Install tensorflow-lite on Penny whistle & Piccolo platform

Table of Contents

Objective:

Install tensorflow-lite on Penny whistle & Piccolo platform

TensorFlow Lite interpreter

The TensorFlow Lite interpreter is a library that takes a model file, executes the operations it defines on input data, and provides access to the output.

The interpreter works across multiple platforms and provides a simple API for running TensorFlow Lite models from Java, Swift, Objective-C, C++, and Python.

https://www.tensorflow.org/lite/guide/get_started

https://www.tensorflow.org/lite/guide/python

Install just the TensorFlow Lite interpreter

To quickly run TensorFlow Lite models with Python, you can install just the TensorFlow Lite interpreter, instead of all TensorFlow packages.

This interpreter-only package is a fraction the size of the full TensorFlow package and includes the bare minimum code required to run inferences with TensorFlow Lite—it includes only the `tf.lite.Interpreter` Python class. This small package is ideal when all you want to do is execute `.tflite` models and avoid wasting disk space with the large TensorFlow library.

To install, run `pip3 install` and pass it the appropriate Python wheel URL from the following table.

For example, if you have **Piccoclo / Penny Whistle** that's running **Python 3.7**, install the Python wheel as follows:

Platform	Python	URL
Linux (ARM 32)	3.5	https://dl.google.com/coral/python/tflite_runtime-2.1.0.post1-cp35-cp35m-linux_armv7l.whl
	3.6	https://dl.google.com/coral/python/tflite_runtime-2.1.0.post1-cp36-cp36m-linux_armv7l.whl
	3.7	https://dl.google.com/coral/python/tflite_runtime-2.1.0.post1-cp37-cp37m-linux_armv7l.whl

Download & Install TensorFlow Lite interpreter:

wget https://dl.google.com/coral/python/tflite_runtime-2.1.0.post1-cp37-cp37m-win_amd64.whl

tflite_runtime-2.1.0.post1-cp37-cp37m-linux_armv7l.whl

copy this file (**tflite_runtime-2.1.0.post1-cp37-cp37m-linux_armv7l.whl**) to piccoclo device / penny whistle device , install with the following command

```
pip3 install ./tflite_runtime-2.1.0.post1-cp37-cp37m-linux_armv7l.whl
```

Run an inference using tflite_runtime

To distinguish this interpreter-only package from the full TensorFlow package (allowing both to be installed, if you choose), the Python module provided in the above wheel is named `tflite_runtime`.

So instead of importing `Interpreter` from the `tensorflow` module, you need to import it from `tflite_runtime`.

For example, after you install the package above, copy and run the `label_image.py` file. It will (probably) fail because you don't have the `tensorflow` library installed. To fix it, edit this line of the file:

Step 1 / Change 1:

```
import tensorflow as tf
```

So it instead reads:

```
import tflite_runtime.interpreter as tflite
```

Step 2 / Change 2:

```
interpreter = tf.lite.Interpreter(model_path=args.model_file)
```

So it instead reads:

```
interpreter = tflite.Interpreter(model_path=args.model_file)
```

Now run **`label_image.py`** again. That's it! You're now executing TensorFlow Lite models.

label_image.py:

https://github.com/tensorflow/tensorflow/tree/master/tensorflow/lite/examples/python/

Step 1: label_image.py

Step 2: Download sample model and image

You can use any compatible model, but the following MobileNet v1 model offers a good demonstration of a model trained to recognize 1,000 different objects.

# Get photo

```
wget https://raw.githubusercontent.com/tensorflow/tensorflow/master/tensorflow/lite/examples/
label_image/testdata/grace_hopper.bmp
```

# Get model

```
wget https://storage.googleapis.com/download.tensorflow.org/models/mobilenet_v1_2018_02_22/
mobilenet_v1_1.0_224.tgz
```

# Get labels

```
wget
https://storage.googleapis.com/download.tensorflow.org/models/mobilenet_v1_1.0_224_frozen.tgz
```

`/media/samsungQVO2TB/alexvatti/jadak/April_23/tensorflow-lite`

```
alexvatti@SYR-DEEPLEARN2:/media/samsungQVO2TB/alexvatti/jadak/April_23/tensorflow-lite$ tree -L  3
.
├── grace_hopper.bmp
├── label_image.py
├── mobilenet_v1_1.0_224
│   ├── mobilenet_v1_1.0_224.ckpt.data-00000-of-00001
│   ├── mobilenet_v1_1.0_224.ckpt.index
│   ├── mobilenet_v1_1.0_224.ckpt.meta
│   ├── mobilenet_v1_1.0_224_info.txt
│   └── mobilenet_v1_1.0_224.tflite
├── mobilenet_v1_1.0_224_frozen
│   └── mobilenet_v1_1.0_224
│       ├── frozen_graph.pb
│       ├── labels.txt
│       └── quantized_graph.pb
├── mobilenet_v1_1.0_224_frozen.tgz
├── mobilenet_v1_1.0_224.tgz
└── tflite_runtime-2.1.0.post1-cp37-cp37m-win_amd64.whl

3 directories, 13 files
```

Run the sample

```
python3 label_image.py \
  --model_file ./mobilenet_v1_1.0_224.tflite \
  --label_file ./labels.txt \
  --image ./grace_hopper.bmp
```

You should see results like this:

```
0.728693: military uniform
0.116163: Windsor tie
0.035517: bow tie
0.014874: mortarboard
0.011758: bolo tie
```

Our Results:

```
0.792127: 653:military uniform

0.084584: 907:Windsor tie

0.021034: 458:bow tie, bow-tie, bowtie

0.009951: 668:mortarboard

0.007782: 514:cornet, horn, trumpet, trump
```

Yocto /Poky Image:

Piccolo device:

/media/samsungQVO2TB/alexvatti/jadak/April_23/test_build_piccolo/JADAK-Piccolo.YoctoPiccolo/build/tmp/deploy/images/jadak-piccolo/core-image-minimal-jadak-piccolo—20200426062341.mender on **Build Machine [172.28.149.86]**

Penny Whistel device:

/media/samsungQVO2TB/alexvatti/jadak/April_23/test_build_pennywhistle/JADAK-Piccolo.YoctoPiccolo/build/tmp/deploy/images/jadak-pennywhistle/core-image-minimal-jadak-pennywhistle--20200426101503.mender

on **Build Machine [172.28.149.86]**