

# Détection d'objets TensorFlow Lite sur Android et Raspberry-Pi



## I. Introduction

TensorFlow Lite est un cadre optimisé pour le déploiement de modèles légers d'apprentissage en profondeur sur des appareils périphériques limités en ressources. Les modèles TensorFlow Lite ont un temps d'inférence plus rapide et nécessitent moins de puissance de traitement, ils peuvent donc être utilisés pour obtenir des performances plus rapides dans les applications en temps réel. Ce guide fournit des instructions pas à pas sur la façon de former un modèle de détection d'objets TensorFlow personnalisé, de le convertir en un format optimisé qui peut être utilisé par TensorFlow Lite et de l'exécuter sur des téléphones Android ou le Raspberry Pi.

### Une note sur les versions

J'ai utilisé TensorFlow v1.13 lors de la création de ce guide, car TF v1.13 est une version stable qui bénéficie d'un excellent support d'Anaconda. Je mettrai périodiquement à jour le guide pour m'assurer qu'il fonctionne avec les nouvelles versions de TensorFlow.

L'équipe TensorFlow est toujours à pied d'œuvre pour publier des versions mises à jour de TensorFlow. Je recommande de choisir une version et de la respecter pour tous vos projets TensorFlow. Chaque partie de ce guide devrait fonctionner avec des versions plus récentes ou plus anciennes, mais vous devrez peut-être utiliser différentes versions des outils nécessaires pour exécuter ou créer TensorFlow (CUDA, cuDNN, bazel, etc.). Google a fourni une liste de configurations de build pour Linux, macOS et Windows qui montrent quelles versions d'outils ont été utilisées pour construire et exécuter chaque version de TensorFlow.

## Section 1 - Comment configurer et exécuter des modèles de détection d'objets TensorFlow Lite sur le Raspberry Pi

La configuration de TensorFlow Lite sur le Raspberry Pi est beaucoup plus facile que TensorFlow ordinaire ! Voici les étapes nécessaires pour configurer TensorFlow Lite :

- Mettre à jour le Raspberry Pi
- Téléchargez ce référentiel et créez un environnement virtuel
- Installer TensorFlow et OpenCV
- Configurer le modèle de détection TensorFlow Lite
- Exécutez le modèle TensorFlow Lite !

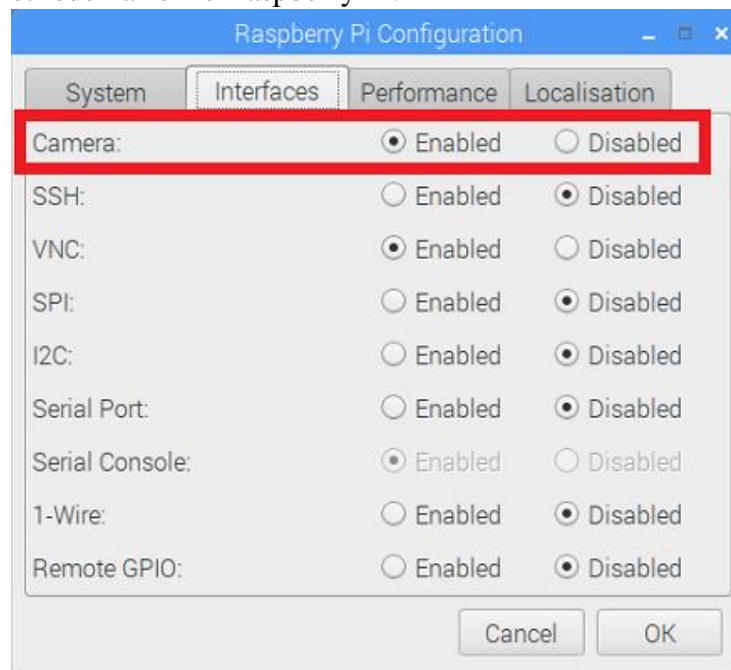
### Étape 1a. Mettre à jour le Raspberry Pi

Tout d'abord, le Raspberry Pi doit être entièrement mis à jour. Ouvrez un terminal et lancez :

```
sudo apt-get update  
sudo apt-get dist-upgrade
```

Selon le temps écoulé depuis la mise à jour de votre Pi, la mise à jour peut durer entre une minute et une heure.

Pendant que nous y sommes, assurons-nous que l'interface de la caméra est activée dans le menu de configuration du Raspberry Pi. Cliquez sur l'icône Pi dans le coin supérieur gauche de l'écran, sélectionnez Préférences -> Configuration du Raspberry Pi, allez dans l'onglet Interfaces et vérifiez que la caméra est définie sur Activé. Si ce n'est pas le cas, activez-le maintenant et redémarrez le Raspberry Pi.



## Étape 1b. Téléchargez ce référentiel et créez un environnement virtuel

Ensuite, clonez ce référentiel GitHub en émettant la commande suivante. Le référentiel contient les scripts que nous utiliserons pour exécuter TensorFlow Lite, ainsi qu'un script shell qui facilitera l'installation de tout. Problème:

```
git clone https://github.com/EdgeElectronics/TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi.git
```

Cela télécharge tout dans un dossier nommé TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi. C'est un peu long à travailler, alors renommez le dossier en "tflite 1" puis cd dedans:

```
mv TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi tflite1  
cd tflite1
```

Nous travaillerons dans ce répertoire / home / pi / tflite1 pour le reste du guide. La prochaine étape consiste à créer un environnement virtuel appelé "tflite1-env."

J'utilise un environnement virtuel pour ce guide car il empêche tout conflit entre les versions des bibliothèques de packages qui peuvent déjà être installées sur votre Pi. Le garder installé dans son propre environnement nous permet d'éviter ce problème. Par exemple, si vous avez déjà installé TensorFlow v1.8 sur le Pi à l'aide de mon autre guide, vous pouvez laisser cette installation telle quelle sans avoir à vous soucier de la remplacer.

Installez virtualenv en émettant :

```
sudo pip3 install virtualenv
```

Ensuite, créez l'environnement virtuel "tflite1-env" en émettant :

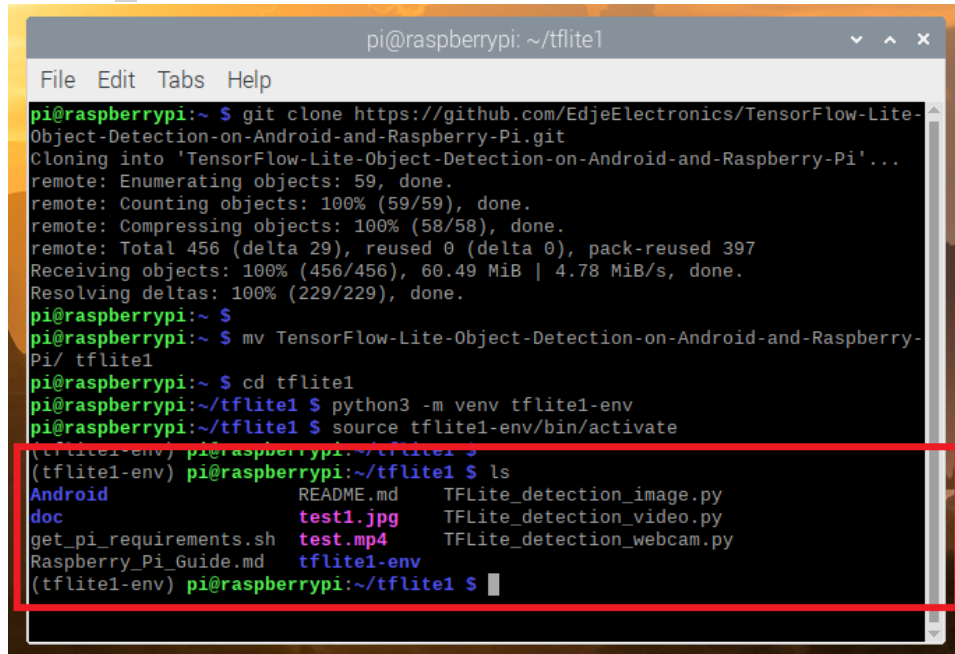
```
python3 -m venv tflite1-env
```

Cela créera un dossier appelé tflite1-env dans le répertoire tflite1. Le dossier tflite1-env contiendra toutes les bibliothèques de packages pour cet environnement. Ensuite, activez l'environnement en émettant :

```
source tflite1-env/bin/activate
```

**Vous devrez émettre la commande `tflite1-env / bin / activate` source depuis le répertoire `/ home / pi / tflite1` pour réactiver l'environnement chaque fois que vous ouvrez une nouvelle fenêtre de terminal. Vous pouvez savoir quand l'environnement est actif en vérifiant si `(tflite1-env)` apparaît avant le chemin dans votre invite de commande, comme indiqué dans la capture d'écran ci-dessous.**

À ce stade, voici à quoi devrait ressembler votre répertoire `tflite1` si vous émettez `ls`.



```
pi@raspberrypi: ~/tflite1
File Edit Tabs Help
pi@raspberrypi:~$ git clone https://github.com/EdgeElectronics/TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi.git
Cloning into 'TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi'...
remote: Enumerating objects: 59, done.
remote: Counting objects: 100% (59/59), done.
remote: Compressing objects: 100% (58/58), done.
remote: Total 456 (delta 29), reused 0 (delta 0), pack-reused 397
Receiving objects: 100% (456/456), 60.49 MiB | 4.78 MiB/s, done.
Resolving deltas: 100% (229/229), done.
pi@raspberrypi:~$
pi@raspberrypi:~$ mv TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi/ tflite1
pi@raspberrypi:~$ cd tflite1
pi@raspberrypi:~/tflite1$ python3 -m venv tflite1-env
pi@raspberrypi:~/tflite1$ source tflite1-env/bin/activate
(tflite1-env) pi@raspberrypi:~/tflite1$
(tflite1-env) pi@raspberrypi:~/tflite1$ ls
Android          README.md        TFLite_detection_image.py
doc              test1.jpg        TFLite_detection_video.py
get_pi_requirements.sh  test.mp4         TFLite_detection_webcam.py
Raspberry_Pi_Guide.md  tflite1-env
(tflite1-env) pi@raspberrypi:~/tflite1$
```

Si votre répertoire semble bon, il est temps de passer à l'étape 1c!

## Étape 1c. Installer les dépendances TensorFlow Lite et OpenCV

Ensuite, nous installerons TensorFlow, OpenCV et toutes les dépendances nécessaires pour les deux packages. OpenCV n'est pas nécessaire pour exécuter TensorFlow Lite, mais les scripts de détection d'objets de ce référentiel l'utilisent pour saisir des images et dessiner des résultats de détection dessus.

Pour faciliter les choses, j'ai écrit un script shell qui téléchargera et installera automatiquement tous les packages et dépendances. Exécutez-le en émettant:

```
bash get_pi_requirements.sh
```

Cela télécharge environ 400 Mo de fichiers d'installation, cela prendra donc un certain temps. Allez prendre une tasse de café pendant que ça marche ! Si vous souhaitez voir tout ce qui est installé, ouvrez simplement `get_pi_dependencies.sh` pour afficher la liste des packages.

REMARQUE : Si vous obtenez une erreur lors de l'exécution de la commande `bash get_pi_requirements.sh`, c'est probablement parce que votre connexion Internet a expiré ou parce que les données du package téléchargé ont été corrompues. Si vous obtenez une erreur, essayez de réexécuter la commande plusieurs fois.

AUTRE REMARQUE: Le script shell installe automatiquement la dernière version de TensorFlow. Si vous souhaitez installer une version spécifique, lancez `pip3 install tensorflow == X.XX` (où X.XX est remplacé par la version que vous souhaitez installer) après avoir exécuté le script. Cela remplacera l'installation existante par la version spécifiée.

C'était facile! Passez à l'étape suivante.

### **Étape 1d. Configurer le modèle de détection TensorFlow Lite**

Ensuite, nous allons configurer le modèle de détection qui sera utilisé avec TensorFlow Lite. Ce guide montre comment télécharger un exemple de modèle TFLite fourni par Google, ou comment utiliser un modèle que vous avez vous-même formé en suivant la partie 1 de ma série de didacticiels TensorFlow Lite.

Un modèle de détection est associé à deux fichiers : un fichier `detect.tflite` (qui est le modèle lui-même) et un fichier `labelmap.txt` (qui fournit une carte d'étiquettes pour le modèle). Ma façon préférée d'organiser les fichiers de modèle est de créer un dossier (tel que "BirdSquirrelRaccoon\_TFLite\_model") et de conserver à la fois le `detect.tflite` et le `labelmap.txt` dans ce dossier. C'est également de cette façon que l'exemple de modèle téléchargeable TFLite de Google est organisé.

### **Option 1. Utilisation de l'exemple de modèle TFLite de Google**

Google fournit un exemple de modèle de détection d'objets SSDLite-MobileNet-v2 quantifié qui est formé à partir du jeu de données MSCOCO et converti pour s'exécuter sur TensorFlow Lite. Il peut détecter et identifier 80 objets communs différents, tels que des personnes, des voitures, des tasses, etc.

Téléchargez l'exemple de modèle (qui se trouve sur la page Détection d'objets du site Web officiel de TensorFlow) en publiant:

```
wget  
https://storage.googleapis.com/download.tensorflow.org/models/tflite/coco\_ssd\_mobilenet\_v1\_1.0\_quant\_2018\_06\_29.zip
```

Décompressez-le dans un dossier appelé "Sample\_TFLite\_model" en émettant (cette commande crée automatiquement le dossier):

```
Unzip          coco_ssd_mobilenet_v1_1.0_quant_2018_06_29.zip-d
Sample_TFLite_model
```

D'accord, l'exemple de modèle est prêt à l'emploi!

## **Option 2: utiliser votre propre modèle personnalisé**

Vous pouvez également utiliser un modèle de détection d'objet personnalisé en déplaçant le dossier du modèle dans le répertoire / home / pi / tflite. Si vous avez suivi la partie 1 de mon guide TensorFlow Lite pour former et convertir un modèle TFLite sur votre PC, vous devriez avoir un dossier nommé "TFLite\_model" avec un fichier detect.tflite et labelmap.txt. (Il aura également un fichier tflite\_graph.pb et tflite\_graph.pbtxt, qui ne sont pas nécessaires à TensorFlow Lite mais peuvent être laissés dans le dossier.)

Vous pouvez simplement copier ce dossier sur une clé USB, insérer la clé USB dans votre Raspberry Pi et déplacer le dossier dans le répertoire / home / pi / tflite1. (Ou vous pouvez vous l'envoyer par e-mail, ou le mettre sur Google Drive, ou faire votre méthode de transfert de fichiers préférée.) Voici un exemple de ce à quoi ressemble mon dossier "BirdSquirrelRaccoon\_TFLite\_model" dans mon répertoire / home / pi / tflite1:

( Ajouter une image de BirdSquirrelRaccoon\_TFLite\_model dans mon répertoire / home / pi / tflite1)

Maintenant, votre modèle personnalisé est prêt à l'emploi!

## **Étape 1e. Exécutez le modèle TensorFlow Lite!**

Il est temps de voir le modèle de détection d'objets TFLite en action ! Tout d'abord, libérez de la mémoire et de la puissance de traitement en fermant toutes les applications que vous n'utilisez pas. Assurez-vous également que votre webcam ou Picamera est branchée.

Exécutez le script de détection de webcam en temps réel en exécutant la commande suivante à partir du répertoire / home / pi / tflite1. (Avant d'exécuter la commande, assurez-vous que l'environnement tflite1-env est actif en vérifiant que (tflite1-env) apparaît devant l'invite de commande.) **Le script TFLite\_detection\_webcam.py fonctionnera avec une Picamera ou une webcam USB.**

```
python3 TFLite_detection_webcam.py --modeldir=Sample_TFLite_model
```

Si votre dossier de modèles porte un nom différent de "Sample\_TFLite\_model", utilisez plutôt ce nom. Par exemple, j'utiliserais `-modeldir = BirdSquirrelRaccoon_TFLite_model` pour exécuter mon modèle de détection d'oiseaux, d'écureuils et de rats laveurs personnalisés. Après quelques instants d'initialisation, une fenêtre apparaîtra montrant le flux webcam. Les objets détectés auront des boîtes englobantes et des étiquettes affichées en temps réel.

La partie 3 de mon guide de formation TensorFlow Lite donne des instructions sur l'utilisation des scripts `TFLite_detection_image.py` et `TFLite_detection_video.py`. Assurez-vous d'utiliser python3 plutôt que python lors de l'exécution des scripts.

## **Section 2 - Exécuter des modèles de détection d'objets TPU Edge sur le Raspberry Pi à l'aide de l'accélérateur USB Coral**

L'accélérateur USB Coral est un accessoire matériel USB pour accélérer les modèles TensorFlow. Vous pouvez en acheter un [ici](#) (lien Amazon Associate).

(Ajouter une image de l'accélérateur USB et de la puce Edge TPU)

L'accélérateur USB utilise l'Edge TPU (unité de traitement du tenseur), qui est une puce ASIC (circuit intégré spécifique à l'application) spécialement conçue avec des ALU hautement parallélisées (unités logiques arithmétiques). Alors que les GPU (unités de traitement graphique) ont également de nombreuses ALU parallélisées, le TPU a une différence clé : les ALU sont directement connectés entre elles. La sortie d'une ALU peut être directement passée à l'entrée de la prochaine ALU sans avoir à être stockée et récupérée à partir d'un tampon de mémoire. La parallélisation extrême et la suppression du goulot d'étranglement de la mémoire signifie que le TPU peut effectuer jusqu'à 4 billions d'opérations arithmétiques par seconde ! C'est parfait pour exécuter des réseaux de neurones profonds, qui nécessitent des millions d'opérations de multiplication-accumulation pour générer des sorties à partir d'un seul lot de données d'entrée.

Ma maîtrise était en conception ASIC, donc le Edge TPU est très intéressant pour moi ! Si vous êtes un nerd de l'architecture informatique comme moi et que vous souhaitez en savoir plus sur le Edge TPU, voici un excellent article qui explique comment il fonctionne.

Il rend les modèles de détection d'objets plus rapides et plus faciles à configurer. Voici les étapes que nous allons parcourir pour configurer l'accélérateur USB Coral:

2a. Installez la bibliothèque `libedgetpu` 2b. Configurez le modèle de détection Edge TPU 2c. Exécutez une détection ultra-rapide!



Cette section du guide suppose que vous avez déjà terminé la section 1 pour configurer la détection d'objets TFLite sur le Pi. Si vous n'avez pas fait cette partie, faites défiler vers le haut et parcourez-la d'abord.

## Étape 2a. Installer la bibliothèque libedgetpu

Tout d'abord, nous allons télécharger et installer le runtime Edge TPU, qui est la bibliothèque nécessaire pour s'interfacer avec l'accélérateur USB. Ces instructions suivent le guide de configuration de l'accélérateur USB du site Web officiel de Coral.

Ouvrez un terminal de commande et accédez au répertoire / home / pi / tflite1 et activez l'environnement virtuel tflite1-env en émettant :

```
cd /home/pi/tflite1
source tflite1-env/bin/activate
```



Ajoutez le référentiel de packages Coral à votre liste de distribution apt-get en émettant les commandes suivantes :

```
echo "deb https://packages.cloud.google.com/apt coral-edgetpu-
stable main" | sudo tee /etc/apt/sources.list.d/coral-edgetpu.list
curl https://packages.cloud.google.com/apt/doc/apt-key.gpg |
sudo apt-key add-
sudo apt-get update
```

Installez la bibliothèque libedgetpu en émettant :

```
sudo apt-get install libedgetpu1-std
```

Vous pouvez également installer la bibliothèque libedgetpu1-max, qui exécute l'accélérateur USB à une fréquence overclockée, ce qui lui permet d'obtenir des fréquences d'images encore plus rapides. Cependant, cela accélère également l'accélérateur USB. Voici les fréquences d'images que j'obtiens lors de l'exécution de TFLite\_detection\_webcam.py avec une résolution de 1280x720 pour chaque option avec un modèle Raspberry Pi 4 4 Go:

 libedgetpu1-std: 22,6 FPS  
 libedgetpu1-max: 26,1 FPS

Je n'ai pas mesuré la température de l'accélérateur USB, mais il devient un peu plus chaud au toucher avec la version libedgetpu1-max. Cependant, il ne semblait pas assez chaud pour être dangereux ou dangereux pour l'électronique.



Si vous souhaitez utiliser la bibliothèque libedgetpu-max, installez-la en utilisant `sudo apt-get install libedgetpu1-max`. (Vous ne pouvez pas installer les bibliothèques -std et -max. Si vous installez la bibliothèque -max, la bibliothèque -std sera automatiquement désinstallée.)

Bien! Maintenant que le runtime libedgetpu est installé, il est temps de configurer un modèle de détection Edge TPU avec lequel l'utiliser.

### **Étape 2b. Configurer le modèle de détection Edge TPU**

Les modèles Edge TPU sont des modèles TensorFlow Lite qui ont été compilés spécifiquement pour fonctionner sur des périphériques Edge TPU comme l'accélérateur USB Coral. Ils résident dans un fichier .tflite et sont utilisés de la même manière qu'un modèle TF Lite normal. Ma méthode préférée consiste à conserver le fichier Edge TPU dans le même dossier de modèle que le modèle TFLite à partir duquel il a été compilé, et à le nommer "edgetpu.tflite."

Je vais vous montrer deux options pour configurer un modèle Edge TPU: en utilisant l'exemple de modèle de Google ou en utilisant un modèle personnalisé que vous avez compilé vous-même.

### **Option 1. Utilisation de l'exemple de modèle EdgeTPU de Google**

Google fournit un exemple de modèle TPU Edge qui est compilé à partir du SSDLite-MobileNet-v2 quantifié que nous avons utilisé à l'étape 1e. Téléchargez-le et déplacez-le dans le dossier Sample\_TFLite\_model (tout en le renommant simultanément "edgetpu.tflite") en lançant ces commandes:

```
wget
https://dl.google.com/coral/canned_models/mobilenet_ssd_v2_c
oco_quant_postprocess_edgetpu.tflite

mv  mobilenet_ssd_v2_coco_quant_postprocess_edgetpu.tflite
Sample_TFLite_model/edgetpu.tflite
```

Maintenant, l'exemple de modèle Edge TPU est prêt à l'emploi. Il utilisera le même fichier labelmap.txt que le modèle TFLite, qui devrait déjà se trouver dans le dossier Sample\_TFLite\_model.

### **Option 2. Utilisation de votre propre modèle EdgeTPU personnalisé**

Si vous avez formé un modèle de détection TFLite personnalisé, vous pouvez le compiler pour l'utiliser avec le TPU Edge. Malheureusement, le

package `edgetpu-compiler` ne fonctionne pas sur le Raspberry Pi: vous avez besoin d'un PC Linux pour l'utiliser. La section 3 de ce guide donnera quelques options pour compiler votre propre modèle si vous n'avez pas de boîte Linux. Pendant que je travaille à l'écriture, voici les instructions officielles qui montrent comment compiler un modèle Edge TPU à partir d'un modèle TFLite.

En supposant que vous avez été en mesure de compiler votre modèle TFLite dans un modèle EdgeTPU, vous pouvez simplement copier le fichier `.tflite` sur une clé USB et le transférer dans le dossier du modèle sur votre Raspberry Pi. Pour mon exemple `"BirdSquirrelRaccoon_TFLite_model"` de l'étape 1e, je peux compiler mon `"BirdSquirrelRaccoon_TFLite_model"` sur un PC Linux, mettre le fichier `edgetpu.tflite` résultant sur un USB, transférer l'USB sur mon Pi et déplacer le fichier `edgetpu.tflite` dans le / dossier `home / pi / tflite1 / BirdSquirrelRaccoon_TFLite_model`. Il utilisera le même fichier `labelmap.txt` qui existe déjà dans le dossier pour obtenir ses étiquettes.

Une fois le fichier `edgetpu.tflite` déplacé dans le dossier modèle, il est prêt à partir!

### Étape 2c. Exécutez la détection avec Edge TPU!

Maintenant que tout est configuré, il est temps de tester la vitesse de détection ultra-rapide du Coral! Assurez-vous de libérer de la mémoire et de la puissance de traitement en fermant tous les programmes que vous n'utilisez pas. Assurez-vous qu'une webcam est branchée.

Branchez votre accélérateur USB Coral sur l'un des ports USB du Raspberry Pi. Si vous utilisez un Pi 4, assurez-vous de le brancher sur l'un des ports USB 3.0 bleus.

Insérez ici l'image de l'accélérateur USB Coral branché sur le Raspberry Pi!

Assurez-vous que l'environnement `tflite1-env` est activé en vérifiant que (`tflite1-env`) apparaît devant l'invite de commande dans votre terminal. Ensuite, exécutez le script de détection de webcam en temps réel avec l'argument `--edgetpu`:

```
python3 TFLite_detection_webcam.py --modeldir=Sample_TFLite_model
--edgetpu
```

The `--edgetpu` argument tells the script to use the Coral USB Accelerator and the EdgeTPU-compiled `.tflite` file. If your model folder has a different name than `"Sample_TFLite_model"`, use that name instead.

After a brief initialization period, a window will appear showing the webcam feed with detections drawn on each from. The detection will run SIGNIFICANTLY faster with the Coral USB Accelerator.

If you'd like to run the video or image detection scripts with the Accelerator, use these commands:

```
python3 TFLite_detection_video.py --modeldir=Sample_TFLite_model --edgetpu
python3 TFLite_detection_image.py --modeldir=Sample_TFLite_model --edgetpu
```

Amusez-vous avec les vitesses de détection fulgurantes de l'accélérateur USB Coral!

## Section 3 - Compilation de modèles de détection d'objets TPU Edge personnalisés

### Annexe : Erreurs courantes

Cette annexe répertorie les erreurs courantes rencontrées par les utilisateurs à la suite de ce guide, ainsi que les solutions montrant comment les résoudre.

**N'hésitez pas à créer des demandes de tirage pour ajouter vos propres erreurs et résolutions ! J'apprécierais toute aide.**

#### **1 .TypeError: l'argument int () doit être une chaîne, un objet de type octets ou un nombre, pas 'NoneType'**

L'erreur «NoneType» signifie que le programme a reçu un tableau vide de la webcam, ce qui signifie généralement que quelque chose ne va pas avec la webcam ou l'interface avec la webcam. Essayez de brancher et de rebrancher la webcam plusieurs fois et / ou de redémarrer le Raspberry Pi, et voyez si cela fonctionne. Sinon, vous devrez peut-être essayer d'utiliser une nouvelle webcam.

#### **2 .ImportError: aucun module nommé «cv2»**

Cette erreur se produit lorsque vous essayez d'exécuter l'un des scripts TFLite\_detection sans activer d'abord le 'tflite1-env'. Cela se produit car Python ne peut pas trouver le chemin d'accès à la bibliothèque OpenCV (cv2) pour l'importer.

Résolvez le problème en fermant la fenêtre de votre terminal, en la rouvrant et en émettant :

```
cd tflite1
source tflite1-env/bin/activate
```

Ensuite, essayez de réexécuter le script comme décrit à l'étape 1e.

### **3 .CES FORFAITS NE CORRESPONDENT PAS AUX HACHTS DU FICHIER DES EXIGENCES**

Cette erreur peut se produire lorsque vous exécutez la commande `bash get_pi_requirements.sh` à l'étape 1c. Cela se produit car les données du package ont été corrompues lors du téléchargement. Vous pouvez résoudre l'erreur en réexécutant la commande `bash get_pi_requirements.sh` plusieurs fois jusqu'à ce qu'elle se termine avec succès sans signaler cette erreur.