```
import imutils
import numpy as np
import cv2
from google.colab.patches import cv2_imshow
from IPython.display import display, Javascript
from google.colab.output import eval_js
from base64 import b64decode
```

Double-click (or enter) to edit

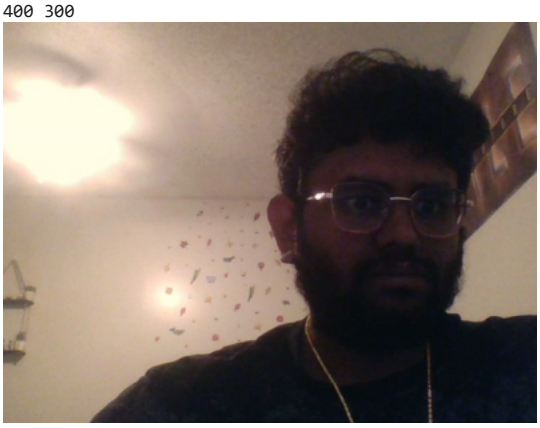
```
Start coding or generate with AI.
```

```
def take_photo(filename='photo.jpg', quality=0.8):
  js = Javascript('''
    async function takePhoto(quality) {
      const div = document.createElement('div');
      const capture = document.createElement('button');
      capture.textContent = 'Capture';
      div.appendChild(capture);

      const video = document.createElement('video');
      video.style.display = 'block';
      const stream = await navigator.mediaDevices.getUserMedia({video: true});

      document.body.appendChild(div);
      div.appendChild(video);
      video.srcObject = stream;
      await video.play();


      google.colab.output.setIframeHeight(document.documentElement.scrollHeight, true);


      await new Promise((resolve) => capture.onclick = resolve);

      const canvas = document.createElement('canvas');
      canvas.width = video.videoWidth;
      canvas.height = video.videoHeight;
      canvas.getContext('2d').drawImage(video, 0, 0);
      stream.getVideoTracks()[0].stop();
      div.remove();
      return canvas.toDataURL('image/jpeg', quality);
    }
    ''')
  display(js)
  data = eval_js('takePhoto({})'.format(quality))
  binary = b64decode(data.split(',')[1])
  with open(filename, 'wb') as f:
    f.write(binary)
  return filename
```

```
image_file = take_photo()
```

```
image = cv2.imread(image_file)
image = imutils.resize(image, width=400)
(h, w) = image.shape[:2]
print(w,h)
cv2_imshow(image)
```

```
400 300
```



```
!wget -N https://raw.githubusercontent.com/opencv/opencv/master/samples/dnn/face_detector/deploy.prototxt
!wget -N https://raw.githubusercontent.com/opencv/opencv_3rdparty/dnn_samples_face_detector_20170830/res10_300x300_ssd_iter_140000.caffe
```

```
--2025-04-12 19:16:21--  https://raw.githubusercontent.com/opencv/opencv/master/samples/dnn/face_detector/deploy.prototxt
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 28104 (27K) [text/plain]
Saving to: ‘deploy.prototxt’

deploy.prototxt     100%[===================>]  27.45K  --.-KB/s    in 0.002s

Last-modified header missing -- time-stamps turned off.
2025-04-12 19:16:21 (11.1 MB/s) - ‘deploy.prototxt’ saved [28104/28104]

--2025-04-12 19:16:22--  https://raw.githubusercontent.com/opencv/opencv_3rdparty/dnn_samples_face_detector_20170830/res10_300x300_s
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.109.133, 185.199.111.133, 185.199.108.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.109.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 10666211 (10M) [application/octet-stream]
Saving to: ‘res10_300x300_ssd_iter_140000.caffemodel’

res10_300x300_ssd_i 100%[===================>]  10.17M  --.-KB/s    in 0.1s

Last-modified header missing -- time-stamps turned off.
2025-04-12 19:16:22 (100 MB/s) - ‘res10_300x300_ssd_iter_140000.caffemodel’ saved [10666211/10666211]
```

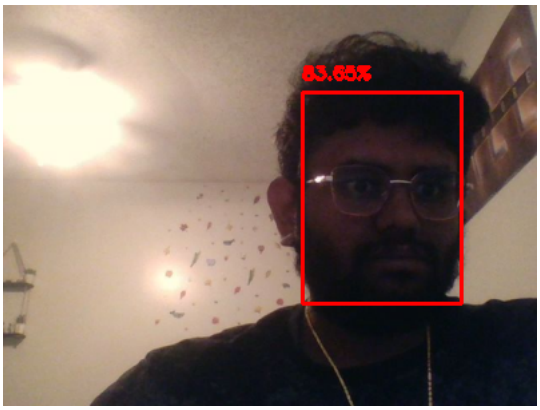
```
print("[INFO] loading model...")
prototxt = 'deploy.prototxt'
model = 'res10_300x300_ssd_iter_140000.caffemodel'
net = cv2.dnn.readNetFromCaffe(prototxt, model)
```

```
[INFO] loading model...
```

```
image = imutils.resize(image, width=400)
blob = cv2.dnn.blobFromImage(cv2.resize(image, (300, 300)), 1.0, (300, 300), (104.0, 177.0, 123.0))


print("[INFO] computing object detections...")
net.setInput(blob)
detections = net.forward()
```

```
[INFO] computing object detections...
```

```
for i in range(0, detections.shape[2]):

    # extract the confidence (i.e., probability) associated with the prediction
    confidence = detections[0, 0, i, 2]

    if confidence > 0.5:

        box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
        (startX, startY, endX, endY) = box.astype("int")

        text = "{:.2f}%".format(confidence * 100)
        y = startY - 10 if startY - 10 > 10 else startY + 10
        cv2.rectangle(image, (startX, startY), (endX, endY), (0, 0, 255), 2)
        cv2.putText(image, text, (startX, y),
            cv2.FONT_HERSHEY_SIMPLEX, 0.45, (0, 0, 255), 2)


cv2_imshow(image)
```




```
import cv2
!pip install deepface
from deepface import DeepFace
import matplotlib.pyplot as plt


predictions = DeepFace.analyze(img_path=image, enforce_detection=False)
```

```
100%|██████████| 539M/539M [00:07<00:00, 67.6MB/s]
Action: gender:  50%|█████     | 2/4 [00:12<00:14,  7.47s/it]25-04-12 19:18:17 - gender_model_weights.h5 will be downloaded...
Downloading...
From: https://github.com/serengil/deepface_models/releases/download/v1.0/gender_model_weights.h5
To: /root/.deepface/weights/gender_model_weights.h5

  0%|          | 0.00/537M [00:00<?, ?B/s]
  2%|▏         | 11.0M/537M [00:00<00:04, 109MB/s]
  7%|▋         | 39.8M/537M [00:00<00:02, 213MB/s]
 13%|█▎        | 70.8M/537M [00:00<00:01, 255MB/s]
 19%|█▉        | 101M/537M [00:00<00:01, 273MB/s]
 24%|██▍       | 129M/537M [00:00<00:01, 265MB/s]
 29%|██▉       | 156M/537M [00:00<00:01, 253MB/s]
 34%|███▍      | 182M/537M [00:00<00:01, 258MB/s]
 39%|███▉      | 210M/537M [00:00<00:01, 263MB/s]
 44%|████▍     | 239M/537M [00:00<00:01, 268MB/s]
 50%|████▉     | 266M/537M [00:01<00:01, 270MB/s]
 55%|█████▍    | 295M/537M [00:01<00:00, 273MB/s]
 60%|██████    | 322M/537M [00:01<00:00, 241MB/s]
 65%|██████▍   | 348M/537M [00:01<00:00, 244MB/s]
 70%|██████▉   | 375M/537M [00:01<00:00, 250MB/s]
 75%|███████▍  | 402M/537M [00:01<00:00, 249MB/s]
 80%|███████▉  | 428M/537M [00:01<00:00, 248MB/s]
 84%|████████▍ | 453M/537M [00:01<00:00, 233MB/s]
 89%|████████▉ | 477M/537M [00:01<00:00, 231MB/s]
 93%|█████████▎| 500M/537M [00:02<00:00, 227MB/s]
100%|██████████| 537M/537M [00:02<00:00, 245MB/s]
Action: race:  75%|███████▌  | 3/4 [00:18<00:06,  6.81s/it]  25-04-12 19:18:23 - race_model_single_batch.h5 will be downloaded...
Downloading...
From: https://github.com/serengil/deepface_models/releases/download/v1.0/race_model_single_batch.h5
To: /root/.deepface/weights/race_model_single_batch.h5

  0%|          | 0.00/537M [00:00<?, ?B/s]
  2%|▏         | 8.91M/537M [00:00<00:05, 88.9MB/s]
  6%|▌         | 32.0M/537M [00:00<00:02, 171MB/s]
 11%|█▏        | 60.8M/537M [00:00<00:02, 223MB/s]
 17%|█▋        | 89.1M/537M [00:00<00:01, 245MB/s]
 21%|██        | 114M/537M [00:00<00:01, 236MB/s]
 26%|██▌       | 138M/537M [00:00<00:01, 229MB/s]
 31%|███       | 165M/537M [00:00<00:01, 242MB/s]
 35%|███▌      | 190M/537M [00:00<00:01, 232MB/s]
 40%|███▉      | 213M/537M [00:00<00:01, 225MB/s]
 44%|████▍     | 236M/537M [00:01<00:01, 216MB/s]
 48%|████▊     | 259M/537M [00:01<00:01, 218MB/s]
 52%|█████▏    | 281M/537M [00:01<00:01, 212MB/s]
 57%|█████▋    | 305M/537M [00:01<00:01, 220MB/s]
 61%|██████    | 328M/537M [00:01<00:01, 201MB/s]
 66%|██████▌   | 352M/537M [00:01<00:00, 213MB/s]
 70%|███████   | 377M/537M [00:01<00:00, 223MB/s]
 75%|███████▍  | 401M/537M [00:01<00:00, 226MB/s]
 79%|███████▉  | 424M/537M [00:01<00:00, 203MB/s]
 83%|████████▎ | 445M/537M [00:02<00:00, 178MB/s]
 86%|████████▌ | 464M/537M [00:02<00:00, 179MB/s]
 90%|████████▉ | 483M/537M [00:02<00:00, 180MB/s]
 93%|█████████▎| 502M/537M [00:02<00:00, 182MB/s]
100%|██████████| 537M/537M [00:02<00:00, 203MB/s]
Action: race: 100%|██████████| 4/4 [00:25<00:00,  6.39s/it]
```

```
predictions = DeepFace.analyze(image, enforce_detection=False)
```

```
Action: race: 100%|██████████| 4/4 [00:02<00:00,  1.74it/s]
```

```
predictions
```

```
[{'emotion': {'angry': np.float32(3.2504482e-05),
   'disgust': np.float32(2.7033524e-14),
   'fear': np.float32(0.03614552),
   'happy': np.float32(2.560246e-09),
   'sad': np.float32(99.16178),
   'surprise': np.float32(7.802905e-10),
   'neutral': np.float32(0.8020294)},
  'dominant_emotion': 'sad',
  'region': {'x': 0,
   'y': 0,
   'w': 399,
   'h': 299,
   'left_eye': None,
   'right_eye': None},
  'face_confidence': 0,
  'age': 32,
  'gender': {'Woman': np.float32(6.2580123), 'Man': np.float32(93.74199)},
  'dominant_gender': 'Man',
  'race': {'asian': np.float32(11.376387),
   'indian': np.float32(0.9555027),
   'black': np.float32(0.51942307),
   'white': np.float32(66.63144),
   'middle eastern': np.float32(10.180746),
   'latino hispanic': np.float32(10.3364935)},
  'dominant_race': 'white'}]
```

```
Start coding or generate with AI.
```