RAAGAVENDRAN SUNDAR - E0123016

Computer Vision Face Detection Lab - Student Notebook

In this lab, you will use Amazon Rekognition to perform a facial detection of a known face.

The high-level steps you will perform in this lab are:

1. Create a collection.

2. Upload an image of a face that you want to detect to the Amazon SageMaker notebook.

3. Add the image to the collection.

4. View the bounding box that was created for the image.

5. List the faces in the collection.

6. Use the collection to find a face.

7. View the bounding box of the face that was found.

8. Delete the collection.

Importing Python packages

Start by importing the Python packages that you need.

In the following code:

- *matplotlib* provides plotting functions
- *skimage* represents scikit-image, which provides several useful image manipulation tools
- *boto3* represents the AWS SDK for Python (Boto3), which is the Python library for AWS
- *numpy* represents NumPy, which is a library for manipulating data
- *PIL* represents the Python Imaging Library, which contains a set of tools for drawing images

```
In [1]:   from skimage import io
          from skimage.transform import rescale
          from matplotlib import pyplot as plt
```

```
import boto3

import numpy as np

from PIL import Image, ImageDraw, ImageColor, ImageOps
```

Task 1: Creating a collection

In this task, you create a collection in Amazon Rekognition.

You only need to run this step once.

In [2]:
```
client = boto3.client('rekognition')
collection_id = 'Collection'
response = client.create_collection(CollectionId=collection_id)
print('Collection ARN: ' + response['CollectionArn'])
print('Status Code:' + str(response['StatusCode']))
print('Done...')
```

```
Collection ARN: aws:rekognition:us-east-1:058264254780:collection/Collection
Status Code:200
Done...
```

Task 2: Uploading an image to search

Use the provided sample image, which is named *mum.jpg*, and upload it to this notebook.

Then, look at the image by running the cell.

In [3]:
```
filename = "mum.jpg"

faceimage = io.imread(filename)

plt.imshow(faceimage)
```

Out[3]: <matplotlib.image.AxesImage at 0x7f30fd53ab90>

Make sure that the image size is less than 4096 x 4096 pixels. If the image is larger than this size, then you must resize it by using the following code:

```
faceimage = rescale(faceimage, 0.50, mode='constant')
```

Note: The numeric value represents the factor to scale by. A value of *0.5* will scale the image to 50 percent of the original.

When the image is resized, save the file.

```
io.imsave(filename, faceimage)
```

Tip: You must copy the code into a code cell and run it.

Task 3: Adding the image to the collection

Add the image to the collection that you created earlier.

```
In [4]:  externalimageid = filename

         with open(filename, 'rb') as fimage:
             response = client.index_faces(CollectionId = collection_id,
                                            Image={'Bytes': fimage.read()},
                                            ExternalImageId=externalimageid,
                                            MaxFaces=1,
                                            QualityFilter="AUTO",
                                            DetectionAttributes=['ALL'])
```

```python
print('Results for ' + filename)
print('Faces indexed:')
for faceRecord in response['FaceRecords']:
    print('  Face ID: ' + faceRecord['Face']['FaceId'])
    print('  Location: {}'.format(faceRecord['Face']['BoundingBox']))

print('Faces not indexed:')
for unindexedFace in response['UnindexedFaces']:
    print(' Location: {}'.format(unindexedFace['FaceDetail']['BoundingBox']))
    print(' Reasons:')
    for reason in unindexedFace['Reasons']:
        print('    ' + reason)
```

```
Results for mum.jpg
Faces indexed:
  Face ID: e463550b-7d1b-4cdc-995e-c84af0db670e
  Location: {'Width': 0.3483645021915436, 'Height': 0.35312092304229736, 'Left':
0.30951350927352905, 'Top': 0.11020112037658691}
Faces not indexed:
```

Task 4: Viewing the bounding box for the detected face

If a face was found, the results should include the location of the face that was detected. Examine the bounding box on the image.

To do this, use the PIL library, which you imported earlier in this lab. By extracting the BoundingBox, you can draw a set of lines around the image.

In [5]:
```python
img = Image.open(filename)
imgWidth, imgHeight = img.size

draw = ImageDraw.Draw(img)
for faceRecord in response['FaceRecords']:
    box = faceRecord['Face']['BoundingBox']
    left = imgWidth * box['Left']
    top = imgHeight * box['Top']
    width = imgWidth * box['Width']
    height = imgHeight * box['Height']

    points = ((left,top),(left+width,top),(left+width,top+height),(left,top+height)

    draw.line(points,fill='#00d400', width=15)

plt.imshow(img)
```
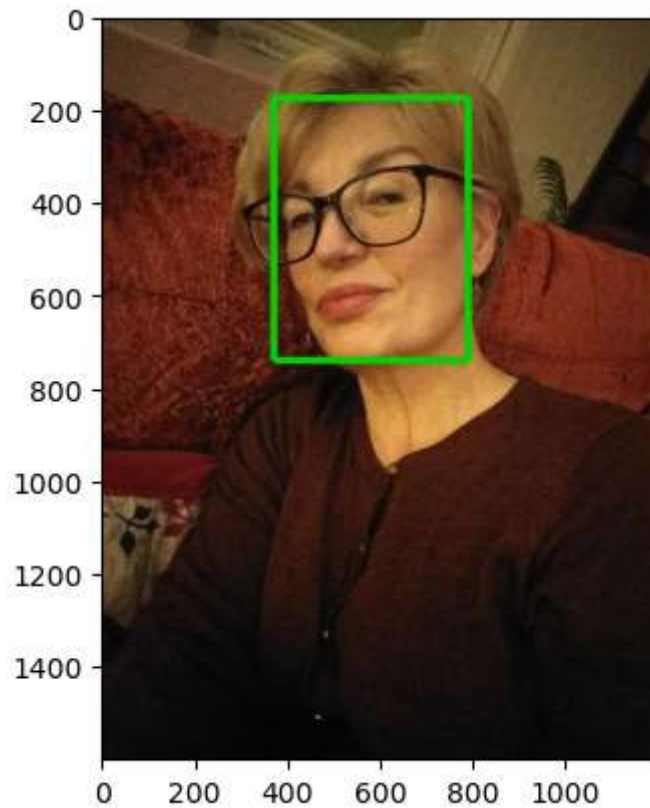
Out[5]: <matplotlib.image.AxesImage at 0x7f30fd3deb60>

Task 5: Listing faces in the collection

Examine the image that you have in the collection.

```
In [6]:  maxResults=2
         faces_count=0
         tokens=True

         response=client.list_faces(CollectionId=collection_id,
                                      MaxResults=maxResults)

         print('Faces in collection ' + collection_id)

         while tokens:

             faces=response['Faces']

             for face in faces:
                 print (face)
                 faces_count+=1
             if 'NextToken' in response:
                 nextToken=response['NextToken']
                 response=client.list_faces(CollectionId=collection_id,
                                              NextToken=nextToken,MaxResults=maxResults)
             else:
                 tokens=False
```

Faces in collection Collection

{'FaceId': 'e463550b-7d1b-4cdc-995e-c84af0db670e', 'BoundingBox': {'Width': 0.3483
6500883102417, 'Height': 0.3531210124492645, 'Left': 0.30951398611068726, 'Top':
0.11020100116729736}, 'ImageId': '59d87d59-b733-306a-94e8-0670b0a8dcdd', 'External
ImageId': 'mum.jpg', 'Confidence': 99.99430084228516, 'IndexFacesModelVersion':
'7.0'}

Task 6: Finding a face by using the collection

In this step, you will use the collection to detect a face in an image.

Use the provided sample image that's named *target.jpg* and upload it to this notebook.

In [7]:
```python
targetfilename = "target.jpg"

targetimage = Image.open(targetfilename)
plt.imshow(targetimage)
```

Out[7]: <matplotlib.image.AxesImage at 0x7f30fd4a12a0>

Next, call the `search_faces_by_image` operation and see if you get a match.

In [8]:
```python
threshold = 70
maxFaces=2

with open(targetfilename, 'rb') as timage:
    response2=client.search_faces_by_image(CollectionId=collection_id,
                            Image={'Bytes': timage.read()},
                            FaceMatchThreshold=threshold,
```

```
                           MaxFaces=maxFaces)

faceMatches=response2['FaceMatches']
print ('Matching faces')
for match in faceMatches:
        print ('FaceId:' + match['Face']['FaceId'])
        print ('Similarity: ' + "{:.2f}".format(match['Similarity']) + "%")
        print ('ExternalImageId: ' + match['Face']['ExternalImageId'])
        print
```

```
Matching faces
FaceId:e463550b-7d1b-4cdc-995e-c84af0db670e
Similarity: 99.91%
ExternalImageId: mum.jpg
```

Task 7: Drawing a bounding box around the discovered face

Draw a bounding box around the discovered face.

In [9]:
```python
imgWidth, imgHeight = targetimage.size

draw = ImageDraw.Draw(targetimage)

box = response2['SearchedFaceBoundingBox']
left = imgWidth * box['Left']
top = imgHeight * box['Top']
width = imgWidth * box['Width']
height = imgHeight * box['Height']

points = ((left,top),(left+width,top),(left+width,top+height),(left,top+height),(le
draw.line(points,fill='#00d400', width=15)

plt.imshow(targetimage)
```
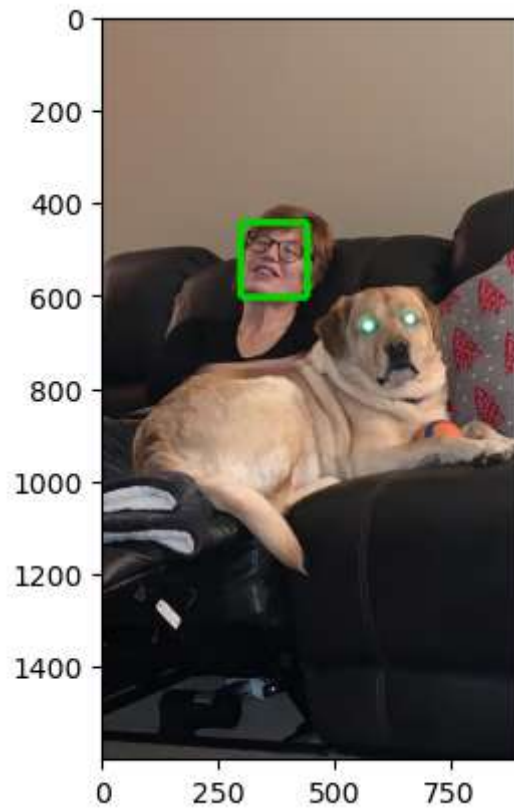
Out[9]:   <matplotlib.image.AxesImage at 0x7f30fc33e6b0>

Task 8: Deleting the collection

When you are finished, delete the collection. To do this, run the following code.

```
In [10]: print('Attempting to delete collection ' + collection_id)
         status_code=0
         try:
             response=client.delete_collection(CollectionId=collection_id)
             status_code=response['StatusCode']
             print('All done!')
             print(status_code)

         except ClientError as e:
             if e.response['Error']['Code'] == 'ResourceNotFoundException':
                 print ('The collection ' + collection_id + ' was not found ')
             else:
                 print ('Error other than Not Found occurred: ' + e.response['Error']['Messa
             status_code=e.response['ResponseMetadata']['HTTPStatusCode']
```

```
Attempting to delete collection Collection
All done!
200
```

Congratulations!

You have completed this lab, and you can now end the lab by following the lab guide instructions.