CEC Continuing Education Center

DesignNews

Machine Learning Application Design using STM32 MCU’s

DAY 5 : Running an Inference on Target

Sponsored by

Digi-Key ELECTRONICS

informa markets

Webinar Logistics

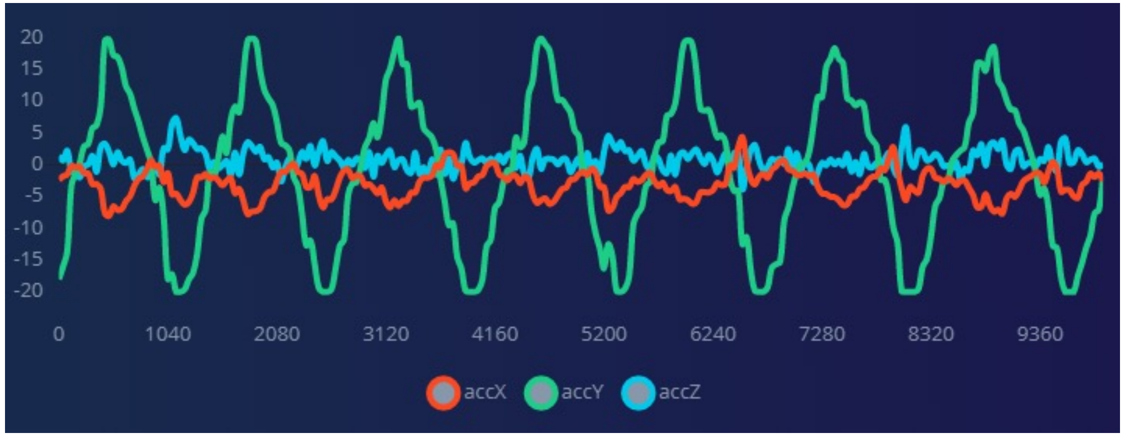
- Turn on your system sound to hear the streaming presentation.
- If you have technical problems, click “Help” or submit a question asking for assistance.
- Participate in ‘Group Chat’ by maximizing the chat widget in your dock.
- Submit questions for the lecturer using the Q&A widget. They will follow-up after the lecture portion concludes.

Course Sessions

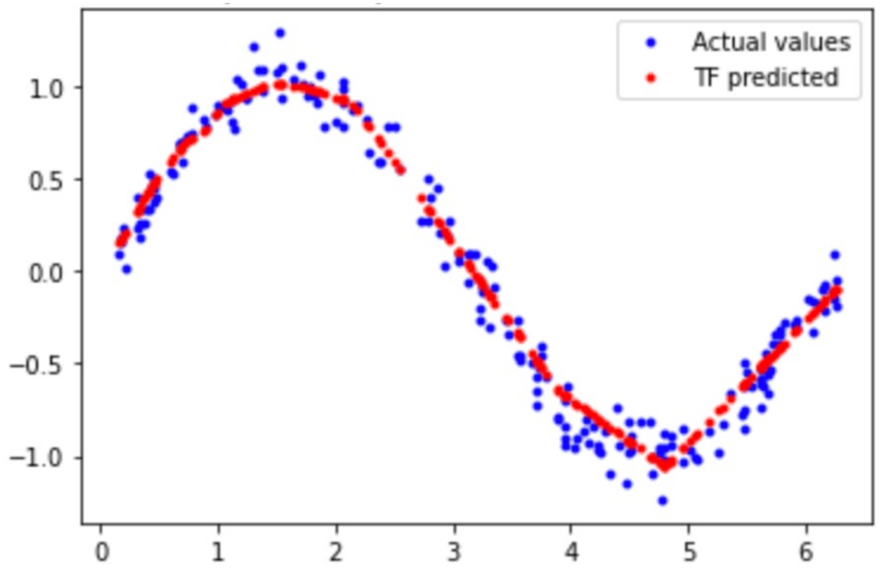
- Introduction to Machine Learning on MCU’s
- Capturing, Cleaning and Digital Signal Processing Data
- Training a Neural Network Part 1
- Training a Neural Network Part 2
- **Running an Inference on Target**

Two Models to Deploy

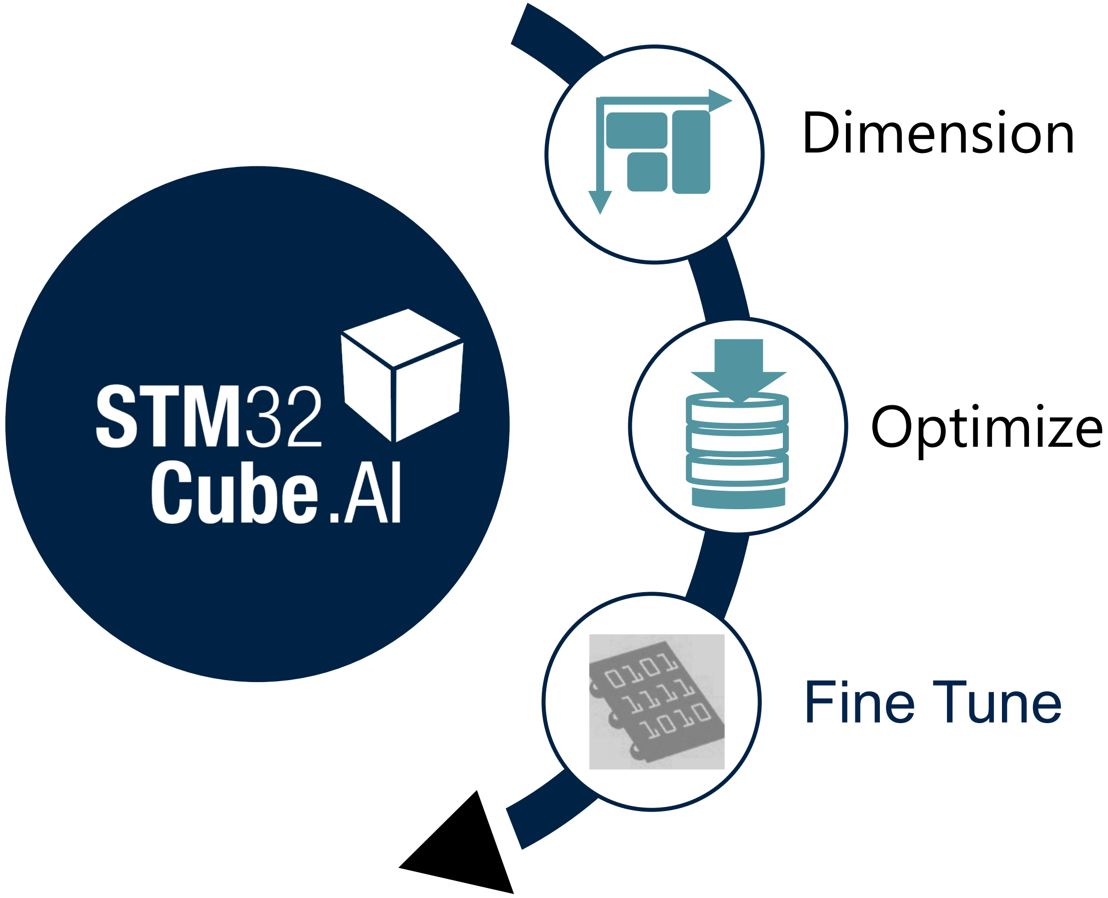
1) Gesture Detection



2) Hello World – Sine Wave



STM32Cube.AI Overview



Dimension

- ✓ Quickly assess model footprint requirements
- ✓ Select and configure MCU in STM32CubeMX
- ✓ Review model layers in STM32Cube.AI

Optimize

- ✓ Generate C-code for pre-trained model
- ✓ Support quantized models to reduce RAM, flash and latency with minimal loss of accuracy
- ✓ Use light run-time libraries
- ✓ Optimize for performance

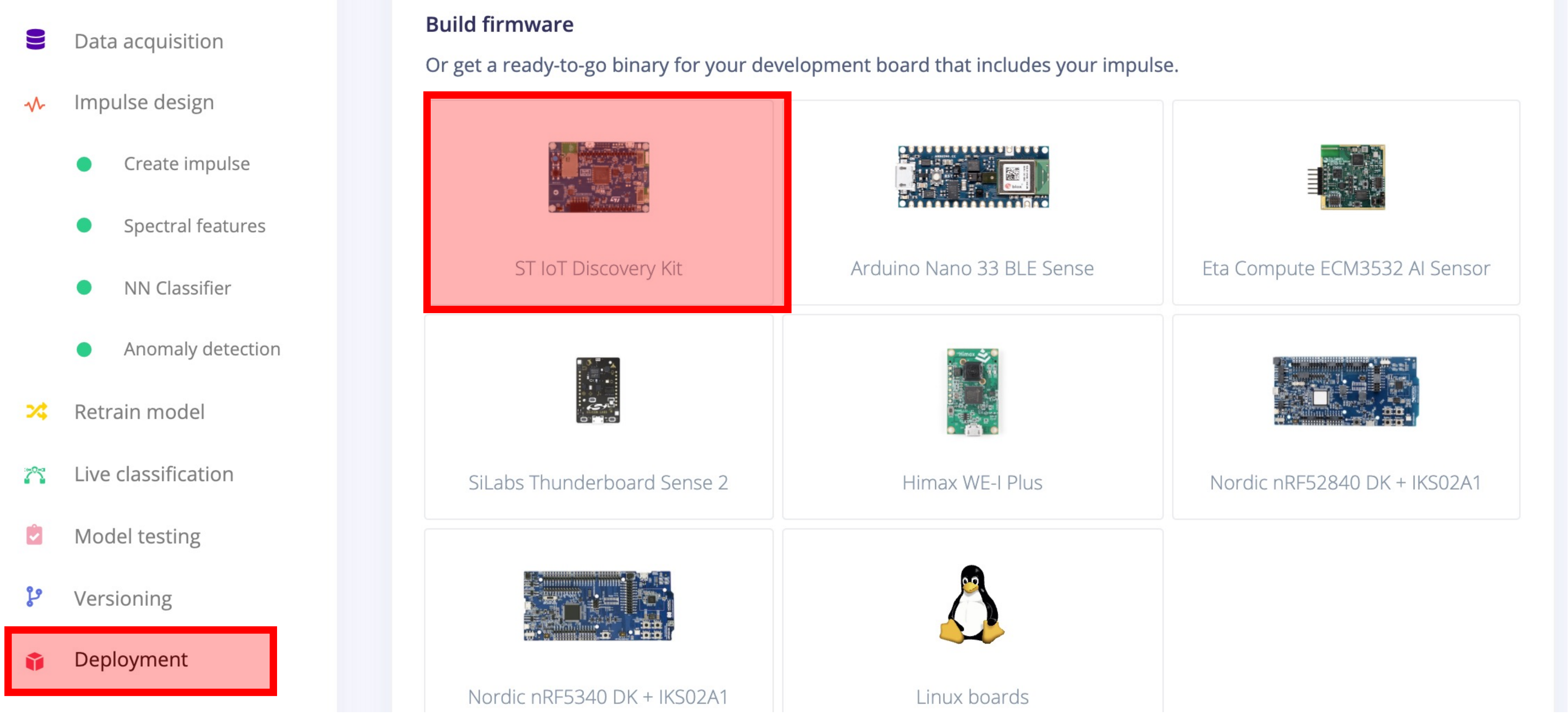
Fine Tune

- ✓ Optimize memory allocation
- ✓ Fine control of weight mapping
- ✓ Split between internal and external memory
- ✓ Update model without full FW update

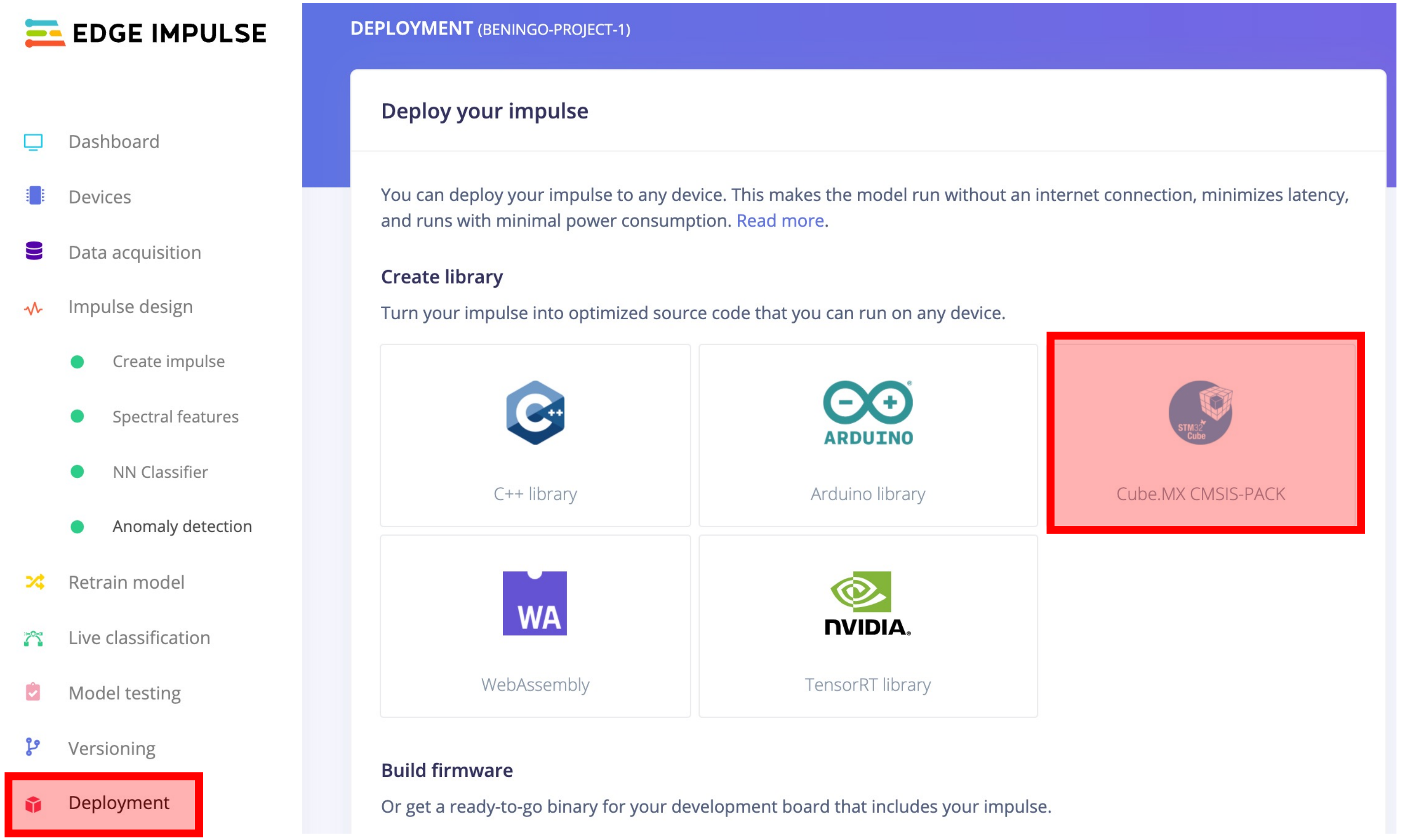
And quickly iterate thanks to on-target validation

Gesture Detection Deployment

Exporting an STM32 Binary



Exporting an STM32 AI pack



What method do you prefer for testing?
- Using the prebuilt binary
- Using the pack
- C++ library
- other

Apply Optimization(s)

Available optimizations for NN Classifier

Quantized (int8)

Currently selected

This optimization is recommended for best performance.

RAM USAGE: 1.5K

LATENCY: 1 ms

ROM USAGE: 15.4K

ACCURACY: 95.68%

CONFUSION MATRIX

87.8	7.0	0	0	5.2
0	99.2	0	0	0.8
0	0	100	0	0
-	-	-	-	-

Unoptimized (float32)

Click to select

RAM USAGE: 1.5K

LATENCY: 1 ms

ROM USAGE: 17.7K

ACCURACY: 95.88%

CONFUSION MATRIX

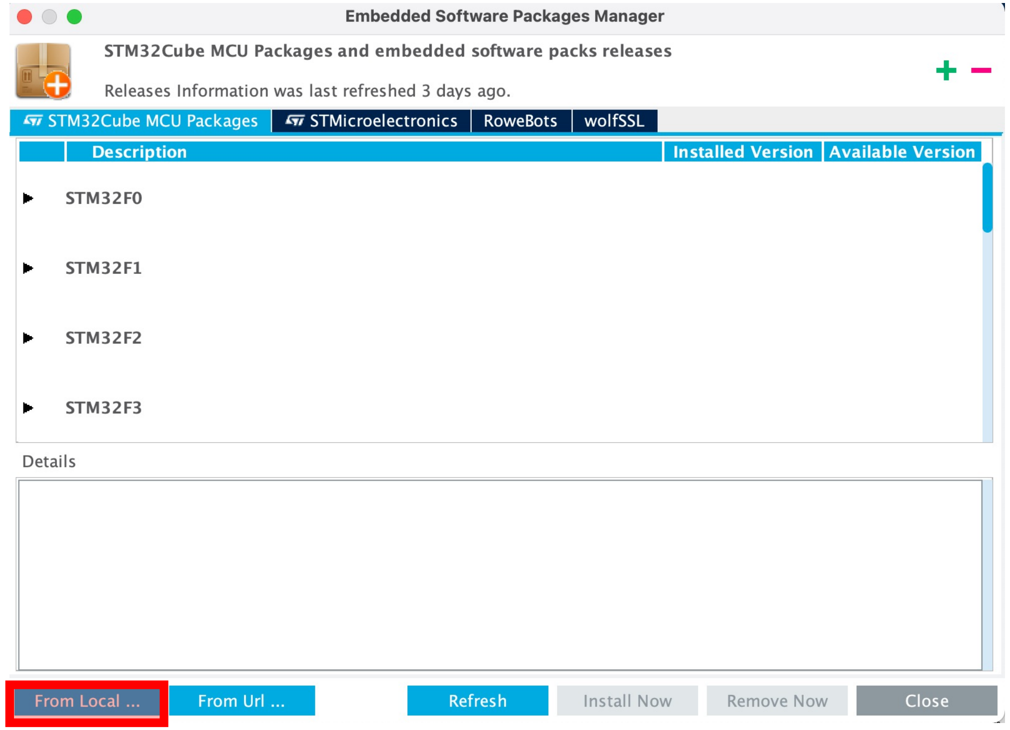
87.6	6.6	0	0	5.8
0	100	0	0	0
0	0	100	0	0
-	-	-	-	-

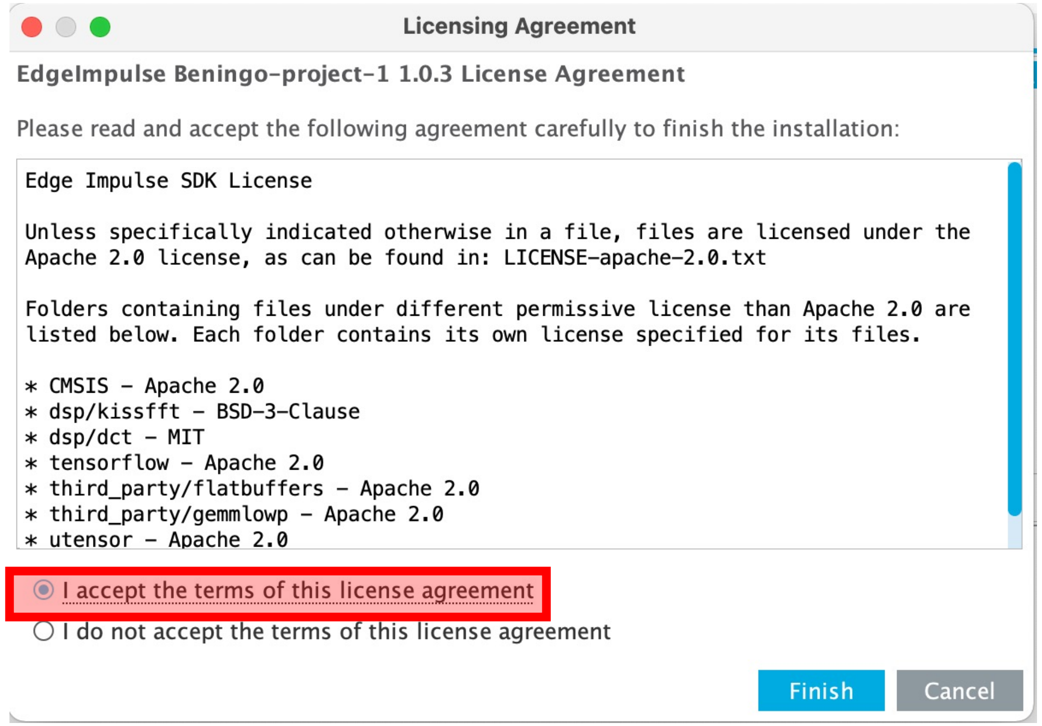
Estimate for Cortex-M4F 80MHz (ST IoT Discovery Kit)

Build

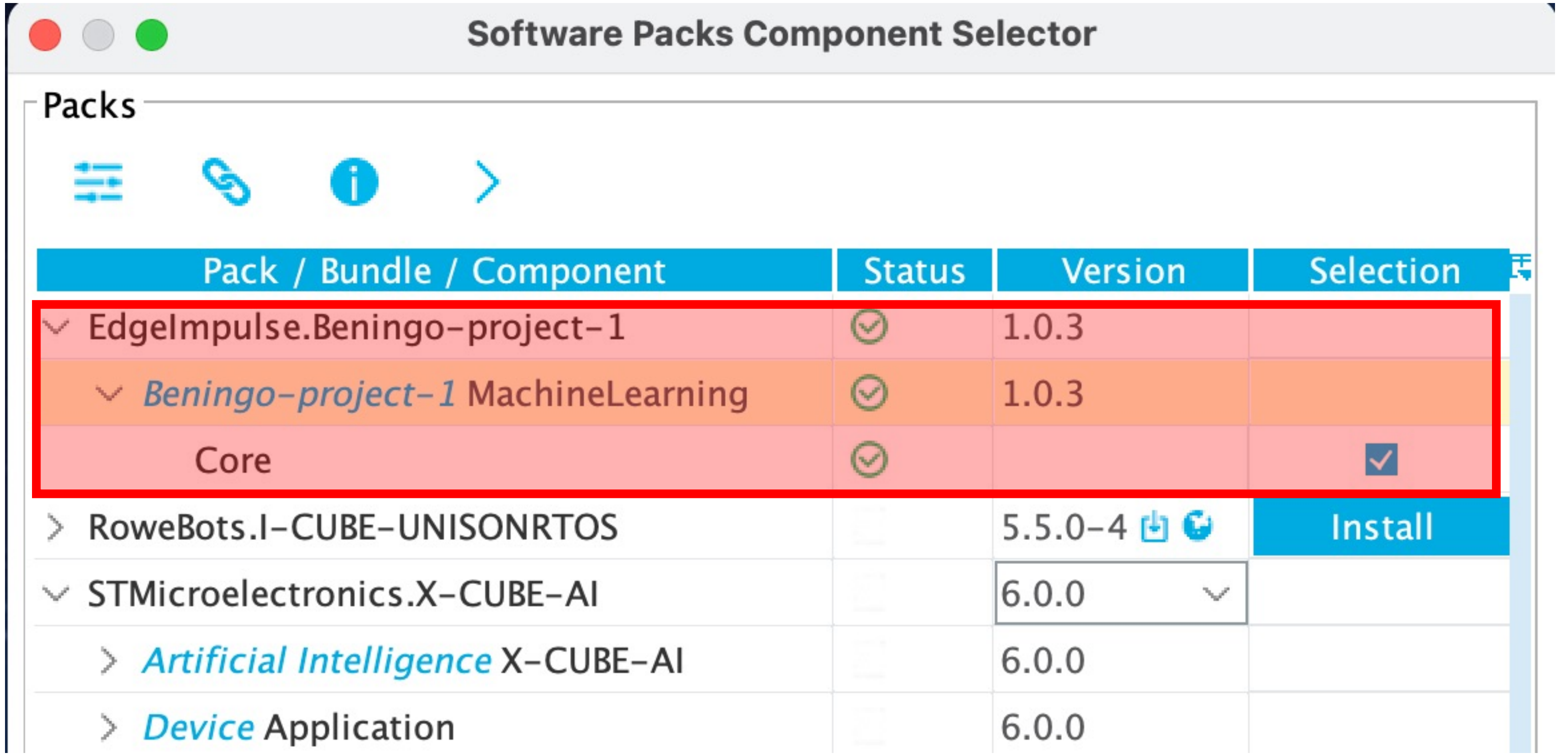
Install Pack into STM32Mx Project

Help -> Manage Embedded Software Packages

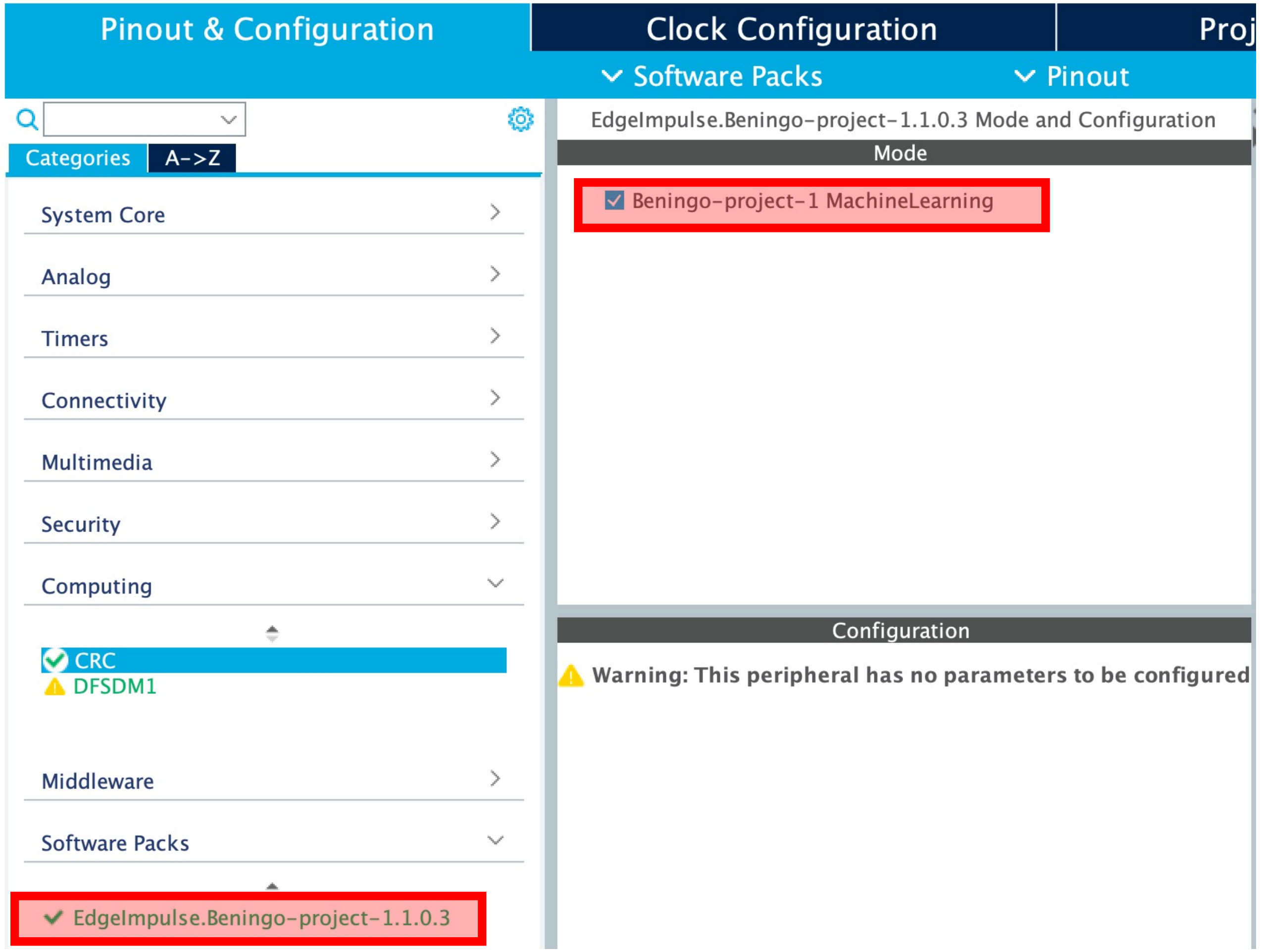




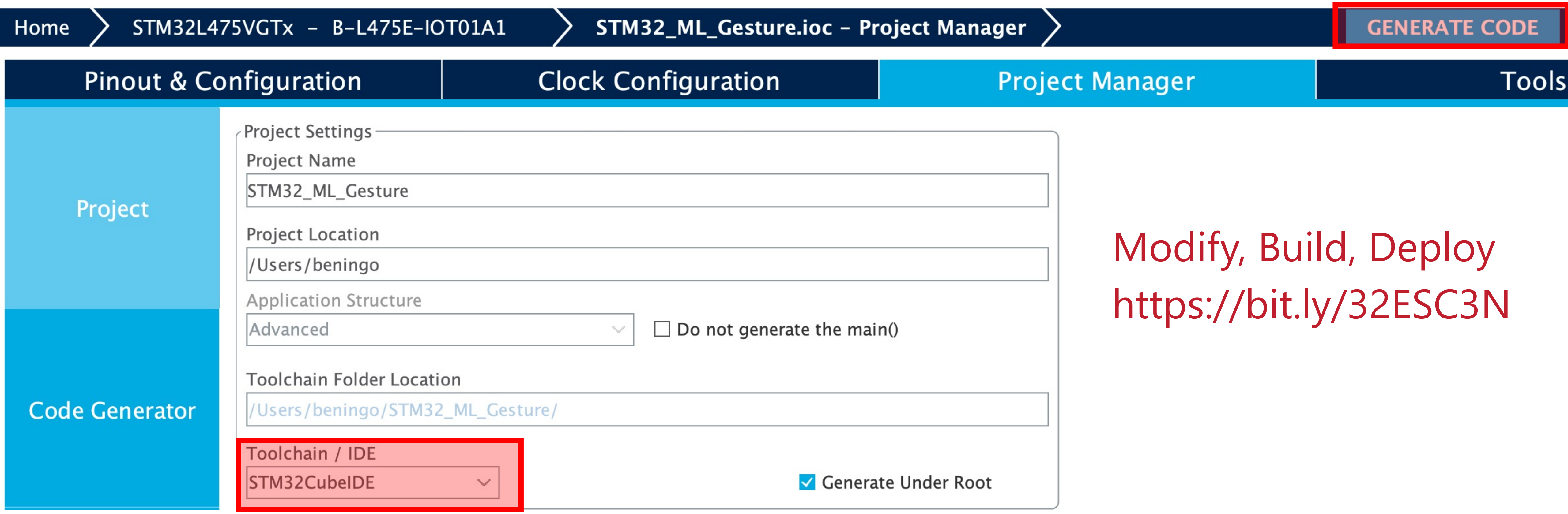
Install Pack into STM32Mx Project



Install Pack into STM32Mx Project



Install Pack into STM32Mx Project



Modify, Build, Deploy
https://bit.ly/32ESC3N

Running the Model

In a terminal, run the command: edge-impulse-run-impulse

```
Starting inferencing in 2 seconds...
Sampling... Storing in file name: /fs/device-classification.116
Predictions (DSP: 14 ms., Classification: 1 ms., Anomaly: 1 ms.):
    Circle: 0.99609
    Updown: 0.00000
    Wave: 0.00000
    anomaly score: -0.026
Finished inferencing, raw data is stored in '/fs/device-classification.116'. Use AT+UPLOADFILE to send back to Edge Impulse.
```

```
Starting inferencing in 2 seconds...
Sampling... Storing in file name: /fs/device-classification.121
Predictions (DSP: 15 ms., Classification: 0 ms., Anomaly: 2 ms.):
    Circle: 0.00000
    Updown: 0.00000
    Wave: 0.99609
    anomaly score: -0.132
Finished inferencing, raw data is stored in '/fs/device-classification.121'. Use AT+UPLOADFILE to send back to Edge Impulse.
```

Running the Model

```
Starting inferencing in 2 seconds...
Sampling... Storing in file name: /fs/device-classification.118
Predictions (DSP: 15 ms., Classification: 0 ms., Anomaly: 2 ms.):
    Circle: 0.01172
    Updown: 0.98828
    Wave: 0.00000
    anomaly score: -0.141
Finished inferencing, raw data is stored in '/fs/device-classification.118'. Use AT+UPLOADFILE to send back to Edge Impulse.
Starting inferencing in 2 seconds...
Sampling... Storing in file name: /fs/device-classification.119
Predictions (DSP: 14 ms., Classification: 1 ms., Anomaly: 1 ms.):
    Circle: 0.21094
    Updown: 0.78906
    Wave: 0.00000
    anomaly score: -0.164
Finished inferencing, raw data is stored in '/fs/device-classification.119'. Use AT+UPLOADFILE to send back to Edge Impulse.
```

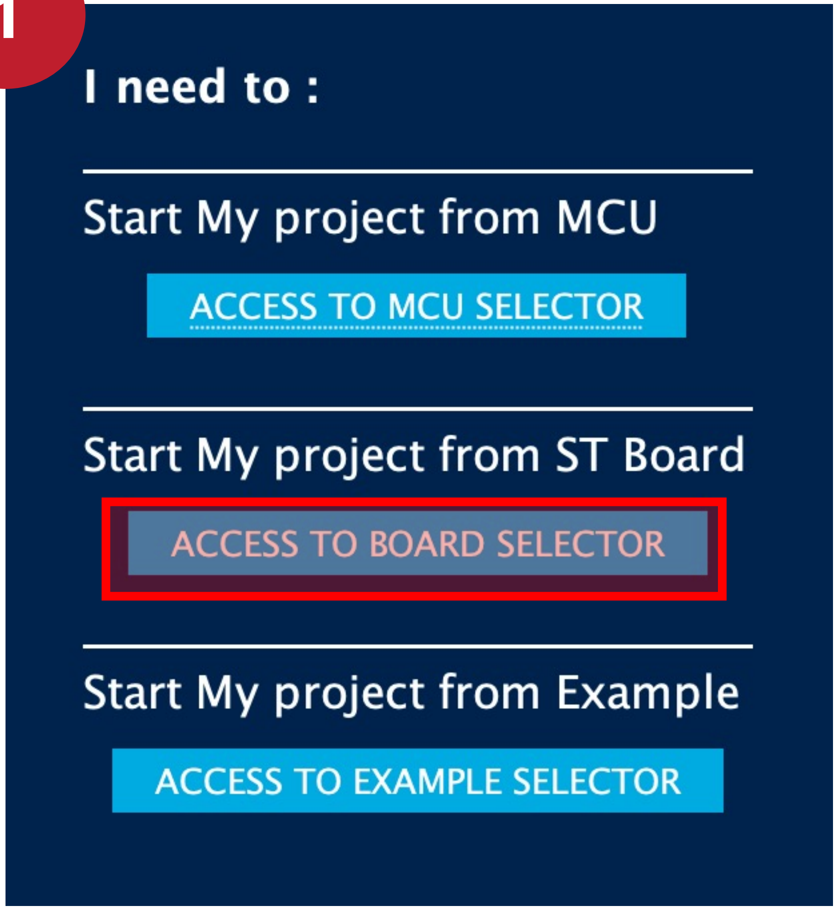
What methods can be used to improve classifaction ?

- Running average on the output
- Monitor the anomaly value
- Set a minimum classification percentage
- All the above
- Other

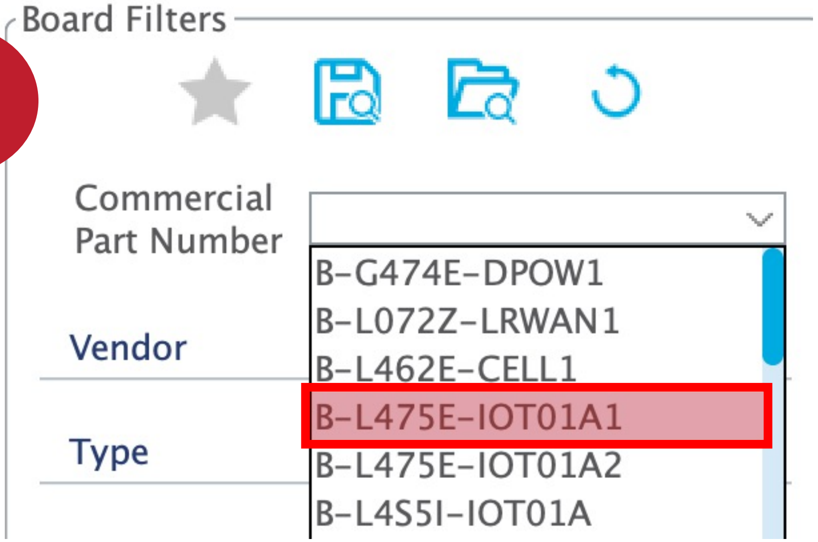
Hello World Deployment

Create a New STM32CubeMx Project

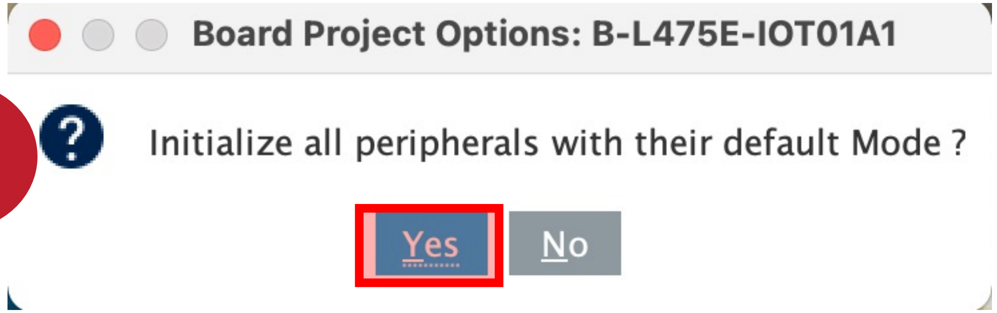
1



2



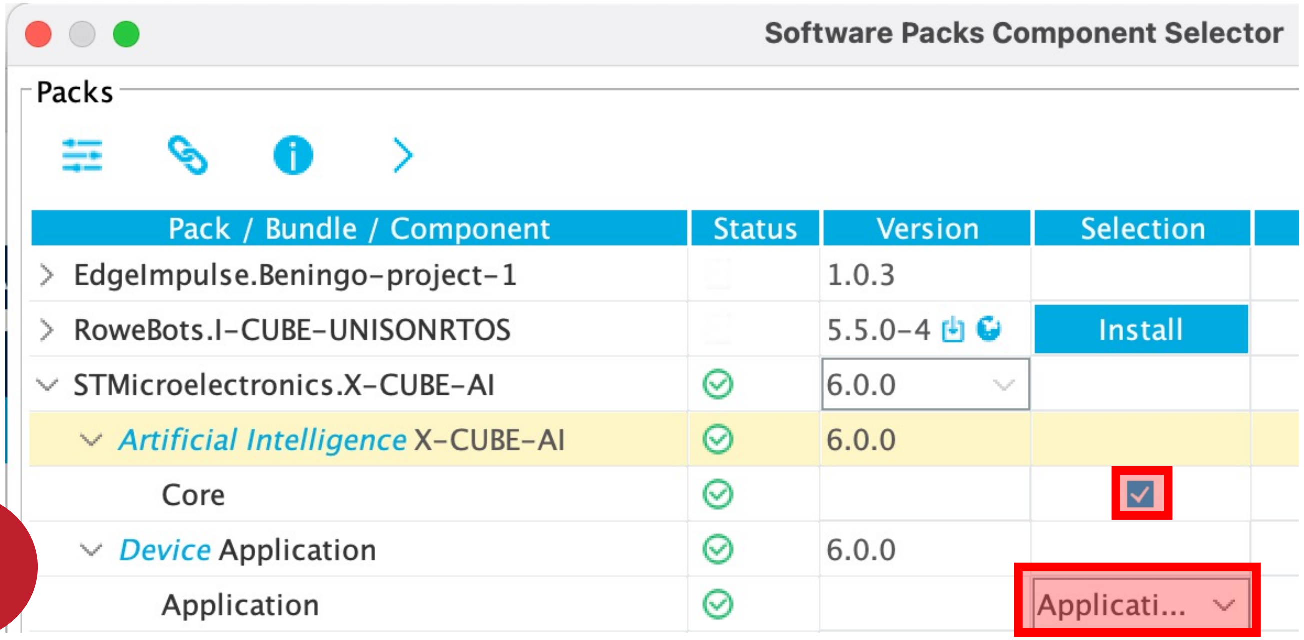
3



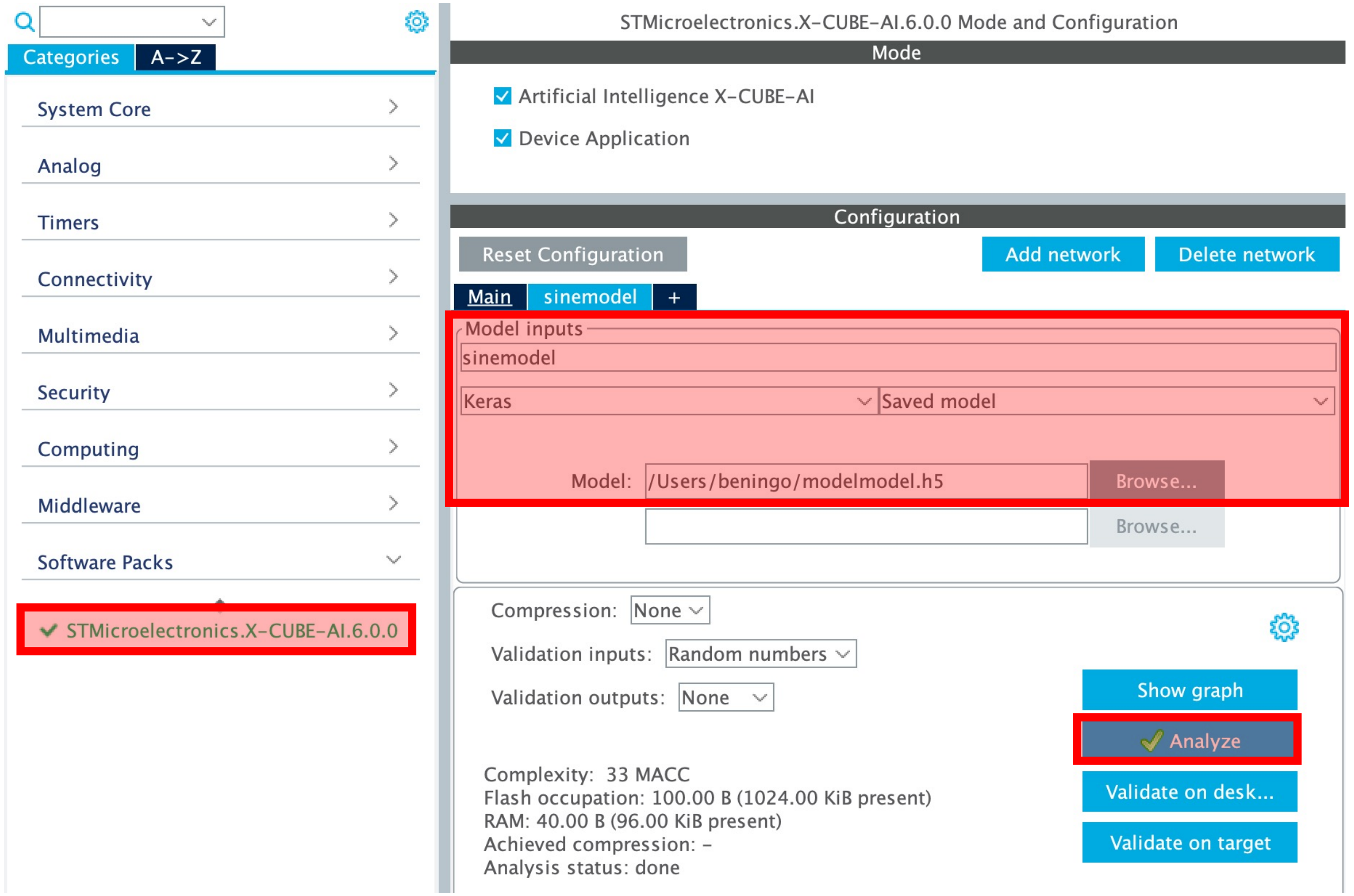
Add the AI Pack to the Project

1

2



Setup and Analyze the Keras Model

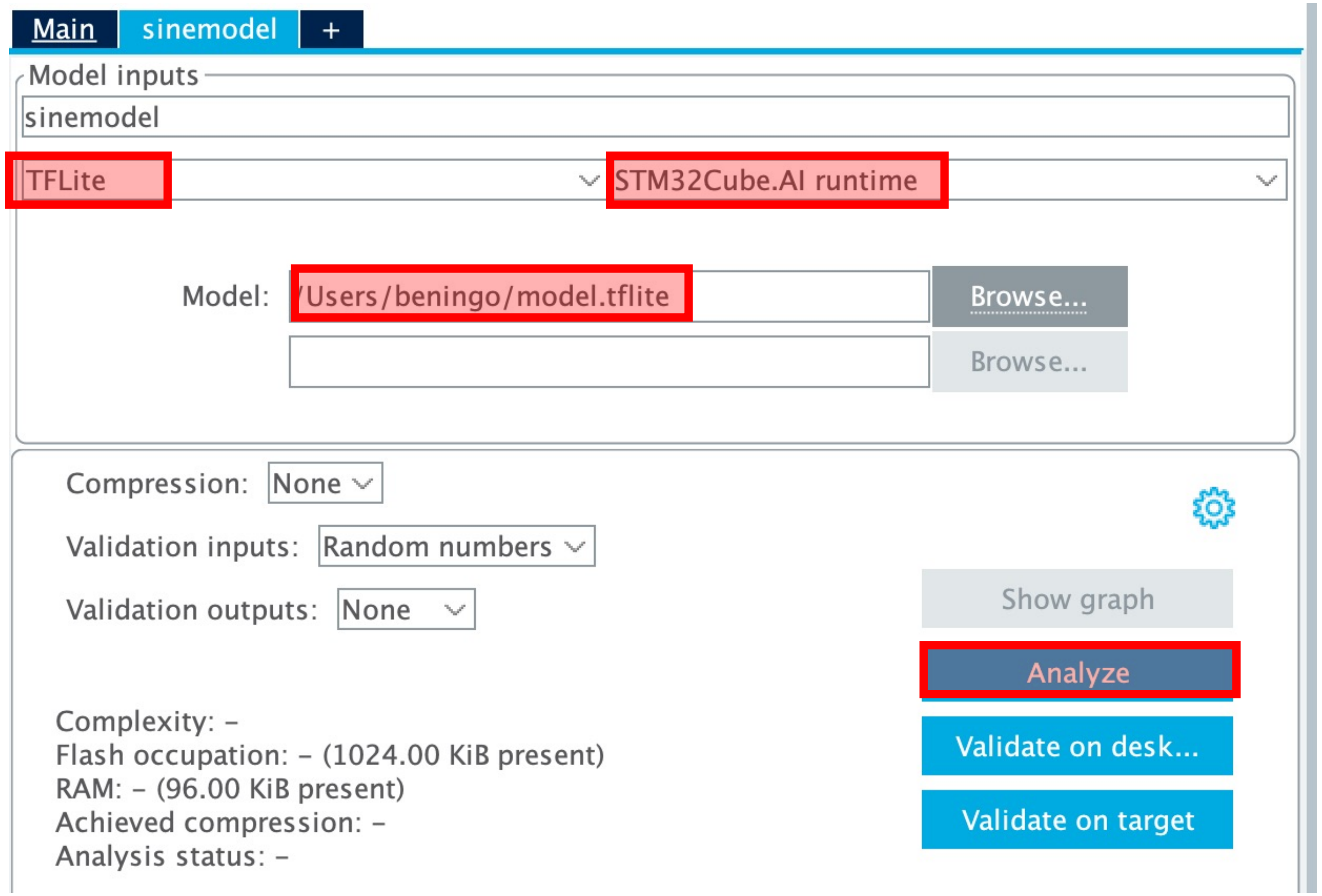


Keras Model Analysis

Analyzing Network

```
workspace dir        : /private/var/folders/l3/4x_x6z9x0zx47b1m2f05zvzw0000gn/T/mxAI_workspace17346681197813975593504316215803
output dir           : /Users/beningo/.stm32cubemx

model_name           : modelmodel
model_hash           : de2ced29f4c4bafd7d1dcf4c98240154
input                : input_0 [1 items, 4 B, ai_float, FLOAT32, (1, 1, 1)]
inputs (total)       : 4 B
output               : dense_1 [1 items, 4 B, ai_float, FLOAT32, (1, 1, 1)]
outputs (total)      : 4 B
params #             : 25 items (100 B)
macc                 : 33
weights (ro)         : 100 B (100 B)
activations (rw)     : 32 B (32 B)
ram (total)          : 40 B (40 B) = 32 + 4 + 4

Model name - modelmodel ['input_0'] ['dense_1']
-----------------------------------------------------------------------------------------------------------------
id   layer (type)              shape    param/size    macc    connected to    |   c_size   c_macc   c_type
-----------------------------------------------------------------------------------------------------------------
0    input_0 (Input)           (c:1)                                          |
     dense (Dense)             (c:8)    16/64         16      input_0         |                     dense()[0]
     dense_nl (Nonlinearity)   (c:8)                  8       dense           |                     nl()[1]
-----------------------------------------------------------------------------------------------------------------
1    dense_1 (Dense)           (c:1)    9/36          9       dense_nl        |                     dense()/o[2]
-----------------------------------------------------------------------------------------------------------------
model/c-model: macc=33/33  weights=100/100  activations=--/32 io=--/8

Complexity report per layer - macc=33 weights=100 act=32 ram_io=8
-------------------------------------------------------------------------
id   name        c_macc                            c_rom                       c_id
-------------------------------------------------------------------------
0    dense       ||||||||||||||||   48.5%    ||||||||||||||||   64.0%   [0]
0    dense_nl    ||||||||           24.2%    |                   0.0%   [1]
1    dense_1     |||||||||          27.3%    |||||||||          36.0%   [2]
```

Setup and Analyze the TensorFlow Lite Model



TFLite Model Analysis

```
model_name          : model
model_hash          : 1c2d5a21b889b8e12b4284a72cfb10fb
input               : serving_default_dense_2_input0_int8 [1 items, 1 B, ai_i8, scale=0.024573976173996925, zero_point=-128
inputs (total)      : 1 B
output              : dense_2 [1 items, 1 B, ai_i8, scale=0.008472034707665443, zero_point=4, (1, 1, 1)]
outputs (total)     : 1 B
params #            : 321 items (420 B)
macc                : 321
weights (ro)        : 420 B (420 B)
activations (rw)    : 32 B (32 B)
ram (total)         : 34 B (34 B) = 32 + 1 + 1

Model name - model ['serving_default_dense_2_input0_int8'] ['dense_2']
------------------------------------------------------------------------------------------------------------------------
id   layer (type)                                shape     param/size   macc    connected to                          |
------------------------------------------------------------------------------------------------------------------------
0    serving_default_dense_2_input0_int8 (Input) (c:1)                                                                |
     dense_0 (Dense)                             (c:16)    32/80        32      serving_default_dense_2_input0_int8   |
     nl_0_nl (Nonlinearity)                      (c:16)                 16      dense_0                               |
------------------------------------------------------------------------------------------------------------------------
1    dense_1 (Dense)                             (c:16)    272/320      272     nl_0_nl                               |
     nl_1_nl (Nonlinearity)                      (c:16)                 16      dense_1                               |
------------------------------------------------------------------------------------------------------------------------
2    dense_2 (Dense)                             (c:1)     17/20        17      nl_1_nl                               |
------------------------------------------------------------------------------------------------------------------------
model/c-model: macc=353/321 -32(-9.1%) weights=420/420  activations=--/32 io=--/2

Complexity report per layer - macc=321 weights=420 act=32 ram_io=2
------------------------------------------------------------------------
id   name      c_macc                        c_rom                    c_id
------------------------------------------------------------------------
0    dense_0   ||                 10.0%      ||||              19.0%   [0]
1    dense_1   ||||||||||||||||   84.7%      ||||||||||||||||  76.2%   [1]
2    dense_2   |                   5.3%      |                  4.8%   [2]
```

Cross Platform Validation Reports

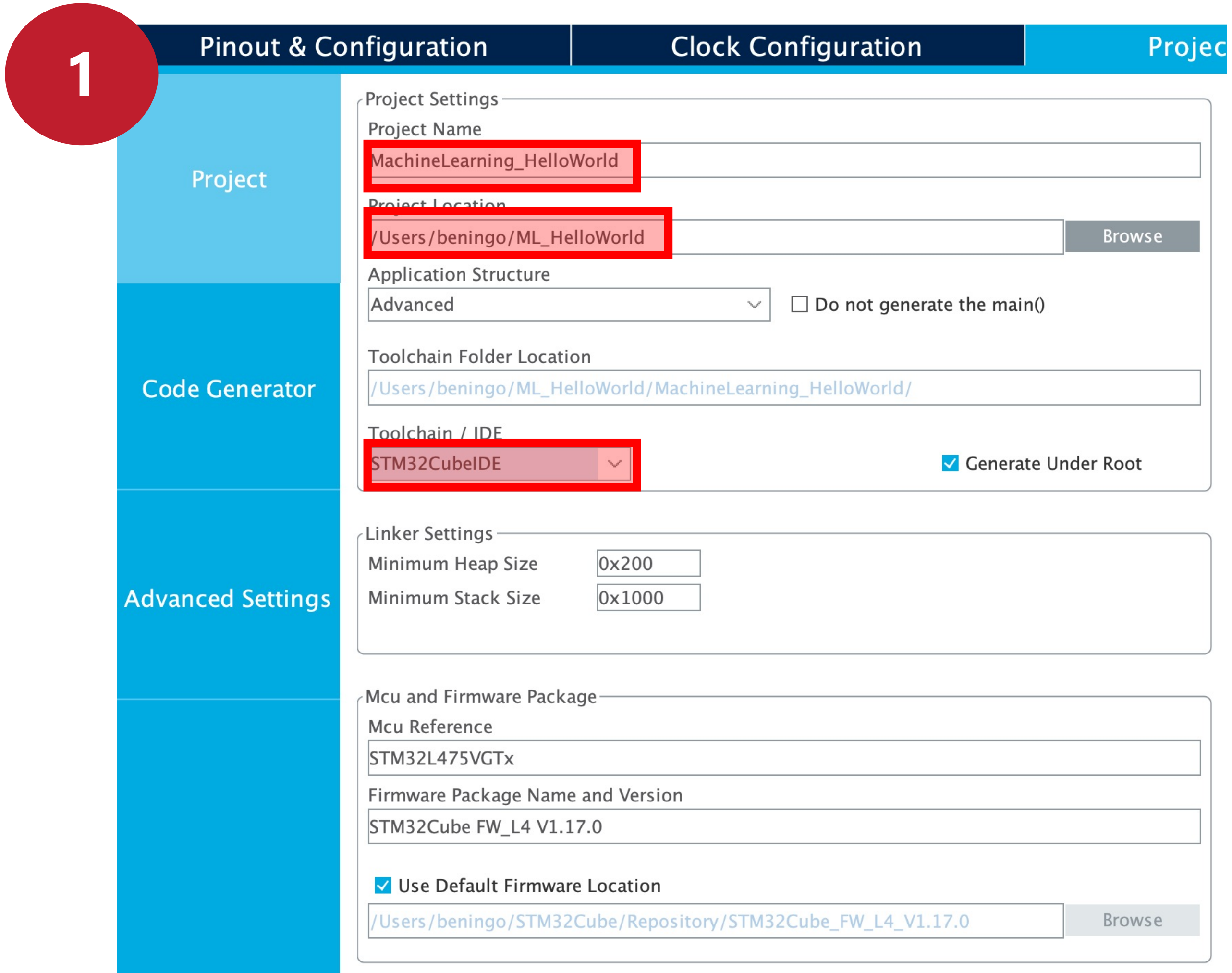
Keras

```
Cross accuracy report #1 (reference vs C-model)
----------------------------------------------------------------------------------------------------
NOTE: the output of the reference model is used as ground truth/reference value
NOTE: ACC metric is not computed ("--classifier" option can be used to force it)

 acc=n.a., rmse=0.000000128, mae=0.000000104, l2r=0.000000225

Evaluation report (summary)
------------------------------------------------------------------------------------------------------
Mode         acc    rmse          mae           l2r           tensor
------------------------------------------------------------------------------------------------------
X-cross #1   n.a.   0.000000128   0.000000104   0.000000225   dense_5, ai_float, [(1, 1, 1)], m_id=[2]
------------------------------------------------------------------------------------------------------

X-cross (l2r) #1 error : 2.25302543e-07 (expected to be < 0.01)
```

TF Lite

```
Cross accuracy report #1 (reference vs C-model)
----------------------------------------------------------------------------------------------------
NOTE: the output of the reference model is used as ground truth/reference value
NOTE: ACC metric is not computed ("--classifier" option can be used to force it)

 acc=n.a., rmse=0.000000000, mae=0.000000000, l2r=0.000000000

Evaluation report (summary)
----------------------------------------------------------------------------------------------------
Mode         acc    rmse          mae           l2r           tensor
----------------------------------------------------------------------------------------------------
X-cross #1   n.a.   0.000000000   0.000000000   0.000000000   dense_2, ai_i8, [(1, 1, 1)], m_id=[2]
----------------------------------------------------------------------------------------------------

X-cross (rmse) #1 error : 0.00000000e+00 (expected to be < 0.01)

Creating txt report file /Users/beningo/.stm32cubemx/sinemodel_validate_report.txt
elapsed time (validate): 1.453s
```

Generate the Model



The Application Template

```
    /* Initialize all configured peripherals */
    MX_GPIO_Init();
    MX_CRC_Init();
    MX_DFSDM1_Init();
    MX_I2C2_Init();
    MX_QUADSPI_Init();
    MX_SPI3_Init();
    MX_USART1_UART_Init();
    MX_USART3_UART_Init();
    MX_USB_OTG_FS_PCD_Init();
    MX_X_CUBE_AI_Init();
    /* USER CODE BEGIN 2 */

    /* USER CODE END 2 */

    /* Infinite loop */
    /* USER CODE BEGIN WHILE */
    while (1)
    {
      /* USER CODE END WHILE */

    MX_X_CUBE_AI_Process();
      /* USER CODE BEGIN 3 */
    }
    /* USER CODE END 3 */
}
```

```
/* Private user code ---------------------------------------------------------*/
/* USER CODE BEGIN 0 */
int _write(int file, char *ptr, int len)
{
    HAL_UART_Transmit(&huart1, (uint8_t *)ptr, len, 0xffffff); // send message via UART

    return len;
}
```

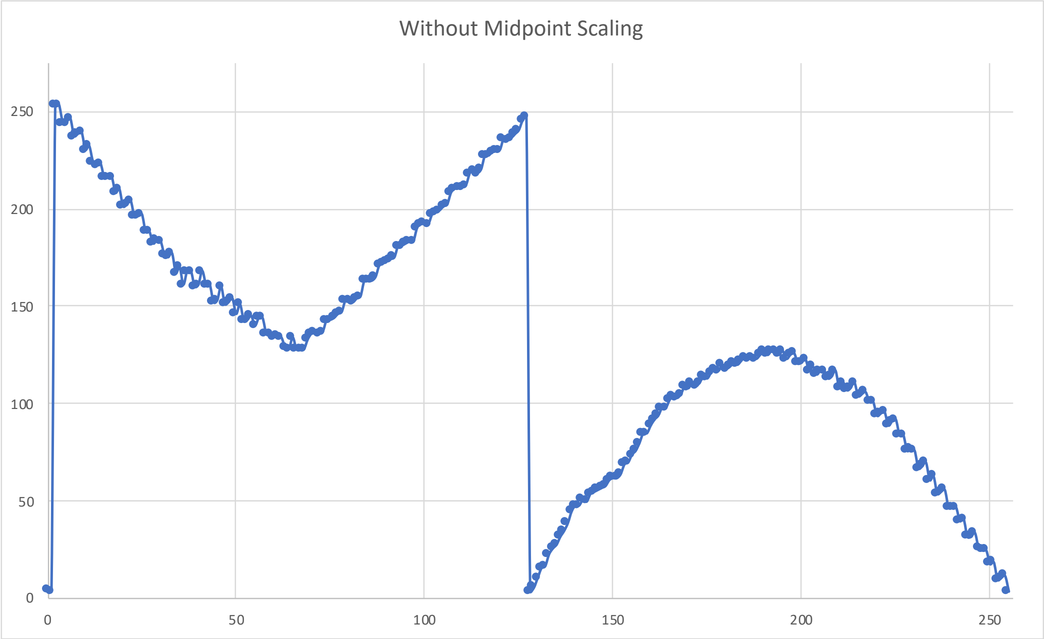
```
    /* 2 - main loop */
    do {
      /* 1 - acquire and pre-process input data */
      res = acquire_and_process_data(in_data);
      /* 2 - process the data - call inference engine */
      if (res == 0)
      {
        res = ai_run(in_data, out_data);
      }

      /* 3- post-process the predictions */
      if (res == 0)
      {
        res = post_process(out_data);
      }
    } while (res==0);
```

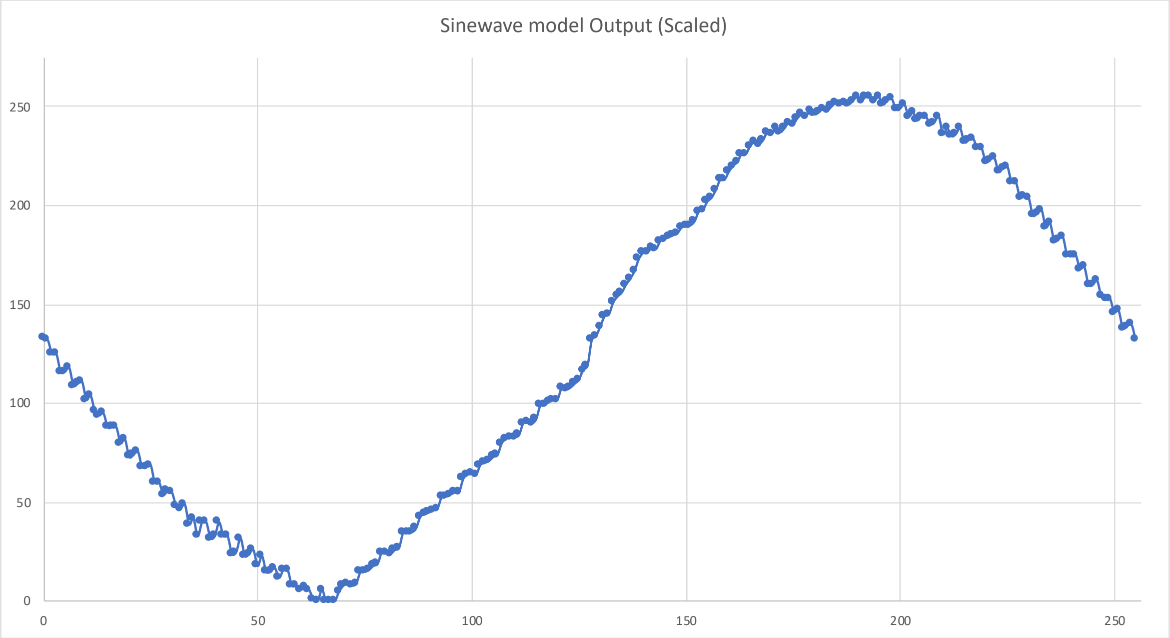
The Details

```c
/* USER CODE BEGIN 2 */
int acquire_and_process_data(void * data)
{
  static uint8_t position = 0;
  uint8_t * Value = data;

  * Value = position;

  position++;

  return 0;
}
```

```c
int post_process(void * data)
{
  uint8_t * Value = data;

  if(*Value >= 128)
  {
      *Value -= 128;
  }
  else
  {
      *Value += 128;
  }

  return 0;
}
```

Results

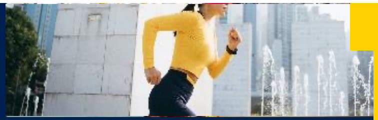

Without Midpoint Scaling
250
200
150
100
50
0
0
50
100
150
200
250


Sinewave model Output (Scaled)
250
200
150
100
50
0
0
50
100
150
200
250

Next Steps

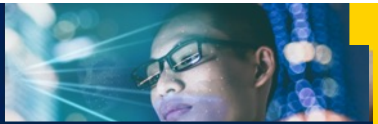
- Connect the output to a PWM LED channel
- Setup a DAC and drive an output voltage
- Configure the rate at which the inference runs (frequency control)
- Try and compare the Keras model behavior
- Improve the training model to provide a more accurate sine wave

Going Further


FP-AI-SENSING1
- People activity recognition
- Audio scene classification


FP-AI-NANOEDG1
- Condition-based monitoring


FP-AI-VISION1
- Person presence detection
- Food classification


FP-AI-FACEREC1
- Face recognition

https://bit.ly/3nf99EZ

1 

Introduction to STM32Cube.AI - 1 Marketing introduction
STMicroelectronics

2 

Introduction to STM32Cube.AI - 2 Theory of AI
STMicroelectronics

3 

Introduction to STM32Cube.AI - 3 Out of the box lab
STMicroelectronics

4 

Introduction to STM32Cube.AI - 4 NN Model creation using Keras
STMicroelectronics

5 

Introduction to STM32Cube.AI - 5 STM32Cube.AI labs
STMicroelectronics

What would you like to learn more about?
- Developing Keras based models
- How to develop ML test cases
- Building more complex ML edge applications
- other

Thank you for attending

Please consider the resources below:

- www.beningo.com
    - Blog, White Papers, Courses
    - Embedded Bytes Newsletter
        - http://bit.ly/1BAHYXm

From www.beningo.com under

- Blog > CEC – Machine Learning Application Design using STM32 MCUs

Thank You

Sponsored by