Get started with Google Coral M.2 and USB Accelerators

Embedded Artists

Embedded Artists AB
Jörgen Ankersgatan 12
SE-211 45 Malmö
Sweden

https://www.EmbeddedArtists.com

Disclaimer

Embedded Artists AB makes no representation or warranties with respect to the contents hereof and specifically disclaim any implied warranties or merchantability or fitness for any particular purpose. Information in this publication is subject to change without notice and does not represent a commitment on the part of Embedded Artists AB.

Feedback

We appreciate any feedback you may have for improvements on this document. Send your comments by using the contact form: www.embeddedartists.com/contact.

Trademarks

All brand and product names mentioned herein are trademarks, services marks, registered trademarks, or registered service marks of their respective owners and should be treated as such.

Table of Contents

1 Document Revision History

Revision	*Date*	*Description*
A	2020-09-29	First release

2 Introduction

This document provides you with step-by-step instructions for how to get stared with the Google Coral M.2 and/or USB accelerators on Embedded Artists iMX8 based COM boards.

Coral is a technology from Google that accelerates the processing of TensorFlow models used with **machine learning**. At its core there is the Edge TPU coprocessor that executes neural networks at high speed. This coprocessor is available in multiple form factors such as the M.2 Accelerator with **PCIe** interface or USB accelerator.

More information about Coral is available on the link below.

https://coral.ai/technology/

> **Note**: This document doesn't explain what machine learning is. You need to have a basic understanding of this.

The models used by the examples in this document all come from https://coral.ai/models/ and can be divided into two categories:

- Classification, where the model can recognize a set of objects and the demo scripts will print which (if any) objects are found and with what probability. Examples in 3.6 and 3.8.1
- Detection, where the model can recognize a set of objects and the demo scripts will draw a rectangle in the image/video around the detected object. Examples in 3.7 and 3.8.2

It should be possible to run classification/detection using any of the models from https://coral.ai/models/ but that is left as an exercise for the reader.

2.1 Supported hardware

The instructions in this document work with the following Developer's Kits from Embedded Artists as the board needs to have a 64-bit core.

- iMX8M Developer's Kit V2 (EAK00330)
- iMX8M Mini uCOM Developer's Kit V2 (EAK00347)
- iMX8M Nano uCOM Developer's Kit V2 (EAK00360). **Note**: This board doesn't support PCIe.

You also need a **Coral M.2 Accelerator with A+E key** or a **Coral USB Accelerator**.

https://coral.ai/products/m2-accelerator-ae/

https://eu.mouser.com/new/google-coral/coral-m2-accelerator-ae/

https://coral.ai/products/accelerator

https://eu.mouser.com/ProductDetail/Coral/G950-01456-01?qs=u16ybLDytRbcxxqFKdbhgQ%3D%3D

2.2 Additional documentation

Additional documentation you might need is.

- *Working with Yocto to build Linux* – found on Embedded Artists website.
- *How to use a Camera with an iMX Developer's Kit* – found on Embedded Artists website.

- Getting started guide - https://www.embeddedartists.com/getting-started/
- Get started documentation from Coral: https://coral.ai/docs/m2/get-started/

2.3 Conventions

A number of conventions have been used throughout to help the reader better understand the content of the document.

`Constant width text` – is used for file system paths and command, utility and tool names.

```
$ This field illustrates user input in a terminal running on the
development workstation, i.e., on the workstation where you edit,
configure and build Linux
```

```
# This field illustrates user input on the target hardware, i.e.,
input given to the terminal attached to the COM Board
```

```
This field is used to illustrate example code or excerpt from a
document.
```

3 Get started

3.1 Setup hardware

3.1.1 M.2 Accelerator

> **Note 1**: You need a COM Carrier board V2 for these instructions.
> **Note 2:** The iMX8M Nano doesn't have a PCIe interface. Use the USB accelerator instead.

All you need to do is insert the Coral M.2 Accelerator to the M.2 E-key connector on the COM Carrier board V2 as shown in Figure 1.

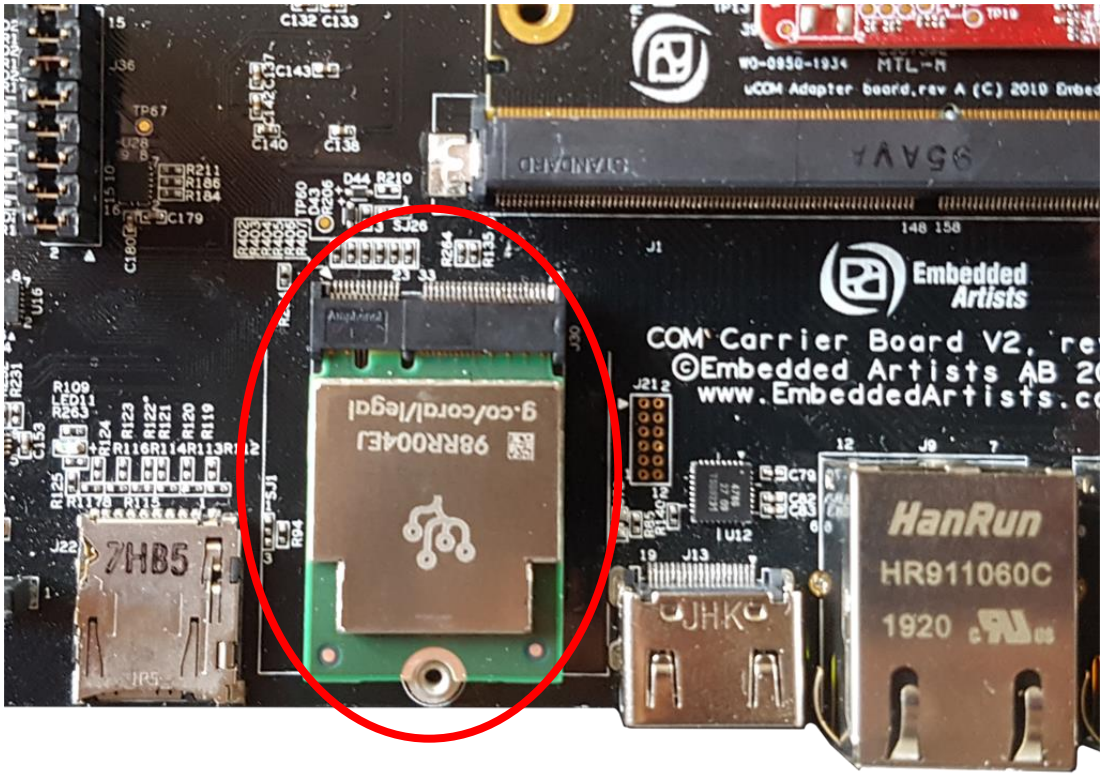

Figure 1 - M.2 Accelerator attached to COM Carrier board V2

3.1.2 USB Accelerator

For iMX8M Mini uCOM and iMX8M Developer's Kits you can connect the USB accelerator directly to USB Host Type A connector (J12) on the carrier board, see Figure 2.

The iMX8M Nano uCOM only has one USB port which by default is connected to USB OTG connector (J11). You need a micro-B (male) to USB A (female) adapter similar to the one shown in Figure 3.

It is possible to change a **slider switch**, S3:8, on the back of the uCOM adapter board and thereby connecting the USB port on the iMX8M Nano to the USB Host Type A connector (J12) on the carrier board. See section 7.2 in the iMX8M Nano uCOM datasheet for details.

> **Note**: If you want to try the **camera** related examples, described in section 3.8 together with the iMX8M Nano you need to change the slider switch. The reason is both the accelerator and the camera need a USB interface.

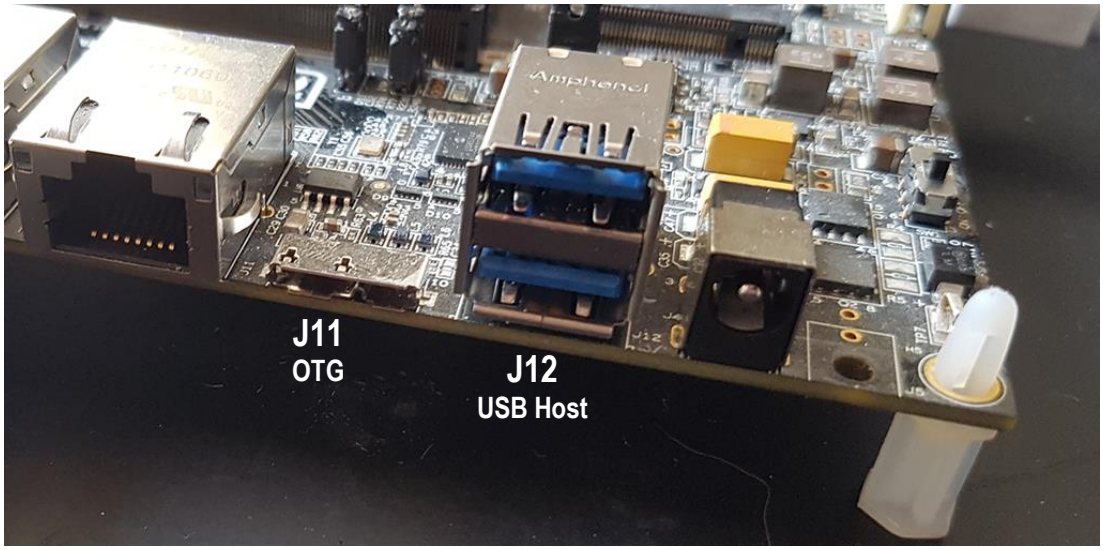

Figure 2 - USB connectors on carrier board

Figure 3 - USB micro-B (male) to USB A (female) adapter

3.2 Download and flash pre-built Yocto image

To be able to setup and use Coral you need software (mostly Python packages) that is not normally installed on a standard `ea-image-base` distribution. Images with as many as possible of the needed packages have been prepared and are available on http://imx.embeddedartists.com. Go to this website and download the zip file that ends with `_coral.zip` for the board you are using.

Unpack this zip file on your computer and setup your iMX Developer's Kit for OTG boot mode (see the document *Working with Yocto* for details). Flash the full image by using UUU and the `full_tar.uuu` script file.

```
> uuu full_tar.uuu
```

3.3 Enable PCIe support

Note: You only need to use the pcie-related dtb file if you use the M.2 Accelerator. If you use the USB accelerator you don't need to change the `fdt_file` variable.

Reset the board and boot into the U-boot console. You need to **hit any key** on your keyboard to stop autoboot. Go to the section below that corresponds to the board you are using.

3.3.1 iMX8M COM board

Change to the device tree file that enables PCIe support.

```
=> setenv fdt_file imx8mq-ea-com-kit_v2-pcie.dtb
=> saveenv
=> boot
```

3.3.2 iMX8M Mini uCOM board

Change to the device tree file that enables PCIe support.

```
=> setenv fdt_file imx8mm-ea-ucom-kit_v2-pcie.dtb
=> saveenv
=> boot
```

3.4 Install Edge TPU runtime

The Edge TPU runtime is available as a Debian package. Support for Debian package manager has been added to the Yocto image, but you also have to add a Google repository so the package can be found.

Note: You need Internet access for this to work.

```
# echo "deb https://packages.cloud.google.com/apt coral-edgetpu-
stable main" | tee /etc/apt/sources.list.d/coral-edgetpu.list
```

Update the package index.

```
# apt-get update
```

You will most likely get several errors while running update since the main package feed probably don't exist. The prebuilt image from section 3.2 was created on a computer with IP address 192.168.1.11 and that is where apt-get will look for the package feed.

You will also get warnings regarding the Google repository about it having an invalid signature. You can ignore this for now.

```
Err:1 http://192.168.1.11:5678/all ./ InRelease
  Could not connect to 192.168.1.11:5678 (192.168.1.11). - connect (111:
Connection refused)
```

```
...
Get:5 https://packages.cloud.google.com/apt coral-edgetpu-stable InRelease [6332
B]
Ign:5 https://packages.cloud.google.com/apt coral-edgetpu-stable InRelease
Get:6 https://packages.cloud.google.com/apt coral-edgetpu-stable/main arm64
Packages [1174 B]
Fetched 7506 B in 0s (8904 B/s)
Reading package lists... Done
W: Failed to fetch http://192.168.1.11:5678/all/./InRelease  Could not connect to
192.168.1.11:5678 (192.168.1.11). - connect (111: Connection refused)
W: Failed to fetch http://192.168.1.11:5678/aarch64/./InRelease  Unable to connect
to 192.168.1.11:5678:
W: Failed to fetch http://192.168.1.11:5678/aarch64-mx8mm/./InRelease  Unable to
connect to 192.168.1.11:5678:
W: Failed to fetch http://192.168.1.11:5678/imx8mmea_ucom/./InRelease  Unable to
connect to 192.168.1.11:5678:
W: Some index files failed to download. They have been ignored, or old ones used
instead.
```

Download and install the Edge TPU runtime environment. **Enter 'y'** to acknowledge that you want to install the package without verification. The warning comes as the signature of the Google repository is not known (could have been fixed by adding / using apt-key).

```
# apt-get install libedgetpu1-std

Reading package lists... Done
Building dependency tree... Done
The following NEW packages will be installed:
  libedgetpu1-std
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 248 kB of archives.
After this operation, 814 kB of additional disk space will be
used.
WARNING: The following packages cannot be authenticated!
  libedgetpu1-std
Install these packages without verification? [y/N]
```

3.5 Install TensorFlow Lite

TensorFlow Lite is used to run machine learning models. Here we will install TensorFlow Lite for Python. There are different packages for different versions of Python and also for different platforms. The original instructions are available at the link below.

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

The Python version installed in the pre-built image for **Linux 5.4.24** is 3.7 and for iMX8 based boards the platform is Linux (ARM 64):

```
# pip3 install https://dl.google.com/coral/python/tflite_runtime-
2.1.0.post1-cp37-cp37m-linux_aarch64.whl
```

The Python version installed in the pre-built image for **Linux 4.14.98** is 3.5 and for iMX8 based boards the platform is Linux (ARM 64):

```
# pip3 install https://dl.google.com/coral/python/tflite_runtime-
2.1.0.post1-cp35-cp35m-linux_aarch64.whl
```

3.6 Example: Image classification

Follow the steps below to install example model that can be used to classify images of certain birds.

```
# cd && mkdir -p coral && cd coral
# git clone https://github.com/google-coral/tflite.git
# cd tflite/python/examples/classification
# bash install_requirements.sh
```

The last instruction will install the example's dependencies. It will also download a photo of a Parrot that will be classified.

The downloaded model files include a text file containing a list of all the birds that the model can classify.

```
# less models/inat_bird_labels.txt
0 Haemorhous cassinii (Cassin's Finch)
1 Aramus guarauna (Limpkin)
...
```

3.6.1 Run example

To classify the parrot in Figure 4 run the instructions below.

Figure 4 - Parrot that will be classified by the example model

First, we must make sure Python can find the Edge TPU library.

```
# export LD_LIBRARY_PATH=/usr/lib/aarch64-linux-gnu/
```

Now run the classify model on the Edge TPU. The script will run the model 5 times and the first time it will be a bit slower since it has to load the model into the Edge TPU memory.

```
# python3 classify_image.py \
  --model models/mobilenet_v2_1.0_224_inat_bird_quant_edgetpu.tflite \
  --labels models/inat_bird_labels.txt \
  --input images/parrot.jpg
```

```
----INFERENCE TIME----
Note: The first inference on Edge TPU is slow because it includes
loading the model into Edge TPU memory.
13.7ms
2.9ms
2.6ms
2.6ms
2.9ms
-------RESULTS--------
Ara macao (Scarlet Macaw): 0.77734
```

To get a sense of how much the Edge TPU accelerates the classification, you can also try to run this model directly on the processor to see the how much slower it is.

```
# python3 classify_image.py \
  --model models/mobilenet_v2_1.0_224_inat_bird_quant.tflite \
  --labels models/inat_bird_labels.txt \
  --input images/parrot.jpg

----INFERENCE TIME----
Note: The first inference on Edge TPU is slow because it includes
loading the model into Edge TPU memory.
148.5ms
147.2ms
147.1ms
147.1ms
147.2ms
-------RESULTS--------
Ara macao (Scarlet Macaw): 0.7773
```

Both runs above correctly detect the bird as an Ara macao with the same probability (77%) which is the expected result given that the input image and the model is the same. The accelerator does it about 50 times faster (2.9ms vs 147.2ms).

We can also try to classify other images of birds. Try for example to download this image of a Rook.

```
# curl
https://upload.wikimedia.org/wikipedia/commons/6/63/Rook_at_Slimbr
idge_Wetland_Centre%2C_Gloucestershire%2C_England_22May2019_arp.jp
g --output images/rook.jpg

# python3 classify_image.py \
  --model models/mobilenet_v2_1.0_224_inat_bird_quant_edgetpu.tflite \
  --labels models/inat_bird_labels.txt \
  --input images/rook.jpg

----INFERENCE TIME----
Note: The first inference on Edge TPU is slow because it includes
loading the model into Edge TPU memory.
13.7ms
3.2ms
2.9ms
3.0ms
3.6ms
-------RESULTS--------
Corvus frugilegus (Rook): 0.78516
```

3.7 Example: Image detection

Follow the steps below to install an example model that can be used to detect objects in an image. The git repository is the same as in the previous example so if you have followed the instructions in 3.6 then skip the git clone command below.

```
# cd && mkdir -p coral && cd coral
# git clone https://github.com/google-coral/tflite.git
# cd tflite/python/examples/detection
# sed -i 's/mobilenet_ssd/ssd_mobilenet/g' install_requirements.sh
# bash install_requirements.sh
```

The `sed` command above will correct the name of the model. That error has been reported and fixed but is not yet released. The last instruction will install the example's dependencies. It will also download a photo of Grace Hopper that will be used for image detection.

The downloaded model files include a text file containing a list of all the objects that the model can detect.

```
# less models/coco_labels.txt
0  person
1  bicycle
2  car
...
```

3.7.1 Run example

To detect objects in Figure 5 run the instructions below.

Figure 5 – Photo of Grace Hopper used by the example model

First, we must make sure Python can find the Edge TPU library.

```
# export LD_LIBRARY_PATH=/usr/lib/aarch64-linux-gnu/
```

Now run the classify model on the Edge TPU. The script will run the model 5 times and the first time it will be a bit slower since it has to load the model into the Edge TPU memory.

```
# python3 detect_image.py \
  --model models/ssd_mobilenet_v2_coco_quant_postprocess_edgetpu.tflite \
  --labels models/coco_labels.txt \
  --input images/grace_hopper.bmp \
  --output images/grace_hopper_processed.bmp

----INFERENCE TIME----
Note: The first inference is slow because it includes loading the
model into Edge TPU memory.
53.21 ms
15.53 ms
17.62 ms
15.66 ms
15.43 ms
-------RESULTS--------
tie
  id:     31
  score:  0.83984375
  bbox:   BBox(xmin=226, ymin=417, xmax=290, ymax=539)
person
  id:     0
  score:  0.83984375
  bbox:   BBox(xmin=2, ymin=5, xmax=507, ymax=590)
```

To get a sense of how much the Edge TPU accelerates the detection, you can also try to run this model directly on the processor to see the how much slower it is.

```
# python3 detect_image.py \
  --model models/ssd_mobilenet_v2_coco_quant_postprocess.tflite \
  --labels models/coco_labels.txt \
  --input images/grace_hopper.bmp \
  --output images/grace_hopper_processed.bmp

----INFERENCE TIME----
Note: The first inference is slow because it includes loading the
model into Edge TPU memory.
318.96 ms
312.84 ms
312.23 ms
312.17 ms
312.41 ms
-------RESULTS--------
person
  id:     0
  score:  0.83984375
  bbox:   BBox(xmin=2, ymin=5, xmax=507, ymax=590)
tie
  id:     31
  score:  0.83984375
  bbox:   BBox(xmin=225, ymin=417, xmax=291, ymax=539)
```

Both runs above correctly detect both the person and the tie in the image with the same probabilities (84%) but the accelerator does it about 20 times faster (15.43ms vs 312.41ms). Download the output image (grace_hopper_processed.bmp) to a PC to view it or, if you have a display connected use the framebuffer image viewer:

```
# fbi -a -T 1 images/grace_hopper_processed.bmp
```

Note 1: On Linux 4.14.98 using the image viewer breaks the desktop. Run the following command after viewing the image to restart the desktop:

```
# systemctl restart Weston
```

Note 2: If nothing is shown on the display when running `fbi` this can be because of an unsupported resolution for the framebuffer. From within u-boot you can set the kernel parameter video like this

```
# setenv extra_bootargs video=1280x720
```

The image will look like this:

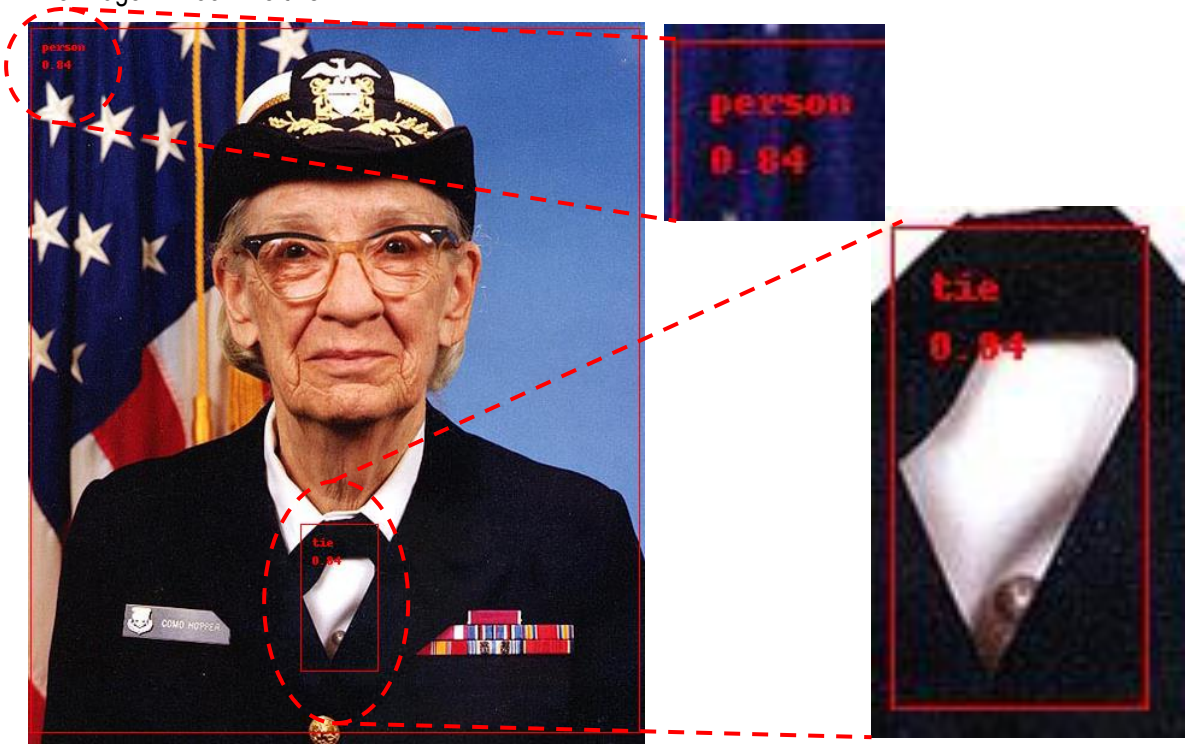

Figure 6 – Processed photo of Grace Hopper showing the found "person" and "tie" objects

We can also try to detect objects from the list in other images. Try for example to download this image of an apple and an orange.

```
# curl
https://upload.wikimedia.org/wikipedia/commons/thumb/a/a8/Apple_and_Orange_-_they_do_not_compare.jpg/220px-Apple_and_Orange_-_they_do_not_compare.jpg --output images/ao.jpg

# python3 detect_image.py \
  --model models/ssd_mobilenet_v2_coco_quant_postprocess_edgetpu.tflite \
  --labels models/coco_labels.txt \
  --input images/ao.jpg \
  --output images/ao_processed.jpg

----INFERENCE TIME----
Note: The first inference is slow because it includes loading the
```

```
model into Edge TPU memory.
53.10 ms
16.93 ms
15.12 ms
16.69 ms
15.81 ms
-------RESULTS--------
orange
  id:     54
  score:  0.96484375
  bbox:   BBox(xmin=109, ymin=29, xmax=213, ymax=134)
apple
  id:     52
  score:  0.93359375
  bbox:   BBox(xmin=41, ymin=18, xmax=121, ymax=105)
```

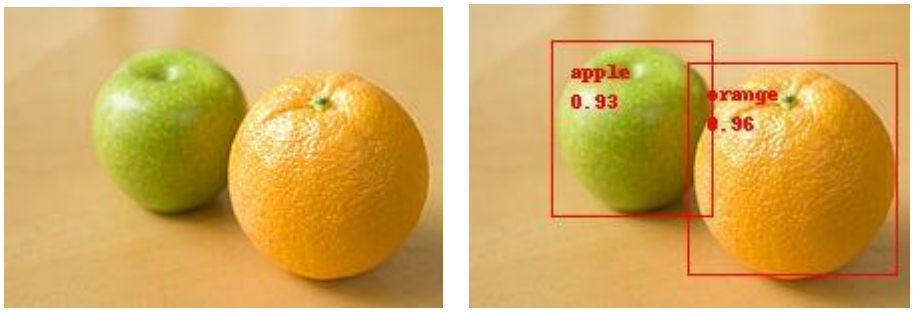

Figure 7 – Image of apple and orange

3.8 Example: Camera

Note: For the iMX8M Nano you need to change a slider switch to be able to connect both the camera and the USB accelerator, see section 3.1.2 above.

This example requires a camera (described in the document *How to use a Camera with an iMX Developer's Kit*) and a display. The instructions in this document were tested on a Logitech QuickCam Pro USB camera and the Embedded Artists 7-inch HDMI Display Kit (https://www.embeddedartists.com/products/7-inch-hdmi-display-kit/) but any HDMI display should work.

Follow the steps below to prepare the example that can be used to classify or detect objects in a live camera feed.

```
# cd && mkdir -p coral && cd coral
# git clone https://github.com/google-coral/examples-camera.git
# cd examples-camera
# sh download_models.sh
# cd gstreamer
# sed -i 's/ximagesink/waylandsink/g' gstreamer.py
# sed -i 's/if detectCoralDev.*/if False:/' gstreamer.py
# pip3 install svgwrite
```

The first `sed` command above is needed as the file system is using Wayland and not the X Window System. The second sed command prevents the `gstreamer.py` script from thinking that the iMX8M COM board is the Google Coral Dev Board.

3.8.1 Run classification example

First, we must make sure Python can find the Edge TPU library.

```
# export LD_LIBRARY_PATH=/usr/lib/aarch64-linux-gnu/
```

Now run the classify model on the Edge TPU. It will take a couple of seconds to start and then the video feed should appear on the display. The console will print out what is found

```
# python3 classify.py

Loading ../all_models/mobilenet_v2_1.0_224_quant_edgetpu.tflite
with ../all_models/imagenet_labels.txt labels
...
  Inference: 4.28 ms FPS: 15 fps score=0.18: mouse, computer mouse
score=0.11: lampshade, lamp shade score=0.10: monitor
  Inference: 4.16 ms FPS: 15 fps score=0.24: mouse, computer mouse
...
```

Stop the demo with Ctrl+C.

To test without the Edge TPU:

```
# python3 classify.py \
  --model ../all_models/mobilenet_v2_1.0_224_quant.tflite

Loading ../all_models/mobilenet_v2_1.0_224_quant.tflite with
../all_models/imagenet_labels.txt labels.
...
  Inference: 141.61 ms FPS: 7 fps score=0.11: mouse, computer
mouse
  Inference: 141.43 ms FPS: 7 fps score=0.13: mouse, computer
mouse
...
```

Note that the inference is 34 times slower without the acceleration (4.16 vs 141.43ms). With the accelerator the frame rate is limited by the camera's capture rate to 15fps but without acceleration the inference time limits the frame rate to a maximum of 1000/141.43 = 7fps.

3.8.2 Run detection example

First, we must make sure Python can find the Edge TPU library.

```
# export LD_LIBRARY_PATH=/usr/lib/aarch64-linux-gnu/
```

Now run the detect model on the Edge TPU. The script will take a couple of seconds to start and then the video feed should appear on the display. The console will only print out statistics

```
# python3 detect.py

Loading
../all_models/mobilenet_ssd_v2_coco_quant_postprocess_edgetpu.tflite with
../all_models/coco_labels.txt labels.
...
Inference: 15.76 ms FPS: 15 fps
Inference: 15.49 ms FPS: 15 fps
Inference: 15.62 ms FPS: 15 fps
...
```

Stop the demo with Ctrl+C.

To test without the Edge TPU:

```
# python3 detect.py --model \
  ../all_models/mobilenet_ssd_v2_coco_quant_postprocess.tflite

Loading ../all_models/mobilenet_ssd_v2_coco_quant_postprocess.tflite with
../all_models/coco_labels.txt labels.
...
Inference: 317.38 ms FPS: 3 fps
Inference: 316.86 ms FPS: 3 fps
Inference: 316.46 ms FPS: 3 fps
...
```

Note that the inference is 20 times slower without the acceleration (15.62 vs 316.46ms). With the accelerator the frame rate is limited by the camera's capture rate to 15fps but without acceleration the inference time limits the frame rate to a maximum of 1000/316.46 = 3fps.

4 Troubleshooting

4.1 Cannot find libedgetpu.so.1

If you get an error message similar to below the most likely problem is that you have forgotten to set `LD_LIBRARY_PATH` as explained in section 3.6.1 above. You could also have forgot or failed to install the TPU Edge runtime library as explained in section 3.4 above.

```
OSError: libedgetpu.so.1: cannot open shared object file: No such
file or directory
```

4.2 ValueError: Failed to load delegate from libedgetpu.so.1

Error message:

```
ValueError: Failed to load delegate from libedgetpu.so.1
```

Reason 1:

PCIe is not enabled. Please make sure you have followed the instructions in section 3.3 above.

Reason 2:

You could also get this error message if the M.2 Accelerator isn't inserted into the M.2 connector on the COM Carrier board V2.

You can check if the M.2 Accelerator is detected on the PCI bus by using `lspci`.

```
# lspci | grep 089a
```

You should get a result similar to below.

```
01:00.0 System peripheral: Device 1ac1:089a
```

You can also check if the PCI driver (Apex) is loaded and available.

```
# lsmod | grep apex
apex                   24576  0
gasket                 94208  1 apex
```

4.3 Cannot identify device '/dev/video0'

Error message:

```
Error: gst-resource-error-quark: Cannot identify device
'/dev/video0'. (3): ../../../git/sys/v4l2/v4l2_calls.c(656):
gst_v4l2_open (): /GstPipeline:pipeline0/GstV4l2Src:v4l2src0:

system error: No such file or directory
```

Reason 1:

The camera is not connected. Check that the camera is connected correctly, reboot and test again

Reason 2:

The camera is not mapped as `/dev/video0`. Check which video device is in use:

```
# ls /dev/video*
/dev/video1
```

Use the found device when running the camera examples by passing the `-videosrc` parameter like this:

```
# python3 detect.py --videosrc /dev/video1
```

5 Yocto image

This chapter shows how the pre-built Yocto image has been created in case you need to build it yourself.

5.1 Debian package manager

Support for the Debian package manager (apt) is needed in order to install the Edge TPU runtime.

Do the following modifications to your `conf/local.conf`.

Change package class from rpm to deb.

```
#PACKAGE_CLASSES ?= "package_rpm"
PACKAGE_CLASSES ?= "package_deb"
```

Enable package-management.

```
IMAGE_FEATURES_append = " package-management"
```

You can also set a default package feed. Replace the IP address below with your own.

```
PACKAGE_FEED_URIS = "http://192.168.1.11:5678"
```

Add `apt` to the image.

This program is added in `IMAGE_INSTALL_append`, see section 5.2 below.

5.2 Additional packages

Also make sure the following packages are added to your build.

```
IMAGE_INSTALL_append = "\
   apt \
   git \
   curl \
   python3 \
   python3-numpy \
   python3-pip \
   python3-dev \
   python3-pygobject \
   python3-pyparsing \
   libjpeg-turbo-dev \
   packagegroup-core-buildessential \
   gnupg \
   glib-2.0 \
   gtk+3 \
"
```