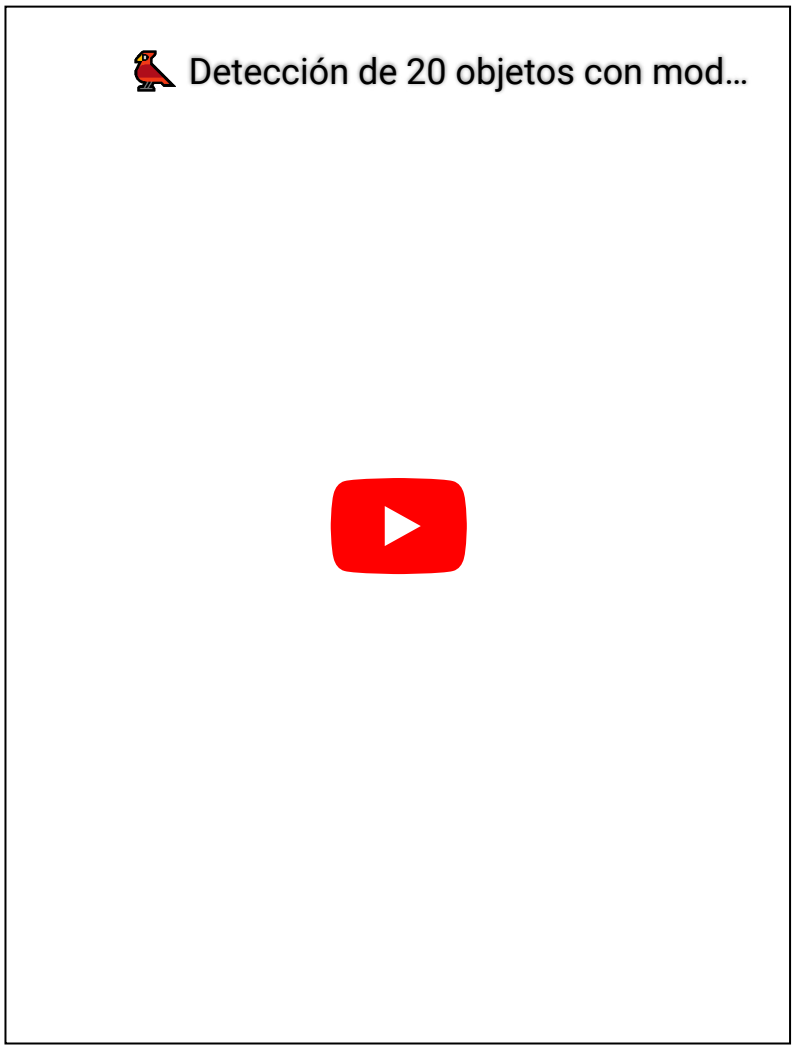
26
2022

entrenado usando DNN de OpenCV | Python

Por ADMINISTRADOR



En el anterior tutorial vimos como detectar rostros en una imagen con ayuda del módulo dnn de OpenCV, con este pudimos leer un modelo pre entrenado y ubicar el rostro en la imagen. En este tutorial veremos algo similar, solo que ha diferencia de las caras, podremos detectar varios objetos a la vez.

CONTENIDO

- Detectando objetos usando el módulo dnn de OpenCV
  - Instalación de packages

OpenCV

Nuestro objetivo en esta ocasión será detectar objetos dentro de una imagen o fotograma. Para ello usaremos la red MobileNetSSD. Los pesos asignados para esta aplicación han sido entrenados por terceros y nos permitirán determinar la ubicación de los objetos en la imagen, y además asignar las etiquetas que le corresponden.

Esta red pre entrenada nos permitirá detectar 20 objetos:

- Aviones
- Bicicletas
- Pájaros
- Barcos
- Botellas
- Autobuses
- Autos
- Gatos
- Sillas
- Vacas
- Comedores
- Perros
- Caballos
- Motos
- Personas
- Plantas en macetas
- Ovejas
- Sofás
- Trenes
- Televisores

Así que vamos a empezar por la instalación de los packages que necesitamos.

Instalación de packages

Te preguntarás, ¿de dónde sale Numpy?, si no lo habíamos instalado. Pues bien, al momento de instalar OpenCV, también se instala Numpy.

IMPORTANTE: Hay que tomar en cuenta que para el uso del módulo dnn, debemos contar con versiones igual o superiores a OpenCV 3.3.

Descargar arquitectura y pesos del modelo para detectar objetos con dnn

Para descargar la arquitectura (`MobileNetSSD_deploy.prototxt.txt`) y pesos pre entrenados (`MobileNetSSD_deploy.caffemodel`) de la red por favor dirígete a este link.

¡Vamos con la programación!

Para una explicación más detallada de los programas que veremos a continuación, por favor dirígete al video que he preparado en mi canal, en donde explico paso a pasito cada procedimiento efectuado. ¡Anímate a verlo 😉!.

El primer programa que veremos es para aplicar el detector de objetos sobre una imagen (recuerda que para probarlo tendrás que cambiar los paths de la imagen, la arquitectura y los pesos):

```
1.  import cv2
2.
```

```
            15:"person", 16:"pottedplant",
            17:"sheep", 18:"sofa",
            19:"train", 20:"tvmonitor"}

 # Load the model
 net = cv2.dnn.readNetFromCaffe(prototxt, model)

 # ----------- READ THE IMAGE AND PREPROCESSING -----------
 image = cv2.imread("ImagesVideos/imagen_0004.jpg")
 height, width, _ = image.shape
 image_resized = cv2.resize(image, (300, 300))

 # Create a blob
 blob = cv2.dnn.blobFromImage(image_resized, 0.007843, (300, 300),
(127.5, 127.5, 127.5))
 print("blob.shape:", blob.shape)

 # ----------- DETECTIONS AND PREDICTIONS -----------
 net.setInput(blob)
 detections = net.forward()

 for detection in detections[0][0]:
      print(detection)

      if detection[2] > 0.45:
           label = classes[detection[1]]
           print("Label:", label)
           box = detection[3:7] * [width, height, width, height]
           x_start, y_start, x_end, y_end = int(box[0]),
int(box[1]), int(box[2]), int(box[3])

           cv2.rectangle(image, (x_start, y_start), (x_end, y_end),
(0, 255, 0), 2)
           cv2.putText(image, "Conf: {:.2f}".format(detection[2] *
100), (x_start, y_start - 5), 1, 1.2, (255, 0, 0), 2)
           cv2.putText(image, label, (x_start, y_start - 25), 1,
1.2, (255, 0, 0), 2)
 cv2.imshow("Image", image)
 cv2.waitKey(0)
 cv2.destroyAllWindows()
```

Luego tenemos el programa para que puedas probar este detector de objetos en video:

```
 import cv2

 # ----------- READ DNN MODEL -----------
 # Model architecture
 prototxt = "model/MobileNetSSD_deploy.prototxt.txt"
 # Weights
 model = "model/MobileNetSSD_deploy.caffemodel"
 # Class labels
```

```
net = cv2.dnn.readNetFromCaffe(prototxt, model)

# ----------- READ THE IMAGE AND PREPROCESSING -----------
cap = cv2.VideoCapture("ImagesVideos/video_0004.mp4")

while True:
    ret, frame, = cap.read()
    if ret == False:
        break

    height, width, _ = frame.shape
    frame_resized = cv2.resize(frame, (300, 300))

    # Create a blob
    blob = cv2.dnn.blobFromImage(frame_resized, 0.007843, (300, 300), (127.5, 127.5, 127.5))
    #print("blob.shape:", blob.shape)

    # ----------- DETECTIONS AND PREDICTIONS -----------
    net.setInput(blob)
    detections = net.forward()

    for detection in detections[0][0]:
        #print(detection)

        if detection[2] > 0.45:
            label = classes[detection[1]]
            #print("Label:", label)
            box = detection[3:7] * [width, height, width, height]
            x_start, y_start, x_end, y_end = int(box[0]), int(box[1]), int(box[2]), int(box[3])

            cv2.rectangle(frame, (x_start, y_start), (x_end, y_end), (0, 255, 0), 2)
            cv2.putText(frame, "Conf: {:.2f}".format(detection[2] * 100), (x_start, y_start - 5), 1, 1.2, (255, 0, 0), 2)
            cv2.putText(frame, label, (x_start, y_start - 25), 1, 1.5, (0, 255, 255), 2)


    cv2.imshow("Frame", frame)
    if cv2.waitKey(1) & 0xFF == 27:
        break
cap.release()
cv2.destroyAllWindows()
```

Si pruebas estos programas podrás obtener algo como lo siguiente:

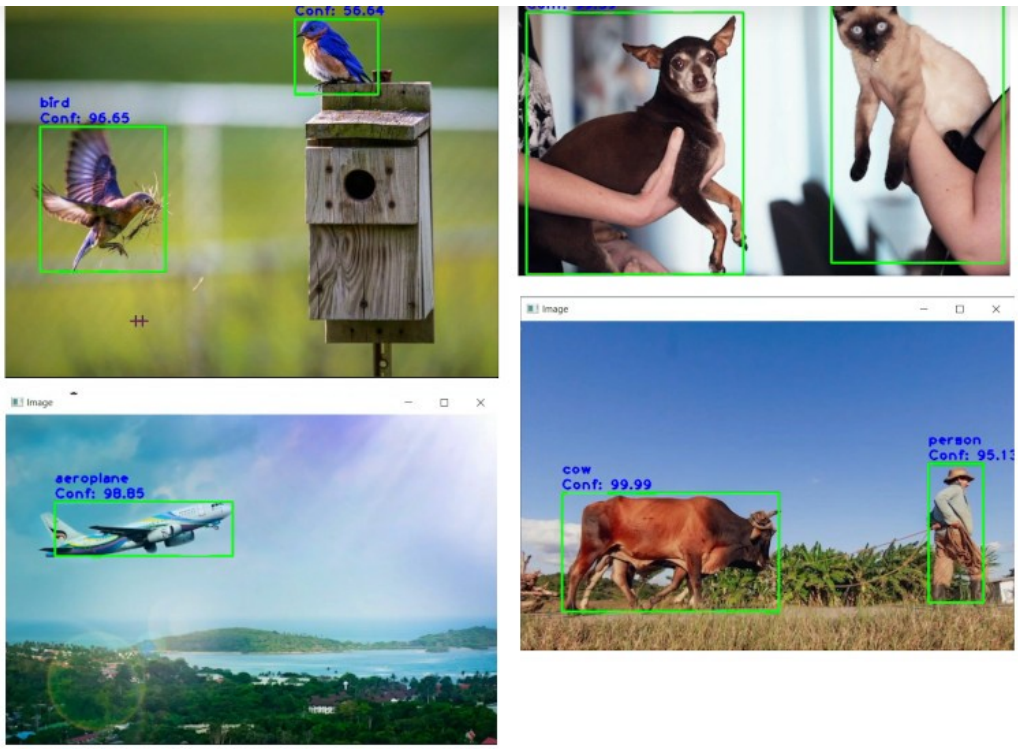


Figura 2: Detección de objetos con dnn de OpenCV, en imágenes.

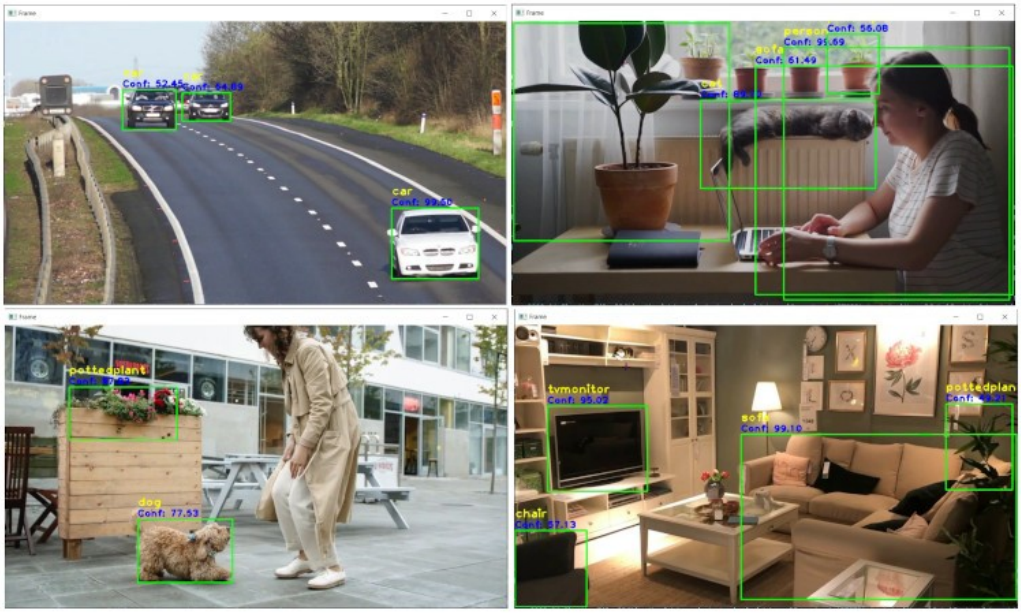


Figura 3: Detección de objetos con dnn de OpenCV, en videos.

Y bien, esto ha sido todo por el tutorial de hoy. ¡Espero que te haya gustado! 😊 Nos vemos en el siguiente... ¡Qué te vaya súper bien!.

https://github.com/opencv/opencv/blob/4.x/samples/data/dnn/object_detection_classes_pascal_voc.txt

https://github.com/opencv/opencv/wiki/Deep-Learning-in-OpenCV

Publicado en *Blog*

ANTERIOR

« Detectar rostros con modelos pre-entrenados con DNN de OpenCV | Python

SIGUIENTE

Detección de objetos con YOLOv3 (80 categorías) usando DNN de OpenCV | Python »

Un comentario en "Detección de 20 objetos con modelo pre-entrenado usando DNN de OpenCV | Python"

Carlos dice:

12 octubre, 2022 a las 2:00 pm

Hola Omes-va, que tal?

Tengo unas preguntas: cual es el tamaño mínimo q detecta el programa? Cual es el tamano minimo q se podria programar para detectar imágenes ?

Atte Carlos

Responder

Deja una respuesta

Tu dirección de correo electrónico no será publicada. Los campos obligatorios están marcados con *

Comentario *

Nombre *

Correo electrónico *

Web

Guarda mi nombre, correo electrónico y web en este navegador para la próxima vez que comente.

No soy un robot

reCAPTCHA
Privacidad - Términos

Publicar el comentario

GABRIELA SOLANO

He construido este blog para que podamos aprender juntos sobre visión por computador, recuerda que también tengo un canal de youtube donde estará información complementaría de la mayoría de los posts. ¡Vamos a divertirnos!

POPULAR POSTS

? Reconocimiento Facial ? | Python – OpenCV

26 mayo, 2020 *97*

Como crear tu propio DETECTOR DE OBJETOS con Haar Cascade | Python y OpenCV

29 julio, 2020 *58*

?? Reconocimiento de emociones ?? (EigenFaces, FisherFaces, LBPH)| Python – OpenCV

2 julio, 2020 *54*

? DETECCIÓN DE MOVIMIENTO en CIERTA ÁREA | Python – OpenCV

SOBRE EL SITIO

- Home
- Sobre mí
- Política de cookies
- Política de privacidad
- Aviso Legal

CONTENIDO

- Blog
- Tutoriales
- Videos

Sígueme