X-LINUX-AI - image classification using Coral Edge TPU TensorFlow Lite Python runtime

Contents

Stable: 01.01.2021 - 12:47 / Revision: 01.01.2021 - 12:48

A quality version of this page, approved on *1 July 2021*, was based off this revision.

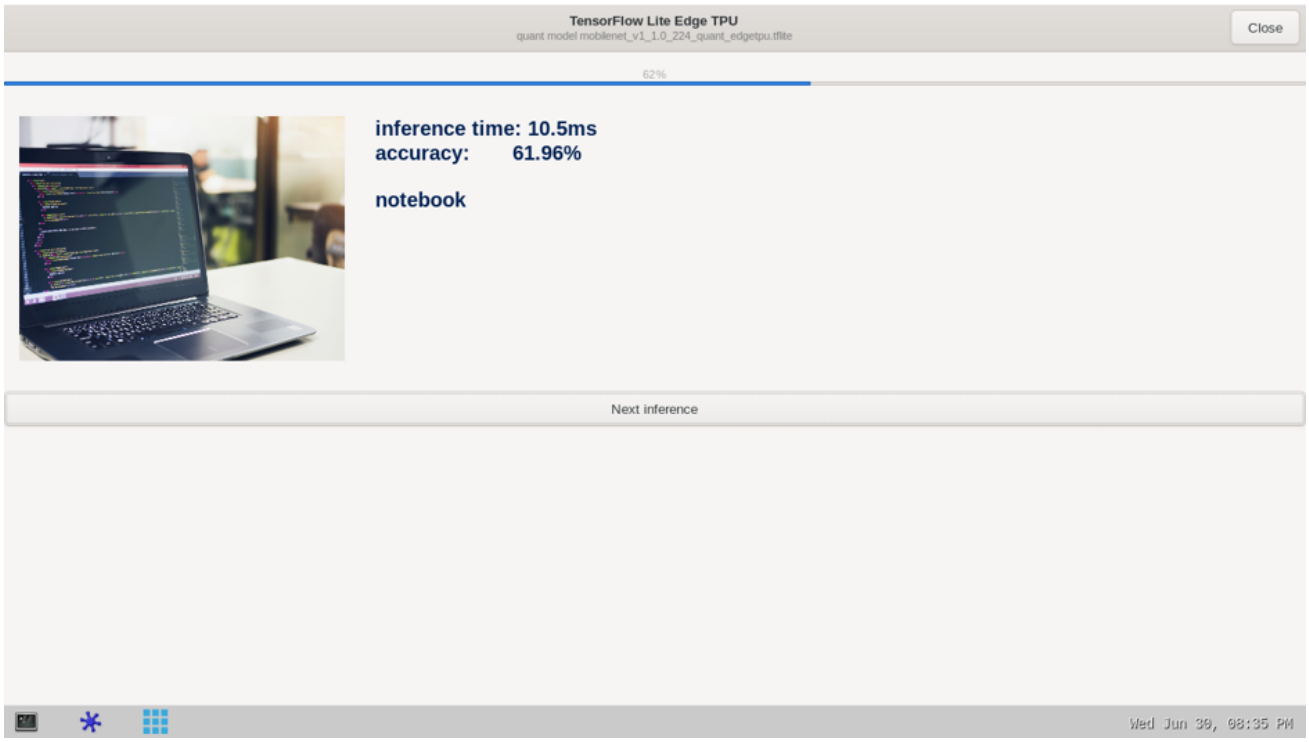
This article explains how to experiment with **Coral Edge TPU**[1] applications for image classification based on the MobileNet v1 model using TensorFlow Lite Python runtime.

> Python applications are good for prototyping but are less efficient than C/C++ applications

Contents

1 Description

The **image classification**[2] neural network model allows identification of the subject represented by an image. It classifies an image into various classes.



The application enables OpenCV camera streaming (or test data picture) and the Coral Edge TPU[1] TensorFlow Lite[3] interpreter runing the NN inference based on the camera (or test data pictures) inputs.
The user interface is implemented using Python GTK.

The model used with this application is the MobileNet v1 downloaded from the **Coral GitHub testing models**[4].

2 Installation

2.1 Install from the OpenSTLinux AI package repository

> ***The software package is provided AS IS, and by downloading it, you agree to be bound to the terms of the software license agreement (SLA). The detailed content licenses can be found here.***

After having configured the AI OpenSTLinux package you can install the X-LINUX-AI components for this application:

```
Board $> apt-get install tflite-cv-apps-edgetpu-image-classification-python
```

Then restart the demo launcher:

```
Board $> systemctl restart weston@root
```

2.2 Source code location

The **label_tfl_edgetpu.py** Python script is available:

- in the Openembedded OpenSTLinux Distribution with X-LINUX-AI Expansion Package:

  <Distribution Package installation directory>/layers/meta-st/meta-st-stm32mpu-ai/recipes-samples/tflite-cv-apps-edgetpu/files/image-classification/python/label_tfl_edgetpu.py

- on the target:

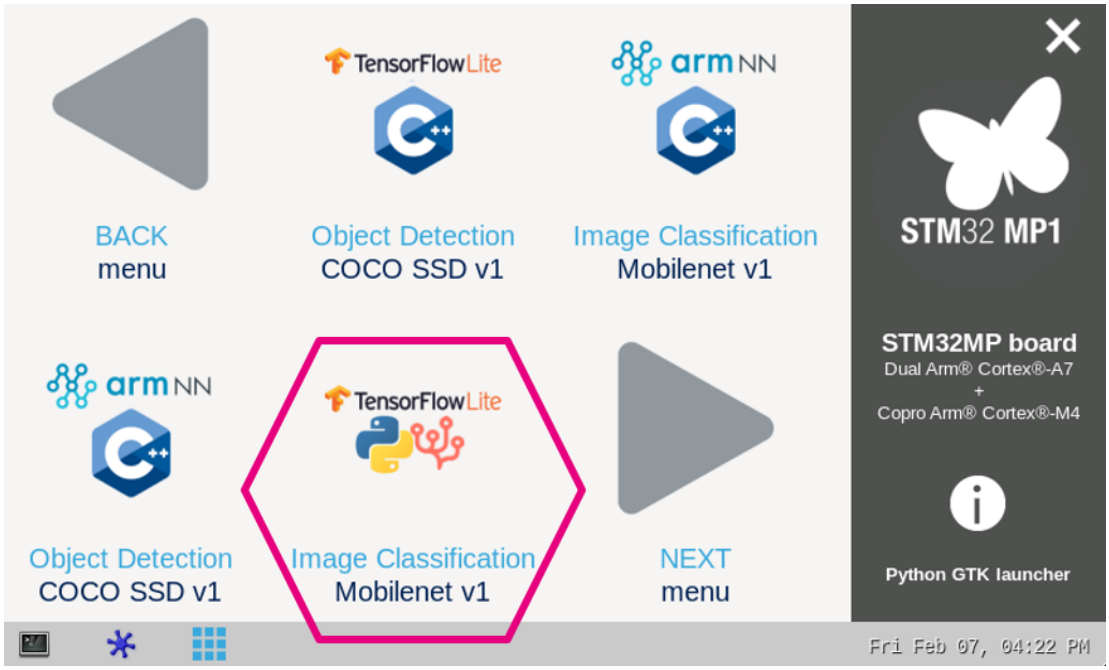
  /usr/local/demo-ai/computer-vision/tflite-image-classification-edgetpu/python/label_tfl_edgetpu.py

- on GitHub:

  https://github.com/STMicroelectronics/meta-st-stm32mpu-ai/tree/dunfell/recipes-samples/tflite-cv-apps-edgetpu/files/image-classification/python/label_tfl_edgetpu.py

3 How to use the application

3.1 Launching via the demo launcher



3.2 Executing with the command line

The Python script **label_tfl_edgetpu.py** application is located in the userfs partition:

```
/usr/local/demo-ai/computer-vision/tflite-image-classification-edgetpu/python
/label_tfl_edgetpu.py
```

It accepts the following input parameters:

```
usage: label_tfl_edgetpu.py [-h] [-i IMAGE] [-v VIDEO_DEVICE] [--frame_width FRAME_WIDTH]
[--frame_height FRAME_HEIGHT]
                            [--framerate FRAMERATE] [-m MODEL_FILE] [-l LABEL_FILE] [--
lib_edgetpu {max,throttled}]
                            [--input_mean INPUT_MEAN] [--input_std INPUT_STD] [--top_k
TOP_K]


optional
arguments:

  -h, --help            show this help message and
exit
  -i IMAGE, --image
IMAGE

                        image directory with image to be
classified
```

```
  -v VIDEO_DEVICE, --video_device
VIDEO_DEVICE

                        video device (default /dev
/video0)
  --frame_width
FRAME_WIDTH

                        width of the camera frame (default is
320)
  --frame_height
FRAME_HEIGHT

                        height of the camera frame (default is
240)
  --framerate
FRAMERATE

                        framerate of the camera (default is
15fps)
  -m MODEL_FILE, --model_file
MODEL_FILE

                        .tflite model to be
executed
  -l LABEL_FILE, --label_file
LABEL_FILE

                        name of file containing
labels
  --lib_edgetpu {max,
throttled}

                        Choose the version of your EdgeTPU
runtime
  --input_mean
INPUT_MEAN

                        input
mean

  --input_std
INPUT_STD

                        input standard
deviation

  --top_k TOP_K         The top_k classes to show
```

3.3 Testing with MobileNet V1

The model used for test is the **mobilenet_v1_1.0_224_quant_edgetpu.tflite** downloaded from **Coral GitHub testing models**[4].

> The different objects the neural network is able to classify are listed in the labels.txt file located in the target:
>
> /usr/local/demo-ai/computer-vision/models/mobilenet/labels.txt

To ease launching of the Python script, two shell scripts are available:

- launch image classification based on camera frame inputs

```
Board $> /usr/local/demo-ai/computer-vision/tflite-image-classification-edgetpu/python
/launch_python_label_tfl_edgetpu_mobilenet.sh
```

- launch image classification based on the pictures located in **/usr/local/demo-ai/computer-vision/models/mobilenet/testdata** directory:

```
Board $> /usr/local/demo-ai/computer-vision/tflite-image-classification-edgetpu/python
/launch_python_label_tfl_edgetpu_mobilenet_testdata.sh
```

Note that you need to populate the testdata directory with your own data sets.

The pictures are then randomly read from the testdata directory

4 References

- 1.01.1 Coral Edge TPU
- TFLite image classification overview
- TensorFlow Lite
- 4.04.1 Coral GitHub testing models

Neural Network

Artificial Intelligence