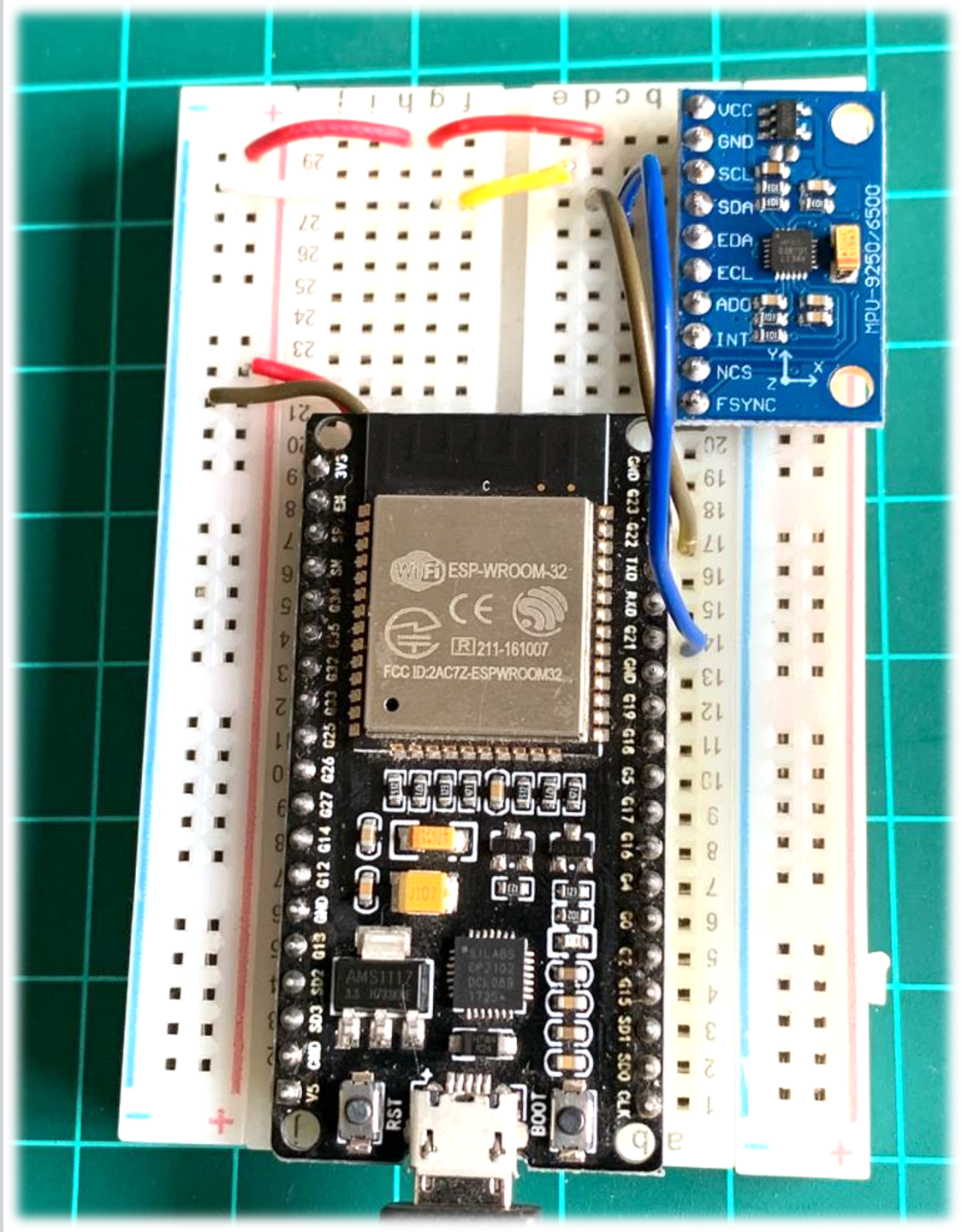
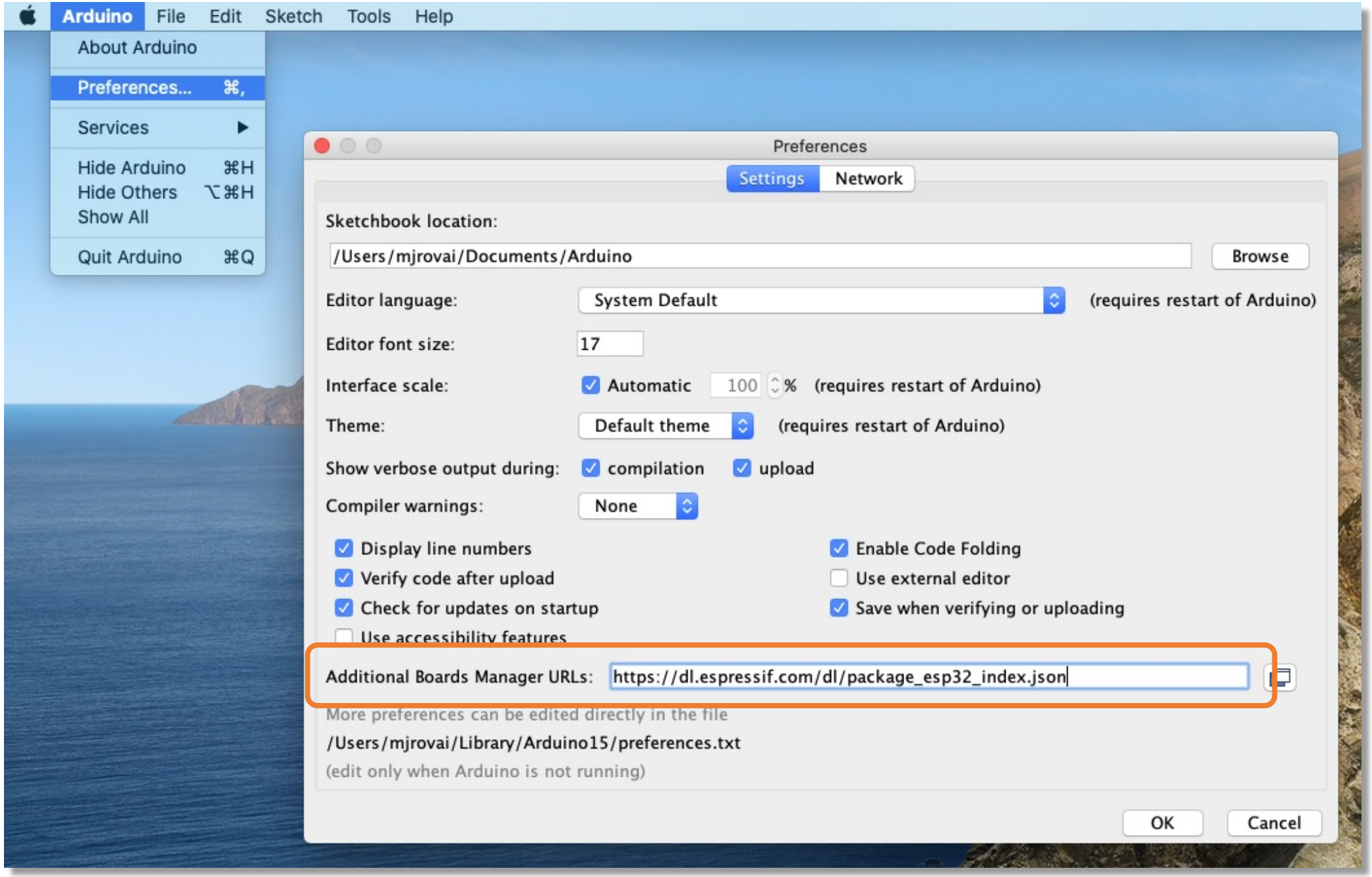
IESTI01 – TinyML

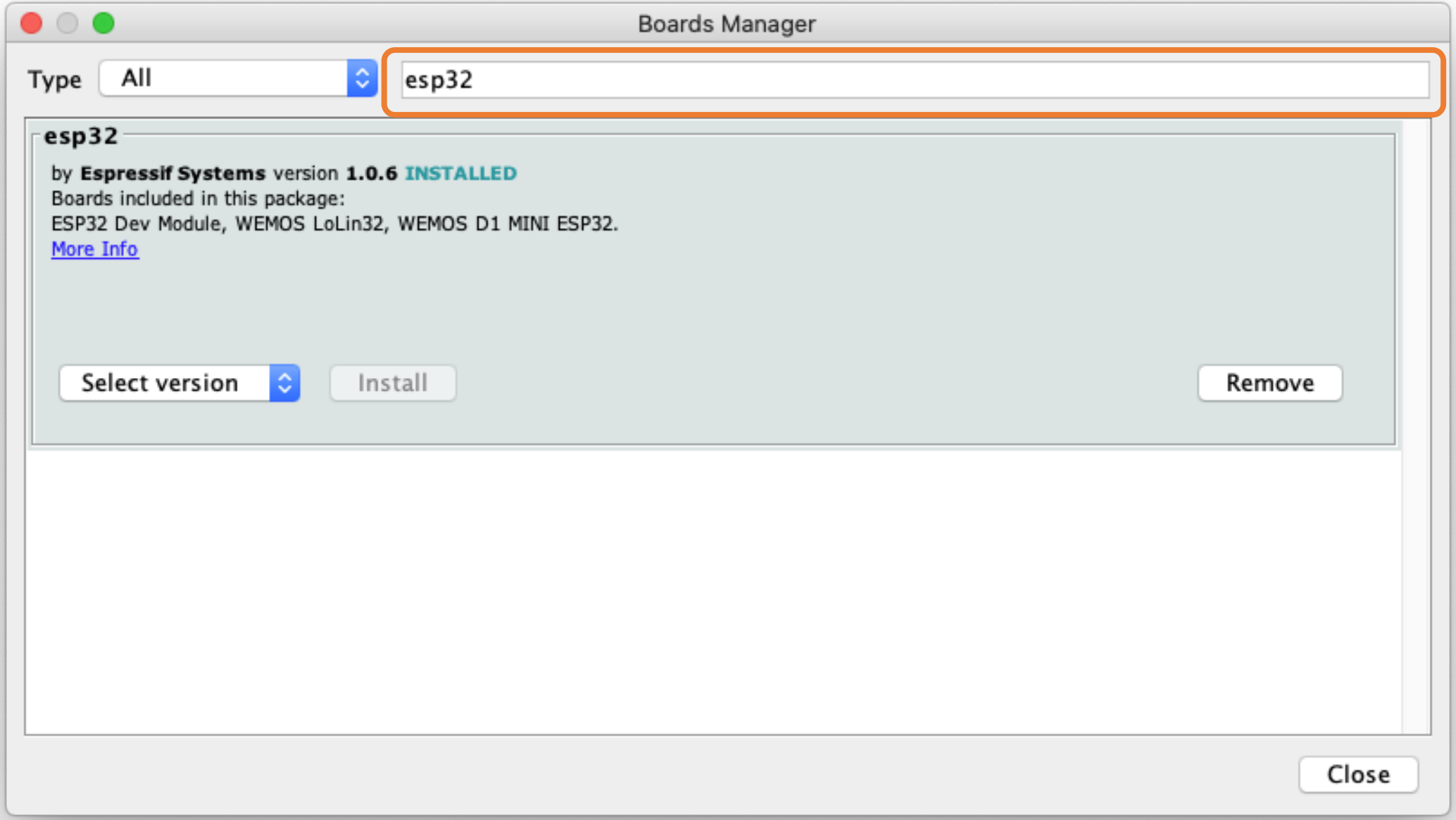
Embedded Machine Learning

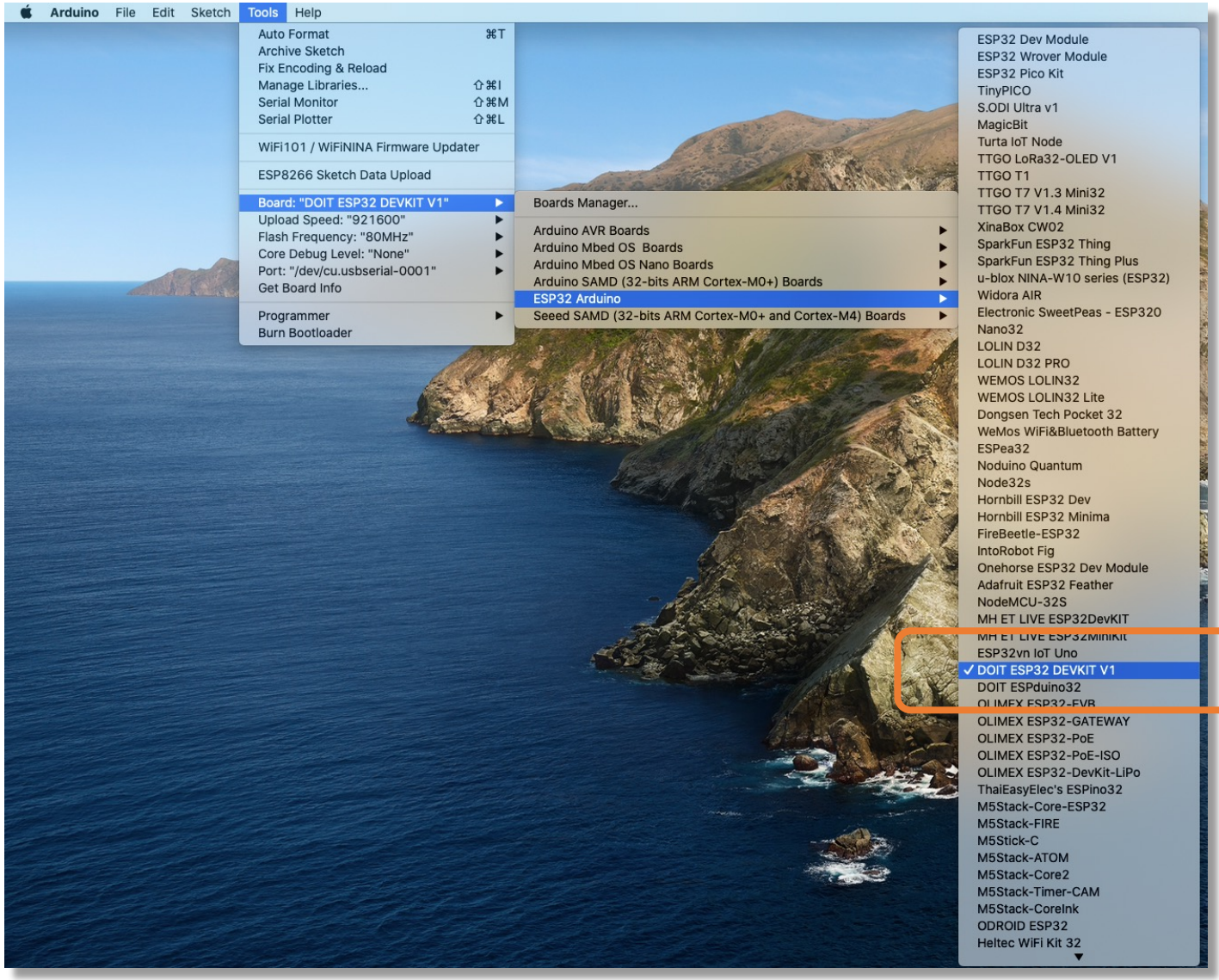
19.a Motion Classification - ESP32

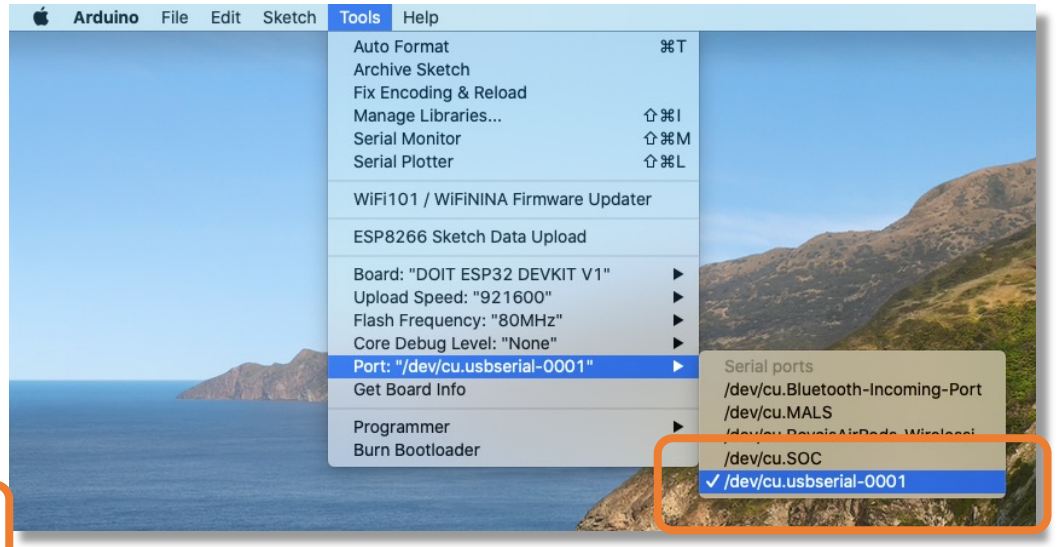


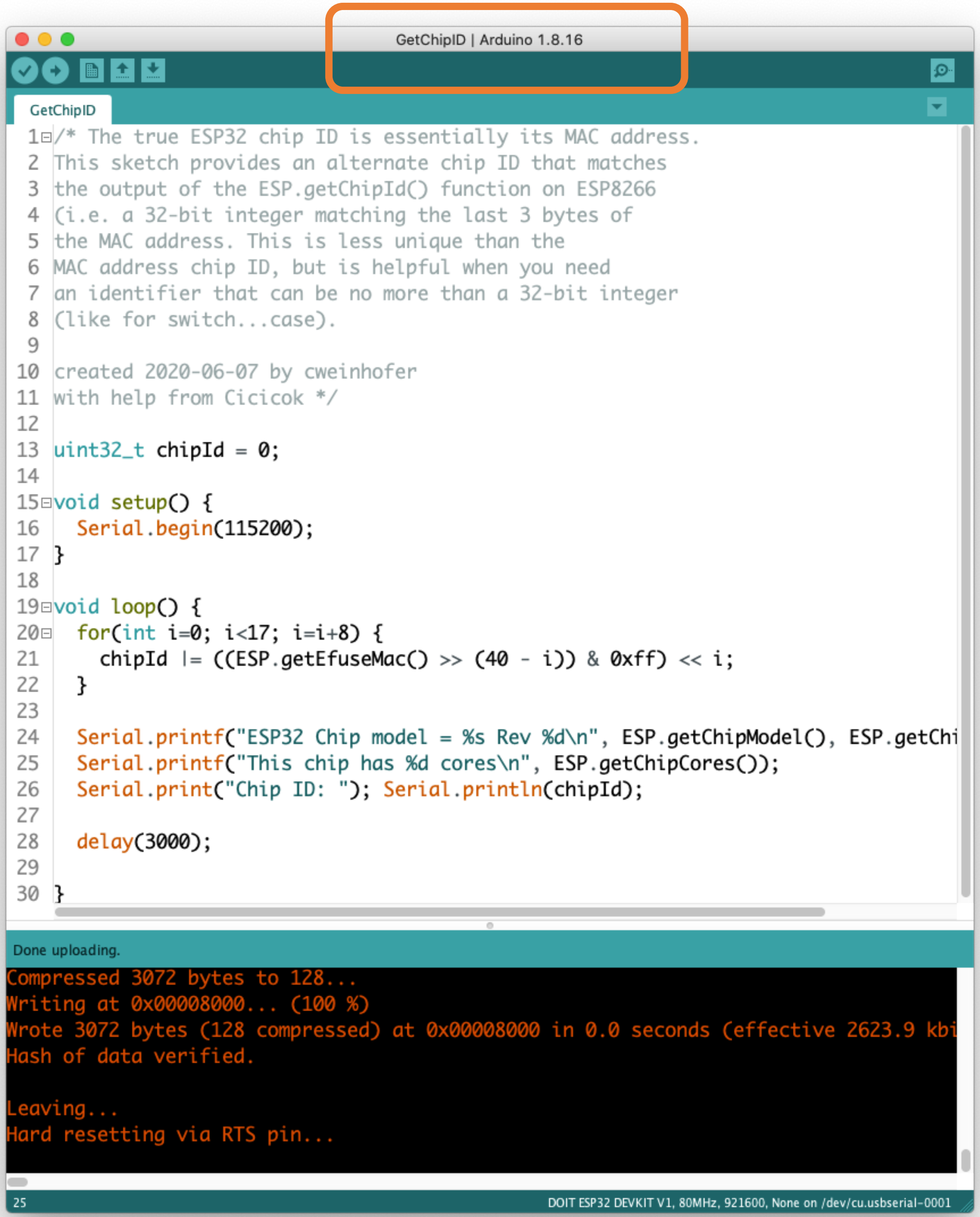
Prof. Marcelo Rovai
UNIFEI


Arduino
File
Edit
Sketch
Tools
Help
About Arduino
Preferences... ⌘,
Services
Hide Arduino ⌘H
Hide Others ⌥⌘H
Show All
Quit Arduino ⌘Q
Preferences
Settings
Network
Sketchbook location:
/Users/mjrovai/Documents/Arduino
Browse
Editor language:
System Default
(requires restart of Arduino)
Editor font size:
17
Interface scale:
Automatic
100
%
(requires restart of Arduino)
Theme:
Default theme
(requires restart of Arduino)
Show verbose output during:
compilation
upload
Compiler warnings:
None
Display line numbers
Enable Code Folding
Verify code after upload
Use external editor
Check for updates on startup
Save when verifying or uploading
Use accessibility features
Additional Boards Manager URLs:
https://dl.espressif.com/dl/package_esp32_index.json
More preferences can be edited directly in the file
/Users/mjrovai/Library/Arduino15/preferences.txt
(edit only when Arduino is not running)
OK
Cancel


Boards Manager
Type
All
esp32
esp32
by Espressif Systems version 1.0.6 INSTALLED
Boards included in this package:
ESP32 Dev Module, WEMOS LoLin32, WEMOS D1 MINI ESP32.
More Info
Select version
Install
Remove
Close


Arduino File Edit Sketch Tools Help
Auto Format ⌘T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... ⇧⌘I
Serial Monitor ⇧⌘M
Serial Plotter ⇧⌘L
WiFi101 / WiFiNINA Firmware Updater
ESP8266 Sketch Data Upload
Board: "DOIT ESP32 DEVKIT V1"
Upload Speed: "921600"
Flash Frequency: "80MHz"
Core Debug Level: "None"
Port: "/dev/cu.usbserial-0001"
Get Board Info
Programmer
Burn Bootloader
Boards Manager...
Arduino AVR Boards
Arduino Mbed OS Boards
Arduino Mbed OS Nano Boards
Arduino SAMD (32-bits ARM Cortex-M0+) Boards
ESP32 Arduino
Seeed SAMD (32-bits ARM Cortex-M0+ and Cortex-M4) Boards
ESP32 Dev Module
ESP32 Wrover Module
ESP32 Pico Kit
TinyPICO
S.ODI Ultra v1
MagicBit
Turta IoT Node
TTGO LoRa32-OLED V1
TTGO T1
TTGO T7 V1.3 Mini32
TTGO T7 V1.4 Mini32
XinaBox CW02
SparkFun ESP32 Thing
SparkFun ESP32 Thing Plus
u-blox NINA-W10 series (ESP32)
Widora AIR
Electronic SweetPeas - ESP320
Nano32
LOLIN D32
LOLIN D32 PRO
WEMOS LOLIN32
WEMOS LOLIN32 Lite
Dongsen Tech Pocket 32
WeMos WiFi&Bluetooth Battery
ESPea32
Noduino Quantum
Node32s
Hornbill ESP32 Dev
Hornbill ESP32 Minima
FireBeetle-ESP32
IntoRobot Fig
Onehorse ESP32 Dev Module
Adafruit ESP32 Feather
NodeMCU-32S
MH ET LIVE ESP32DevKIT
ESP32vn IoT Uno
DOIT ESP32 DEVKIT V1
DOIT ESPduino32
OLIMEX ESP32-GATEWAY
OLIMEX ESP32-PoE
OLIMEX ESP32-PoE-ISO
OLIMEX ESP32-DevKit-LiPo
ThaiEasyElec's ESPino32
M5Stack-Core-ESP32
M5Stack-FIRE
M5Stick-C
M5Stack-ATOM
M5Stack-Core2
M5Stack-Timer-CAM
M5Stack-CoreInk
ODROID ESP32
Heltec WiFi Kit 32


Arduino File Edit Sketch Tools Help
Auto Format ⌘T
Archive Sketch
Fix Encoding & Reload
Manage Libraries... ⇧⌘I
Serial Monitor ⇧⌘M
Serial Plotter ⇧⌘L
WiFi101 / WiFiNINA Firmware Updater
ESP8266 Sketch Data Upload
Board: "DOIT ESP32 DEVKIT V1"
Upload Speed: "921600"
Flash Frequency: "80MHz"
Core Debug Level: "None"
Port: "/dev/cu.usbserial-0001"
Get Board Info
Programmer
Burn Bootloader
Serial ports
/dev/cu.Bluetooth-Incoming-Port
/dev/cu.MALS
/dev/cu.SOC
/dev/cu.usbserial-0001


GetChipID | Arduino 1.8.16
GetChipID
/* The true ESP32 chip ID is essentially its MAC address.
This sketch provides an alternate chip ID that matches
the output of the ESP.getChipId() function on ESP8266
(i.e. a 32-bit integer matching the last 3 bytes of
the MAC address. This is less unique than the
MAC address chip ID, but is helpful when you need
an identifier that can be no more than a 32-bit integer
(like for switch...case).

created 2020-06-07 by cweinhofer
with help from Cicicok */

uint32_t chipId = 0;

void setup() {
Serial.begin(115200);
}

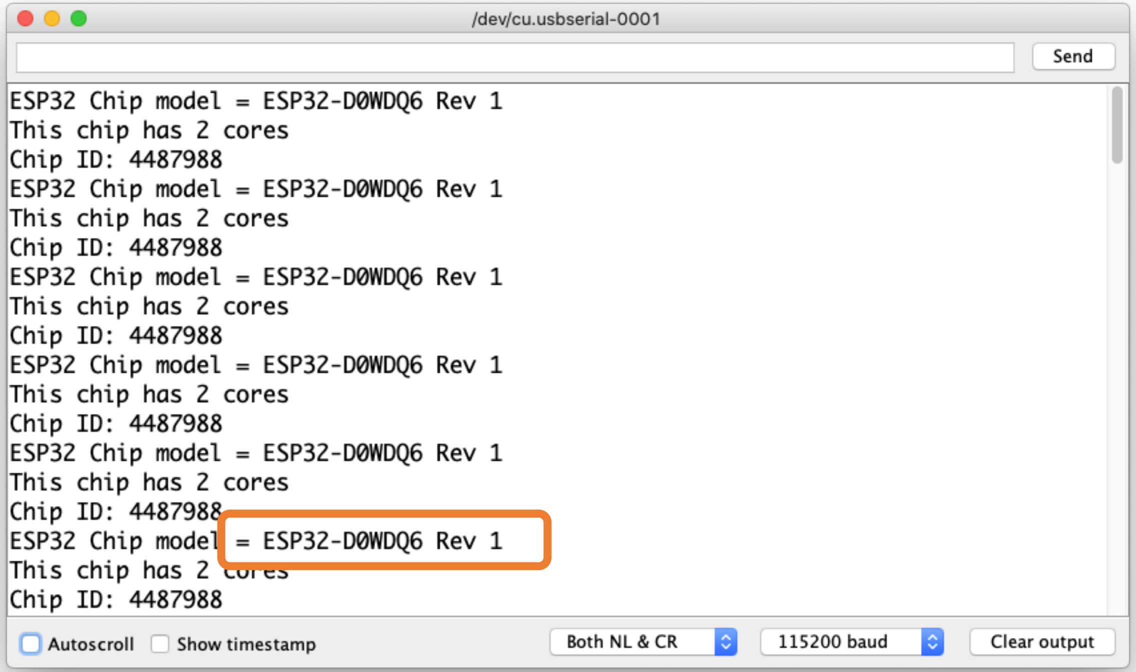
void loop() {
for(int i=0; i<17; i=i+8) {
chipId |= ((ESP.getEfuseMac() >> (40 - i)) & 0xff) << i;
}

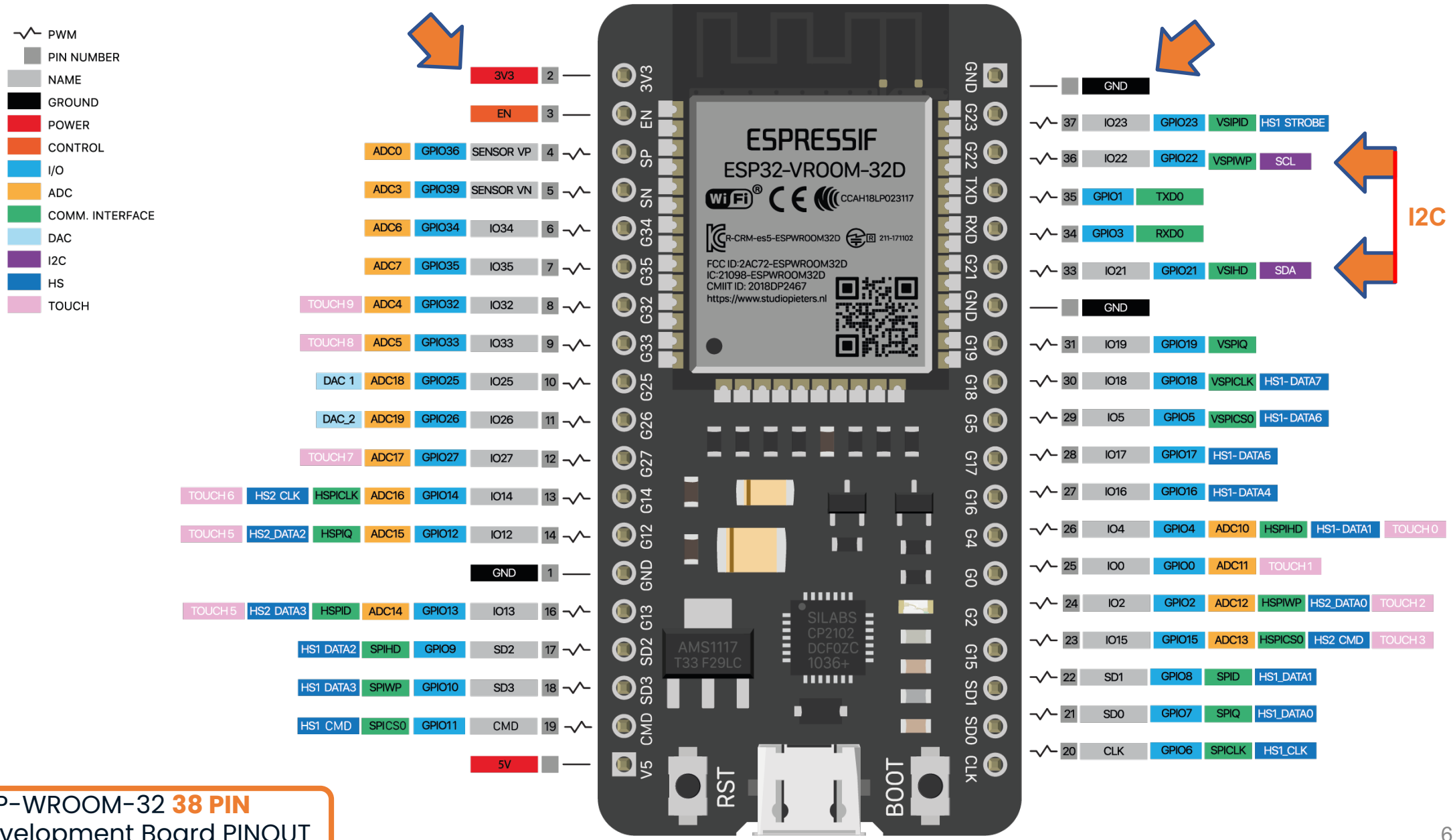
Serial.printf("ESP32 Chip model = %s Rev %d\n", ESP.getChipModel(), ESP.getChi
Serial.printf("This chip has %d cores\n", ESP.getChipCores());
Serial.print("Chip ID: "); Serial.println(chipId);

delay(3000);

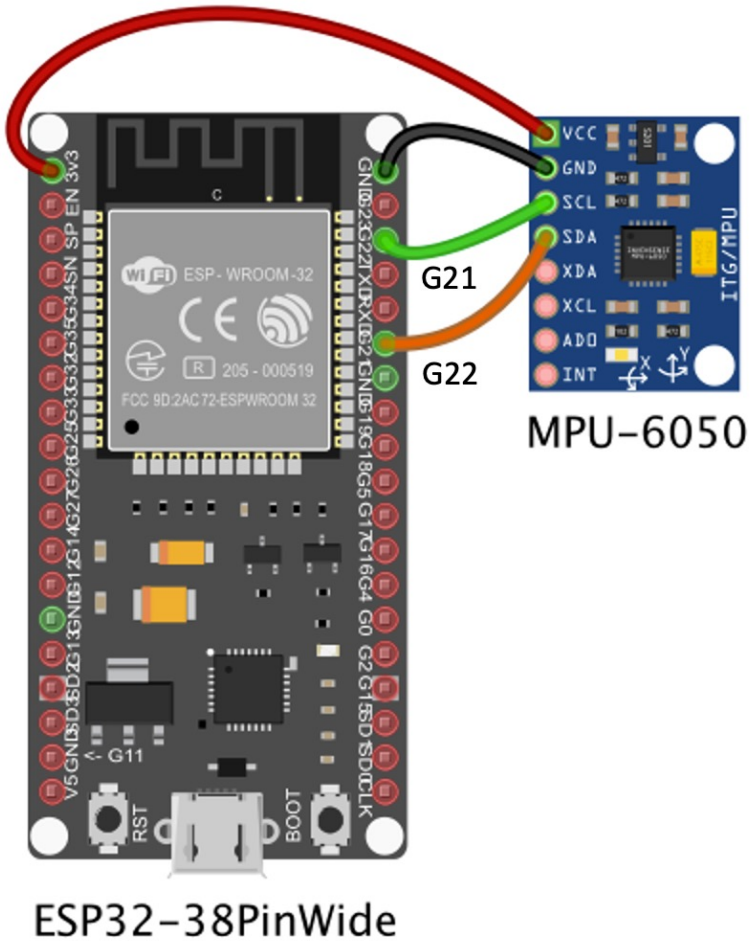
}
Done uploading.
Compressed 3072 bytes to 128...
Writing at 0x00008000... (100 %)
Wrote 3072 bytes (128 compressed) at 0x00008000 in 0.0 seconds (effective 2623.9 kbi
Hash of data verified.

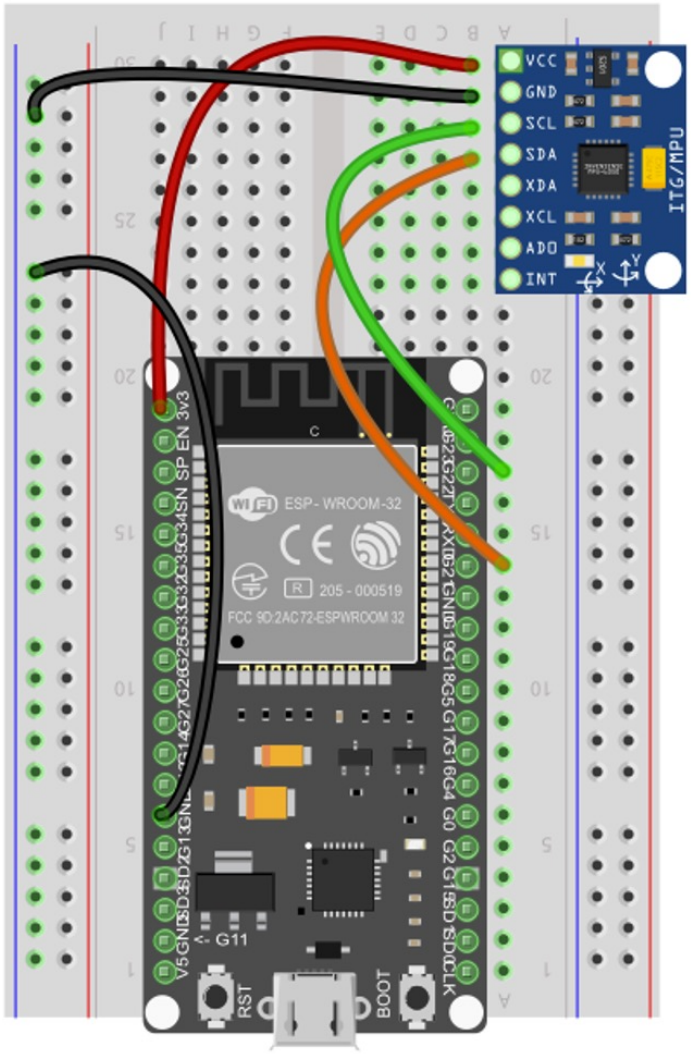
Leaving...
Hard resetting via RTS pin...
25
DOIT ESP32 DEVKIT V1, 80MHz, 921600, None on /dev/cu.usbserial-0001

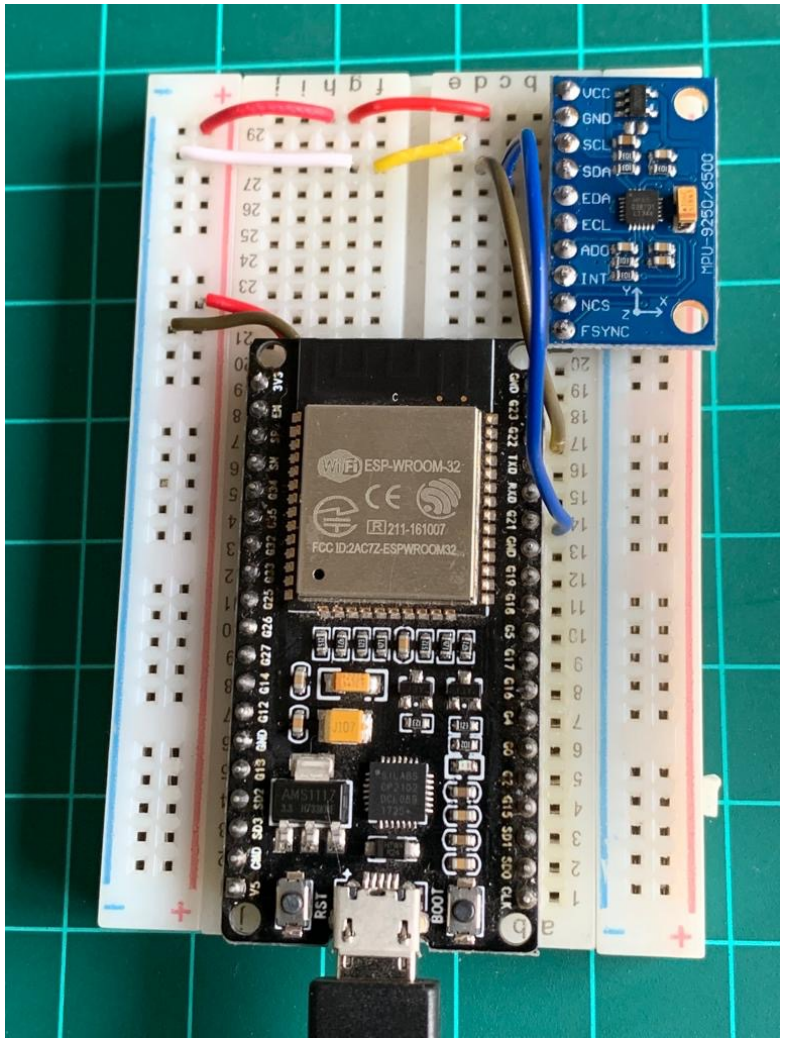

/dev/cu.usbserial-0001
Send
ESP32 Chip model = ESP32-D0WDQ6 Rev 1
This chip has 2 cores
Chip ID: 4487988
ESP32 Chip model = ESP32-D0WDQ6 Rev 1
This chip has 2 cores
Chip ID: 4487988
ESP32 Chip model = ESP32-D0WDQ6 Rev 1
This chip has 2 cores
Chip ID: 4487988
ESP32 Chip model = ESP32-D0WDQ6 Rev 1
This chip has 2 cores
Chip ID: 4487988
ESP32 Chip model = ESP32-D0WDQ6 Rev 1
This chip has 2 cores
Chip ID: 4487988
ESP32 Chip model = ESP32-D0WDQ6 Rev 1
This chip has 2 cores
Chip ID: 4487988
Autoscroll
Show timestamp
Both NL & CR
115200 baud
Clear output



ESP-WROOM-32 **38 PIN**
Development Board PINOUT


VCC
GND
SCL
SDA
XDA
XCL
ADO
INT
ITG/MPU
G21
G22
MPU-6050
ESP-WROOM-32
205 - 000519
FCC 9D:2AC72-ESPWROOM 32
<- G11
RST
BOOT
ESP32-38PinWide


VCC
GND
SCL
SDA
XDA
XCL
ADO
INT
ITG/MPU
ESP-WROOM-32
205 - 000519
FCC 9D:2AC72-ESPWROOM 32
<- G11
RST
BOOT


VCC
GND
SCL
SDA
EDA
ECL
ADO
INT
NCS
FSYNC
MPU-9250/9500
ESP-WROOM-32
211-161007
FCC ID:2AC7Z-ESPWROOM32
RST
BOOT

MPU6050_Acc_Data_Acquisition | Arduino 1.8.16

MPU6050_Acc_Data_Acquisition

```
/*
 * Based on I2C device class (I2Cdev) Arduino sketch for MPU6050 class by Jeff Rowberg <jeff@rowberg.net>
 * and Edge Impulse Data Forwarder Exampe (Arduino) - https://docs.edgeimpulse.com/docs/cli-data-forwarder
 *
 * Developed by M.Rovai for IESTI01 TinyML Course
 */

#include "I2Cdev.h"
#include "MPU6050.h"
#include "Wire.h"

#define FREQUENCY_HZ  50
#define INTERVAL_MS   (1000 / (FREQUENCY_HZ + 1))
static unsigned long last_interval_ms = 0;

#define OUTPUT_READABLE_ACCELGYRO
MPU6050 accelgyro;
int16_t ax, ay, az;

void setup() {

    Wire.begin();
    Serial.begin(115200);

    // initialize device
    Serial.println("Initializing I2C devices...");
    accelgyro.initialize();

    // verify connection
    Serial.println("Testing device connections...");
    Serial.println(accelgyro.testConnection() ? "MPU6050 connection successful" : "MPU6050 connection failed");
}

void loop() {

      if (millis() > last_interval_ms + INTERVAL_MS) {
        last_interval_ms = millis();

        // read raw accel/gyro measurements from device
        accelgyro.getAcceleration(&ax, &ay, &az);

        Serial.print(ax); Serial.print("\t");
        Serial.print(ay); Serial.print("\t");
        Serial.println(az);
      }
}
```

Done Saving.

```
Leaving...
Hard resetting via RTS pin...
```

18 DOIT ESP32 DEVKIT V1, 80MHz, 921600, None on /dev/cu.usbserial-0001

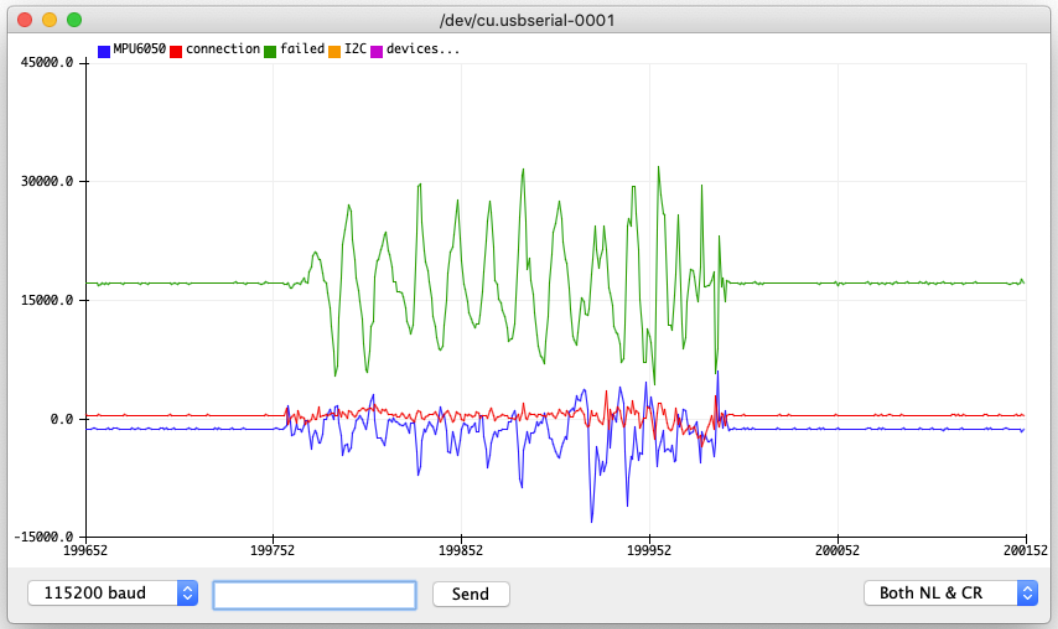
/dev/cu.usbserial-0001

Send

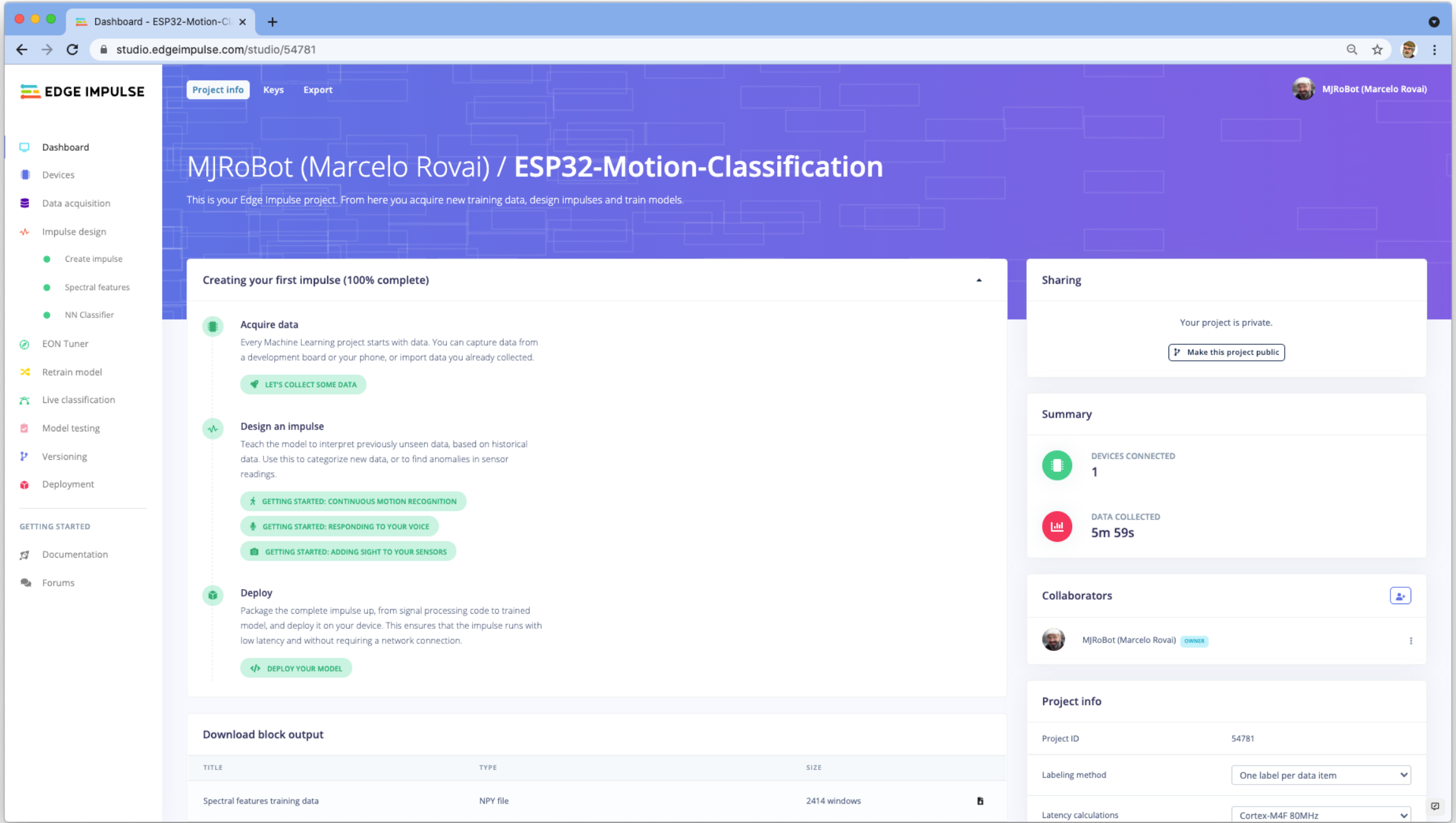
```
-1240   444     17136
-1276   428     17180
-1296   412     17212
-1296   432     17164
-1272   472     17180
-1284   432     17188
-1312   436     17260
-1280   504     17160
-1288   484     17224
-1192   468     17276
-1316   428     17224
-1252   460     17108
-1248   440     17200
-1388   420     17308
-1156   448     17184
-1328   400     17236
```

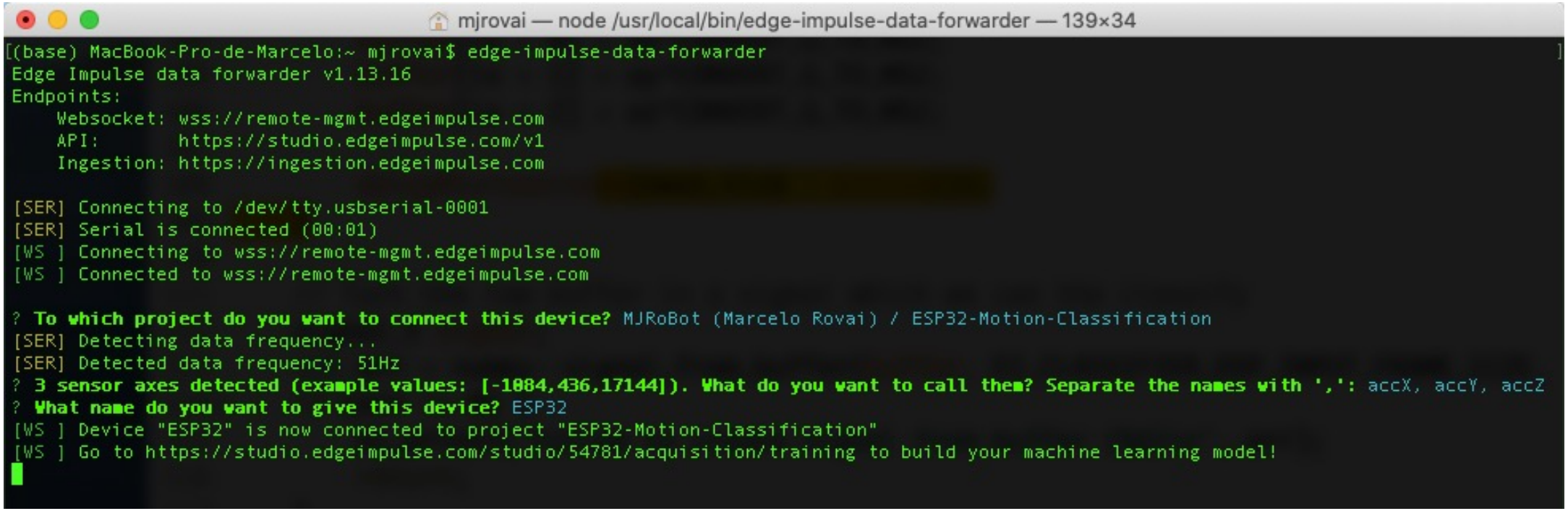
Autoscroll Show timestamp Both NL & CR 115200 baud Clear output

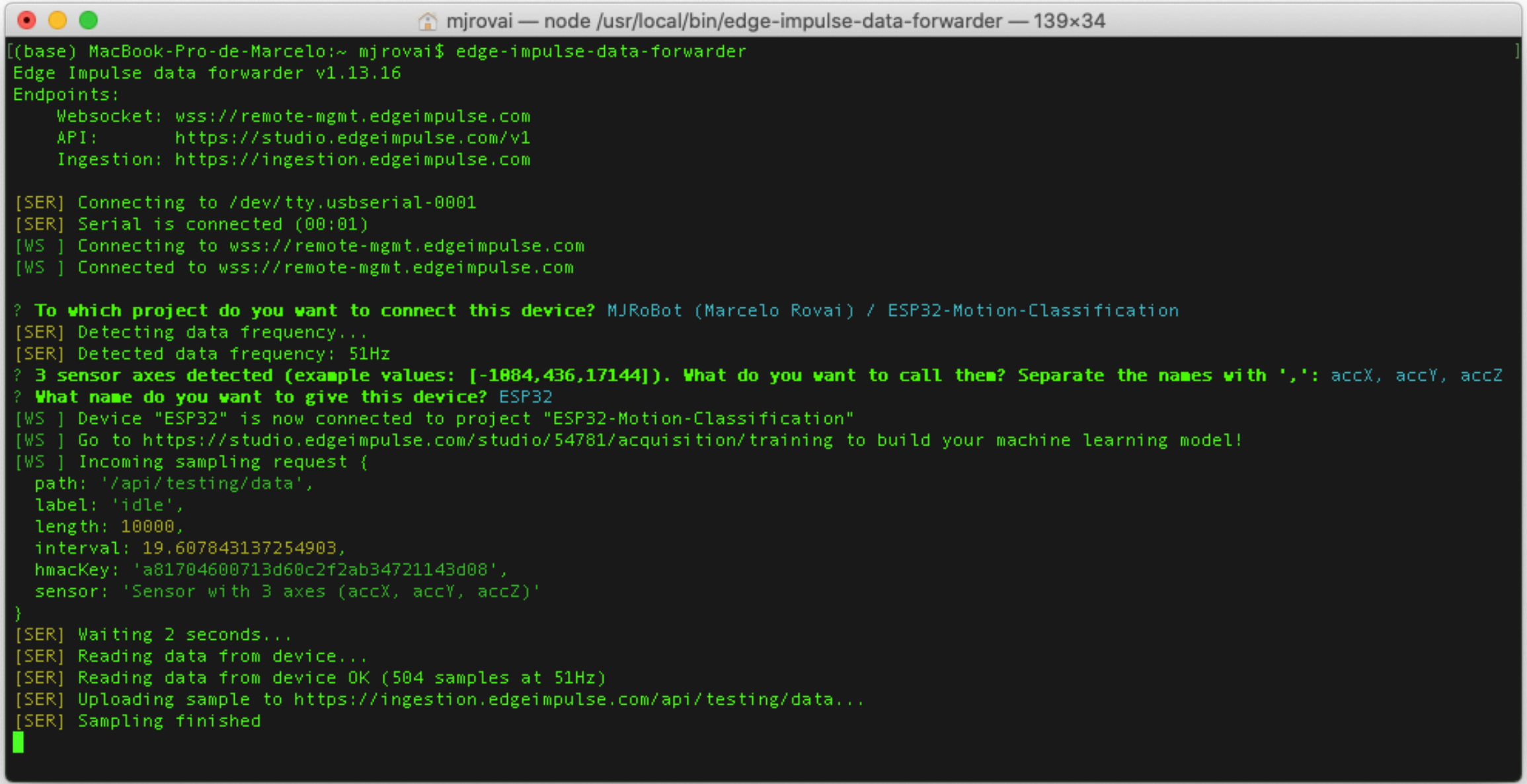
/dev/cu.usbserial-0001

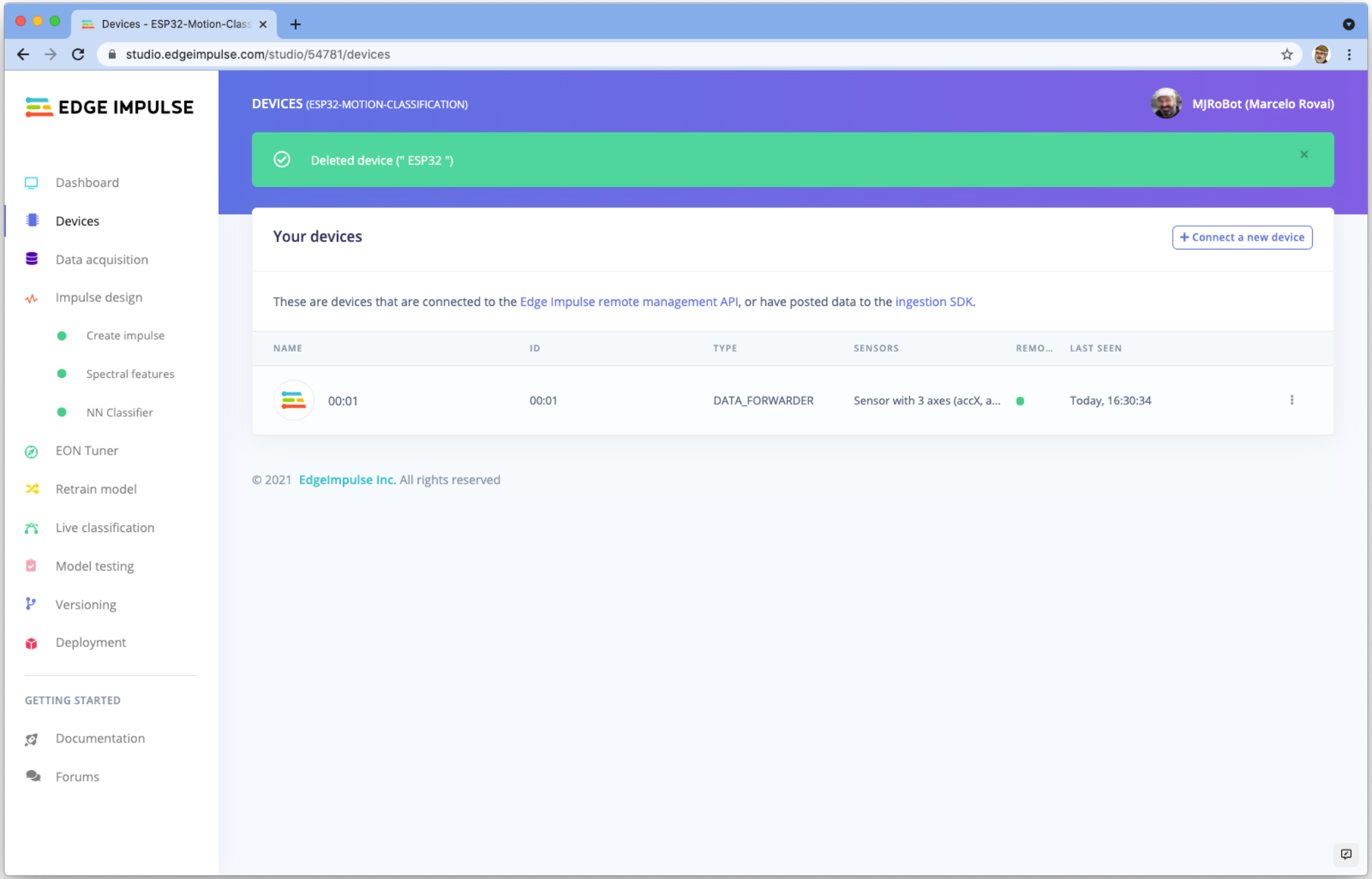


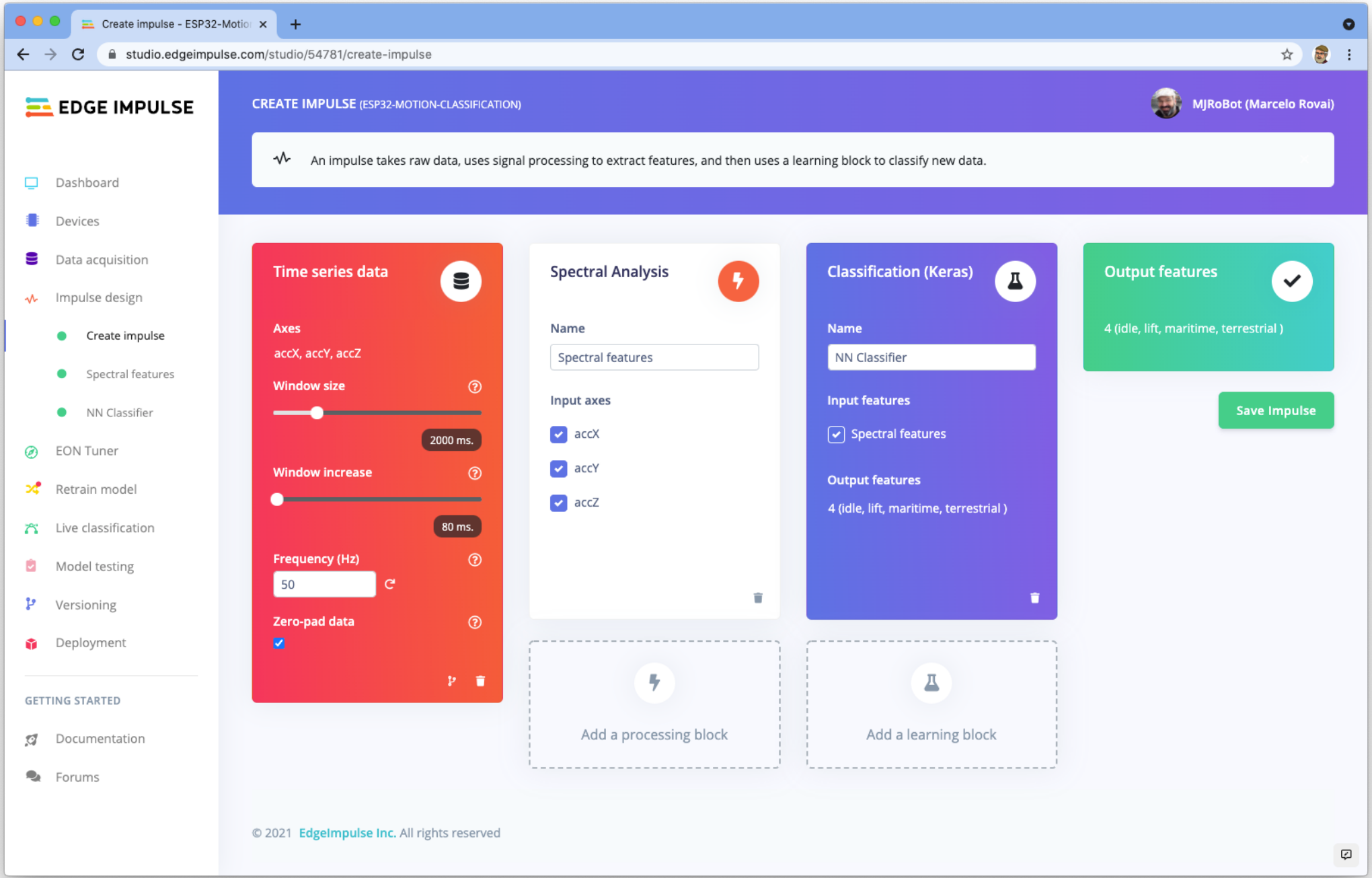
115200 baud Send Both NL & CR


Dashboard - ESP32-Motion-Cl
studio.edgeimpulse.com/studio/54781
EDGE IMPULSE
Project info
Keys
Export
MJRoBot (Marcelo Rovai)
Dashboard
Devices
Data acquisition
Impulse design
Create impulse
Spectral features
NN Classifier
EON Tuner
Retrain model
Live classification
Model testing
Versioning
Deployment
GETTING STARTED
Documentation
Forums
MJRoBot (Marcelo Rovai) / ESP32-Motion-Classification
This is your Edge Impulse project. From here you acquire new training data, design impulses and train models.
Creating your first impulse (100% complete)
Acquire data
Every Machine Learning project starts with data. You can capture data from a development board or your phone, or import data you already collected.
LET'S COLLECT SOME DATA
Design an impulse
Teach the model to interpret previously unseen data, based on historical data. Use this to categorize new data, or to find anomalies in sensor readings.
GETTING STARTED: CONTINUOUS MOTION RECOGNITION
GETTING STARTED: RESPONDING TO YOUR VOICE
GETTING STARTED: ADDING SIGHT TO YOUR SENSORS
Deploy
Package the complete impulse up, from signal processing code to trained model, and deploy it on your device. This ensures that the impulse runs with low latency and without requiring a network connection.
DEPLOY YOUR MODEL
Download block output
TITLE
TYPE
SIZE
Spectral features training data
NPY file
2414 windows
Sharing
Your project is private.
Make this project public
Summary
DEVICES CONNECTED
1
DATA COLLECTED
5m 59s
Collaborators
MJRoBot (Marcelo Rovai)
OWNER
Project info
Project ID
54781
Labeling method
One label per data item
Latency calculations
Cortex-M4F 80MHz


mjrovai — node /usr/local/bin/edge-impulse-data-forwarder — 139×34
[(base) MacBook-Pro-de-Marcelo:~ mjrovai$ edge-impulse-data-forwarder
Edge Impulse data forwarder v1.13.16
Endpoints:
Websocket: wss://remote-mgmt.edgeimpulse.com
API: https://studio.edgeimpulse.com/v1
Ingestion: https://ingestion.edgeimpulse.com
[SER] Connecting to /dev/tty.usbserial-0001
[SER] Serial is connected (00:01)
[WS] Connecting to wss://remote-mgmt.edgeimpulse.com
[WS] Connected to wss://remote-mgmt.edgeimpulse.com
? To which project do you want to connect this device? MJRoBot (Marcelo Rovai) / ESP32-Motion-Classification
[SER] Detecting data frequency...
[SER] Detected data frequency: 51Hz
? 3 sensor axes detected (example values: [-1084,436,17144]). What do you want to call them? Separate the names with ',': accX, accY, accZ
? What name do you want to give this device? ESP32
[WS] Device "ESP32" is now connected to project "ESP32-Motion-Classification"
[WS] Go to https://studio.edgeimpulse.com/studio/54781/acquisition/training to build your machine learning model!


mjrovai — node /usr/local/bin/edge-impulse-data-forwarder — 139×34
[(base) MacBook-Pro-de-Marcelo:~ mjrovai$ edge-impulse-data-forwarder
Edge Impulse data forwarder v1.13.16
Endpoints:
Websocket: wss://remote-mgmt.edgeimpulse.com
API: https://studio.edgeimpulse.com/v1
Ingestion: https://ingestion.edgeimpulse.com
[SER] Connecting to /dev/tty.usbserial-0001
[SER] Serial is connected (00:01)
[WS] Connecting to wss://remote-mgmt.edgeimpulse.com
[WS] Connected to wss://remote-mgmt.edgeimpulse.com
? To which project do you want to connect this device? MJRoBot (Marcelo Rovai) / ESP32-Motion-Classification
[SER] Detecting data frequency...
[SER] Detected data frequency: 51Hz
? 3 sensor axes detected (example values: [-1884,436,17144]). What do you want to call them? Separate the names with ',': accX, accY, accZ
? What name do you want to give this device? ESP32
[WS] Device "ESP32" is now connected to project "ESP32-Motion-Classification"
[WS] Go to https://studio.edgeimpulse.com/studio/54781/acquisition/training to build your machine learning model!
[WS] Incoming sampling request {
path: '/api/testing/data',
label: 'idle',
length: 10000,
interval: 19.607843137254903,
hmacKey: 'a81704600713d60c2f2ab34721143d08',
sensor: 'Sensor with 3 axes (accX, accY, accZ)'
}
[SER] Waiting 2 seconds...
[SER] Reading data from device...
[SER] Reading data from device OK (504 samples at 51Hz)
[SER] Uploading sample to https://ingestion.edgeimpulse.com/api/testing/data...
[SER] Sampling finished


Devices - ESP32-Motion-Class
studio.edgeimpulse.com/studio/54781/devices
EDGE IMPULSE
Dashboard
Devices
Data acquisition
Impulse design
Create impulse
Spectral features
NN Classifier
EON Tuner
Retrain model
Live classification
Model testing
Versioning
Deployment
GETTING STARTED
Documentation
Forums
DEVICES (ESP32-MOTION-CLASSIFICATION)
MJRoBot (Marcelo Rovai)
Deleted device (" ESP32 ")
Your devices
+ Connect a new device
These are devices that are connected to the Edge Impulse remote management API, or have posted data to the ingestion SDK.
NAME
ID
TYPE
SENSORS
REMO...
LAST SEEN
00:01
00:01
DATA_FORWARDER
Sensor with 3 axes (accX, a...
Today, 16:30:34
© 2021 EdgeImpulse Inc. All rights reserved


Create impulse - ESP32-Motio
studio.edgeimpulse.com/studio/54781/create-impulse
EDGE IMPULSE
Dashboard
Devices
Data acquisition
Impulse design
Create impulse
Spectral features
NN Classifier
EON Tuner
Retrain model
Live classification
Model testing
Versioning
Deployment
GETTING STARTED
Documentation
Forums
CREATE IMPULSE (ESP32-MOTION-CLASSIFICATION)
MJRoBot (Marcelo Rovai)
An impulse takes raw data, uses signal processing to extract features, and then uses a learning block to classify new data.
Time series data
Axes
accX, accY, accZ
Window size
2000 ms.
Window increase
80 ms.
Frequency (Hz)
50
Zero-pad data
Spectral Analysis
Name
Spectral features
Input axes
accX
accY
accZ
Classification (Keras)
Name
NN Classifier
Input features
Spectral features
Output features
4 (idle, lift, maritime, terrestrial)
Output features
4 (idle, lift, maritime, terrestrial)
Save Impulse
Add a processing block
Add a learning block
© 2021 EdgeImpulse Inc. All rights reserved

Spectral features - ESP32-Mot

studio.edgeimpulse.com/studio/54781/dsp/spectral-analysis/3/generate-features

EDGE IMPULSE

- Dashboard
- Devices
- Data acquisition
- Impulse design
  - Create impulse
  - Spectral features
  - NN Classifier
- EON Tuner
- Retrain model
- Live classification
- Model testing
- Versioning
- Deployment

GETTING STARTED

- Documentation
- Forums

SPECTRAL FEATURES (ESP32-MOTION-CLASSIFICATION)

MJRoBot (Marcelo Rovai)

#1 Click to set a description for this version

Parameters | Generate features

Training set

Data in training set	3m 59s
Classes	4 (idle, lift, maritime, terrestrial)
Window length	2000 ms.
Window increase	80 ms.
Training windows	2,414

Generate features

Feature generation output

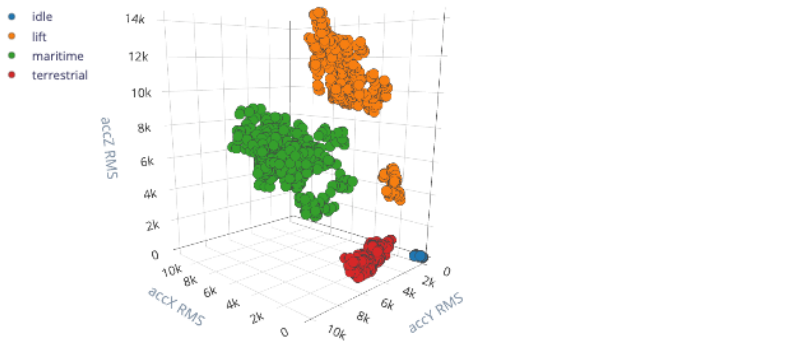
```
Scheduling job in cluster...
Job started
Creating windows from 24 files...
[ 0/24] Creating windows from files...
[ 1/24] Creating windows from files...
[24/24] Creating windows from files...
Created 2414 windows: idle: 596, lift: 606, maritime: 606, terrestrial : 606

Creating features
[   1/2414] Creating features...
[1046/2414] Creating features...
[2076/2414] Creating features...
[2414/2414] Creating features...
Created features

Job completed
```

Feature explorer (2,414 samples)

X Axis: accX RMS | Y Axis: accY RMS | Z Axis: accZ RMS



NN CLASSIFIER (ESP32-MOTION-CLASSIFICATION)

#1 Click to set a description for this version

Neural Network settings

Training settings

Number of training cycles: 30

Learning rate: 0.0005

Neural network architecture

Input layer (33 features)

Dense layer (20 neurons)

Dense layer (10 neurons)

Add an extra layer

Output layer (4 features)

Start training

Model

Model version: Quantized (int8)

Last training performance (validation set)

ACCURACY
96.9%

LOSS
0.95

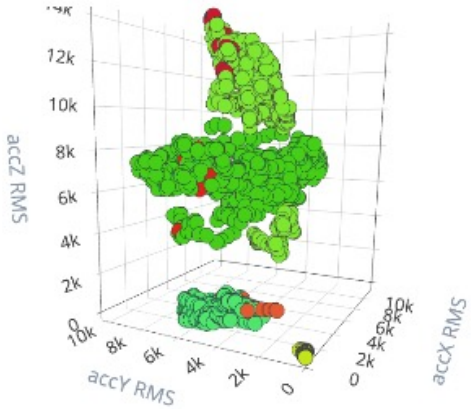
Confusion matrix (validation set)

	IDLE	LIFT	MARITIME	TERRESTRIAL
IDLE	100%	0%	0%	0%
LIFT	3.3%	96.7%	0%	0%
MARITIME	1.7%	2.6%	95.7%	0%
TERRESTRIAL	0.8%	4.1%	0%	95.0%
F1 SCORE	0.97	0.95	0.98	0.97

Feature explorer (full training set)

accX RMS | accY RMS | accZ RMS

- idle - correct
- lift - correct
- maritime - correct
- terrestrial - correct
- lift - incorrect
- maritime - incorrect
- terrestrial - incorrect



On-device performance

INFERENCING TIME
1 ms.

PEAK RAM USAGE
1.7K

FLASH USAGE
19.0K

Model testing - ESP32-Motion

studio.edgeimpulse.com/studio/54781/validation

EDGE IMPULSE

- Dashboard
- Devices
- Data acquisition
- Impulse design
  - Create impulse
  - Spectral features
  - NN Classifier
- EON Tuner
- Retrain model
- Live classification
- Model testing
- Versioning
- Deployment

GETTING STARTED

- Documentation
- Forums

Test data

Classify all

Set the 'expected outcome' for each sample to the desired outcome to automatically score the impulse.

SAMPLE NAME	EXPECTED OUTCOME	LENGTH	ACCURACY	RESULT
idle.json.2jb4ipqd	idle	10s	89%	90 idle, 6 lift, 4 uncertain, 1 maritime
testing.json.2jb2djqg	maritime	10s	98%	99 maritime, 2 idle
testing.json.2jb2cdi2	terrestrial	10s		101 terrestrial
testing.json.2jb29lqg	lift	10s	70%	71 lift, 30 maritime
idle.json.2jb1mdjv	idle	10s	18%	76 maritime, 19 idle, 4 lift, 2 uncert...
idle.json.2jb1j3th	idle	10s	100%	101 idle
lift.json.2jb1hmvf	lift	10s	85%	86 lift, 15 maritime
terrestrial.json.2jb1gu...	terrestrial	10s		101 terrestrial
maritime.json.2jb1e80t	maritime	10s	100%	101 maritime
maritime.json.2jb1c79k	maritime	10s	100%	101 maritime
terrestrial .json.2jb18...	terrestrial	10s		101 terrestrial
lift.json.2jb130ml	lift	10s	94%	95 lift, 6 maritime

Model testing output

```
Classifying data for NN Classifier...
Copying features from processing blocks...
Copying features from DSP block...
Copying features from DSP block OK
Copying features from processing blocks OK

Classifying data for float32 model...
Scheduling job in cluster...
Job started
Classifying data for NN Classifier OK


Job completed
```

Model testing results

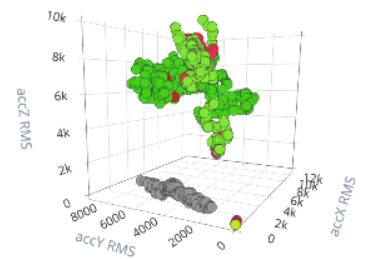
ACCURACY
83.94%

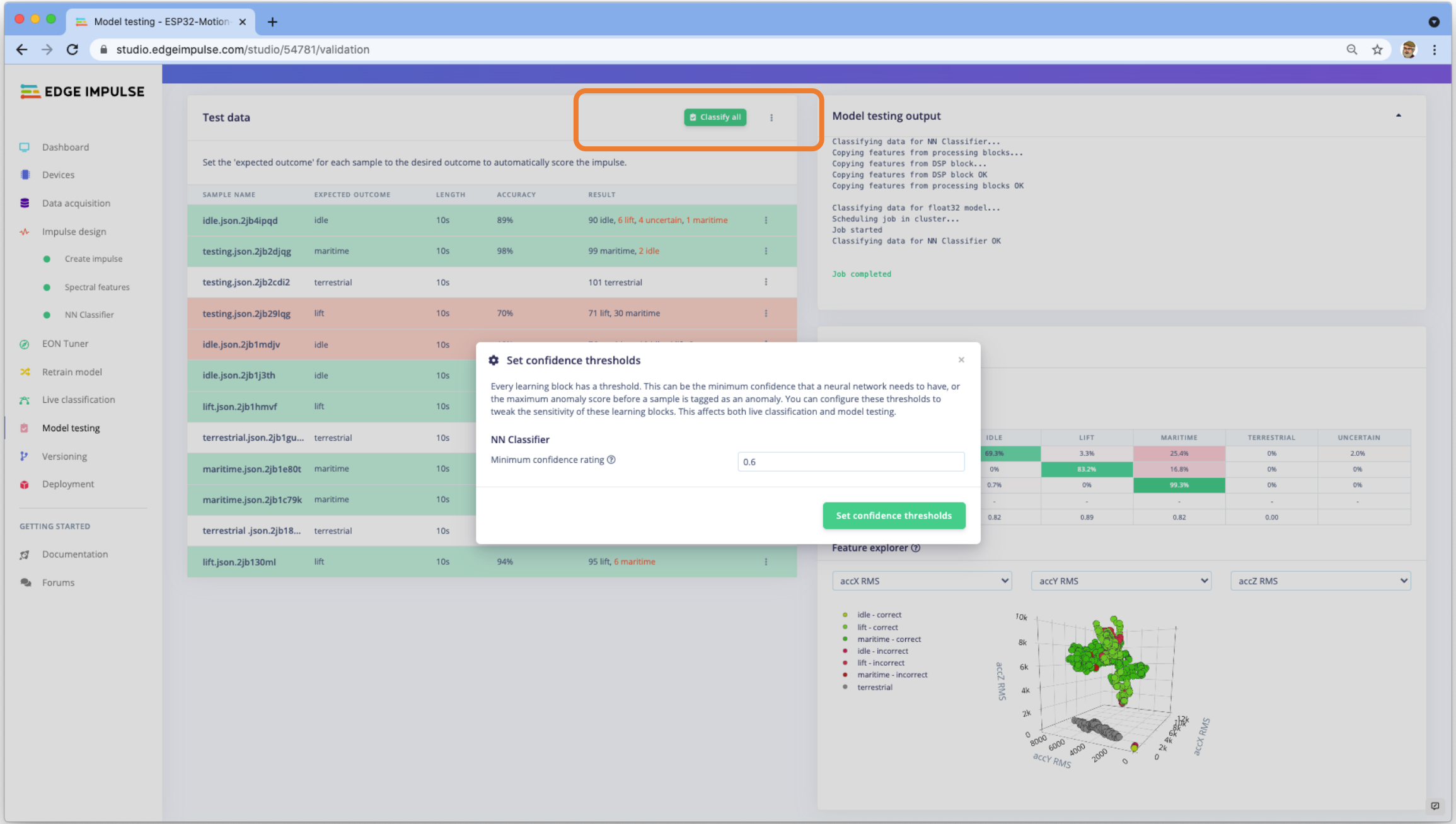
	IDLE	LIFT	MARITIME	TERRESTRIAL	UNCERTAIN
IDLE	69.3%	3.3%	25.4%	0%	2.0%
LIFT	0%	83.2%	16.8%	0%	0%
MARITIME	0.7%	0%	99.3%	0%	0%
TERRESTRIAL	-	-	-	-	-
F1 SCORE	0.82	0.89	0.82	0.00	

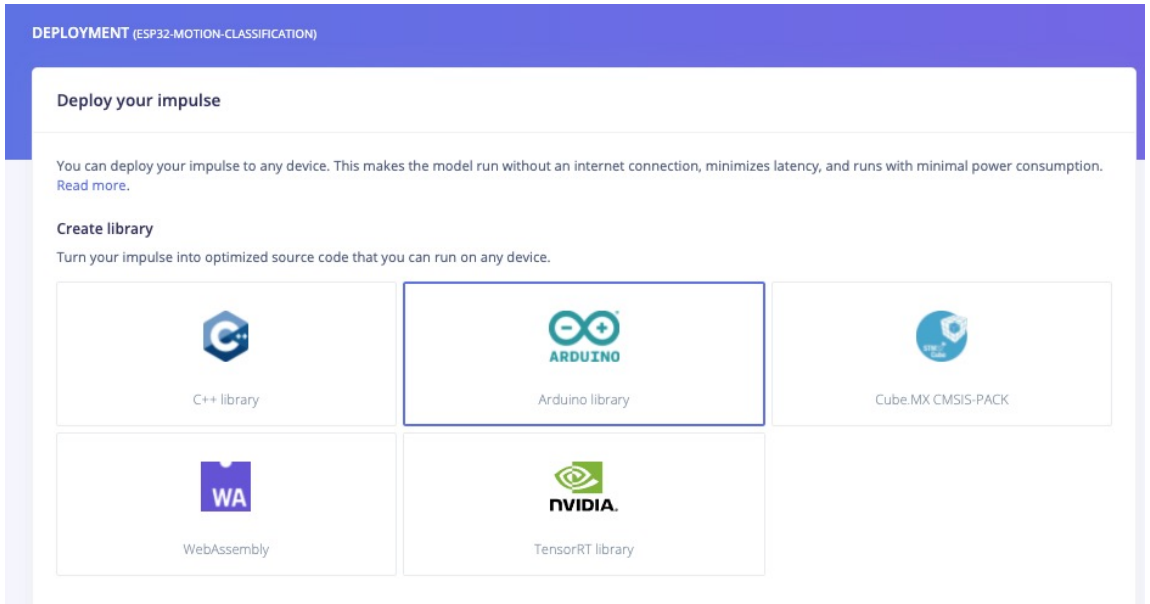
Feature explorer

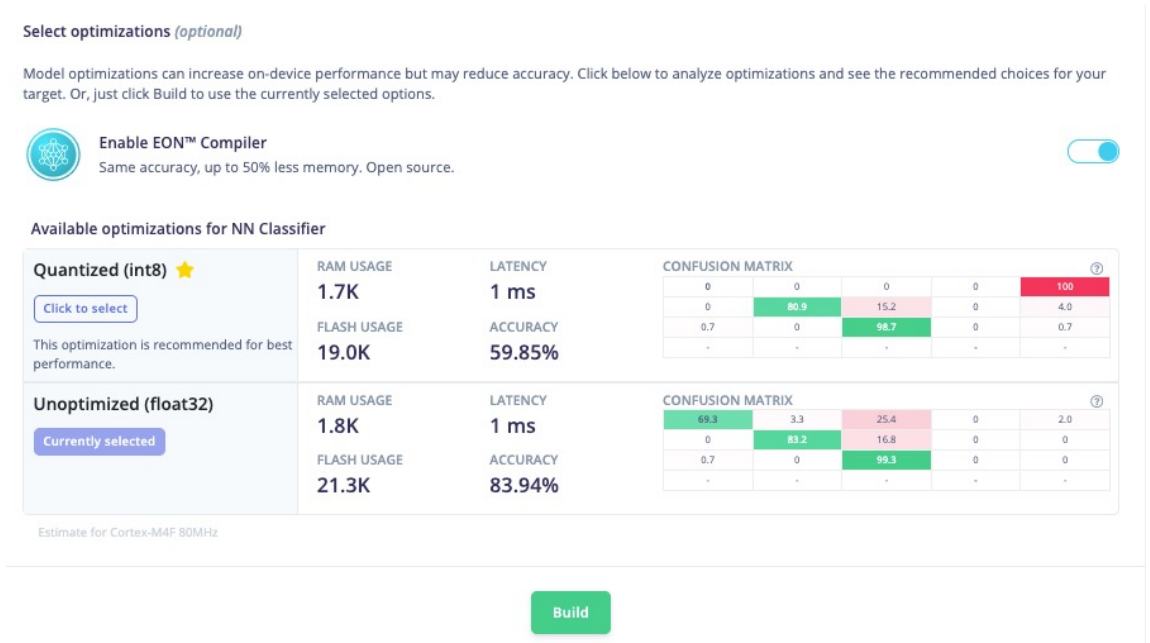
accX RMS | accY RMS | accZ RMS

- idle - correct
- lift - correct
- maritime - correct
- idle - incorrect
- lift - incorrect
- maritime - incorrect
- terrestrial




Model testing - ESP32-Motion
studio.edgeimpulse.com/studio/54781/validation
EDGE IMPULSE
Dashboard
Devices
Data acquisition
Impulse design
Create impulse
Spectral features
NN Classifier
EON Tuner
Retrain model
Live classification
Model testing
Versioning
Deployment
GETTING STARTED
Documentation
Forums
Test data
Classify all
Set the 'expected outcome' for each sample to the desired outcome to automatically score the impulse.
SAMPLE NAME
EXPECTED OUTCOME
LENGTH
ACCURACY
RESULT
idle.json.2jb4ipqd idle 10s 89% 90 idle, 6 lift, 4 uncertain, 1 maritime
testing.json.2jb2djqg maritime 10s 98% 99 maritime, 2 idle
testing.json.2jb2cdi2 terrestrial 10s 101 terrestrial
testing.json.2jb29lqg lift 10s 70% 71 lift, 30 maritime
idle.json.2jb1mdjv idle 10s
idle.json.2jb1j3th idle 10s
lift.json.2jb1hmvf lift 10s
terrestrial.json.2jb1gu... terrestrial 10s
maritime.json.2jb1e80t maritime 10s
maritime.json.2jb1c79k maritime 10s
terrestrial .json.2jb18... terrestrial 10s
lift.json.2jb130ml lift 10s 94% 95 lift, 6 maritime
Set confidence thresholds
Every learning block has a threshold. This can be the minimum confidence that a neural network needs to have, or the maximum anomaly score before a sample is tagged as an anomaly. You can configure these thresholds to tweak the sensitivity of these learning blocks. This affects both live classification and model testing.
NN Classifier
Minimum confidence rating
0.6
Set confidence thresholds
Model testing output
Classifying data for NN Classifier...
Copying features from processing blocks...
Copying features from DSP block...
Copying features from DSP block OK
Copying features from processing blocks OK
Classifying data for float32 model...
Scheduling job in cluster...
Job started
Classifying data for NN Classifier OK
Job completed
IDLE
LIFT
MARITIME
TERRESTRIAL
UNCERTAIN
69.3% 3.3% 25.4% 0% 2.0%
0% 83.2% 16.8% 0% 0%
0.7% 0% 99.3% 0% 0%
0.82 0.89 0.82 0.00
Feature explorer
accX RMS
accY RMS
accZ RMS
idle - correct
lift - correct
maritime - correct
idle - incorrect
lift - incorrect
maritime - incorrect
terrestrial
accZ RMS
accY RMS
accX RMS


DEPLOYMENT (ESP32-MOTION-CLASSIFICATION)
Deploy your impulse
You can deploy your impulse to any device. This makes the model run without an internet connection, minimizes latency, and runs with minimal power consumption.
Read more.
Create library
Turn your impulse into optimized source code that you can run on any device.
C++ library
ARDUINO
Arduino library
Cube.MX CMSIS-PACK
WA
WebAssembly
NVIDIA.
TensorRT library


Select optimizations (optional)
Model optimizations can increase on-device performance but may reduce accuracy. Click below to analyze optimizations and see the recommended choices for your target. Or, just click Build to use the currently selected options.
Enable EON™ Compiler
Same accuracy, up to 50% less memory. Open source.
Available optimizations for NN Classifier
Quantized (int8)
Click to select
This optimization is recommended for best performance.
RAM USAGE
1.7K
LATENCY
1 ms
FLASH USAGE
19.0K
ACCURACY
59.85%
CONFUSION MATRIX
Unoptimized (float32)
Currently selected
RAM USAGE
1.8K
LATENCY
1 ms
FLASH USAGE
21.3K
ACCURACY
83.94%
CONFUSION MATRIX
Estimate for Cortex-M4F 80MHz
Build

esp32_sense_accelerometer | Arduino 1.8.16

esp32_sense_accelerometer

```
/* Includes ---------------------------------------------------------------- */
#include <ESP32-Motion-Classification_inferencing.h>
#include "I2Cdev.h"
#include "MPU6050.h"
#include "Wire.h"

/* Constant defines -------------------------------------------------------- */
#define CONVERT_G_TO_MS2    1.0f
#define OUTPUT_READABLE_ACCELGYRO
MPU6050 accelgyro;
int16_t ax, ay, az;

/* Private variables ------------------------------------------------------- */
static bool debug_nn = false; // Set this to true to see e.g. features generated from the raw signal

/**
* @brief      Arduino setup function
*/
void setup()
{
    // put your setup code here, to run once:
    Serial.begin(115200);
    Serial.println("Edge Impulse Inferencing - ESP32-Motion Classification");

    Wire.begin();

    // initialize device
    Serial.println("Initializing I2C devices...");
    accelgyro.initialize();

    // verify connection
    Serial.println("Testing device connections...");
    Serial.println(accelgyro.testConnection() ? "MPU6050 connection successful" : "MPU6050 connection failed");

    if (EI_CLASSIFIER_RAW_SAMPLES_PER_FRAME != 3) {
        ei_printf("ERR: EI_CLASSIFIER_RAW_SAMPLES_PER_FRAME should be equal to 3 (the 3 sensor axes)\n");
        return;
    }
}
```

```
Writing at 0x00008000... (100 %)
Wrote 3072 bytes (128 compressed) at 0x00008000 in 0.0 seconds (effective 2165.5 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
```

22 DOIT ESP32 DEVKIT V1, 80MHz, 921600, None on /dev/cu.usbserial-0001

esp32_sense_accelerometer | Arduino 1.8.16

esp32_sense_accelerometer §

```
void loop() {
    ei_printf("\nStarting inferencing in 2 seconds...\n");
    delay(2000);
    ei_printf("Sampling...\n");

    // Allocate a buffer here for the values we'll read from the IMU
    float buffer[EI_CLASSIFIER_DSP_INPUT_FRAME_SIZE] = { 0 };

    for (size_t ix = 0; ix < EI_CLASSIFIER_DSP_INPUT_FRAME_SIZE; ix += 3) {
        // Determine the next tick (and then sleep later)
        uint64_t next_tick = micros() + (EI_CLASSIFIER_INTERVAL_MS * 1000);

        accelgyro.getAcceleration(&ax, &ay, &az);
        buffer[ix + 0] = ax*CONVERT_G_TO_MS2;
        buffer[ix + 1] = ay*CONVERT_G_TO_MS2;
        buffer[ix + 2] = az*CONVERT_G_TO_MS2;

        delayMicroseconds(next_tick - micros());
    }
    // Turn the raw buffer in a signal which we can the classify
    signal_t signal;
    int err = numpy::signal_from_buffer(buffer, EI_CLASSIFIER_DSP_INPUT_FRAME_SIZE, &signal);
    if (err != 0) {
        ei_printf("Failed to create signal from buffer (%d)\n", err);
        return;
    }

    // Run the classifier
    ei_impulse_result_t result = { 0 };

    err = run_classifier(&signal, &result, debug_nn);
    if (err != EI_IMPULSE_OK) {
        ei_printf("ERR: Failed to run classifier (%d)\n", err);
        return;
    }

    // print the predictions
    ei_printf("Predictions ");
    ei_printf("(DSP: %d ms., Classification: %d ms., Anomaly: %d ms.)",
        result.timing.dsp, result.timing.classification, result.timing.anomaly);
    ei_printf(": \n");
    for (size_t ix = 0; ix < EI_CLASSIFIER_LABEL_COUNT; ix++) {
        ei_printf("    %s: %.5f\n", result.classification[ix].label, result.classification[ix].value);
    }
#if EI_CLASSIFIER_HAS_ANOMALY == 1
    ei_printf("    anomaly score: %.3f\n", result.anomaly);
#endif
}

#if !defined(EI_CLASSIFIER_SENSOR) || EI_CLASSIFIER_SENSOR != EI_CLASSIFIER_SENSOR_ACCELEROMETER
#error "Invalid model for current sensor"
#endif
```

104 DOIT ESP32 DEVKIT V1, 80MHz, 921600, None on /dev/cu.usbserial-0001

/dev/cu.usbserial-0001

Send

```
    lift: 0.02348
    maritime: 0.14678
    terrestrial : 0.00562

Starting inferencing in 2 seconds...
Sampling...
Predictions (DSP: 19 ms., Classification: 0 ms., Anomaly: 0 ms.):
    idle: 0.88469
    lift: 0.05896
    maritime: 0.03198
    terrestrial : 0.02436

Starting inferencing in 2 seconds...
Sampling...
Predictions (DSP: 19 ms., Classification: 0 ms., Anomaly: 0 ms.):
    idle: 0.86179
    lift: 0.03734
    maritime: 0.08310
    terrestrial : 0.01776

Starting inferencing in 2 seconds...
```

Autoscroll Show timestamp Both NL & CR 115200 baud Clear output

/dev/cu.usbserial-0001

Send

```
    maritime: 0.18839
    terrestrial : 0.00147

Starting inferencing in 2 seconds...
Sampling...
Predictions (DSP: 19 ms., Classification: 0 ms., Anomaly: 0 ms.):
    idle: 0.00000
    lift: 1.00000
    maritime: 0.00000
    terrestrial : 0.00000

Starting inferencing in 2 seconds...
Sampling...
Predictions (DSP: 19 ms., Classification: 0 ms., Anomaly: 0 ms.):
    idle: 0.00000
    lift: 0.00000
    maritime: 1.00000
    terrestrial : 0.00000

Starting inferencing in 2 seconds...
Sampling...
```

Autoscroll Show timestamp Both NL & CR 115200 baud Clear output

Main references

- Harvard School of Engineering and Applied Sciences - CS249r: Tiny Machine Learning
- Professional Certificate in Tiny Machine Learning (TinyML) – edX/Harvard
- Introduction to Embedded Machine Learning (Coursera)
- Text Book: "TinyML" by Pete Warden, Daniel Situnayake

I want to thank Shawn Hymel and Edge Impulse, Pete Warden and Laurence Moroney from Google, and especially Harvard professor Vijay Janapa Reddi, Ph.D. student Brian Plancher and their staff for preparing the excellent material on TinyML that is the basis of this course at UNIFEI.

The IESTI01 course is part of the TinyML4D, an initiative to make TinyML education available to everyone globally.

Thanks

And stay safe!

UNIFEI