SmartBridge

IoT Industrial Internship

Face Mask Detection System based access control system for shopping malls

(Project Report)

Guided by

Mr. Gnaneshwar Bandari

Project by

Gowtham B (gowtham.18bce7143@vitap.ac.in)

H Subramani (subramani.18bcd7181@vitap.ac.in)

K Viswanath Naveen (viswanathnaveen3@gmail.com)

M Anil Kumar (anil.18bcd7141@vitap.ac.in)

1. Introduction

a. Overview

The rising of COVI-19 cases challenged the world in many aspects. Every country has taken many measures to prevent the COVID-19 cases. As we are all aware the spread of COVID is high when people refuse to wear the mask and follow social distance. Because of this social distancing and wearing face masks became one of the important measures taken against the COVID-19. Though few people followed the COVID precautions strictly, few refused to wear them because of breathing issues, physical looks and not interest in hiding their face in the public.

b. Purpose

At beginning of this pandemic, there were 10 million cases recorded within a year. To overcome this situation many countries utilized the available technologies like IoT, AI, Big Data, Machine Learning to ensure the safety of people in a crowded place and keep track of the covid patient primary, secondary contacts and so on. Since technology can't provide a complete solution, there's a lot of responsibilities on society to follow precautions and safeguard themselves, their families and their country. Hence, people are forced by laws to wear face masks in public.

2. Literature Survey

a. Existing Problem:

Today, life the midst of the pandemic have become really hard for many people. There are many safety rules and precautions we need to adhere to. The shop keepers and malls really struggle to ensure their customers follow rules and wear face mask. Many malls and shops inside are even fined and closed because of their customers not wearing face mask. So having an automatic face mask detection system in these malls and shops will help them to focus more on their business and ensure rules and followed by everyone.

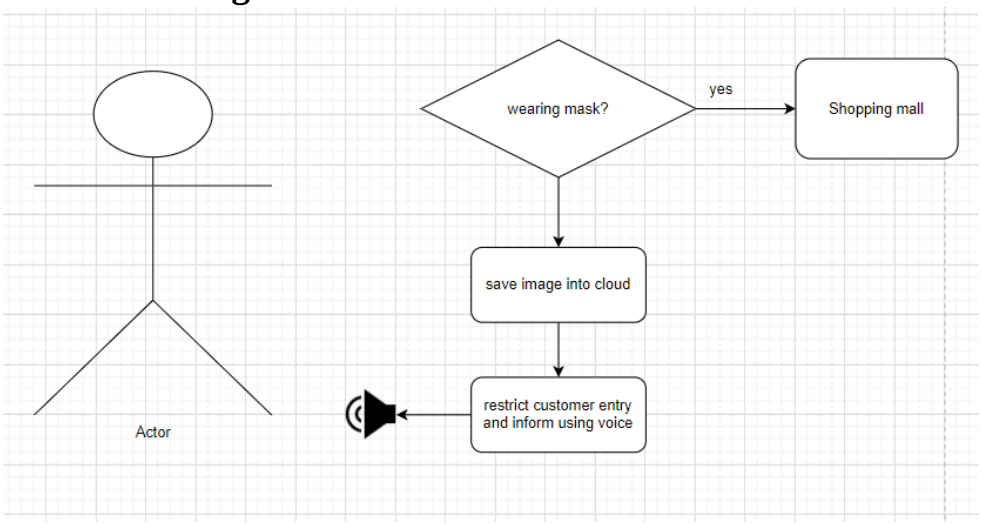
b. Proposed Solution:

Here, we use technologies like python, opencv, IBM Cloud and Node Red to develop a face mask detection system for malls. In this system, if a person enters a mall without wearing a face mask, he automatically gets an alarm saying to wear face mask and also the person's face not wearing a face mask

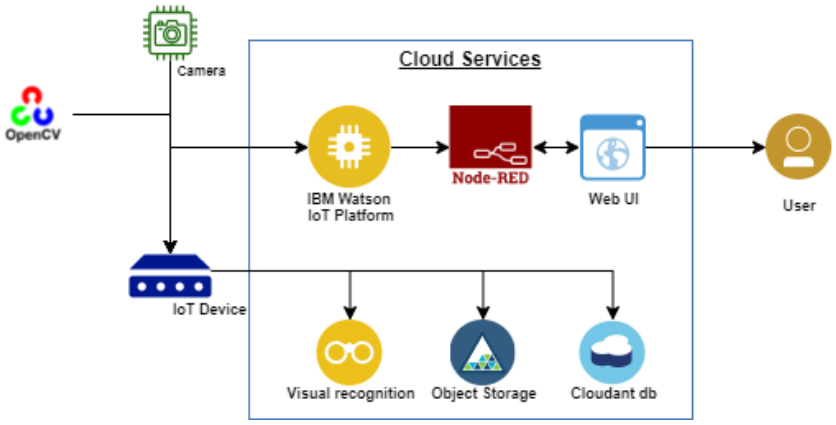
will be sent and notified to the admin so he can monitor him and advise to follow safety rules.

3. Theoretical Analysis

a. Block diagram



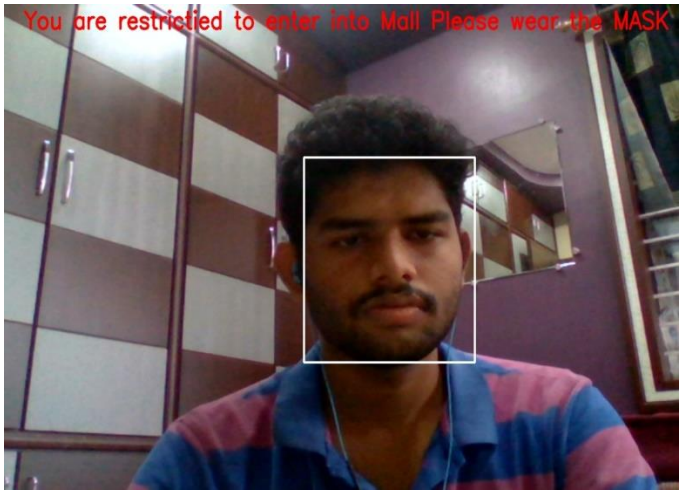
b. Software Design



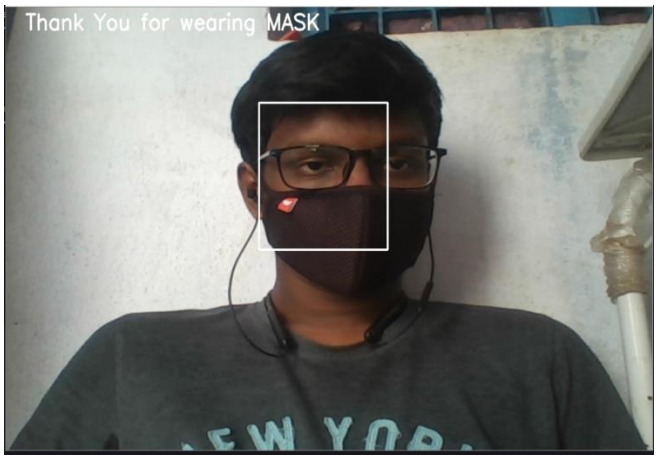
4. Flowchart

5. # Result

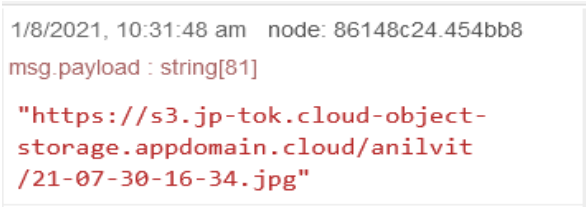
a. Without Mask in OpenCV



b. With mask in OpenCV

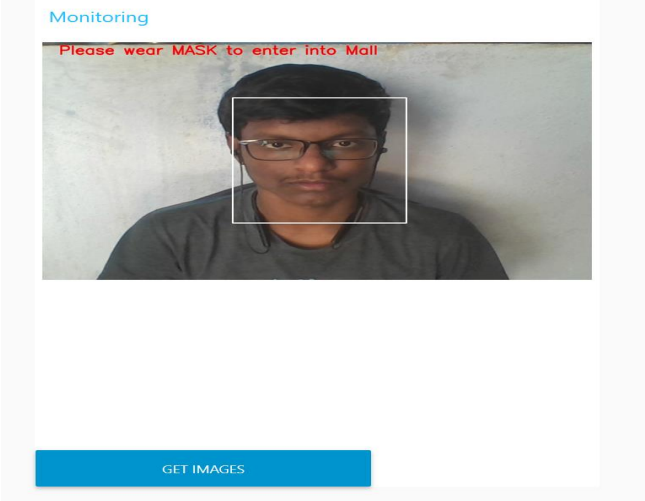


c. Node debug Output



d. CloudantDB data using HTTP request

e. Web UI



6. Advantages and Disadvantages

The proposed system will help malls and shops to focus on their business instead worrying about safety measures followed by customers. In a place where hundreds of people come and go, its hard to manually check if people are wearing face masks. So this system solves this problem by automatically checking if people are wearing mask. It also sends photo to people not wearing the mask to the admin so he/she can make sure the person is taken care.

The proposed system also has a few disadvantages. Firstly, since we use something similar to facial recognition, we are not implementing any measures to ensure safety and privacy of people. Secondly since the system detects face masks in live time, there is some delay in detection and notification to admin. Also some fancy masks are less accurately detected.

7. Applications:

The system can be used to detected people not wearing masks in malls where a lot of people come and go and it makes it easy to check if people are wearing mask instead of manually checking. It makes things easy of people running malls and shopkeepers in it. Also the system sends photos of people not wearing mask to the admin automatically.

8. Conclusion:

Today a lot of AI and cloud based tools are used to make life easier for us in the midst of this pandemic and this is one such tool which will hugely benefit the stakeholders. The system achieves maximum accuracy and can help malls ensure people wear masks and follow. Hence if its deployed and scaled in real time, it can make a huge impact and it will also make more and more people to follow the rules and wear masks.

9. Future Scope:

The system can be improved to ensure privacy and safety concerns are taken into account. Also we can further expand to also see if people are socially distanced also. Using robust scaling techniques can also make the system work faster and reduce any delay between detecting and notification. It can also be applied to many other scenarios other than malls, like in parks, roads, railway stations and airports where many people come and go and its hard to manually check if people are following rules or not.

10. Bibliography:

https://docs.opencv.org/4.5.3/
https://docs.python.org/3/
https://realpython.com/face-recognition-with-python/
https://www.pyimagesearch.com/2020/05/04/covid-19-face-mask-detector-with-opencv-keras-tensorflow-and-deep-learning/
https://cloud.ibm.com/

11. Appendix

a. Source code:

Python:

```
import numpy as np
import cv2
import random
import datetime
import ibm_boto3
from ibm_botocore.client import Config, ClientError
from ibmcloudant.cloudant_v1 import CloudantV1
from ibmcloudant import CouchDbSessionAuthenticator
from ibm_cloud_sdk_core.authenticators import BasicAuthenticator
#Text to Speech Conversion
from ibm_watson import TextToSpeechV1
from ibm_cloud_sdk_core.authenticators import IAMAuthenticator
import playsound
# Constants for IBM COS values
```

```
COS_ENDPOINT = "https://s3.jp-tok.cloud-object-storage.appdomain.cloud" #
        Current list avaiable at https://control.cloud-object-
        storage.cloud.ibm.com/v2/endpoints
COS_API_KEY_ID = "GUWZxUx1joJG-llklaaRqQ1IuiA-YBzJ8gYfsCd7mFKK"
        # eg "W00YixxxxxxxxxxMB-odB-2ySfTrFBIQQWanc--P3byk"
COS_INSTANCE_CRN = "crn:v1:bluemix:public:cloud-object-
        storage:global:a/06b7b01ca0b641caabedc1fedacea9f8:02cda9ba-fd76-41b7-
        a26f-2498dac7f9ae::"
# Create resource
cos = ibm_boto3.resource("s3",
    ibm_api_key_id=COS_API_KEY_ID,
    ibm_service_instance_id=COS_INSTANCE_CRN,
    config=Config(signature_version="oauth"),
    endpoint_url=COS_ENDPOINT
)

authenticator = BasicAuthenticator('apikey-v2-
        2wv8tuh4z6vre8dp3xtcg53padwxovi1vcbmonu59csv',
        'ab2ed6687b575ddd7f6201c6f1694701')
service = CloudantV1(authenticator=authenticator)
service.set_service_url('https://apikey-v2-
        2wv8tuh4z6vre8dp3xtcg53padwxovi1vcbmonu59csv:ab2ed6687b575ddd7f62
        01c6f1694701@437c4eb8-b45c-4a02-bc43-5eb60a35022c-
        bluemix.cloudantnosqldb.appdomain.cloud')
bucket = "anilvit"
def multi_part_upload(bucket_name, item_name, file_path):
    try:
        print("Starting file transfer for {0} to bucket: {1}\n".format(item_name,
          bucket_name))
        # set 5 MB chunks
        part_size = 1024 * 1024 * 5
        # set threadhold to 15 MB
        file_threshold = 1024 * 1024 * 15
        # set the transfer threshold and chunk size
        transfer_config = ibm_boto3.s3.transfer.TransferConfig(
            multipart_threshold=file_threshold,
            multipart_chunksize=part_size
        )
        # the upload_fileobj method will automatically execute a multi-part upload
        # in 5 MB chunks for all files over 15 MB
        with open(file_path, "rb") as file_data:
            cos.Object(bucket_name, item_name).upload_fileobj(
                Fileobj=file_data,
                Config=transfer_config
            )
        print("Transfer for {0} Complete!\n".format(item_name))
```

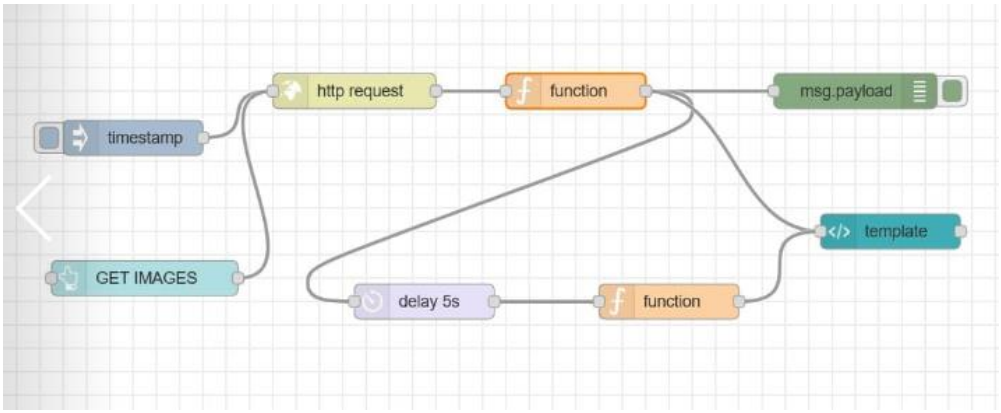
```
    except ClientError as be:
        print("CLIENT ERROR: {0}\n".format(be))
    except Exception as e:
        print("Unable to complete multi-part upload: {0}".format(e))
authenticator =
        IAMAuthenticator('90ZND2exsa6Qa96XDSMbt4zilsWbZzxP0CZMCUN3M
        VBD')
text_to_speech = TextToSpeechV1(
    authenticator=authenticator
)
text_to_speech.set_service_url('https://api.eu-gb.text-to-
        speech.watson.cloud.ibm.com/instances/29fa5bbf-695b-440b-a597-
        ff8b094a4abe')
# multiple cascades: https://github.com/Itseez/opencv/tree/master/data/haarcascades
face_cascade = cv2.CascadeClassifier('haarcascade_frontalface_default(1).xml')
eye_cascade = cv2.CascadeClassifier('haarcascade_eye.xml')
mouth_cascade = cv2.CascadeClassifier('haarcascade_mcs_mouth.xml')
upper_body = cv2.CascadeClassifier('haarcascade_upperbody.xml')
# Adjust threshold value in range 80 to 105 based on your light.
bw_threshold = 80
# User message
font = cv2.FONT_HERSHEY_SIMPLEX
org = (30, 30)
weared_mask_font_color = (255, 255, 255)
not_weared_mask_font_color = (0, 0, 255)
thickness = 2
font_scale = 1
weared_mask = "Thank You for wearing MASK"
not_weared_mask = "Please wear MASK to enter into Mall"
# Read video
cap = cv2.VideoCapture(0,cv2.CAP_DSHOW)
while 1:
    # Get individual frame
    ret, img = cap.read()
    #img = cv2.flip(img,1)
    img=cv2.resize(img,(1000,600))
    # Convert Image into gray
    gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
    # Convert image in black and white
    (thresh, black_and_white) = cv2.threshold(gray, bw_threshold, 255,
        cv2.THRESH_BINARY)
    #cv2.imshow('black_and_white', black_and_white)
    # detect face
    faces = face_cascade.detectMultiScale(gray, 1.1, 4)
    # Face prediction for black and white
    faces_bw = face_cascade.detectMultiScale(black_and_white, 1.1, 4)
```

```
if(len(faces) == 0 and len(faces_bw) == 0):
    cv2.putText(img, "No face found...", org, font, font_scale,
     weared_mask_font_color, thickness, cv2.LINE_AA)
elif(len(faces) == 0 and len(faces_bw) == 1):
    # It has been observed that for white mask covering mouth, with gray image
     face prediction is not happening
    cv2.putText(img, weared_mask, org, font, font_scale,
     weared_mask_font_color, thickness, cv2.LINE_AA)
else:
    # Draw rectangle on gace
    for (x, y, w, h) in faces:
        cv2.rectangle(img, (x, y), (x + w, y + h), (255, 255, 255), 2)
        roi_gray = gray[y:y + h, x:x + w]
        roi_color = img[y:y + h, x:x + w]
        # Detect lips counters
        mouth_rects = mouth_cascade.detectMultiScale(gray, 1.5, 5)
    # Face detected but Lips not detected which means person is wearing mask
    if(len(mouth_rects) == 0):
        cv2.putText(img, weared_mask, org, font, font_scale,
     weared_mask_font_color, thickness, cv2.LINE_AA)
    else:
        for (mx, my, mw, mh) in mouth_rects:
            if(y < my < y + h):
                 # Face and Lips are detected but lips coordinates are within face
     cordinates which `means lips prediction is true and
                # person is not waring mask
                cv2.putText(img, not_weared_mask, org, font, font_scale,
     not_weared_mask_font_color, thickness, cv2.LINE_AA)
                # here tts.wav is our file name and wb is write byte by byte in particular
     file as audio file as type it is python file handling procedure
                picname=datetime.datetime.now().strftime("%y-%m-%d-%H-%M")
                cv2.imwrite(picname+".jpg",img)
                multi_part_upload(bucket, picname+'.jpg', picname+'.jpg')

     json_document={"link":COS_ENDPOINT+'/'+bucket+'/'+picname+'.jpg'}
                response = service.post_document(db='mask-detection',
     document=json_document).get_result()
                print('Playing.....')
                playsound.playsound('mall.mp3')
                print('stopped!')
                #cv2.rectangle(img, (mx, my), (mx + mh, my + mw), (0, 0, 255),
                break
# Show frame with results
cv2.imshow('Mask Detection', img)
if cv2.waitKey(1) & 0xFF == ord('q'):
    break
```

```
# Release video
cap.release()
cv2.destroyAllWindows()
```

Node RED Flow



b. UI Output