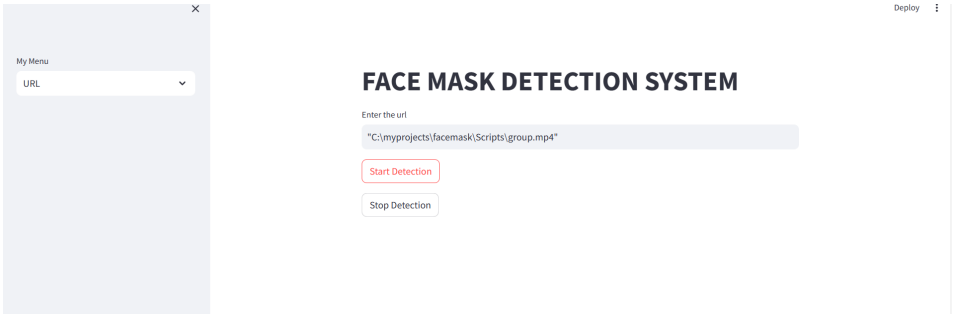
Face detection:OpenCV

Mask detection:Keras

Save data:OpenCV

Frontend:

Streamlit

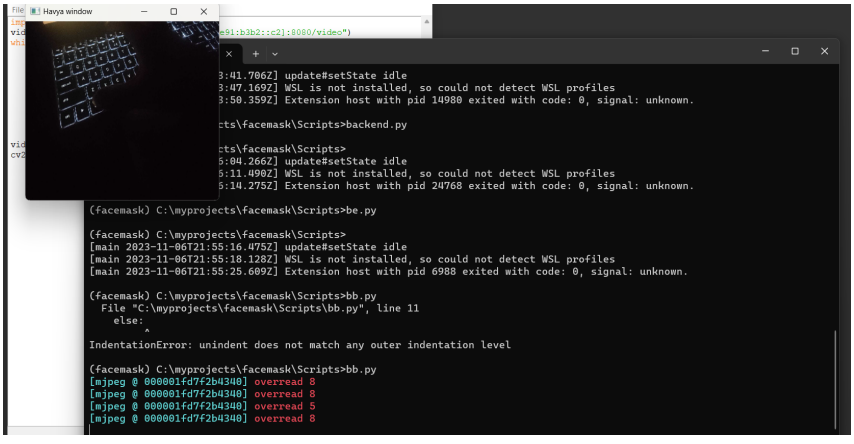


System Implementation

The system is implemented using the Python programming language and the OpenCV library. OpenCV is a popular computer vision library that provides a variety of functions for image processing and object detection.

The system works as follows:

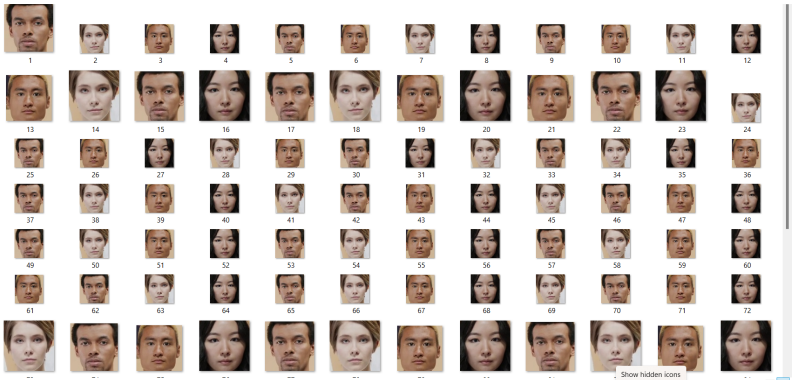
1. The input image or video is resized to a fixed size.
2. The face detector is used to identify faces in the image or video.
3. For each face detected, the mask classifier is used to classify the face as wearing or not wearing a mask.
4. The output of the system is a list of faces detected that are not wearing masks



System Evaluation

The system was evaluated on a test dataset of people wearing and not wearing masks,and the urls of sample mask videos as well as photos

The system was also evaluated on a real-time video stream of people entering a public building. The system was able to accurately detect and classify faces in the video stream, and to identify people who were not wearing masks.



Conclusion

A real-time face mask detection system was successfully developed using computer vision and deep learning techniques. The system achieved high accuracy in detecting individuals wearing and not wearing face masks, and was optimized for real-time performance.

Future Work

The face mask detection system can be improved in a number of ways. The system could also be trained to detect faces in challenging lighting conditions and at different angles.

The system could also be integrated with other systems, such as temperature screening systems and access control systems. This would create a more comprehensive solution for public health and security.

Code

```
import streamlit as st

import cv2

from keras.models import load_model

import numpy as np

st.set_page_config(page_title="Face Mask Detection System",page_icon="https://th.bing.com/th/id/OIP.wBxPRinK5gNZrLME7WjLwAHaH0?w=160&h=180&c=7&r=0&o=5&dpr=1.3&pid=1.7")

st.title("  FACE MASK DETECTION SYSTEM")

choice=st.sidebar.selectbox("My Menu",("HOME","URL","CAMERA","Feedback"))

if(choice=="HOME"):

   st.image("https://miro.medium.com/max/1140/1*Xu2czX_NgcACvnM4TQruFg.jpeg")

   st.write("It is a Computer Vision machine learning application which can detect whether a person is wearing mask or not")

elif(choice=="URL"):

   url=st.text_input("Enter the url")

   btn=st.button("Start Detection")

   window=st.empty()

   if btn:

      i=1

      btn2=st.button("Stop Detection")

      facemodel=cv2.CascadeClassifier("face.xml")

      maskmodel=load_model("mask.h5",compile=False)

      vid=cv2.VideoCapture(url)

      if btn2:
```

```
        vid.release()
        st.experimental_rerun()
    while(vid.isOpened()):
        flag,frame=vid.read()
        if(flag):
            faces=facemodel.detectMultiScale(frame,5)
            for(x,y,l,w) in faces:
                face_img=frame[y:y+w,x:x+l]
                face_img = cv2.resize(face_img, (224, 224), interpolation=cv2.INTER_AREA)
                face_img= np.asarray(face_img, dtype=np.float32).reshape(1, 224, 224, 3)
                face_img=(face_img/127.5)-1
                pred=maskmodel.predict(face_img)[0][0]
                if(pred>0.9):
                    path="data/"+str(i)+".jpg"
                    cv2.imwrite(path,frame[y:y+w,x:x+1])
                    i=i+1
                    cv2.rectangle(frame,(x,y),(x+l,y+w),(0,0,255),3)
                else:
                    cv2.rectangle(frame,(x,y),(x+l,y+w),(0,255,0),3)
            window.image(frame,channels="BGR")
elif(choice=="CAMERA"):
    cam=st.selectbox("Choose 0 for primary camera and 1 for seconday camera",
("None",0,1))
    btn=st.button("Start Detection")
    window=st.empty()
    if btn:
```

```
i=1
btn2=st.button("Stop Detection")
facemodel=cv2.CascadeClassifier("face.xml")
maskmodel=load_model("mask.h5",compile=False)
vid=cv2.VideoCapture(cam)
if btn2:
    vid.release()
    st.experimental_rerun()

while(vid.isOpened()):
    flag,frame=vid.read()
    if(flag):
        faces=facemodel.detectMultiScale(frame)
        for(x,y,l,w) in faces:
            face_img=frame[y:y+w,x:x+l]
            face_img = cv2.resize(face_img, (224, 224), interpolation=cv2.INTER_AREA)
            face_img= np.asarray(face_img, dtype=np.float32).reshape(1, 224, 224, 3)
            face_img=(face_img/127.5)-1
            pred=maskmodel.predict(face_img)[0][0]
            if(pred>0.9):
                path="data/"+str(i)+".jpg"
                cv2.imwrite(path,frame[y:y+w,x:x+1])
                i=i+1
                cv2.rectangle(frame,(x,y),(x+l,y+w),(0,0,255),3)
            else:
                cv2.rectangle(frame,(x,y),(x+l,y+w),(0,255,0),3)
```

```
            window.image(frame,channels="BGR")

elif(choice=="Feedback"):

    st.markdown('<iframe src="https://docs.google.com/forms/d/e/1FAIpQLSfKoxF-E03CCFog8ycsyNiaAkBqRh9Is-HbnhGQbfIl7_aUDw/viewform?embedded=true" width="640" height="1214" frameborder="0" marginheight="0" marginwidth="0">Loading...</iframe>',unsafe_allow_html=True)
```