Face detection using pre-trained model

By SISAYKEO Phonepaserth

We use following blog as a reference: [Face detection with OpenCV and deep learning]

Import required Python libraries

```
1
2 import imutils
3 import numpy as np
4 import cv2
5 from google.colab.patches import cv2_imshow
6 from IPython.display import display, Javascript
7 from google.colab.output import eval_js
8 from base64 import b64decode
```

Start webcam

```
1 def take_photo(filename='photo.jpg', quality=0.8):
2   js = Javascript('''
3     async function takePhoto(quality) {
4       const div = document.createElement('div');
5       const capture = document.createElement('button');
6       capture.textContent = 'Capture';
7       div.appendChild(capture);
8
9       const video = document.createElement('video');
10      video.style.display = 'block';
11      const stream = await navigator.mediaDevices.getUserMedia({video: true});
12
13      document.body.appendChild(div);
14      div.appendChild(video);
15      video.srcObject = stream;
16      await video.play();
17
18      // Resize the output to fit the video element.
19      google.colab.output.setIframeHeight(document.documentElement.scrollHeight, true);
20
21      // Wait for Capture to be clicked.
22      await new Promise((resolve) => capture.onclick = resolve);
23
24      const canvas = document.createElement('canvas');
25      canvas.width = video.videoWidth;
26      canvas.height = video.videoHeight;
27      canvas.getContext('2d').drawImage(video, 0, 0);
28      stream.getVideoTracks()[0].stop();
29      div.remove();
30      return canvas.toDataURL('image/jpeg', quality);
31    }
32    ''')
33  display(js)
34  data = eval_js('takePhoto({})'.format(quality))
35  binary = b64decode(data.split(',')[1])
36  with open(filename, 'wb') as f:
37    f.write(binary)
38  return filename
```

Click 'Capture' to make photo using your webcam.

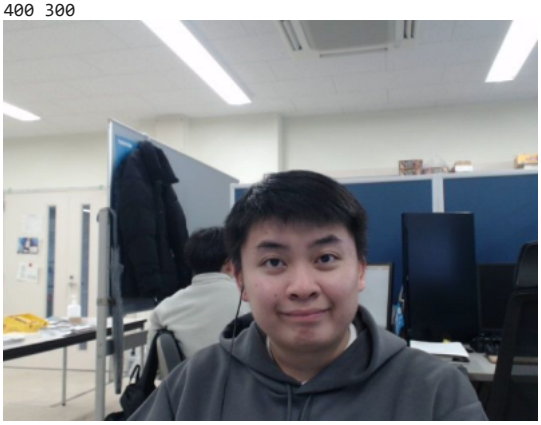
```
1 image_file = take_photo()
```

Read, resize and display the image.

```
1 #image = cv2.imread(image_file, cv2.IMREAD_UNCHANGED)
2 image = cv2.imread(image_file)
```

```
3
4 # resize it to have a maximum width of 400 pixels
5 image = imutils.resize(image, width=400)
6 (h, w) = image.shape[:2]
7 print(w,h)
8 cv2_imshow(image)
```

```
400 300
```



OpenCV's deep learning face detector is based on the Single Shot Detector (SSD) framework with a ResNet base network. The network is defined and trained using the Caffe Deep Learning framework

Download the pre-trained face detection model, consisting of two files:

- The network definition (deploy.prototxt)
- The learned weights (res10_300x300_ssd_iter_140000.caffemodel)

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

```
--2024-01-15 06:15:01--  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 0.001s

Last-modified header missing -- time-stamps turned off.
2024-01-15 06:15:01 (19.5 MB/s) - 'deploy.prototxt' saved [28104/28104]

--2024-01-15 06:15:01--  https://raw.githubusercontent.com/opencv/opencv_3rdparty/dnn_samples_face_detector_20170830/res10_300x300_ssd_i
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.111.133, 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: 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.
2024-01-15 06:15:02 (105 MB/s) - 'res10_300x300_ssd_iter_140000.caffemodel' saved [10666211/10666211]
```

Load the pre-trained face detection network model from disk

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

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

Use the dnn.blobFromImage function to construct an input blob by resizing the image to a fixed 300x300 pixels and then normalizing it.

```
# resize it to have a maximum width of 400 pixels
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))
```

Pass the blob through the neural network and obtain the detections and predictions.

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

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

Loop over the detections and draw boxes around the detected faces

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

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

    # filter out weak detections by ensuring the `confidence` is
    # greater than the minimum confidence threshold
    if confidence > 0.5:
        # compute the (x, y)-coordinates of the bounding box for the object
        box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
        (startX, startY, endX, endY) = box.astype("int")
        # draw the bounding box of the face along with the associated probability
        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)
```

Show the resulting image

```
cv2_imshow(image)
```