19BAI10017

SHANZEH BATOOL

ASSIGNMENT 6

Q. Develop a python code to detect any object using Haar cascade classifier.

```
import cv2
import numpy as np
import datetime


#ObjectStorage
import ibm_boto3
from ibm_botocore.client import Config, ClientError

#CloudantDB
from cloudant.client import Cloudant
from cloudant.error import CloudantException
from cloudant.result import Result, ResultByKey
import requests

face_classifier=cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
eye_classifier=cv2.CascadeClassifier("haarcascade_eye.xml")


# 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 = "_08kgDODk4BV2-e0fsG5SX-cbLuIAkweyYLqf1gQny62o" # eg "W00YiRnLW4a3fTjMB-odB-2ySfTrFBIQQWanc--P3byk"

COS_AUTH_ENDPOINT = "https://iam.cloud.ibm.com/identity/token"

COS_RESOURCE_CRN = "crn:v1:bluemix:public:cloud-object-storage:global:a/b03790c764f74139bbf0115be6990c81:4ae1d3b9-2de4-408a-a343-4536b529d067::" # eg "crn:v1:bluemix:public:cloud-object-storage:global:a/3bf0d9003abfb5d29761c3e97696b71c:d6f04d83-6c4f-4a62-a165-696756d63903::"


# Create resource

cos = ibm_boto3.resource("s3",

    ibm_api_key_id=COS_API_KEY_ID,

    ibm_service_instance_id=COS_RESOURCE_CRN,

    ibm_auth_endpoint=COS_AUTH_ENDPOINT,

    config=Config(signature_version="oauth"),

    endpoint_url=COS_ENDPOINT

)


#Provide CloudantDB credentials such as username,password and url


client = Cloudant("959488fa-a1c4-4c85-8129-ed1ee8360c62-bluemix", "3d9d719012c25813207ab56cb5ac36c7b057529f14f42a83ab6a95e9da1e2e4b", url="https://959488fa-a1c4-4c85-8129-ed1ee8360c62-bluemix:3d9d719012c25813207ab56cb5ac36c7b057529f14f42a83ab6a95e9da1e2e4b@959488fa-a1c4-4c85-8129-ed1ee8360c62-bluemix.cloudantnosqldb.appdomain.cloud")

client.connect()


#Provide your database name
```

```
database_name = "sample"

my_database = client.create_database(database_name)

if my_database.exists():
  print("'{database_name}' successfully created.")



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))



#It will read the first frame/image of the video
video=cv2.VideoCapture(0)

while True:
    #capture the first frame
    check,frame=video.read()
    #frame = cv2.resize(frame, (1000,667))
    gray=cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)



    #detect the faces from the video using detectMultiScale function
    faces=face_classifier.detectMultiScale(gray,1.3,5)
    eyes=eye_classifier.detectMultiScale(gray,1.3,5)
```

```
    print(faces)


    #drawing rectangle boundries for the detected face
    for(x,y,w,h) in faces:
        cv2.rectangle(frame, (x,y), (x+w,y+h), (127,0,255), 2)
        cv2.imshow('Face detection', frame)
        picname=datetime.datetime.now().strftime("%y-%m-%d-%H-%M")
        cv2.imwrite(picname+".jpg",frame)
        multi_part_upload("deepthijidagam3", picname+".jpg", picname+".jpg")
        json_document={"link":COS_ENDPOINT+"/"+"deepthijidagam3"+"/"+picname+".jpg"}
        new_document = my_database.create_document(json_document)
        # Check that the document exists in the database.
        if new_document.exists():
          print(f"Document successfully created.")
        r =
requests.get('https://www.fast2sms.com/dev/bulk?authorization=OMyK5jnSDx9CG40kTNihZ6s
zEpYRqBPJaQAdr7v1bHg2cmLfoUgiV2jnM75hLRKcC6QAS9ePqOWBJ3dy&sender_id=FSTSMS&
message=Some one at door&language=english&route=p&numbers=9030644234')
        print(r.status_code)




    #drawing rectangle boundries for the detected eyes
    for(ex,ey,ew,eh) in eyes:
        cv2.rectangle(frame, (ex,ey), (ex+ew,ey+eh), (127,0,255), 2)
        cv2.imshow('Face detection', frame)


    #waitKey(1)- for every 1 millisecond new frame will be captured
```

```
Key=cv2.waitKey(1)
if Key==ord('q'):
    #release the camera
    video.release()
    #destroy all windows
    cv2.destroyAllWindows()
    break
```