**How to Run TensorFlow Lite Models on Windows**

This guide shows how to set up a TensorFlow Lite Runtime environment on a Windows PC. We'll use [Anaconda](https://www.anaconda.com/) to create a Python environment to install the TFLite Runtime in. It's easy!

**Step 1. Download and Install Anaconda**

First, install [Anaconda](https://www.anaconda.com/), which is a Python environment manager that greatly simplifies Python package management and deployment. Anaconda allows you to create Python virtual environments on your PC without interfering with existing installations of Python. Go to the [Anaconda Downloads page](https://www.anaconda.com/products/distribution) and click the Download button.

When the download finishes, open the downloaded .exe file and step through the installation wizard. Use the default install options.

**Step 2. Set Up Virtual Environment and Directory**

Go to the Start Menu, search for "Anaconda Command Prompt", and click it to open up a command terminal. We'll create a folder called tflite1 directly in the C: drive. (You can use any other folder location you like, just make sure to modify the commands below to use the correct file paths.) Create the folder and move into it by issuing the following commands in the terminal:

mkdir C:\tflite1

cd C:\tflite1

Next, create a Python 3.9 virtual environment by issuing:

conda create --name tflite1-env python=3.9

Enter "y" when it asks if you want to proceed. Activate the environment and install the required packages by issuing the commands below. We'll install TensorFlow, OpenCV, and a downgraded version of protobuf. TensorFlow is a pretty big download (about 450MB), so it will take a while.

conda activate tflite1-env

pip install tensorflow opencv-python protobuf==3.20.\*

Download the detection scripts from this repository by issuing:

curl https://raw.githubusercontent.com/EdjeElectronics/TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi/master/TFLite\_detection\_image.py --output TFLite\_detection\_image.py

curl https://raw.githubusercontent.com/EdjeElectronics/TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi/master/TFLite\_detection\_video.py --output TFLite\_detection\_video.py

curl https://raw.githubusercontent.com/EdjeElectronics/TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi/master/TFLite\_detection\_webcam.py --output TFLite\_detection\_webcam.py

curl https://raw.githubusercontent.com/EdjeElectronics/TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi/master/TFLite\_detection\_stream.py --output TFLite\_detection\_stream.py

**Step 3. Move TFLite Model into Directory**

Next, take the custom TFLite model that was trained and downloaded from the Colab notebook and move it into the C:\tflite1 directory. If you downloaded it from Colab, it should be in a file called custom\_model\_lite.zip. (If you haven't trained a model yet and just want to test one out, download my "change counter" model by clicking this [Dropbox link](https://www.dropbox.com/scl/fi/4fk8ls8s03c94g6sb3ngo/custom_model_lite.zip?rlkey=zqda21sowk0hrw6i3f2dgbsyy&dl=0).) Move that file to the C:\tflite1 directory. Once it's moved, unzip it using:

tar -xf custom\_model\_lite.zip

At this point, you should have a folder at C:\tflite1\custom\_model\_lite which contains at least a detect.tflite and labelmap.txt file.

**Step 4. Run TensorFlow Lite Model!**

Alright! Now that everything is set up, running the TFLite model is easy. Just call one of the detection scripts and point it at your model folder with the --modeldir option. For example, to run your custom\_model\_lite model on a webcam, issue:

python TFLite\_detection\_webcam.py --modeldir=custom\_model\_lite

A window will appear showing detection results drawn on the live webcam feed. For more information on how to use the detection scripts, please see [Step 3 in the main README page](https://github.com/EdjeElectronics/TensorFlow-Lite-Object-Detection-on-Android-and-Raspberry-Pi#step-3-run-tensorflow-lite-models).

Have fun using TensorFlow Lite! Stay tuned for more examples on how to build cool applications around your model.