Dayananda Sagar College of Engineering
Department of Electronics and Communication Engineering

Assignment

Program:	B.E.	**Semester:**	7
Course:	Machine Learning	**Section:**	C
Course Code:	19EC7DEMAL	**Date:**	23-12-2022

A Report on

AWS Cloud - MLOPs setup on the cloud for emotion detection

Submitted by

USN: 1DS19EC116	**NAME: Sachin Bhat G**
USN: 1DS19EC130	**NAME: Sirish Hublikar**
USN: 1DS19EC131	**NAME: Sohan Gowda R**

Faculty In-charge
Dr. Pushpalatha KN

Signature of Faculty In-charge

AWS Cloud - MLOPs setup on the cloud for emotion detection

Introduction

Rekognition is an AWS managed image and video analysis service. We just provide an image or video to the Rekognition API, and the service can identify objects, people, text, scenes, and activities. It can detect any inappropriate content as well. Rekognition also provides highly accurate facial analysis and facial recognition. You can detect, analyze, and compare faces for a wide variety of use cases, including user verification, cataloging, people counting, and public safety.

Other AWS services used:

1. **S3**:

Amazon Simple Storage Service (S3) lets us store objects and is where we'll be storing the images and other files in this project.

2. **Cloud9**:

AWS Cloud9 is a cloud-based IDE that lets us write, run, and debug our code with just a browser.
It includes a code editor, debugger, and terminal. It comes prepackaged with essential tools for popular programming languages and AWS, so we don't have to install files or configure our machine to start new projects.

Algorithm:

Step 1: Basic preparation

Install our dependencies. The libraries we need are included in setup.sh. Run it to install.

- Create an AWS S3 bucket to store the image to be used in next steps. This bucket is also used to store processed image and enable a presigned URL to allow temporary access to the image from a user who doesn't have AWS credentials.
- Create a Rekognition collection to store the facial signature. A facial signature consists of feature vectors sampled by AWS Rekognition service from input image frame and this metadata can be used for matching faces and emotional analysis. AWS Rekognition service groups the signatures about objects including faces into collections.

Create your 33 buckets using the command below, pick a globally unique bucket name.
These bucket name will be used in next steps. Name can be a mixture of lowercase letters and numbers.
If successful, console will prompt: "make_bucket:<your bucket name>" e.g. aws s3 mb s3://rekognition-workshop-simon
Use command 'aws s3 ls' to find and verify the creation of bucket

aws s3 mb s3://<your_raw_images_bucket_name>

Create a rekognition pool
Console will prompt "StatusCode": 200 when successful
e.g. aws rekognition create-collection --collection-id rekognition-workshop-simon

aws rekognition create-collection --collection-id <your_collection_name>

Step 2 - Upload an image to S3 bucket

Ready a photo and save it onto your local machine, make sure there is at least one face in it. AWS Rekognition service can index up to 100 faces at once, here we keep it simple by letting Rekognition index the one prominent face. More details shown in step 3.
You can try this with any image you want but there are some sample images in the folder "sampleImages". If you use these, make sure you drag the images out to where your .py files reside (as below)
Upload it to your Cloud9 IDE working directory: same directory where .py files resides
In case the Cloud9 'Upload Local Files" doesn't work, follow the steps below:

1. Change "IMAGE_PATH" in 'convertImageBase64.py' (e.g. "IMAGE_PATH = "~/workspace/wolverine.jpg") and run it on local machine against the image file

2. In Cloud9, open 'assembleImg.py', paste the base64 string into the designated place, and change the new image name, then run it. An image will be created.

~/workspace $ python convertImageBase64.py

Give the command below to upload the image to the S3 bucket that holds the raw images.

python3 upload_to_s3.py -i <image_name> -b <your_raw_images_bucket_name>

Step 3 - Use the main function to index the image to Rekognition

Modify the variable names in index.py according to the comments.
Our code will perform the tasks below sequentially:

1. Index the face to AWS Rekognition service
2. Process the metadata received from Rekognition service
3. Print the results of facial analysis to the console
4. Put a bounding box around the face on the image
5. Writes the main facial emotion onto the image while processing
6. Send the image to the s3 processed images bucket
7. Save the file and run it

python3 index.py

Python Code:

➢ index.py

```
import cv2
import boto3

# Modify varible names here
RAW_IMAGES_BUCKET = "<YOUR_RAW_IMAGES_S3_BUCKET>"                # e.g.
myrawimages12345
RAW_IMAGE_NAME = "<YOUR_IMAGE>"                                  # e.g. raw_image.jpg
COLLECTION = "<YOUR_REKOGNITION_COLLECTION>"                     # e.g. rekognition-workshop
```

```
PROCESSED_IMAGES_BUCKET = "<YOUR_PROCESSED_IMAGES_S3_BUCKET>"      # e.g.
myprocessedimages12345

# Name the processed image by appending processed
PROCESSED_IMAGE_NAME = "processed-" + RAW_IMAGE_NAME

if __name__=='__main__':

    # Create an s3 client 's3' and a rekognition client 'reko'
    s3 = boto3.resource('s3')
    reko = boto3.client('rekognition')

    # Index the face and receive response from rekognition service
    response = reko.index_faces(
        CollectionId = COLLECTION,
        Image = {
            'S3Object': {
                'Bucket': RAW_IMAGES_BUCKET,
                'Name': RAW_IMAGE_NAME,
            }
        },
        DetectionAttributes = ['ALL'],
        MaxFaces = 1,
        QualityFilter = 'AUTO'
    )
    # print(response)

    # Extract face metadata from the response
    info = response['FaceRecords'][0]['FaceDetail']
    smileConfidence = info['Smile']['Confidence']
    smile = info['Smile']['Value']
    gender = info['Gender']['Value']
```

```
agelow = info['AgeRange']['Low']
agehigh = info['AgeRange']['High']
faceId = response['FaceRecords'][0]['Face']['FaceId']

# Determine emotion by picking the one with highest confidence
emotions = info['Emotions']
emotionType = []
emotionConf = []
count = 0
for allEmotions in emotions:
    emotionType.insert(count, allEmotions['Type'])
    emotionConf.insert(count, allEmotions['Confidence'])
    count += 1
max_value = max(emotionConf)
max_index = emotionConf.index(max_value)
emotion = emotionType[max_index]

print("Information about face ID " + str(faceId) + ":")
print("Smile: " + str(smile) + "\nThe confidence is: " + str(smileConfidence))
print("Gender: " + str(gender))
print("Age between: " + str(agelow) + " to " + str(agehigh))
print("Emotion: " + str(emotion))

# Get the image W x H
img = cv2.imread(RAW_IMAGE_NAME, 1)
imgHeight, imgWidth, channels= img.shape

# Get bounding box values and turn it into drawing coordinates
box = info['BoundingBox']
left = int(imgWidth * box['Left'])
top = int(imgHeight * box['Top'])
width = int(imgWidth * box['Width']) + left
```

```
    height = int(imgHeight * box['Height']) + top

    # Draw on the image
    cv2.rectangle(img, (left, top), (width, height), (0, 255, 0), 2)
    summaryStr = "Smile: " + str(smile) + " | Gender: " + str(gender) + " | Age between: " + str(agelow) + " to " +
str(agehigh) + " | Emotion: " + str(emotion)
    cv2.putText(img, summaryStr, (50, 50), cv2.FONT_HERSHEY_PLAIN , 1.1, (0, 255, 0), 2, cv2.LINE_AA)

    # Save this image and send it to s3
    cv2.imwrite(PROCESSED_IMAGE_NAME, img)
    s3.meta.client.upload_file(PROCESSED_IMAGE_NAME, PROCESSED_IMAGES_BUCKET,
PROCESSED_IMAGE_NAME)
```

- **upload_to_s3.py**

```
import argparse
import boto3

# No need to modify here (unless stuff doesn't work)
BUCKET = ""
IMAGE = ""

ap = argparse.ArgumentParser()
ap.add_argument('-i', '--image', required=True, help = 'path to your image')
ap.add_argument('-b', '--bucket', required=True, help = 'your bucket name')
args = ap.parse_args()
if (args.image != None):
    IMAGE = args.image
if (args.bucket != None):
    BUCKET = args.bucket

# Create an s3 client 's3'
```

```
s3 = boto3.resource('s3')


# Use s3 client to send this image to the s3 bucket
s3.meta.client.upload_file(IMAGE, BUCKET, IMAGE)
print("Image: " + IMAGE + " has been sent to bucket: " + BUCKET)
```

Results:

Sample Input:

Sample output:
EMOTION: HAPPY

Sample output:
EMOTION: DISGUSTED

EMOTION: SURPRISED

EMOTION: ANGRY

References:

1. https://aws.amazon.com/rekognition/
2. https://aws.amazon.com/s3/
3. https://aws.amazon.com/cloud9/