POLITECNICO
MILANO 1863

Hardware Architectures for Embedded and Edge AI

Prof Manuel Roveri – manuel.roveri@polimi.it
Massimo Pavan – massimo.pavan@polimi.it

Exercise session 8 – Taking the network on-device and testing

How do I port my network on device?

How did you train your model?

- With Edge impulse -> deploy with edge impulse (seen in the last lecture)
- With Tensorflow/colab -> We'll see 2 ways:
  - Standard TFlite4micro examples (updated!)
  - Edge impulse loading of an already trained .tflite network

Each of these two methods has its pros and cons that we'll see:

- In general, with TFlite4micro you have more control over what you are doing, with edge impulse you are more «guided».

POLITECNICO
MILANO 1863

Deployment with TFlite4Micro

Working examples and library!

https://github.com/tensorflow/tflite-micro-arduino-examples

Cloning the repo: - Git clone https://github.com/tensorflow/tflite-micro-arduino-examples.git

- A library very similar to the one that we have seen during the lectures

- Drivers for the camera are still not present in this library. You can use the EI method or try to implement the drivers from the other examples.
- The other on-board sensors should work.

TensorFlow Lite Micro Library for Arduino

This repository has the code (including examples) needed to use Tensorflow Lite Micro on an Arduino.

Table of contents

Build Status

Build Type	Status
Arduino CLI on Linux	CI passing
Sync from tflite-micro	(Arduino) Sync from tflite-micro passing

Always start from the hello world example

By starting with the hello world example you can test if the network is working by itself.

- Size TFlite arena
- Check if everything works with all_ops resolver
- Check needed operations with standard op_resolver
- Provide «mock» data
- Test inference
- Provide «real» test data

```
// This pulls in all the operation implementations we need.
// NOLINTNEXTLINE(runtime-global-variables)
static tflite::AllOpsResolver resolver;
```



```
static tflite::MicroMutableOpResolver<9> micro_op_resolver;
micro_op_resolver.AddAveragePool2D();
micro_op_resolver.AddLogistic();
micro_op_resolver.AddConv2D();
micro_op_resolver.AddDepthwiseConv2D();
micro_op_resolver.AddReshape();
micro_op_resolver.AddSoftmax();
micro_op_resolver.AddQuantize();
micro_op_resolver.AddMaxPool2D();
micro_op_resolver.AddMean();
micro_op_resolver.AddFullyConnected();
```

```
// Quantize the input from floating-point to integer
int8_t x_quantized = x / input->params.scale + input->params.zero_point;
// Place the quantized input in the model's input tensor
for (int x=0; x < 1960; x++){
  input->data.int8[x] = x_quantized;
}S
```

Layers_name != operations to be added to micro_op_resolver

```python
from tensorflow.keras import datasets, layers, models

base_model = models.Sequential()
base_model.add(layers.Input(shape=(1960)))
base_model.add(layers.Reshape([49,40,1]))
base_model.add(layers.Conv2D(4, (3, 3), activation='relu'))
base_model.add(layers.GlobalAveragePooling2D())
base_model.add(layers.Dense(4))
```



```
static tflite::MicroMutableOpResolver<10> micro_op_resolver;
micro_op_resolver.AddPack()
micro_op_resolver.AddMean()
micro_op_resolver.AddConv2D()
micro_op_resolver.AddFullyConnected()
micro_op_resolver.AddStridedSlice()
micro_op_resolver.AddSoftmax()
micro_op_resolver.AddReshape()
micro_op_resolver.AddShape()
micro_op_resolver.AddQuantize()
micro_op_resolver.AddAveragePool2D()
```

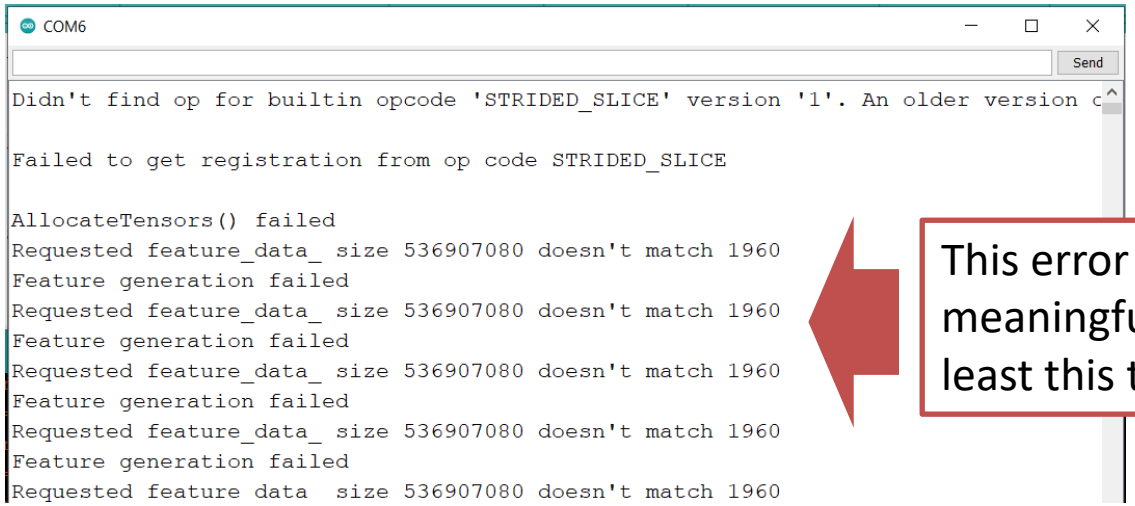
- The right layer name could not correspond 1 to 1 to the names that you need to import
- The Ops could be more than the layers
- Ops could vary from example to example

- … How to know which ops to add?

Adding Ops one by one

```
// The name of this function is important for Arduino compatibili
void setup() {
  delay(10000);
  tflite::InitializeTarget();

  // Map the model into a usable data structure. This doesn't inv
  // copying or parsing, it's a very lightweight operation.
  model = tflite::GetModel(g_model);
```



This error is not meaningful, at least this time

```
}
if (micro_op_resolver.AddStridedSlice() != kTfLiteOk) {
  return;
}
```

- Add a delay before anything else in setup(). This is done to have the time to open the serial Monitor before printing the error
- Until all Operations are added:
  1. Check the name of the required Op
  2. Add the required Op
  3. Load firmware

Expand to the example, or write an input pipeline for your data if you are using other sensors

micro_speech_7 | Arduino 1.8.16

File Edit Sketch Tools Help

micro_speech_7 | arduino_audio_provider.cpp | arduino_command_responder.cpp | arduino_main.cpp | audio_provider.h | command_res

```
int how_many_new_slices = 0;
TfLiteStatus feature_status = feature_provider->PopulateFeatureData(
    previous_time, current_time, &how_many_new_slices);
if (feature_status != kTfLiteOk) {
  MicroPrintf("Feature generation failed");
  return;
}
previous_time += how_many_new_slices * kFeatureSliceStrideMs;
// If no new audio samples have been received since last time, don't bother
// running the network model.
if (how_many_new_slices == 0) {
  return;
}

// Copy feature buffer to input tensor
for (int i = 0; i < kFeatureElementCount; i++) {
  model_input_buffer[i] = feature_buffer[i];
}

// Run the model on the spectrogram input and make sure it succeeds.
TfLiteStatus invoke_status = interpreter->Invoke();
```

Done Saving.

person_detection_6 | Arduino 1.8.16

File Edit Sketch Tools Help

person_detection_6 | arduino_detection_responder.cpp | arduino_image_provider.cpp | arduino_main.cpp | detection_resp

```
    return;
  }
}

// The name of this function is important for Arduino compatibility.
void loop() {
  // Get image from provider.
  if (kTfLiteOk != GetImage(input)) {
    MicroPrintf("Image capture failed.");
  }

  // Run the model on this input and make sure it succeeds.
  if (kTfLiteOk != interpreter->Invoke()) {
    MicroPrintf("Invoke failed.");
  }

  TfLiteTensor* output = interpreter->output(0);
```

Inference get executed, but the predictions are random

- Check that the sensors and input pipeline are working properly.
- Check that the two preprocessing pipelines are equal
- Check that the training data and the data collected by the sensor are at least similar
- Check that the quantization of inputs and outputs are the same for both the training and inference pipelines

Examples: porting the Keyword Spotting

- KWS - Sheila:

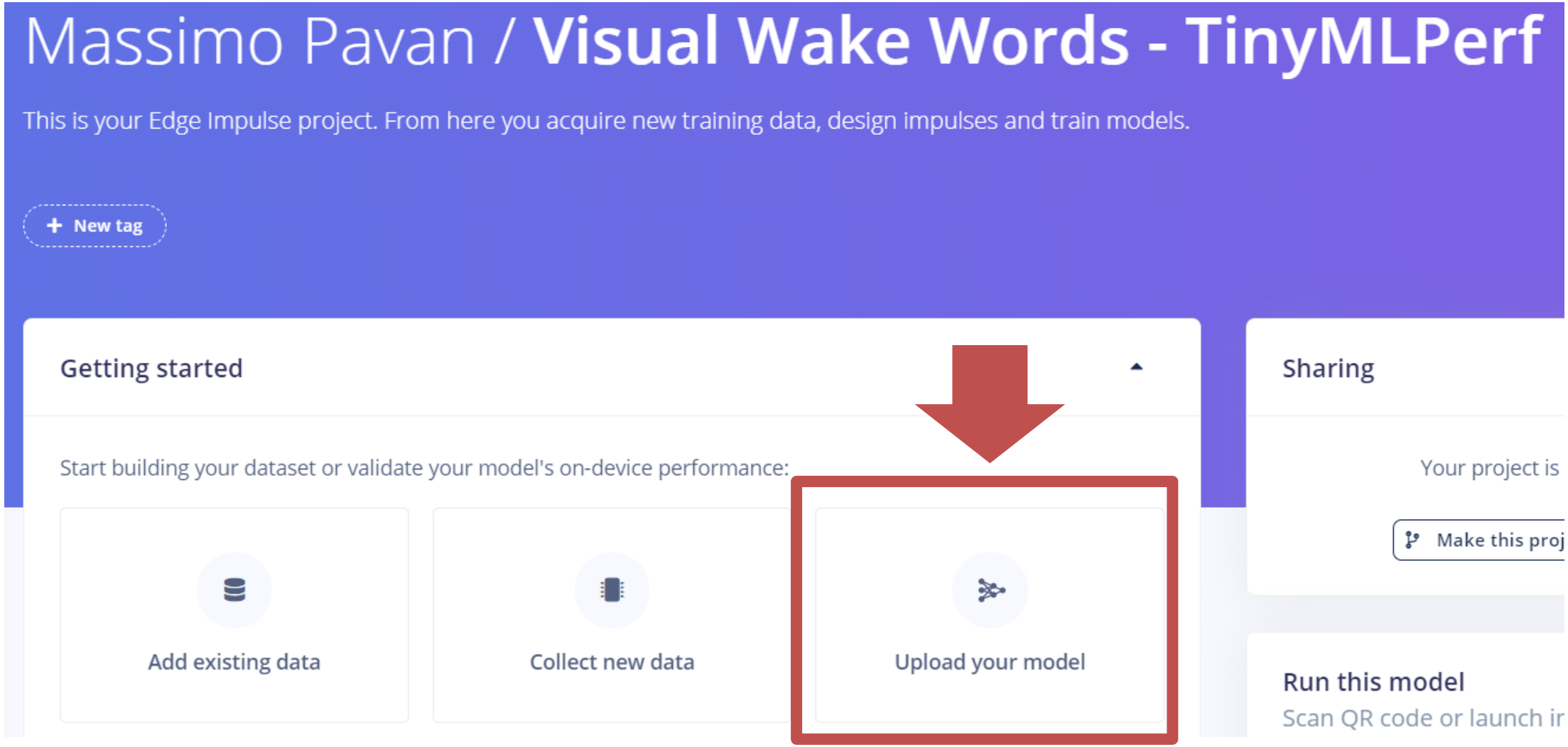
https://colab.research.google.com/drive/1j3mGVMuoQRT-TWRgmyqxb-AeVVlMcwbL?usp=sharing

- KWS - Sheila – trained with the old code:

https://colab.research.google.com/drive/1ncPXAAvn7Bo3b4mn6y_KrT_I_kMXZyMg?usp=sharing

POLITECNICO
MILANO 1863

Deployment with Edge impulse


Massimo Pavan / Visual Wake Words - TinyMLPerf
This is your Edge Impulse project. From here you acquire new training data, design impulses and train models.
+ New tag
Getting started
Start building your dataset or validate your model's on-device performance:
Add existing data
Collect new data
Upload your model
Sharing
Your project is
Make this proj
Run this model
Scan QR code or launch ir

Upload pretrained model - Step 1: Upload a model

1. Upload your trained model

Upload a TensorFlow SavedModel (`saved_model.zip`), ONNX model (`.onnx`) or TensorFlow Lite model (`.tflite`) to get started.

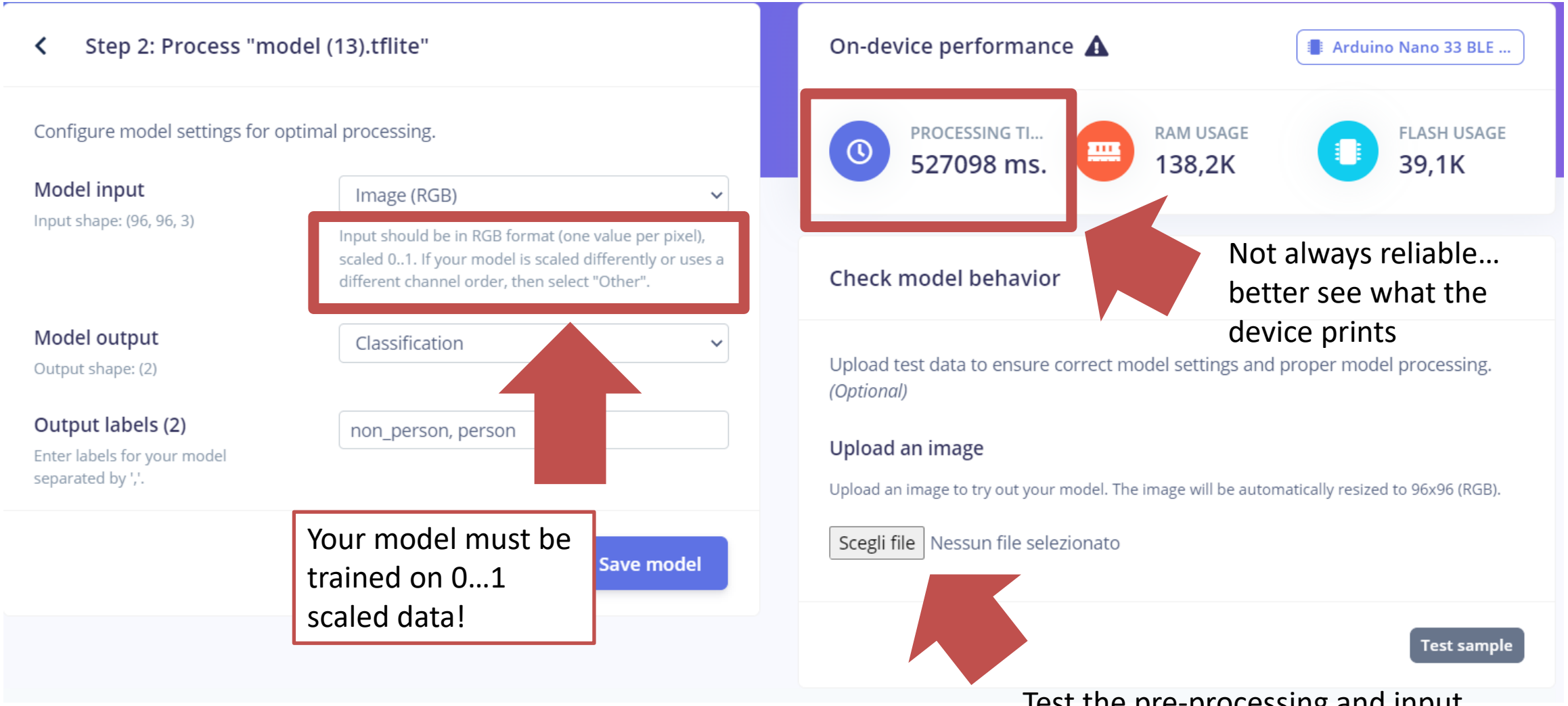
Scegli file Nessun file selezionato

2. Model performance

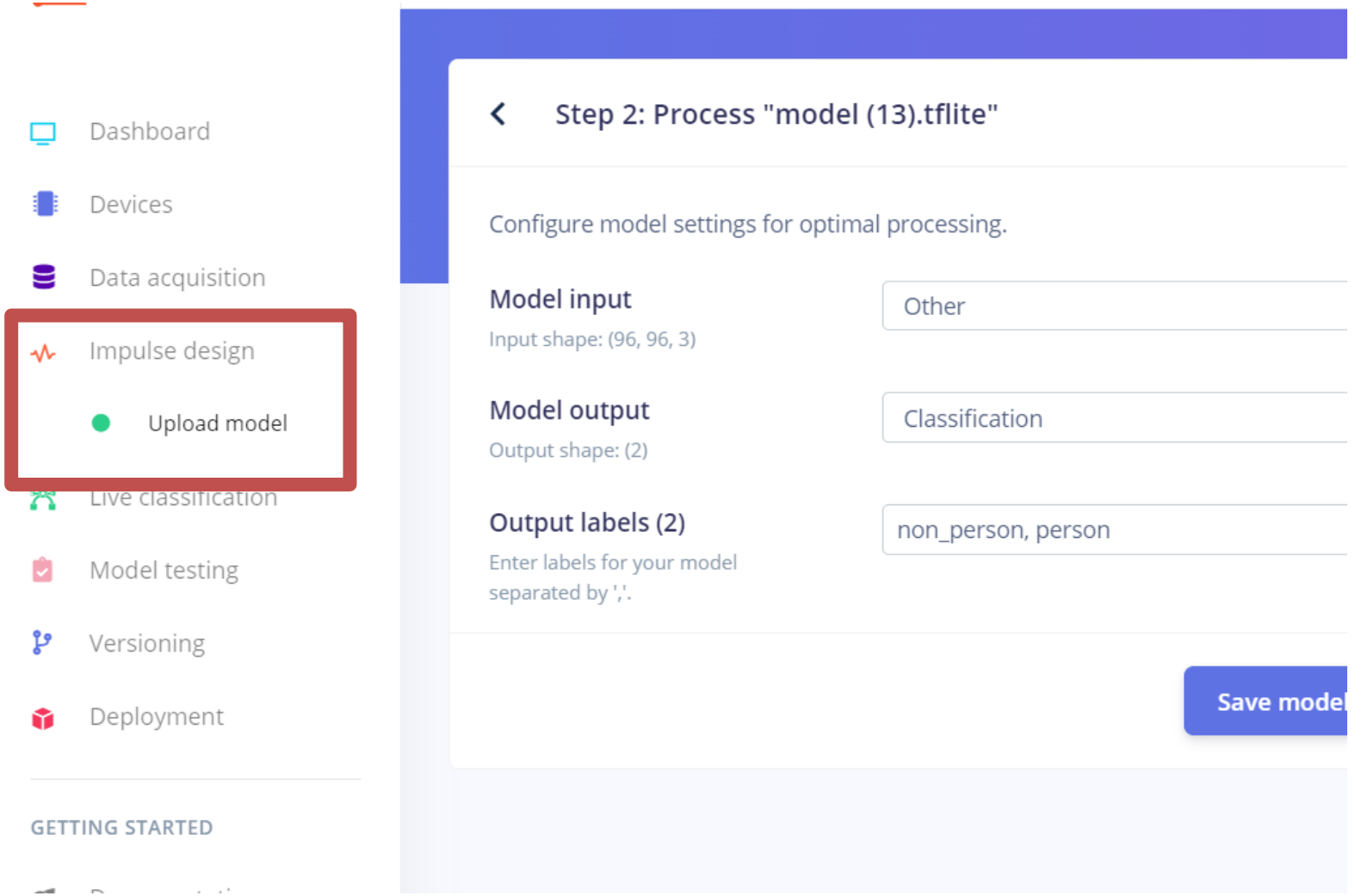
Do you want performance characteristics (latency, RAM and ROM) for a specific device?

- ○ No, show me performance for a range of device types.
- ◉ Yes, run performance profiling for: Arduino Nano 33 BLE Sense (Cortex-M4F 64MHz)

Upload file


Step 2: Process "model (13).tflite"
Configure model settings for optimal processing.
Model input
Input shape: (96, 96, 3)
Image (RGB)
Input should be in RGB format (one value per pixel), scaled 0..1. If your model is scaled differently or uses a different channel order, then select "Other".
Model output
Output shape: (2)
Classification
Output labels (2)
Enter labels for your model separated by ','.
non_person, person
Save model
Your model must be trained on 0…1 scaled data!
On-device performance
Arduino Nano 33 BLE ...
PROCESSING TI...
527098 ms.
RAM USAGE
138,2K
FLASH USAGE
39,1K
Not always reliable… better see what the device prints
Check model behavior
Upload test data to ensure correct model settings and proper model processing. (Optional)
Upload an image
Upload an image to try out your model. The image will be automatically resized to 96x96 (RGB).
Scegli file
Nessun file selezionato
Test sample
Test the pre-processing and input pipeline here!

- When importing a model like this, it is not possible to use pre-processing blocks…
- All must be done inside of the network
- Other pre-processing needed must be written by hand inside the code of the application



The Visual Wake Word Example

Person Detection – VWW Detection:

https://colab.research.google.com/drive/1sJmtTFxHr6faM0RbSE8CzDriFOVs0BWB?usp=sharing

POLITECNICO
MILANO 1863

Deployment Options comparison

Which deployment option should I use?

*These are personal opinions/advices, as long as you are able to make it work, use whatever you like

	TFLite4Micro	Edge Impulse
Experience required	Higher	Lower
Model trained in colab	Ok	Okayish
Model trained in EI	No	Yes
Camera drivers working	Require some work	Yes
Works with other type of sensor	Yes	Yes but with some limitations
Limits to the output type/dimension	No	Yes

POLITECNICO
MILANO 1863

Appendix

Credits and reference

- "TinyML: Machine Learning with TensorFlow Lite on Arduino and Ultra-Low-Power Microcontrollers", Daniel Situnayake, Pete Warden, O'Reilly Media, Inc.
- Online course:
  - https://www.edx.org/professional-certificate/harvardx-tiny-machine-learning
- A lot more material on TinyML:
  - http://tinyml.seas.harvard.edu/

Special thanks to Shalby Hazem who helped me in finding an alternative to the old code/repository

Test model

https://oreil.ly/NN6Mj