UC San Diego Extension
Cloud Services for Machine Learning

Summer 2020
Homework#4

Date Given: July 20, 2020 Due Date: July 26, 2020

==========

Problem#1:
Using your own GCP account generate a Vision API Key file in JSON format. This file will be used in the next problem.

Problem#2:
Using the Vision API key file generated in the previous problem, write Python code in Colab to analyze the images described on the next page by using GCP 'pre-trained' Vision API model.

Your report should include the following features of the images.

1. If an image has human faces, it should recognize the faces and describe the emotions on the faces. It should draw a bounding box around all the faces in that image.

2. If an image has famous Landmarks or Logo, it should recognize those landmarks/logos and provide information about them.

3. It should generate labels of the objects in the image. It should create a bounding box around the objects in the image

4. It should generate the text for the portion of the image that contains text using OCR (Optical Character Recognition).

5. It should show the colors used in the image

6. It should able to tell if the content of the image contains adult material (sexual in nature).

Images to Analyze

Category#1: Faces with emotions

- **01 happy faces.jpg**
- **02 sad face.jpg**
- **03 angry face.jpg**
- **04 surprise face.jpg**

Category#2: Landmarks

- **01 Colosseum Rome.jpg**
- **02 Pyramids.jpg**
- **03 TajMahal.jpg**

Category#3: Objects

- **01 traffic image Autonomous Cars.jpg**

Category#4: Images with Text

- **01 Image Text1.jpg**
- **02 Image Text2.jpg**

==========

Category#1: Faces with emotions

01 happy faces.jpg

1. Labels (and confidence score):

```
==========================================================================
Face (98.63%)
People (96.93%)
Facial expression (94.48%)
Smile (91.85%)
Head (91.25%)
Team (90.78%)
Photography (72.45%)
Fun (70.39%)
Laugh (63.50%)
Portrait photography (56.45%)

==========================================================================
```

2. Emotions

```
anger: VERY_UNLIKELY
joy: VERY_LIKELY
surprise: VERY_UNLIKELY
Face bounds: (8,0),(160,0),(160,167),(8,167)

anger: VERY_UNLIKELY
joy: LIKELY
surprise: VERY_UNLIKELY
Face bounds: (169,158),(340,158),(340,341),(169,341)

anger: VERY_UNLIKELY
joy: VERY_LIKELY
surprise: VERY_UNLIKELY
Face bounds: (352,168),(495,168),(495,341),(352,341)
```

```
anger: VERY_UNLIKELY
joy: VERY_LIKELY
surprise: VERY_UNLIKELY
Face bounds: (346,0),(495,0),(495,182),(346,182)

anger: VERY_UNLIKELY
joy: VERY_LIKELY
surprise: VERY_UNLIKELY
Face bounds: (2,153),(171,153),(171,341),(2,341)

anger: VERY_UNLIKELY
joy: VERY_LIKELY
surprise: VERY_UNLIKELY
Face bounds: (180,0),(332,0),(332,175),(180,175)

================================================================================
Number of objects found: 9

Person (confidence: 0.8644946217536926)
Person (confidence: 0.85704106092453)
Person (confidence: 0.8407533764839172)
Clothing (confidence: 0.8321565389633179)
Person (confidence: 0.8080854415893555)
Person (confidence: 0.7959855794906616)
Person (confidence: 0.7630079984664917)
Clothing (confidence: 0.5840413570404053)
Clothing (confidence: 0.547836184501648)

Safe search:
adult: VERY_UNLIKELY
medical: VERY_UNLIKELY
spoofed: VERY_UNLIKELY
violence: VERY_UNLIKELY
racy: VERY_UNLIKELY
```

02 sad face.jpg

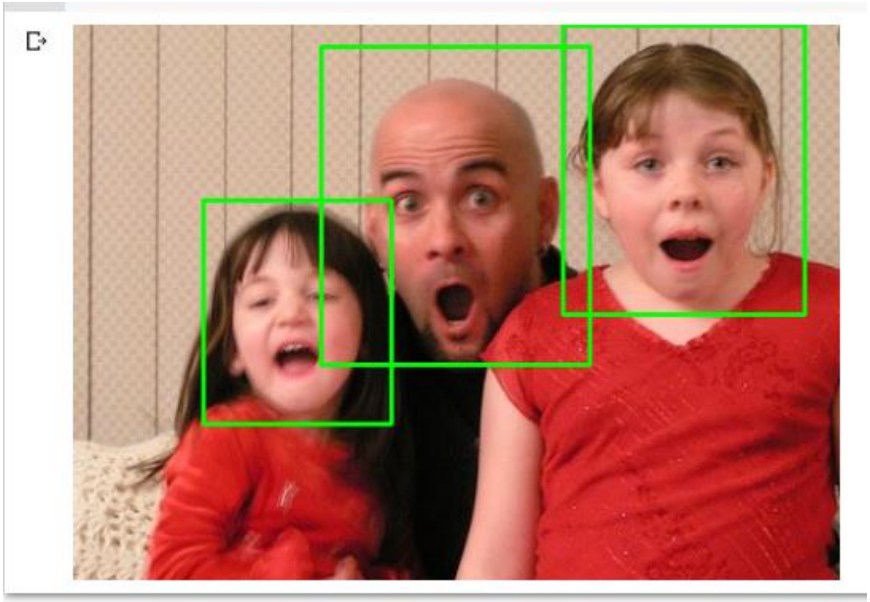
```
Labels (and confidence score):
=================================================================================
Face (99.14%)
Photograph (95.27%)
Nose (94.46%)
Facial expression (94.46%)
Child (92.43%)
Skin (92.00%)
Head (91.66%)
Black-and-white (91.27%)
Cheek (91.11%)
Eyebrow (89.32%)

=================================================================================
anger: VERY_UNLIKELY
joy: VERY_UNLIKELY
surprise: VERY_UNLIKELY
Face bounds: (60,0),(484,0),(484,301),(60,301)
```

03 angry face.jpg

==========
anger: VERY_LIKELY
joy: VERY_UNLIKELY
surprise: VERY_UNLIKELY
Face bounds: (129,4),(294,4),(294,195),(129,195)

04 surprise face.jpg

```
================================================================================
anger: VERY_UNLIKELY
joy: VERY_UNLIKELY
surprise: VERY_UNLIKELY
Face bounds: (87,115),(213,115),(213,262),(87,262)
anger: VERY_UNLIKELY
joy: VERY_UNLIKELY
surprise: VERY_LIKELY
Face bounds: (328,0),(490,0),(490,190),(328,190)
anger: POSSIBLE
joy: VERY_UNLIKELY
surprise: VERY_LIKELY
Face bounds: (166,14),(346,14),(346,223),(166,223)
```



==========

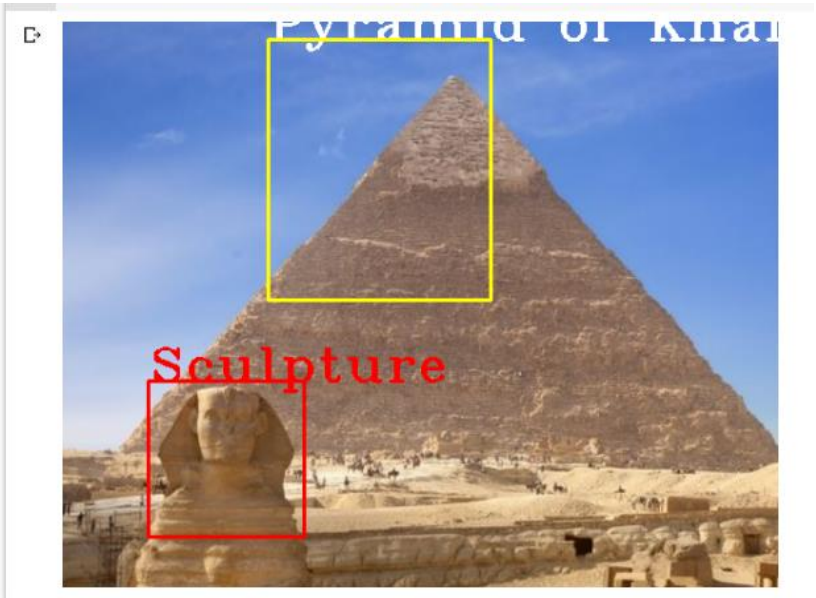
Category#2: Landmarks

01 Colosseum Rome.jpg

```
Labels (and confidence score):
================================================================================
Landmark (98.07%)
Historic site (96.64%)
Ancient roman architecture (96.47%)
Architecture (94.16%)
Amphitheatre (94.14%)
Ancient rome (94.02%)
Ancient history (93.91%)
Building (91.95%)
Human settlement (87.36%)
Classical architecture (86.87%)


landmark :  Palatine Museum on Palatine Hill
landmark :  Colosseum
landmark :  Colosseum
```

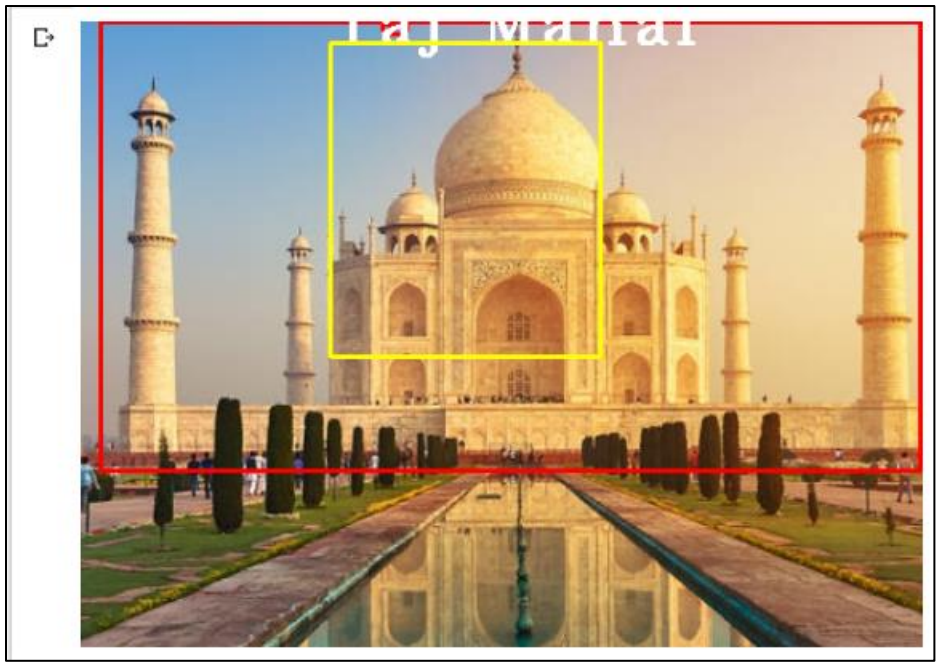
02 Pyramids.jpg

```
Labels (and confidence score):
================================================================================
Pyramid (99.20%)
Ancient history (98.30%)
Monument (98.18%)
Landmark (98.01%)
Historic site (97.76%)
Wonders of the world (93.16%)
Archaeological site (91.16%)
Unesco world heritage site (88.60%)
History (87.42%)
Tourist attraction (85.51%)
```



03 TajMahal.jpg

```
Labels (and confidence score):
================================================================================
Landmark (98.57%)
Wonders of the world (96.32%)
Historic site (95.83%)
Holy places (91.83%)
Tourist attraction (91.25%)
Monument (91.08%)
Mausoleum (87.69%)
Dome (87.08%)
Reflecting pool (86.17%)
Architecture (86.05%)
```



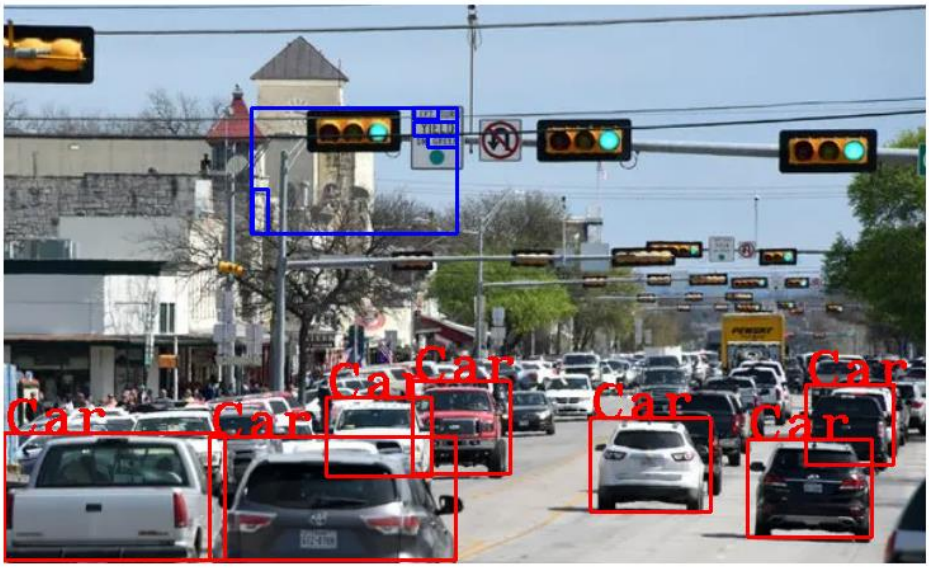

```
landmark :  Taj Mahal
```

==========

Category#3: Objects

01 traffic image Autonomous Cars.jpg

```
Labels (and confidence score):
================================================================================
Traffic (97.40%)
Motor vehicle (96.27%)
signaling device (95.61%)
Traffic light (95.35%)
Lane (93.06%)
Lighting (89.42%)
Transport (89.41%)
Vehicle (88.29%)
Traffic congestion (87.61%)
Mode of transport (87.25%)
```



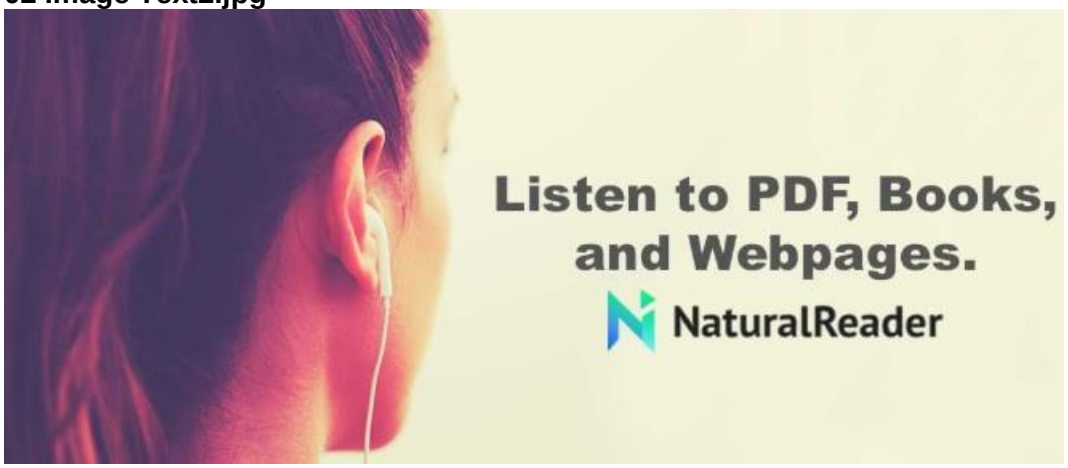
==========

Category#4: Images with Text

01 Image Text1.jpg


```
Labels (and confidence score):
================================================================================
Text (94.40%)
Aqua (94.01%)
Font (92.18%)
Turquoise (91.30%)
Wave (82.80%)
Ocean (81.20%)
Wind wave (72.12%)
Logo (71.54%)
Sea (63.57%)
Banner (63.57%)

"SURF
"
bounds: (16,36),(591,36),(591,255),(16,255)

"SURF"
bounds: (16,36),(591,36),(591,255),(16,255)
```

02 Image Text2.jpg




```
[12] #Text detection
     response = client.text_detection(image=image)
     texts = response.text_annotations
     for text in texts:
       print('\n"{}"'.format(text.description))

       vertices = (['({},{})'.format(vertex.x, vertex.y) for vert
       print('bounds: {}'.format(','.join(vertices)))
       cv2.rectangle(img,(int(vertices[0].split(",")[0].replace("
```

```
"Listen to PDF, Books,
and Webpages.
N NaturalReader
"
bounds: (308,103),(656,103),(656,213),(308,213)

"Listen"
bounds: (308,104),(406,105),(406,127),(308,126)

"to"
bounds: (420,105),(451,105),(451,125),(420,125)

"PDF,"
bounds: (465,103),(533,104),(533,132),(465,131)

"Books,"
bounds: (547,104),(656,105),(656,131),(547,130)

"and"
bounds: (357,140),(415,141),(415,163),(357,162)

"Webpages."
bounds: (427,140),(606,143),(606,171),(427,168)

"N"
bounds: (406,181),(426,181),(426,211),(406,211)

"NaturalReader"
bounds: (428,181),(586,182),(586,213),(428,212)
```

```
from google.colab.patches import cv2_imshow
cv2_imshow(img)
```