Object detection with Tensorflow model and OpenCV

Using a trained model to identify objects on static images and live video

Gabriel Cassimiro · Follow

Published in Towards Data Science · 3 min read · Jul 16, 2021

source

In this article, I'm going to demonstrate how to use a trained model to detect objects in images and videos using two of the best libraries for this kind of problem. For the detection, we need a model capable of predicting multiple classes in an image and returning the location of those objects so that we can place boxes on the image.

The Model

We are going to use a model from the Tensorflow Hub library, which has multiple ready to deploy models trained in all kinds of datasets and to solve all kinds of problems. For our use, I filtered models trained for object detection tasks and models in the TFLite format. This format is usually used for IoT applications, for its small size and faster performance than bigger

models. I choose this format because I intend to use this model on a Rasberry Pi on future projects.

The chosen model was the EfficientDet-Lite2 Object detection model. It was trained on the COCO17 dataset with 91 different labels and optimized for the TFLite application. This model returns:

1. The box boundaries of the detection;
2. The detection scores (probabilities of a given class);
3. The detection classes;
4. The number of detections.

Detecting Objects

I'm going to divide this section into two parts: Detections on static images and detection on live webcam video.

Static Images

We will start by detecting objects in this image from Unsplash:

source

So the first thing we have to do is load this image and process it to the expected format for the TensorFlow model.

```python
import tensorflow_hub as hub
import cv2
import numpy
import pandas as pd
import tensorflow as tf
import matplotlib.pyplot as plt

width = 1028
height = 1028

#Load image by Opencv2
img = cv2.imread('image_2.jpg')
#Resize to respect the input_shape
inp = cv2.resize(img, (width , height ))

#Convert img to RGB
rgb = cv2.cvtColor(inp, cv2.COLOR_BGR2RGB)

# COnverting to uint8
rgb_tensor = tf.convert_to_tensor(rgb, dtype=tf.uint8)

#Add dims to rgb_tensor
rgb_tensor = tf.expand_dims(rgb_tensor , 0)
```

object-detection-1.py hosted with ❤️ by GitHub view raw

Basically, we used OpenCV to load and do a couple of transformations on the raw image to an RGB tensor in the model format.

Now we can load the model and the labels:

```python
import tensorflow_hub as hub
import pandas as pd

# Loading model directly from TensorFlow Hub
detector = hub.load("https://tfhub.dev/tensorflow/efficientdet/lite2/detection/1")

# Loading csv with labels of classes
labels = pd.read_csv('labels.csv', sep=';', index_col='ID')
labels = labels['OBJECT (2017 REL.)']
```

object-detection-2.py hosted with ❤ by GitHub view raw

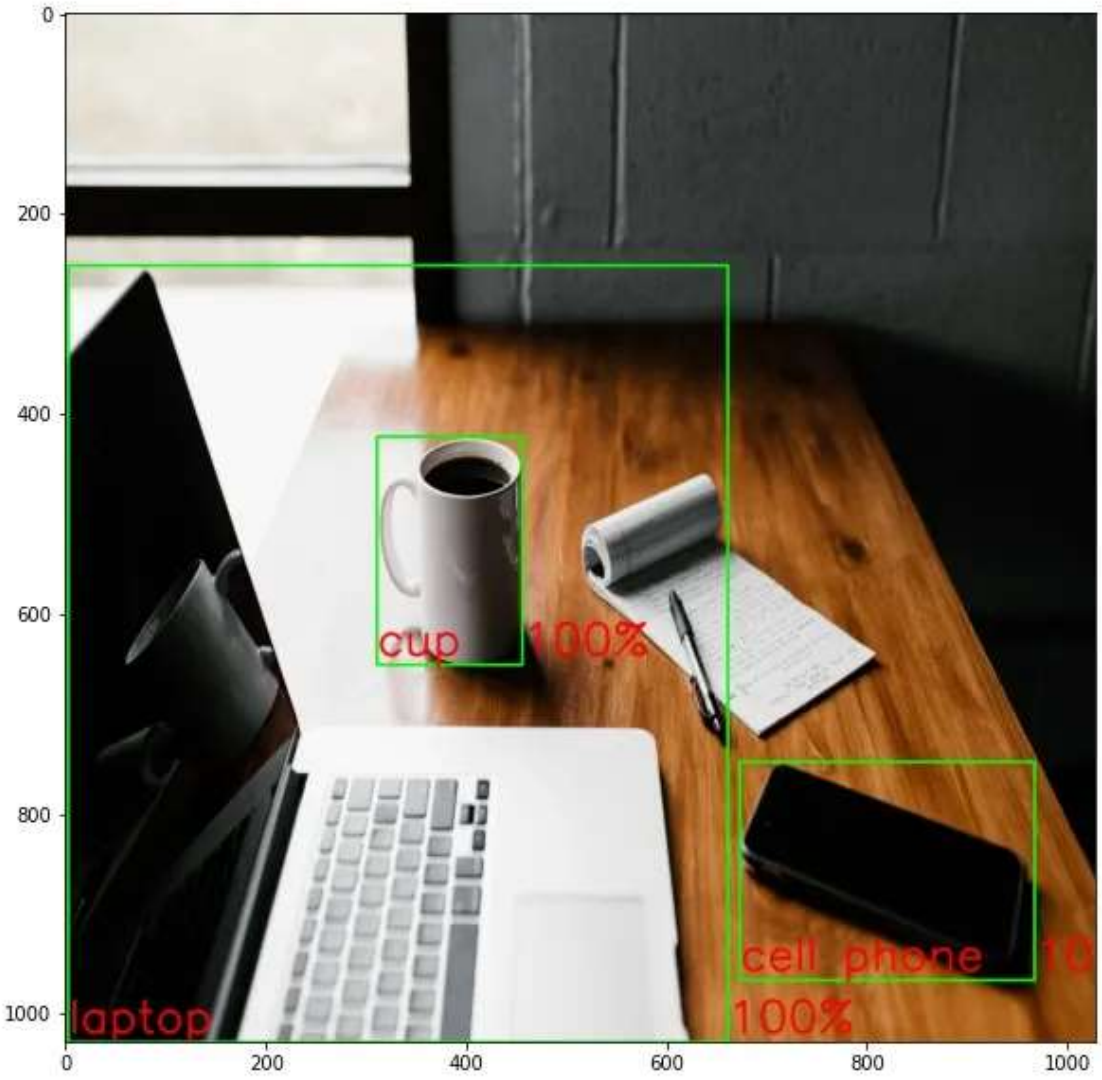
The model is being loaded directly from the website however, you can

Now we can create the predictions and put in the image the boxes and labels found:

```python
# Creating prediction
boxes, scores, classes, num_detections = detector(rgb_tensor)

# Processing outputs
pred_labels = classes.numpy().astype('int')[0]
pred_labels = [labels[i] for i in pred_labels]
pred_boxes = boxes.numpy()[0].astype('int')
pred_scores = scores.numpy()[0]

# Putting the boxes and labels on the image
for score, (ymin,xmin,ymax,xmax), label in zip(pred_scores, pred_boxes, pred_labels):
    if score < 0.5:
        continue

    score_txt = f'{100 * round(score)}%'
    img_boxes = cv2.rectangle(rgb,(xmin, ymax),(xmax, ymin),(0,255,0),2)
    font = cv2.FONT_HERSHEY_SIMPLEX
    cv2.putText(img_boxes, label,(xmin, ymax-10), font, 1.5, (255,0,0), 2, cv2.LINE_AA)
    cv2.putText(img_boxes,score_txt,(xmax, ymax-10), font, 1.5, (255,0,0), 2, cv2.LINE_AA)
```

object-detection-3.py hosted with ❤ by GitHub view raw

Now if we run *plt.imshow(img_boxes)* we get the following output:



source with modifications

Live Webcam Video

Now we can move on to detecting objects live using the webcam on your pc.

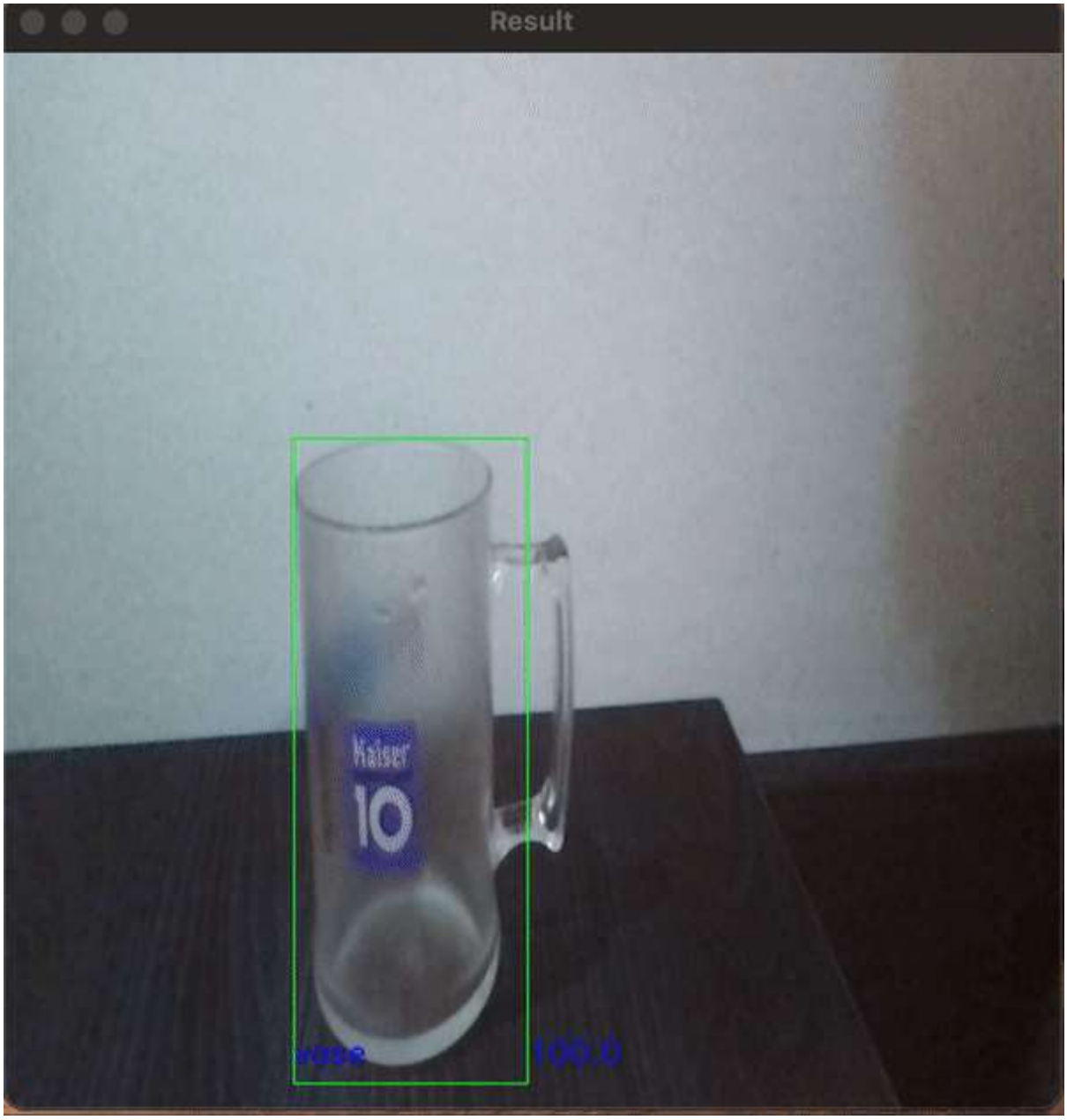
This part is not as hard as it seems, we just have to insert the code we used for one image in a loop:

```
import tensorflow_hub as hub
import cv2
import numpy
import tensorflow as tf
import pandas as pd

# Carregar modelos
detector = hub.load("https://tfhub.dev/tensorflow/efficientdet/lite2/detection/1")
labels = pd.read_csv('labels.csv',sep=';',index_col='ID')
labels = labels['OBJECT (2017 REL.)']

cap = cv2.VideoCapture(0)

width = 512
height = 512

while(True):
    #Capture frame-by-frame
    ret, frame = cap.read()

    #Resize to respect the input_shape
    inp = cv2.resize(frame, (width , height ))

    #Convert img to RGB
    rgb = cv2.cvtColor(inp, cv2.COLOR_BGR2RGB)

    #Is optional but i recommend (float convertion and convert img to tensor image)
    rgb_tensor = tf.convert_to_tensor(rgb, dtype=tf.uint8)

    #Add dims to rgb_tensor
    rgb_tensor = tf.expand_dims(rgb_tensor , 0)

    boxes, scores, classes, num_detections = detector(rgb_tensor)

    pred_labels = classes.numpy().astype('int')[0]

    pred_labels = [labels[i] for i in pred_labels]
    pred_boxes = boxes.numpy()[0].astype('int')
    pred_scores = scores.numpy()[0]

   #loop throughout the detections and place a box around it
    for score, (ymin,xmin,ymax,xmax), label in zip(pred_scores, pred_boxes, pred_labels):
        if score < 0.5:
            continue
```

```
            score_txt = f'{100 * round(score,0)}'
            img_boxes = cv2.rectangle(rgb,(xmin, ymax),(xmax, ymin),(0,255,0),1)
            font = cv2.FONT_HERSHEY_SIMPLEX
            cv2.putText(img_boxes,label,(xmin, ymax-10), font, 0.5, (255,0,0), 1, cv2.LINE_AA)
            cv2.putText(img_boxes,score_txt,(xmax, ymax-10), font, 0.5, (255,0,0), 1, cv2.LINE_AA)



        #Display the resulting frame
        cv2.imshow('black and white',img_boxes)
        if cv2.waitKey(1) & 0xFF == ord('q'):
            break

    # When everything done, release the capture
    cap.release()
    cv2.destroyAllWindows()
```

object-detection-webcam.py hosted with ❤ by GitHub view raw

Then we get:



GIF by Author

We used VideoCapture from open cv to load the video from the computer webcam. Then we did the same processing that we used on the static image and predicted the labels and positions. The main difference is that the image input is continuous so we inserted the code inside a while loop.

All the code and notebooks used are in this repository:

gabrielcassimiro17/raspberry-pi-tensorflow

Contribute to gabrielcassimiro17/raspberry-pi-tensorflow development by creating an account on GitHub.

github.com

In the near future, I will load this into a raspberry pi to create some interactions using a model capable of detecting objects, and post the results here.

If you like the content and want to support me, you can buy me a coffee:

Gabriel Cassimiro is a Data Scientist sharing free content to the community

Hey 👋 I just created a page here. You can now buy me a coffee!

www.buymeacoffee.com

Deep Learning | Object Detection | TensorFlow | Opencv | Artificial Intelligence

Written by Gabriel Cassimiro

Follow

183 Followers · Writer for Towards Data Science

Solving and creating problems with AI. Google Developer Expert in ML

More from Gabriel Cassimiro and Towards Data Science

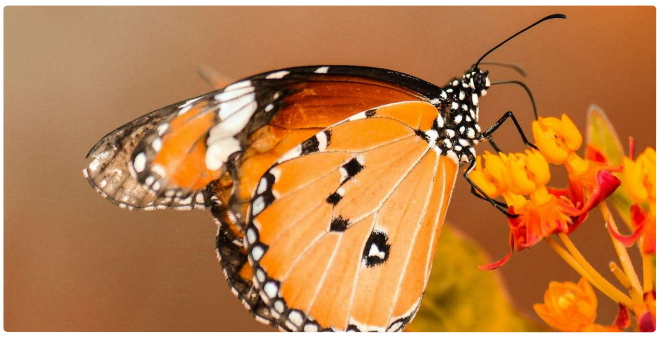
Gabriel Cassimiro in Towards Data Science

LLM Output Parsing Function Calling vs. LangChain

How to consistently parse outputs from LLMs using Function Calling of Open AI API and...

11 min read · Sep 22

233



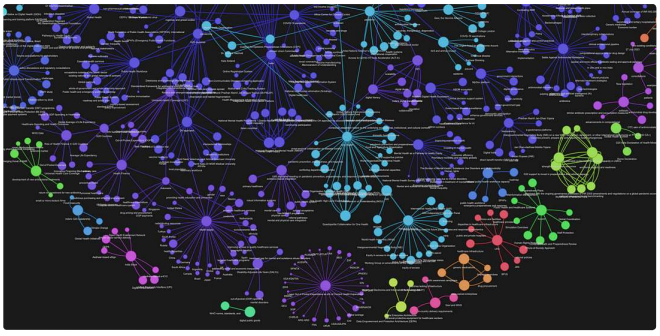
Marco Peixeiro in Towards Data Science

TimeGPT: The First Foundation Model for Time Series Forecasting

Explore the first generative pre-trained forecasting model and apply it in a project...

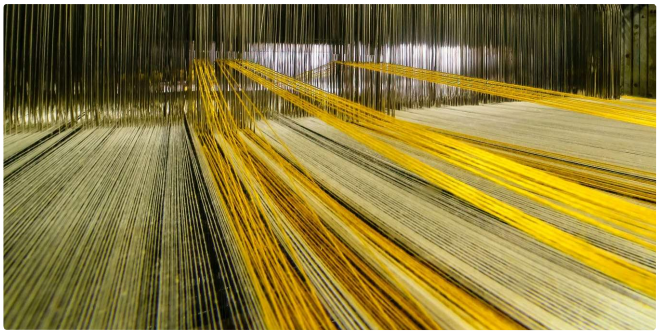
✦ · 12 min read · Oct 24

2.3K 21



Rahul Nayak in Towards Data Science

How to Convert Any Text Into a Graph of Concepts



Gabriel Cassimiro in Towards Data Science

Async calls for Chains with Langchain

A method to convert any text corpus into a Knowledge Graph using Mistral 7B.

12 min read · Nov 10

1.4K 23

How to make Langchain chains work with Async calls to LLMs, speeding up the time it...

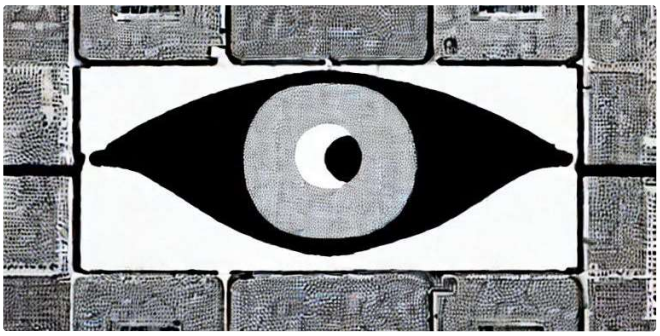
6 min read · Jul 10

357 3

See all from Gabriel Cassimiro

See all from Towards Data Science

Recommended from Medium

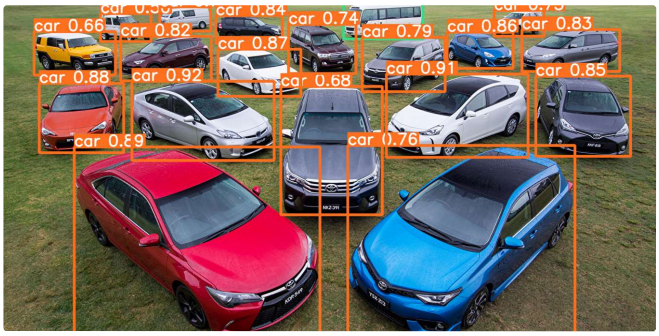


albertqueralto

Installing OpenCV within Docker containers for Computer Vision...

7 min read · Jul 26

6



AI_Pioneer

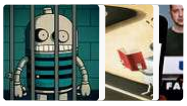
Object Detection with YOLO and OpenCV: A Practical Guide

Object detection is a fundamental computer vision task that involves identifying and...

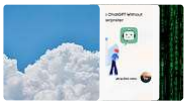
4 min read · Jun 28

30

Lists



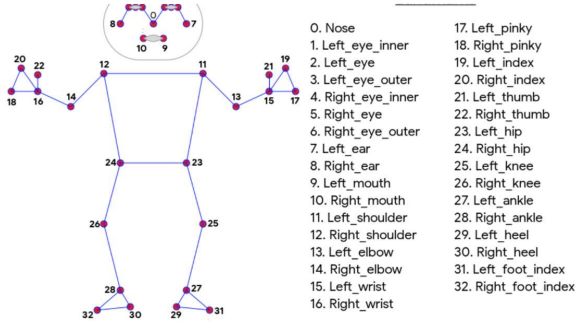
AI Regulation
6 stories · 188 saves



ChatGPT
22 stories · 267 saves

Natural Language Processing
863 stories · 407 saves

ChatGPT prompts
30 stories · 681 saves



Elven Kim

Easy calculation of Angle using OpenCV: Contours and Coordinat...

Detecting an angle of an coloured object is not easy especially with complex backgroun...

3 min read · Aug 9

1

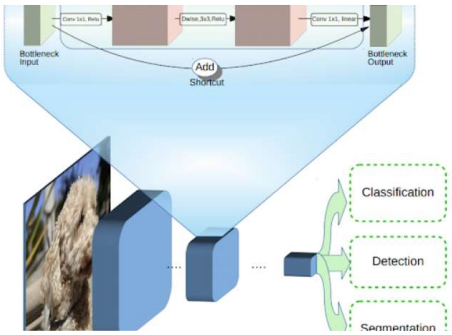
Juan Couto

ML Kit and Pose Detection

These days we are always listening about a new ML app and some of them seem to be...

5 min read · Jun 8

1



Tahera Firdose

Transfer Learning with MobileNet

Deep learning has revolutionized the field of computer vision by enabling machines to...

11 min read · Aug 27

Mustafa Celik

Converting TensorFlow .pb Model to .tflite Model: A Step-by-Step...

Learn how to convert a TensorFlow .pb model to .tflite format effortlessly. Optimize your...

✦ · 2 min read · Aug 11

10

See more recommendations