

IESTI01 – TinyML

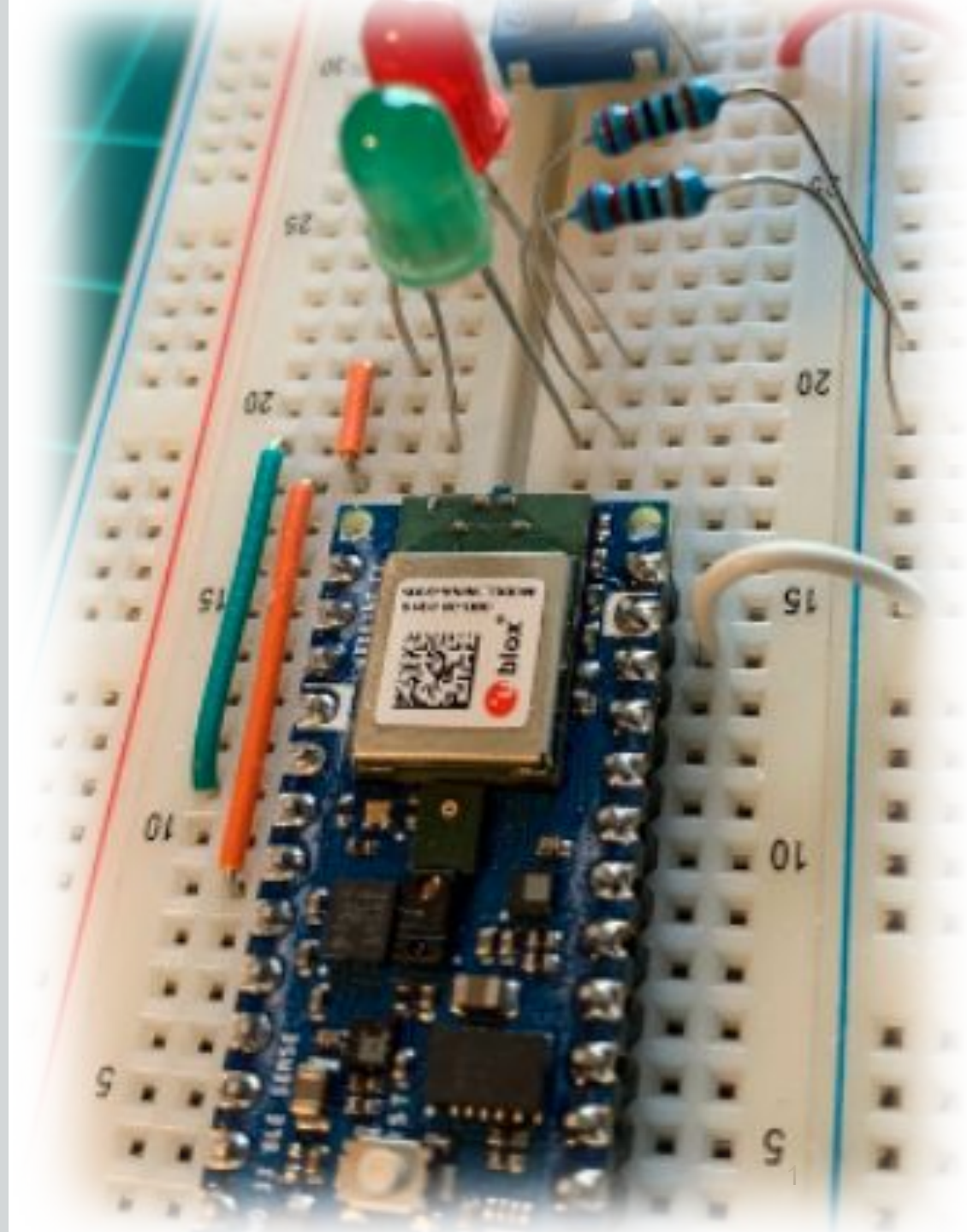
Embedded Machine Learning

27. Collecting Data with Edge Impulse Studio



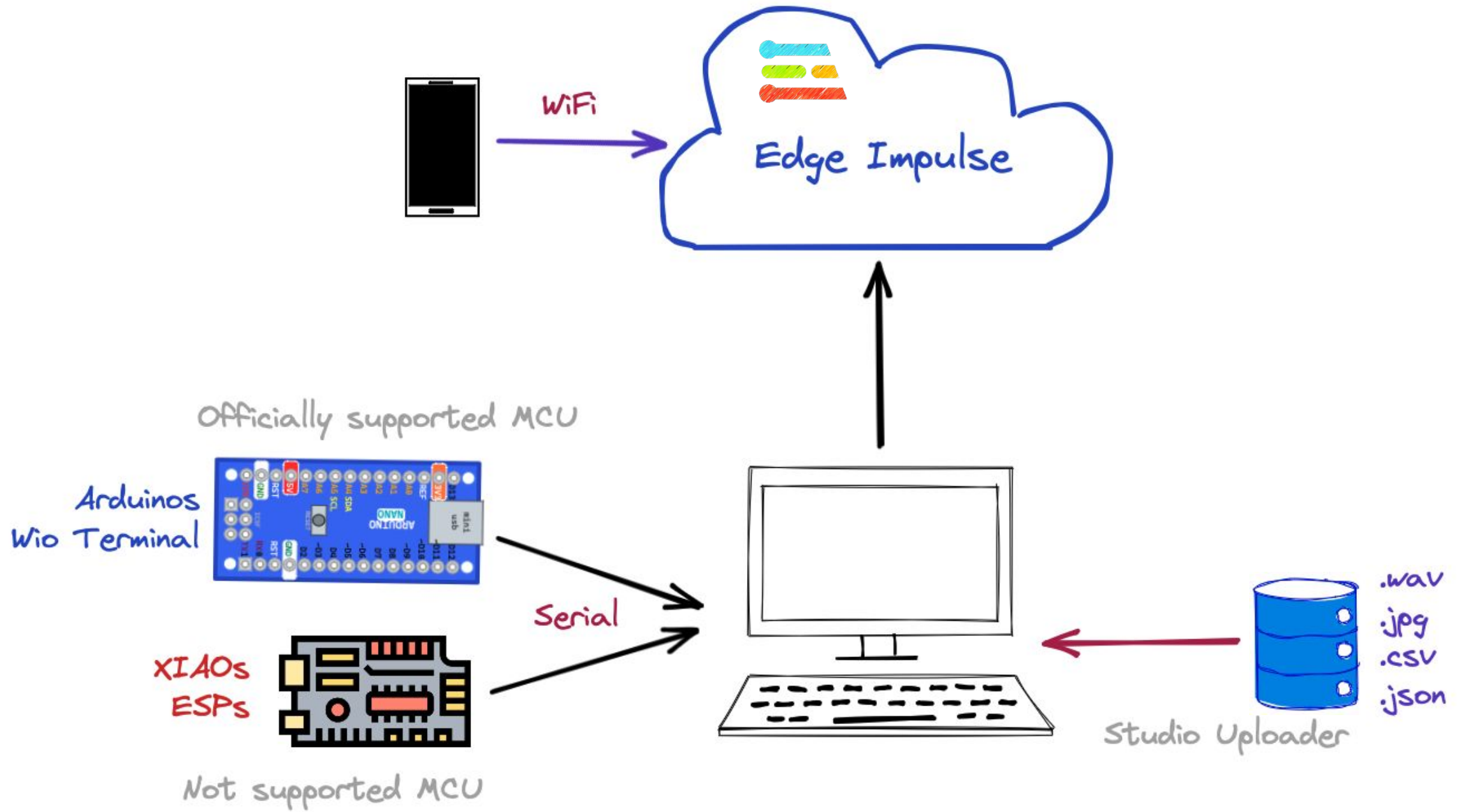
Prof. Marcelo Rovai

UNIFEI



El Studio Data Ingestion

Alternative methods



1. SmartPhone

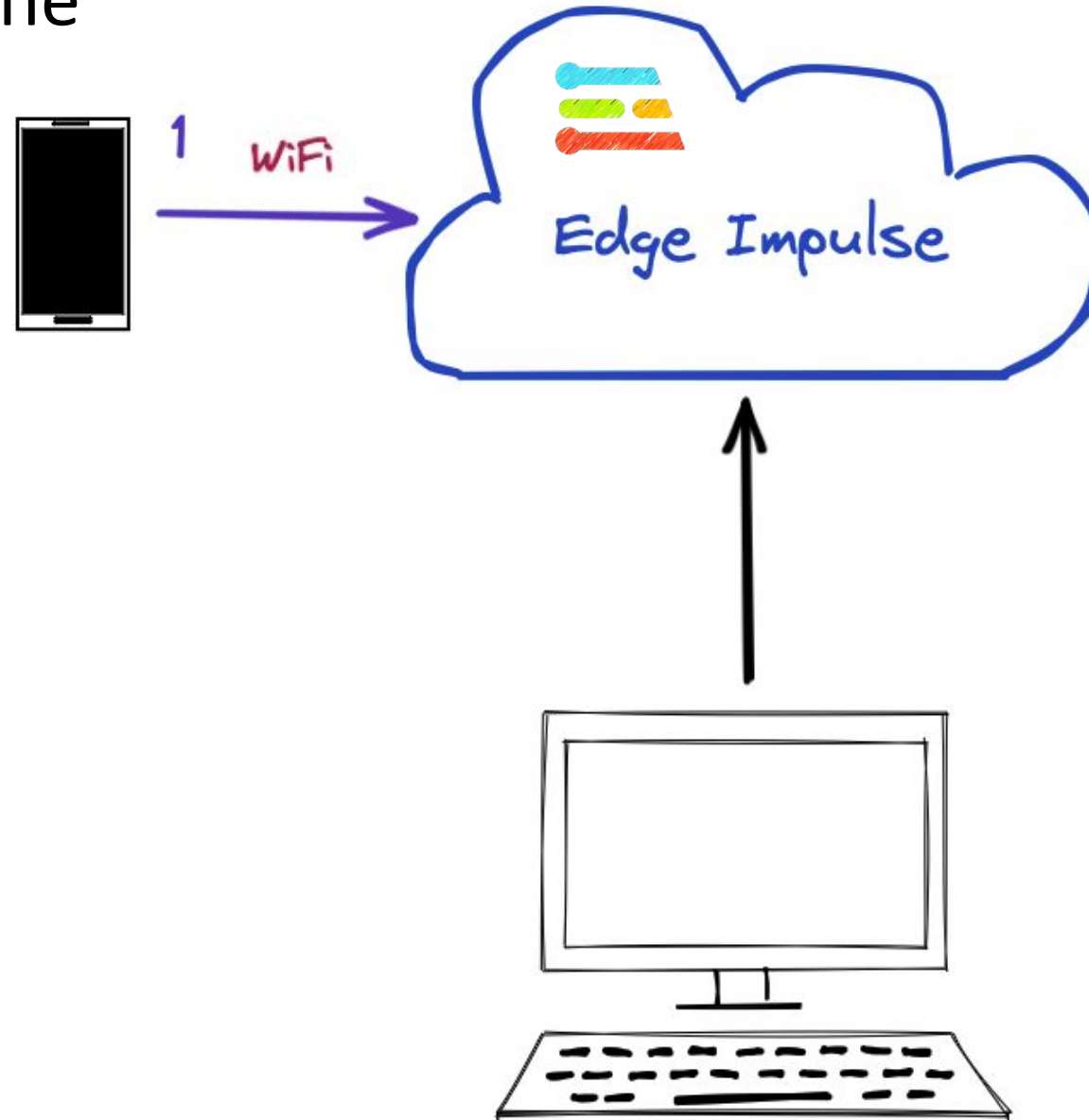
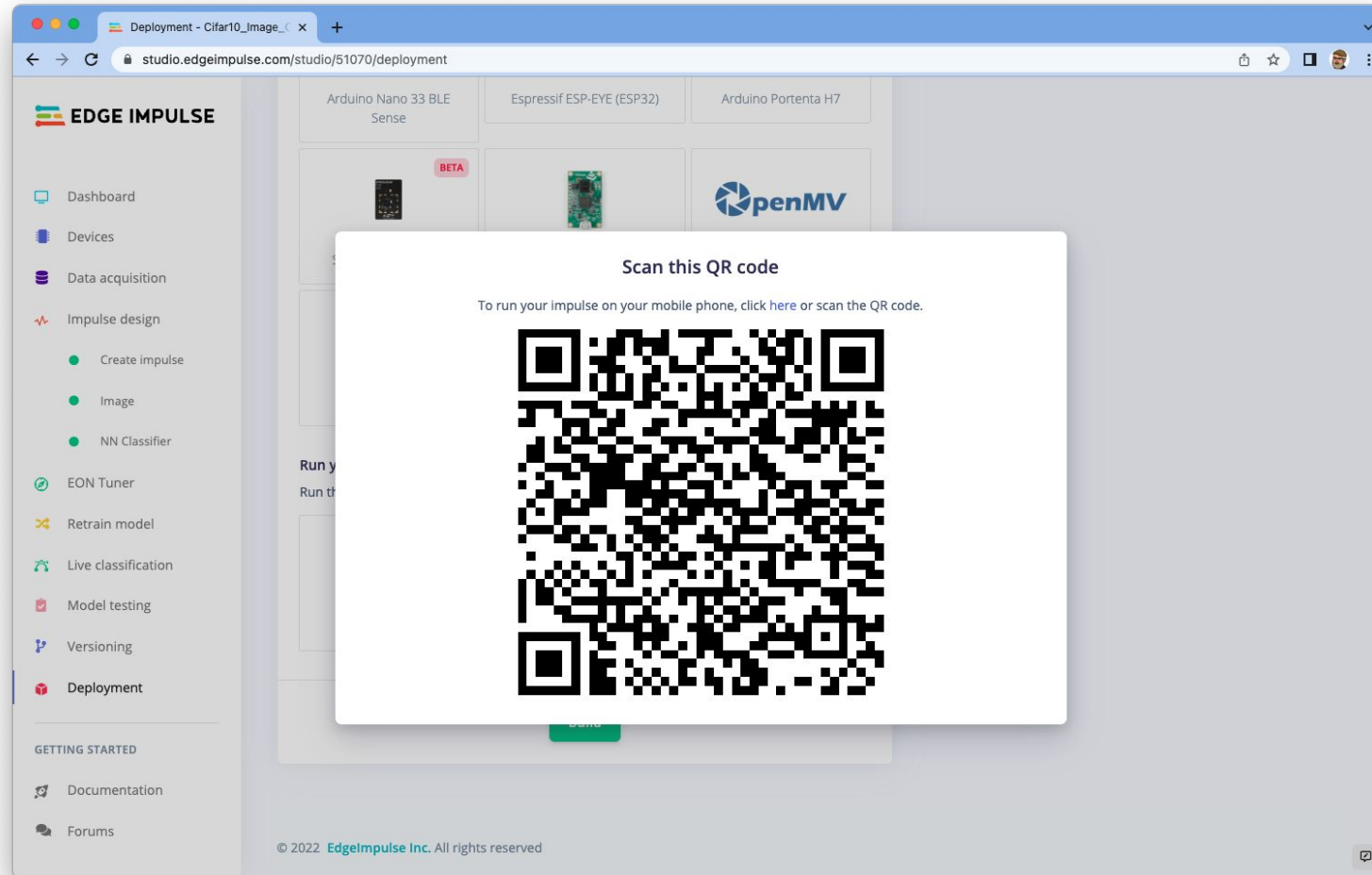


Image Classification using a **smartphone** and
Edge Impulse Studio

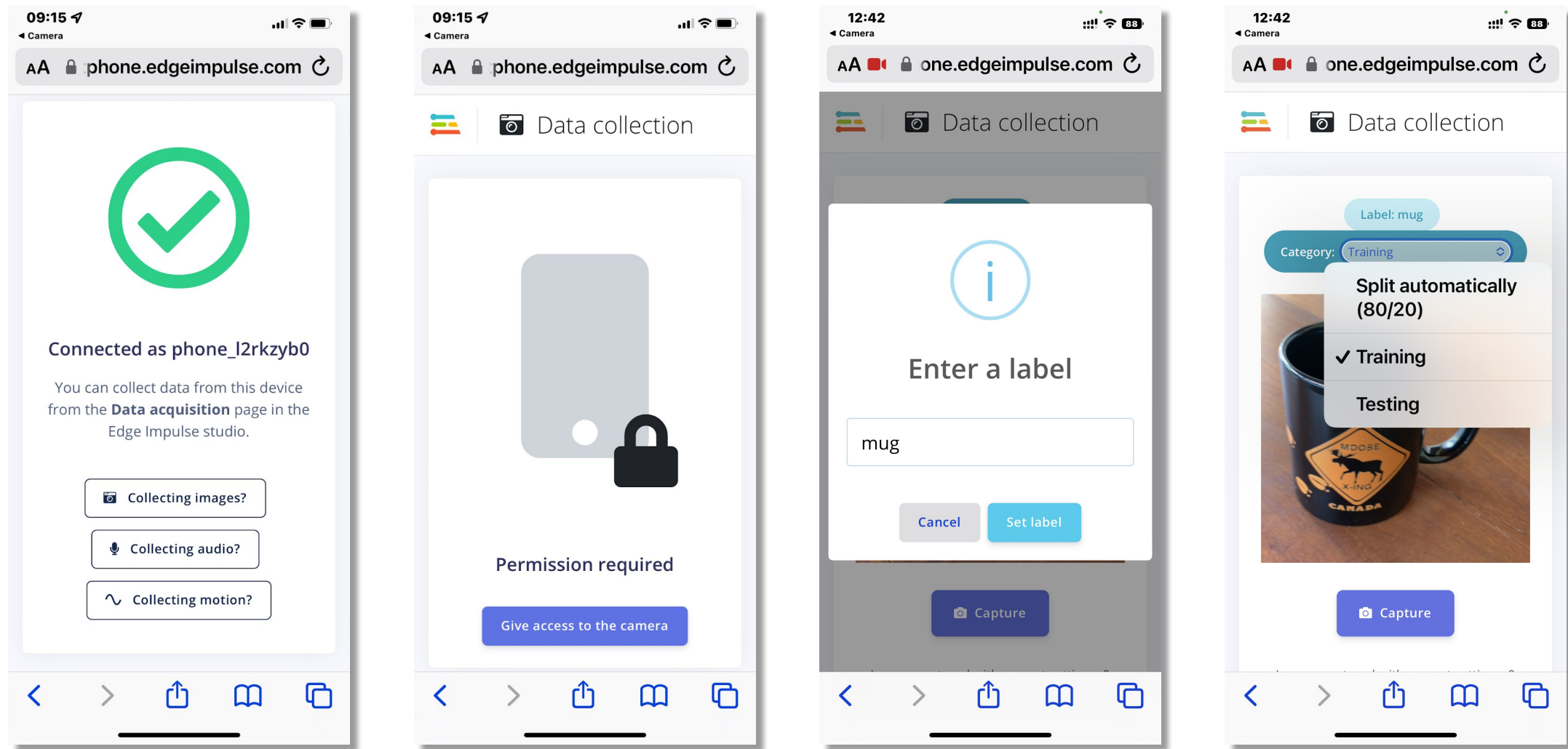
Prof. Marcelo José Ravioli
UNIFEI - Federal University of Itajubá, Brazil
TinyML4U Academic Network Co-Chair



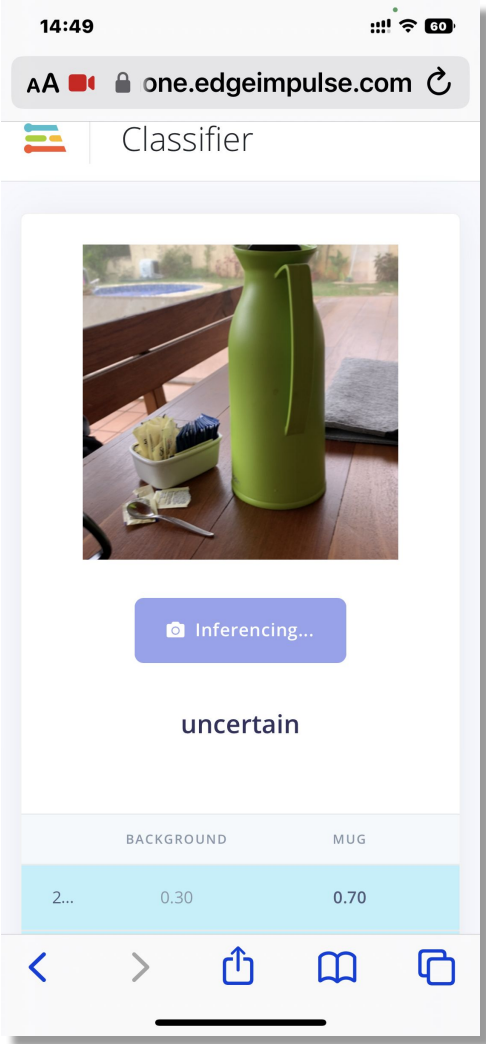
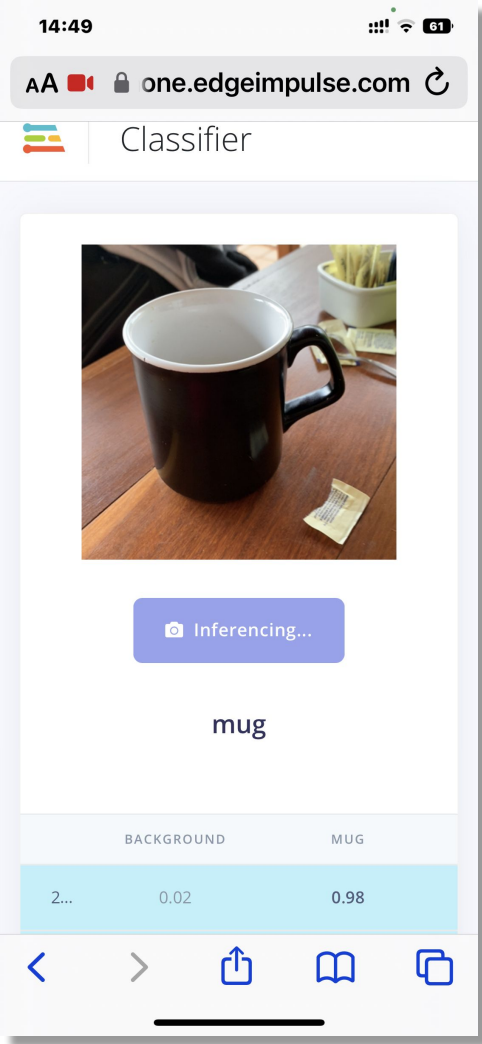
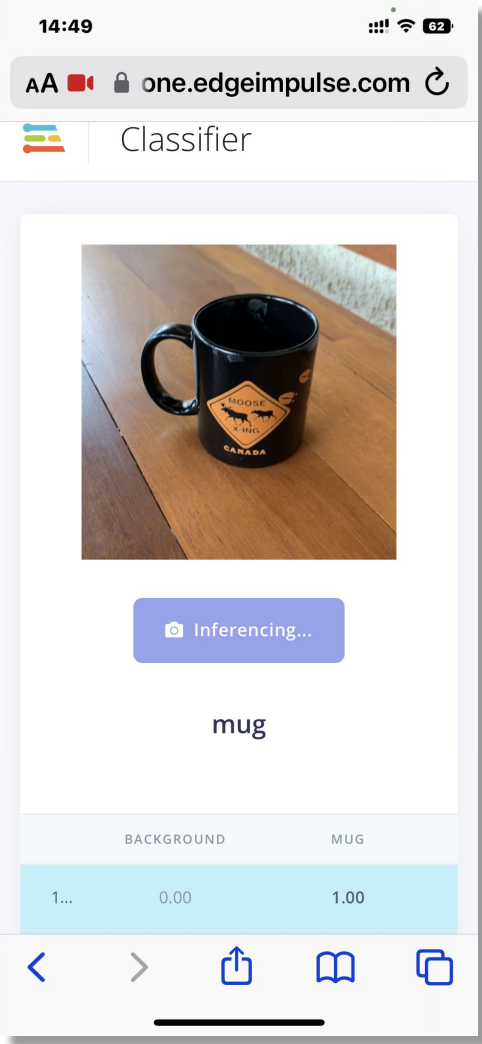
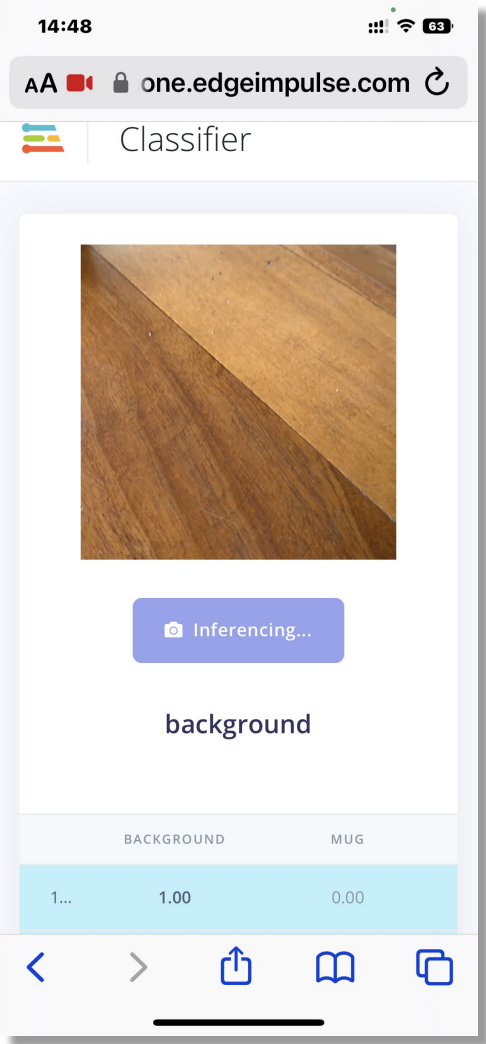
1. Data Ingestion using Smart Phone



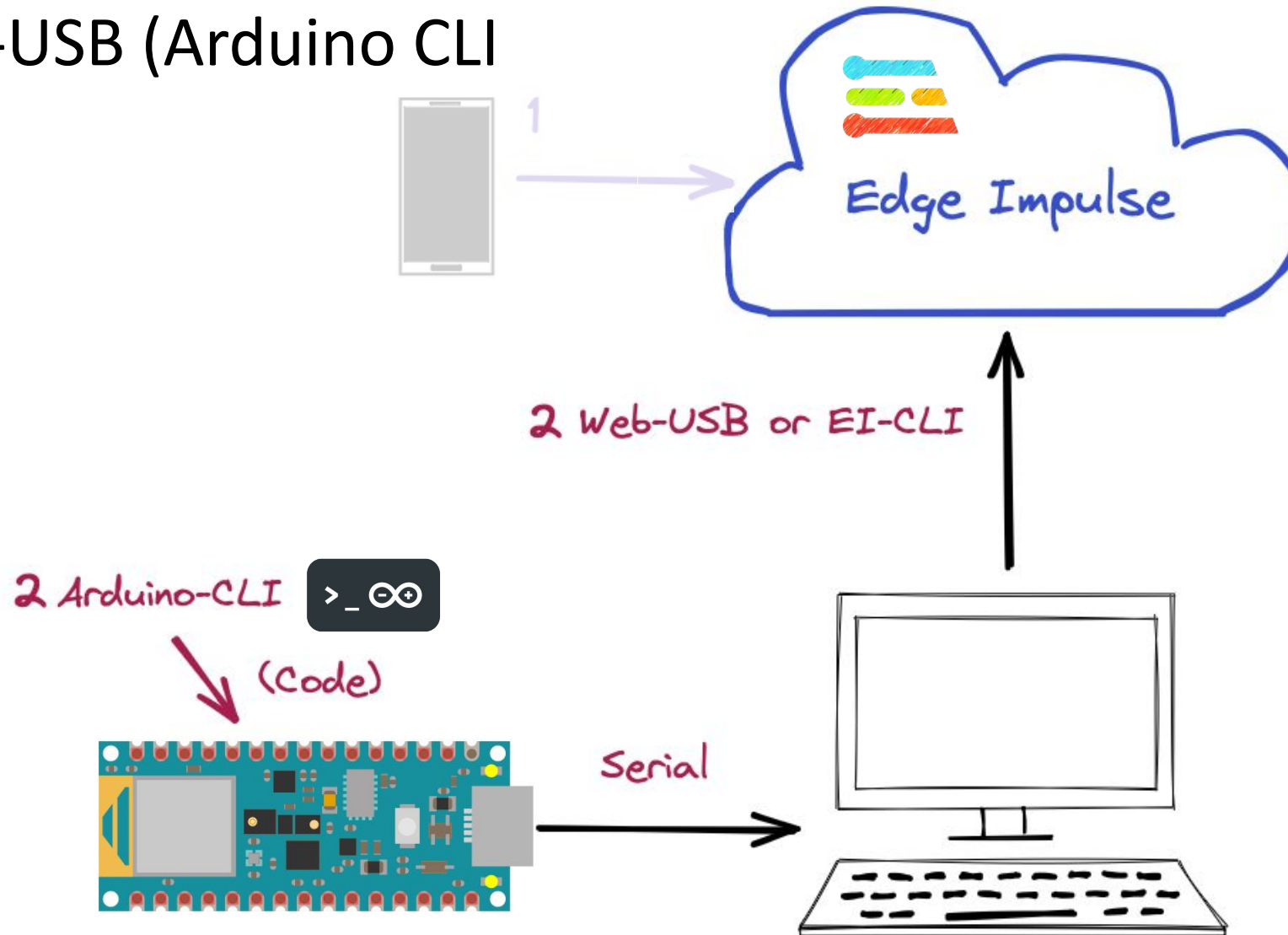
1. Data Capture and model Training



1. Off-Line Inference



2. Web-USB (Arduino CLI)



Issue: Limited MCU and sensors



TinyML Arduino Kit
Connection to **Edge Impulse**

Prof. Marcelo José Ravioli
UNIFEI - Federal University of Itajubá, Brazil
TinyML4D Academic Network Co-Chair



2. Data Ingestion using Arduino-Cli + Web-USB (or EI-CLI)

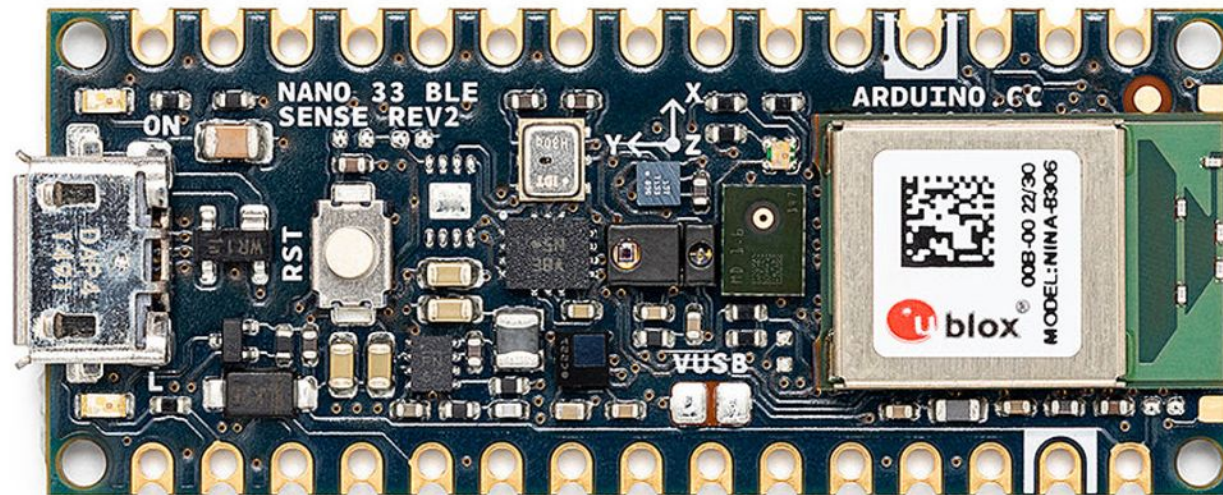
The image is a composite of three screenshots illustrating the data ingestion process:

- File Explorer:** A Windows file explorer window showing the contents of the 'arduino-nano-33-ble-sense' folder. The file 'flash_windows.bat' is highlighted with an orange box.
- Terminal Window:** A command prompt window showing the output of running 'flash_windows.bat'. The output indicates that the Arduino Mbed OS Nano Boards were found, the Arduino Nano 33 BLE was identified at COM11, and the firmware was successfully flashed. The terminal text includes:

```
C:\WINDOWS\system32\cmd.exe
Finding Arduino Mbed core...
arduino:mbd_nano 2.0.0 2.0.0 Arduino Mbed OS Nano Boards
Finding Arduino Mbed core OK
Finding Arduino Nano 33 BLE...
Finding Arduino Nano 33 BLE OK at COM11
arduino:mbd_nano 2.0.0 2.0.0 Arduino Mbed OS Nano Boards
Device : nRF52840-QIAA
Version : Arduino Bootloader (SAM-BA extended) 2.0 [Arduino:IKXYZ]
Address : 0x0
Pages : 256
Page Size : 4096 bytes
Total Size : 1024KB
Planes : 1
Lock Regions : 0
Locked : none
Security : false
Erase flash

Done in 0.002 seconds
Write 525440 bytes to flash (129 pages)
[=====] 100% (129/129 pages)
Done in 22.296 seconds
Flashed your Arduino Nano 33 BLE development board
To set up your development with Edge Impulse, run 'edge-impulse'
To run your impulse on your development board, run 'edge-impulse run'
Pressione qualquer tecla para continuar. . .
```
- Edge Impulse Studio Web Interface:** A screenshot of the Edge Impulse Studio web application. A modal dialog box is open, showing a list of detected USB devices. The device 'Nano 33 BLE [cu.usbmodem144301] - Paired' is selected. The 'Connect' button in the dialog is highlighted with an orange box. In the background, the main interface shows a 'Record new data' section with a 'Connect using WebUSB' button also highlighted with an orange box.

Arduino Nano 33 BLE Sense **Rev2**



- **IMU** - LSM9DS1 - 9 axis → BMI270 – 6 axis + BMM150 - 3 axis
- **Temperature and humidity sensor** - HTS221 → HS3003
- **Microphone** - MP34DT05 → MP34DT06JTR.

Record new data

Device ?

36:17:55:F9:70:F7

Label

lift

Sample length (ms.)

10000

Sensor

Inertial

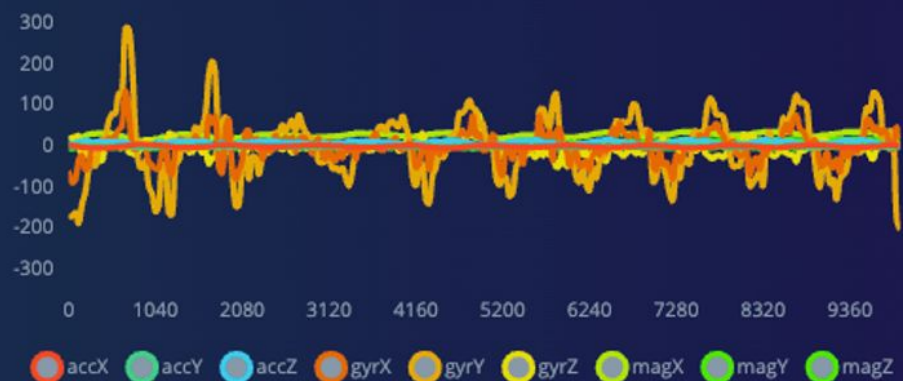
Frequency

62.5Hz

Start sampling

RAW DATA

lift.3soee1ar



Data acquisition

Impulse design

- Create impulse
- Spectral Analysis
- Classifier
- Anomaly detection

EON Tuner

Retrain model

Live classification

Model testing

Versioning

Deployment

GETTING STARTED

Documentation

Forums

Time series data

Input axes (9)

accX, accY, accZ, gyrX, gyrY, gyrZ, magX, magY, magZ

Window size



2000 ms.

Window increase



80 ms.

Frequency (Hz)

62,5

Zero-pad data



Spectral Analysis

Name

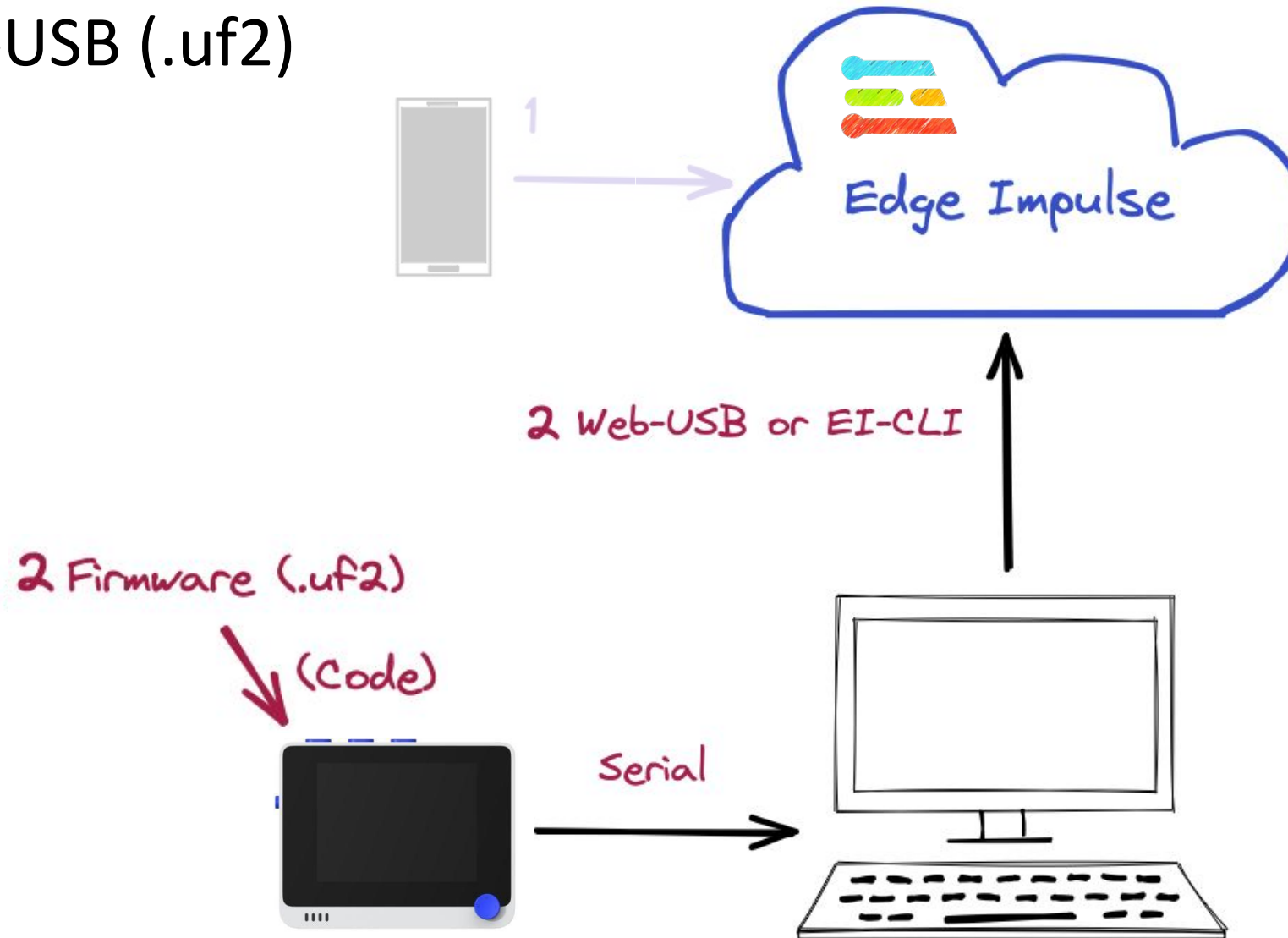
Spectral Analysis

Input axes (3)

- ☒ accX
- ☒ accY
- ☒ accZ
- ☐ gyrX
- ☐ gyrY
- ☐ gyrZ
- ☐ magX
- ☐ magY
- ☐ magZ

Add a processing

2. Web-USB (.uf2)



Issue: Limited MCU and sensors



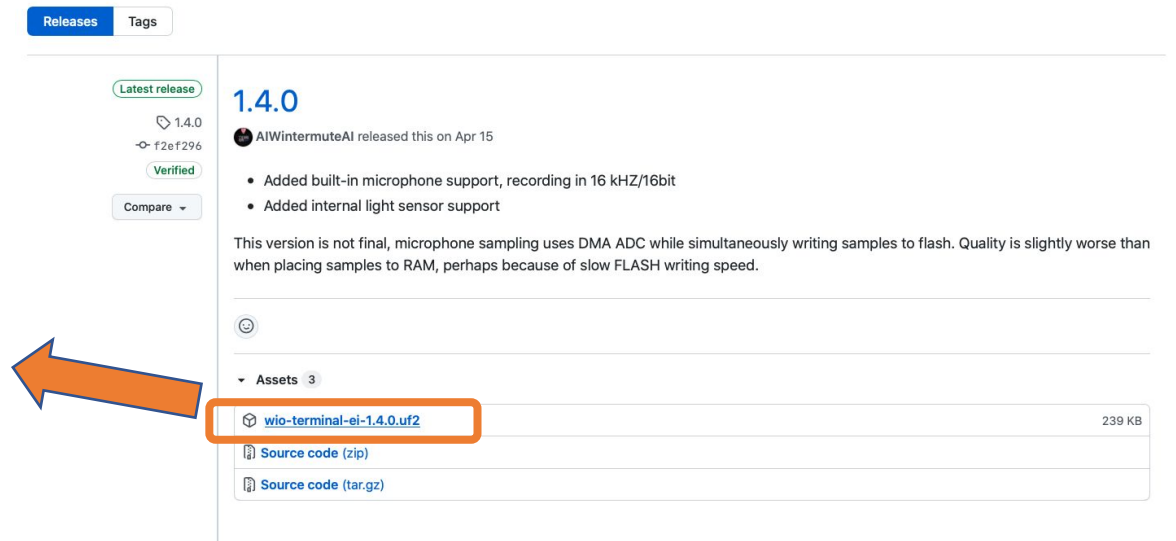
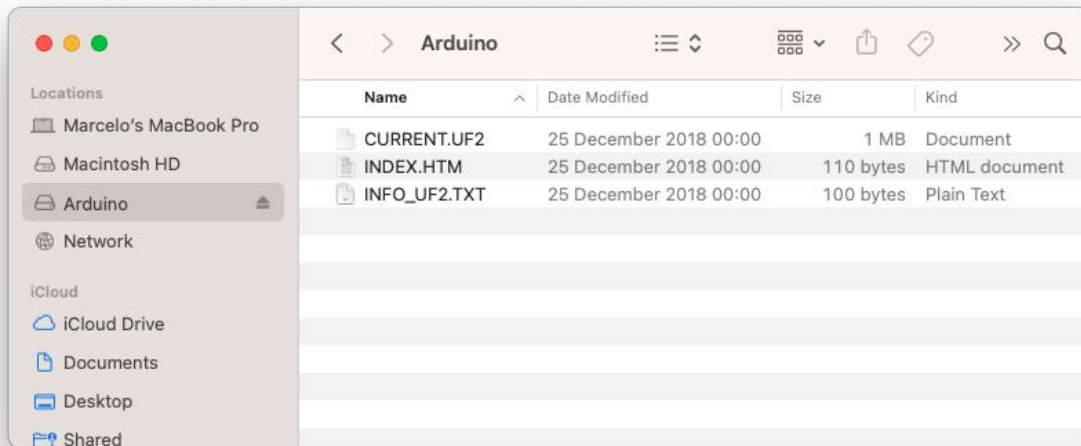
Wio Terminal
Installation & Tests

Prof. Marcelo José Roval
UNIFEI - Federal University of Itajubá, Brazil
ThyML 4D Academic Network Co-Chair

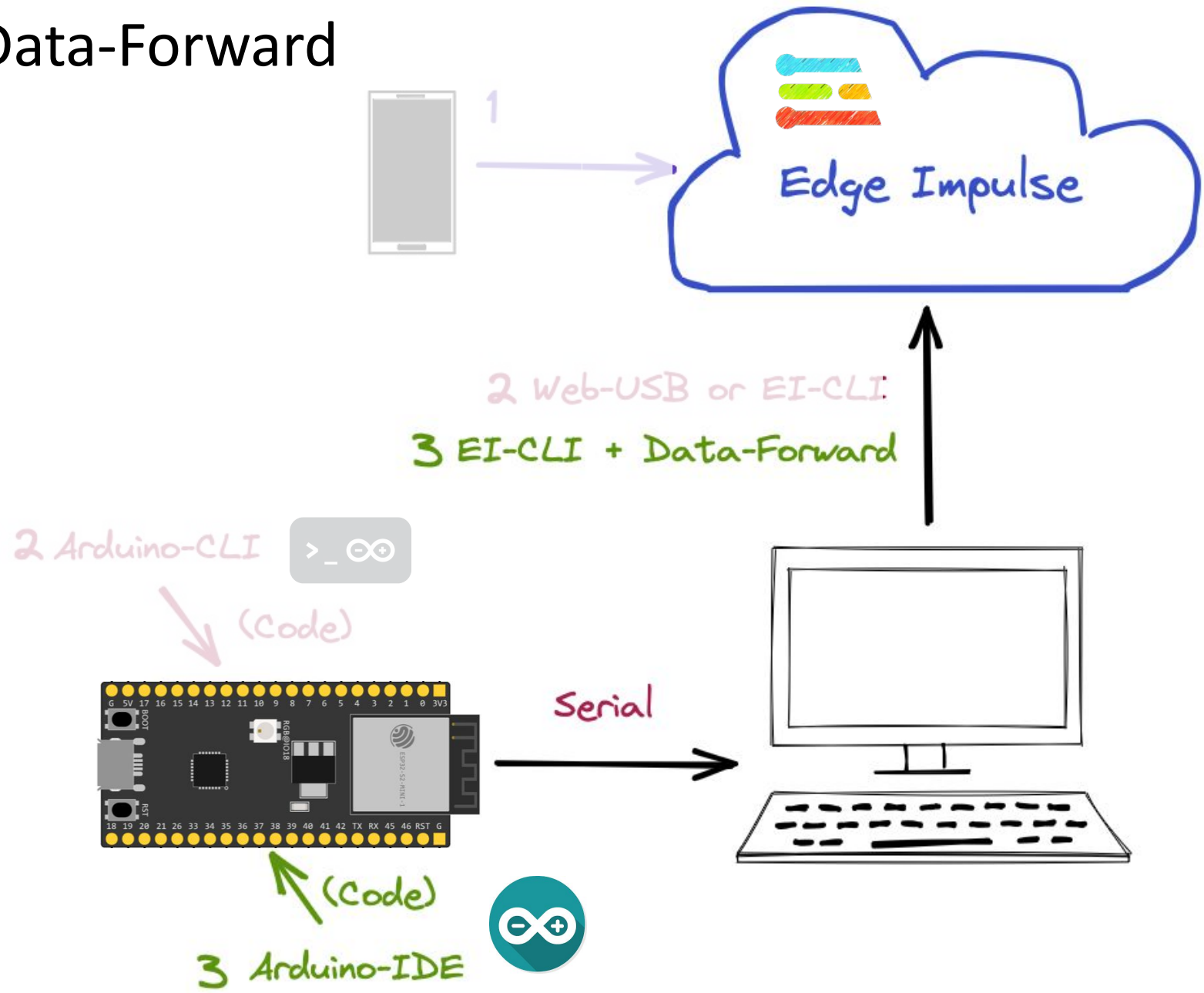


2. (.uf2) Firmware installation

1. Connect Wio Terminal to your computer.
2. Entering the bootloader mode by sliding the power switch twice quickly.
3. An external drive named Arduino should appear in your PC.
4. Drag the the downloaded [Edge Impulse uf2 firmware files](#) to the Arduino drive. Now, Edge Impulse is loaded on Seeeduino Wio Terminal!



3. Data-Forward



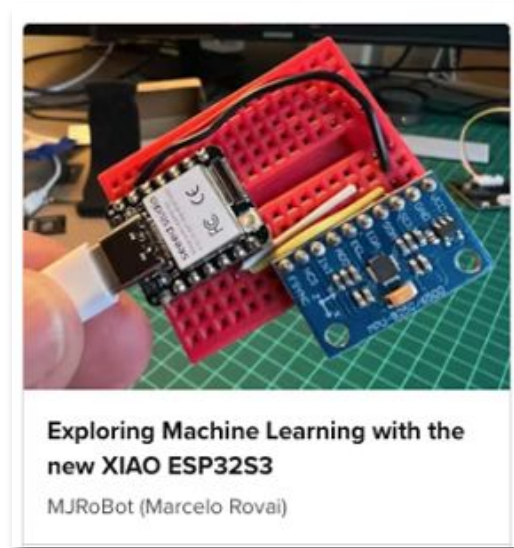
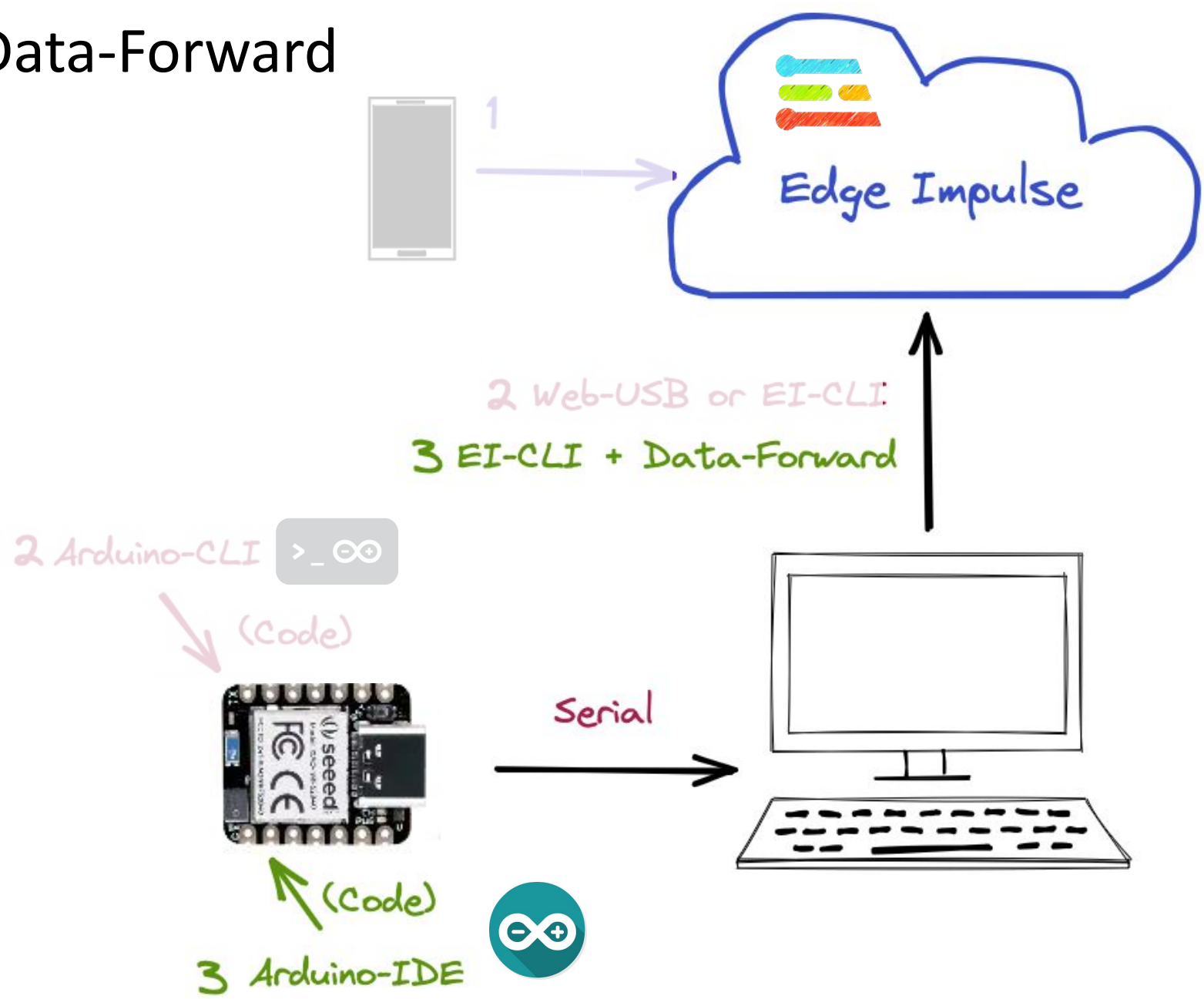


ESP32 - Motion Classification

Prof. Marcelo José Rossi
UNIFEI - Federal University of Itajubá, Brazil
TinyML40 Academic Network Co-Chair



3. Data-Forward



2. Data Ingestion using El-Cli + Data Forward

```
XIAO_BLE_Sense_Accelerometer_Data_Forewarder | Arduino 1.8.19
XIAO_BLE_Sense_Accelerometer_Data_Forewarder $
9  Marcelo Rovai @July2022
10 */
11 #include "LSM6DS3.h"
12 #include "Wire.h"
13
14 //Create an instance of class LSM6DS3
15 LSM6DS3 xIMU(I2C_MODE, 0x6A); //I2C device address 0x6A
16
17 #define CONVERT_G_TO_MS2 9.80665f
18 #define FREQUENCY_HZ 50
19 #define INTERVAL_MS (1000 / (FREQUENCY_HZ + 1))
20 static unsigned long last_interval_ms = 0;
21
22 void setup() {
23   Serial.begin(115200);
24   while (!Serial);
25
26   if (xIMU.begin() != 0) {
27     Serial.println("Device error");
28   } else {
29     Serial.println("Device OK!");
30   }
31   Serial.println("Data Forwarder - Built-in IMU on the XIAO BLE Sense\n");
32 }
33
34 void loop() {
35   float x, y, z;
36   if (millis() > last_interval_ms + INTERVAL_MS) {
37     last_interval_ms = millis();
38     x = xIMU.readFloatAccelX();
39     y = xIMU.readFloatAccelY();
40     z = xIMU.readFloatAccelZ();
41
42     Serial.print(x * CONVERT_G_TO_MS2);
43     Serial.print('\t');
44     Serial.print(y * CONVERT_G_TO_MS2);
45     Serial.print('\t');
46     Serial.println(z * CONVERT_G_TO_MS2);
47   }
48 }
```

\$ edge-impulse-data-forwarder --clean

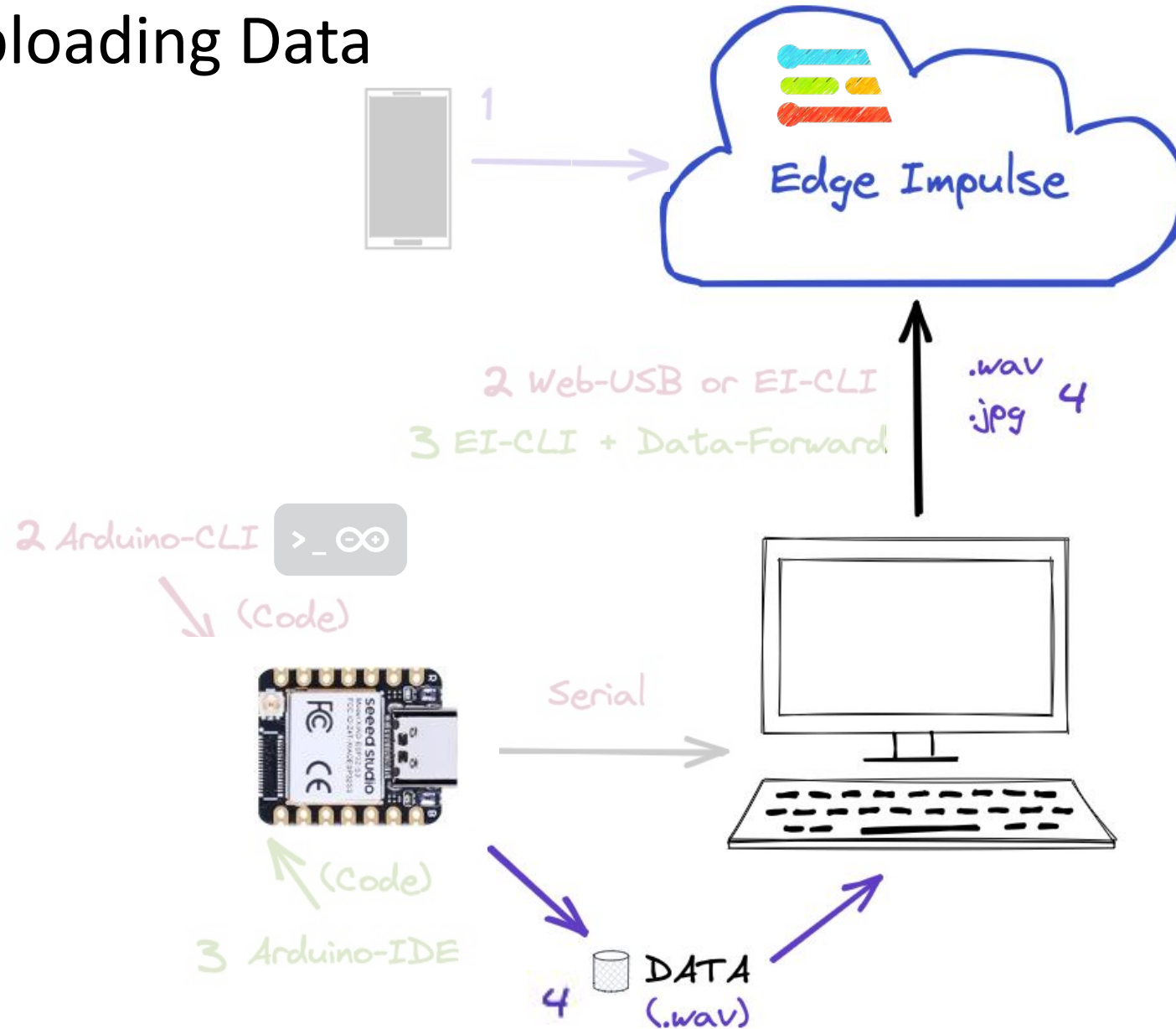


```
mjrovai — -bash — 80x41
[(base) MacBook-Pro-de-Marcelo:~ mjrovai$ edge-impulse-data-forwarder --clean
Edge Impulse data forwarder v1.12.2
[?] What is your user name or e-mail address (edgeimpulse.com)? rovai@mjrobot.org ]
[?] What is your password? [hidden]
Endpoints:
  Websocket: wss://remote-mgmt.edgeimpulse.com
  API:       https://studio.edgeimpulse.com/v1
  Ingestion: https://ingestion.edgeimpulse.com

[SER] Connecting to /dev/tty.usbmodem144301
[SER] Serial is connected (4A:5A:36:17:55:F9:70:F7)
[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) /
  IESTI01_Input_Data_Test
[SER] Detecting data frequency...
[SER] Detected data frequency: 51Hz
[?] 3 sensor axes detected (example values: [-0.08,-0.34,9.82]). What do you want
[to call them? Separate the names with ',': accX, accY,
  accZ
? What name do you want to give this device? nano
[WS ] Device "nano" is now connected to project "IESTI01_Input_Data_Test"
[WS ] Go to https://studio.edgeimpulse.com/studio/39877/acquisition/training to
build your machine learning model!
[WS ] Incoming sampling request {
  path: '/api/training/data',
  label: 'left-right',
  length: 10000,
  interval: 19.607843137254903,
  hmacKey: '6ee929b90e563aa74517f505a3ecb9c8',
  sensor: 'Sensor with 3 axes (accX, accY, accZ)'
}
```

4. Uploading Data



4. Data Ingestion using Upload existing Data

(CBOR, JSON, CSV), or as WAV, JPG or PNG files.

The screenshot displays a web interface for data ingestion. At the top left, a 'LABELS' section shows the number '5' and a circular progress indicator. Below it, a table with columns 'ADDED' and 'LENGTH' is partially visible. A file explorer shows a directory structure with 'data', 'cool', and 'hot' folders. A list of audio files (e.g., 20210710-125854.wav) is shown, with a blue arrow pointing to the '20210710-125854.wav' file. An 'Upload existing data' modal is open, showing options to 'Choose Files' (No file chosen), 'Upload into category' (Training selected), and 'Label' (Enter label: hot). A 'Begin upload' button is at the bottom. To the right, an 'Upload output' section shows a progress log for 14 files, all of which were uploaded successfully.

Labels: 5

Upload existing data

ADDED LENGTH

20210710-125854.wav

20210710-125930.wav

20210710-125956.wav

20210710-130010.wav

20210710-130041.wav

20210710-130100.wav

20210710-130119.wav

20210710-130129.wav

20210710-130142.wav

20210710-130200.wav

20210710-130212.wav

20210710-130224.wav

20210710-130236.wav

20210710-130246.wav

20210710-130256.wav

20210710-130304.wav

20210710-130315.wav

20210710-130329.wav

20210710-130348.wav

20210710-130358.wav

20210710-130408.wav

20210710-130416.wav

UPLOAD DATA (ICTP_PSYCHOACOUSTICS_TEMPERATURE_DEPENDENCE)

Upload existing data

You can upload existing data to your project in the Data Acquisition Format (CBOR, JSON, CSV), or as WAV, JPG or PNG files.

Select files

Choose Files No file chosen

Upload into category

☐ Automatically split between training and testing

☒ Training

☐ Testing

Label

☐ Infer from filename

☒ Enter label:

hot

Begin upload

Upload output

Uploading 14 files...

[1/14] Uploading 20210710-130535.wav OK

[2/14] Uploading 20210710-130603.wav OK

[3/14] Uploading 20210710-130544.wav OK

[4/14] Uploading 20210710-130553.wav OK

[5/14] Uploading 20210710-130738.wav OK

[6/14] Uploading 20210710-130718.wav OK

[7/14] Uploading 20210710-130649.wav OK

[8/14] Uploading 20210710-130700.wav OK

[9/14] Uploading 20210710-130630.wav OK

[10/14] Uploading 20210710-130621.wav OK

[11/14] Uploading 20210710-130709.wav OK

[12/14] Uploading 20210710-130611.wav OK

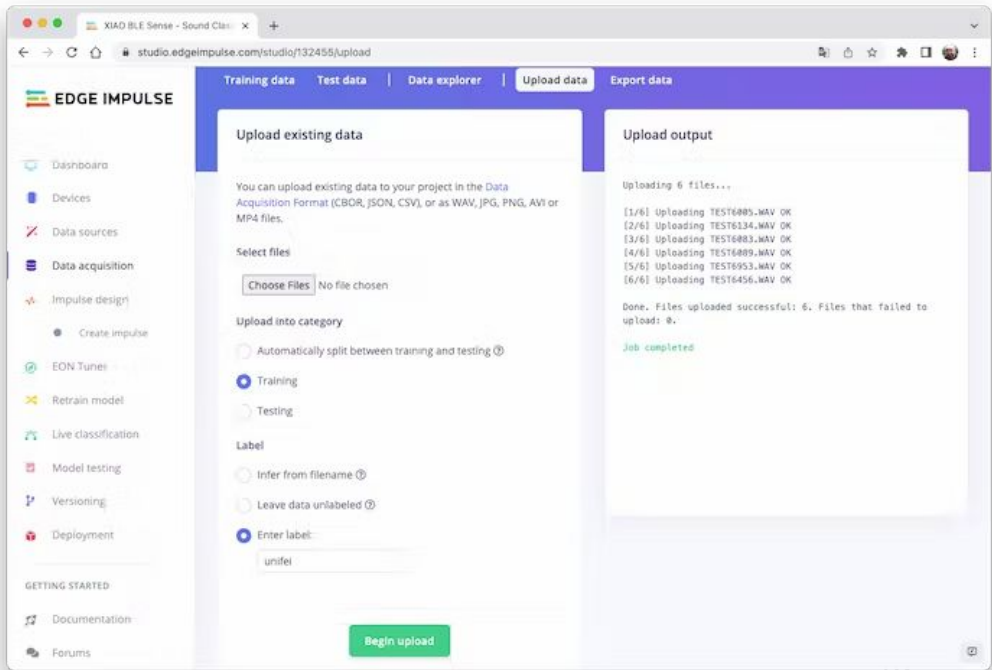
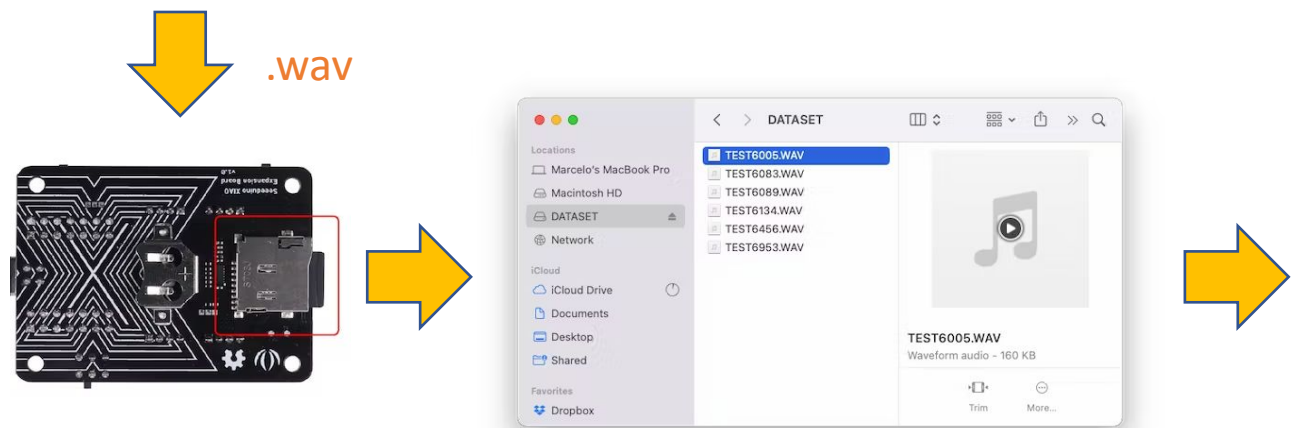
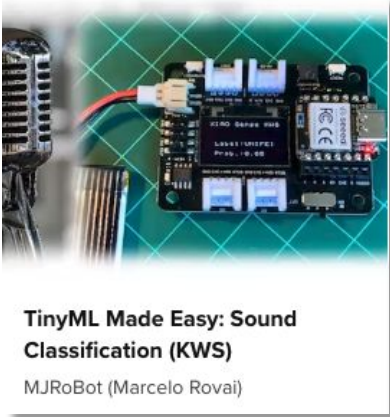
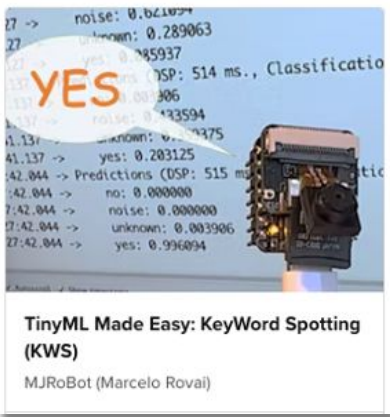
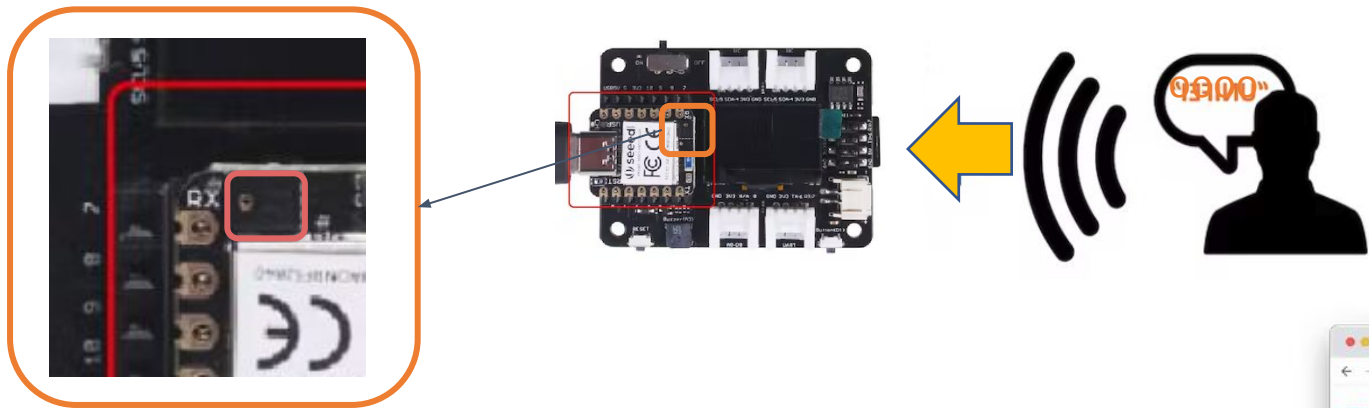
[13/14] Uploading 20210710-130728.wav OK

[14/14] Uploading 20210710-130639.wav OK

Done. Files uploaded successful: 14. Files that failed to upload: 0.

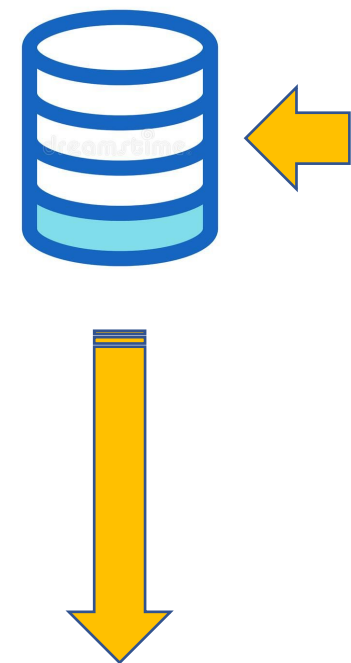
Job completed

4. Uploading .wav data



4. Uploading .jpg data

<https://github.com/YoongiKim/CIFAR-10-images>



master 1 branch 0 tags

YoongiKim Create README.md

test Upload

train Upload

README.md Create README.md

README.md

CIFAR-10-images

CIFAR-10 raw jpeg images

You can just clone this repository to use dataset.

Go to file Add file Code

Clone

HTTPS SSH GitHub CLI

https://github.com/YoongiKim/CIFAR-10-

Use Git or checkout with SVN using the web URL.

Open with GitHub Desktop

Download ZIP

About

CIFAR-10 raw jpeg images

Readme

Releases

No releases published

Packages

dog

CIFAR-10-images-master test

CIFAR-10-images-master.zip train

airplane

automobile

bird

cat

deer

dog

frog

horse

ship

truck

0000.jpg

0001.jpg

0002.jpg

0003.jpg

0004.jpg

0005.jpg

0006.jpg

0007.jpg

0008.jpg

0009.jpg

0010.jpg

0011.jpg

0012.jpg

0013.jpg

0014.jpg

0015.jpg

0016.jpg

0017.jpg

0018.jpg

0019.jpg

0020.jpg

0006.jpg

JPEG image - 919 bytes

Information Show Less

Rotate Left Markup More...

EDGE IMPULSE

Dashboard

Devices

Data acquisition

Impulse design

Create impulse

Image

NN Classifier

UPLOAD DATA (CIFAR10_IMAGE_CLASSIFICATION)

Upload existing data

You can upload existing data to your project in the Data Acquisition Format (CBOR, JSON, CSV), or as WAV, JPG or PNG files.

Select files

Choose Files No file chosen

Upload into category

Automatically split between training and testing

Training

Testing

Label

Infer from filename

Enter label:

dog

begin upload

Workshop on Scientific Use of Machine Learning on Low-Power Devices: Applications and Advanced Topics

Image Classification (Cifar10) using Convolutions (CNN) and Edge Impulse Studio

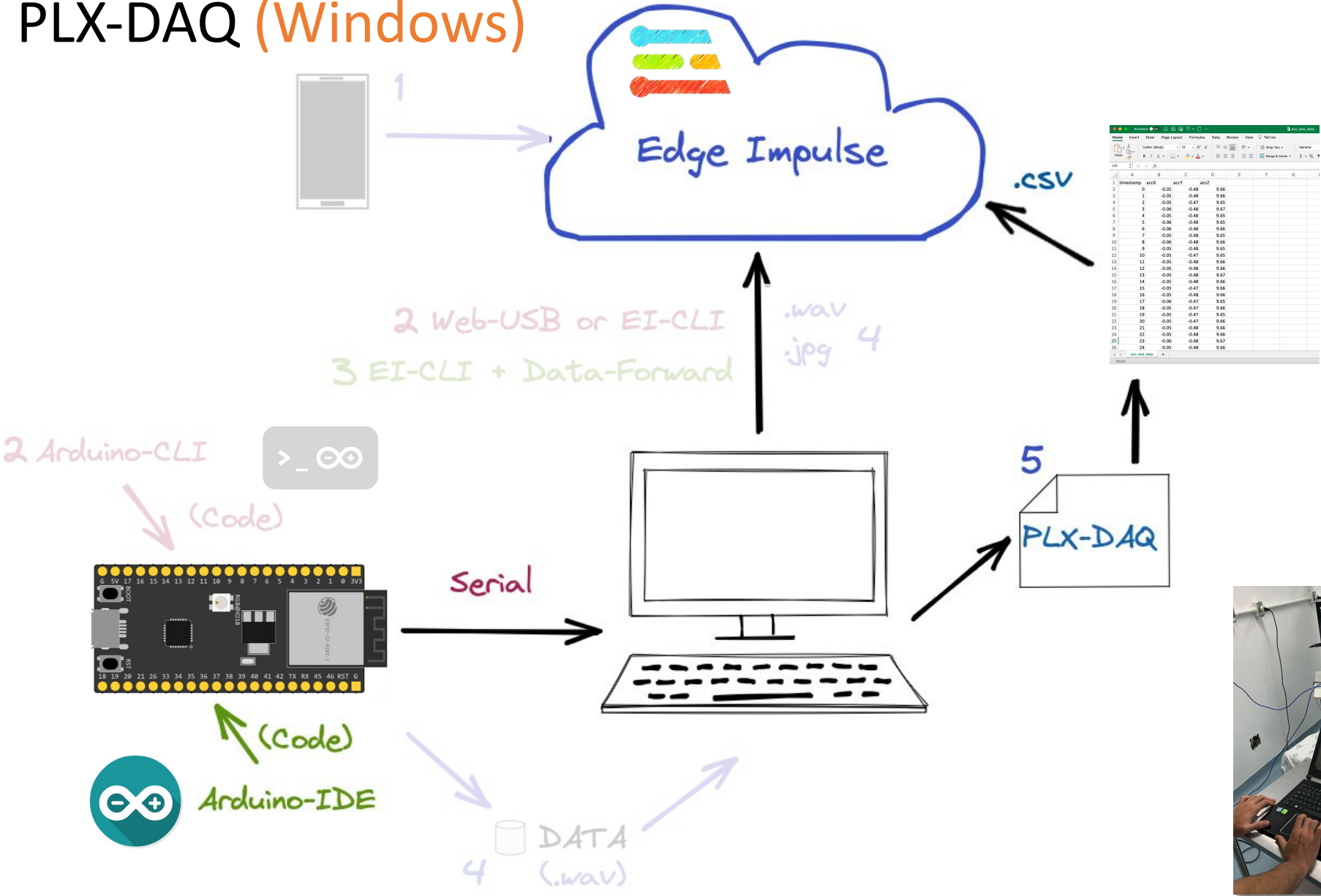
Prof. Marcelo José Rovali

UNIFEI - Federal University of Itajubá, Brazil

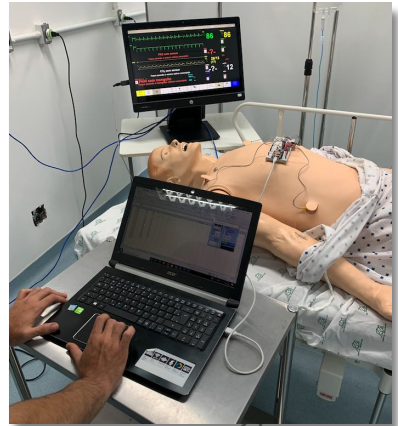
TinyML/60 Academic Network Co-Chair

UNIFEI

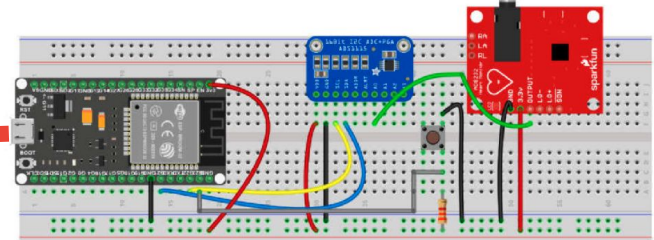
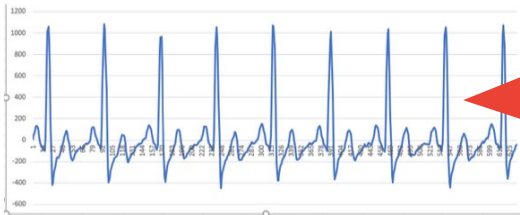
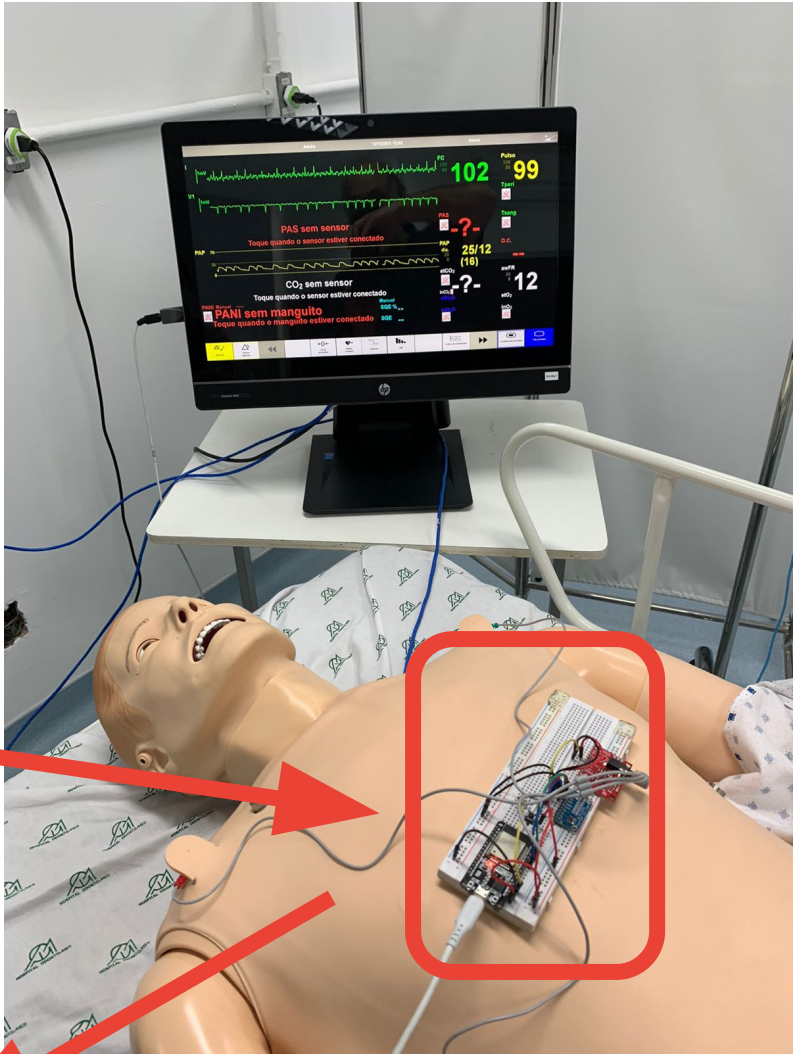
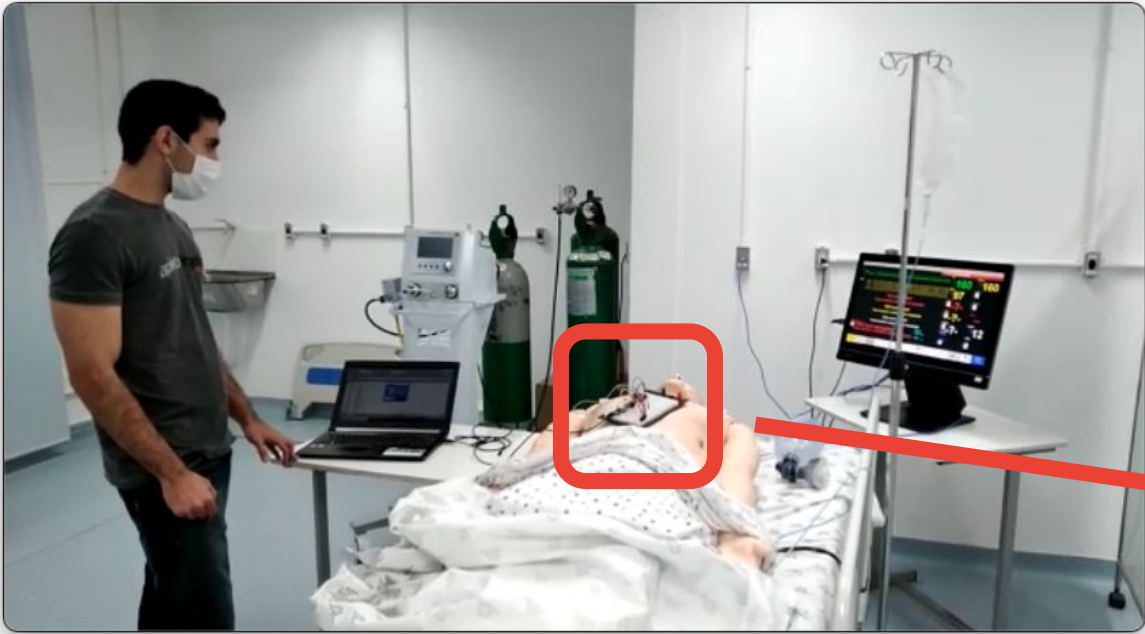
5. (.CSV) PLX-DAQ (Windows)



TimeStamp	INCH	INCH	INCH	INCH	INCH	INCH	INCH
0	-0.05	-0.48	9.66				
1	-0.05	-0.48	9.66				
2	-0.05	-0.47	9.66				
3	-0.06	-0.48	9.67				
4	-0.05	-0.48	9.65				
5	-0.06	-0.48	9.65				
6	-0.06	-0.48	9.66				
7	-0.05	-0.48	9.65				
8	-0.06	-0.48	9.65				
9	-0.05	-0.48	9.65				
10	-0.06	-0.48	9.65				
11	-0.05	-0.48	9.65				
12	-0.05	-0.47	9.65				
13	-0.05	-0.48	9.66				
14	-0.05	-0.48	9.66				
15	-0.05	-0.48	9.67				
16	-0.05	-0.48	9.66				
17	-0.05	-0.47	9.66				
18	-0.05	-0.48	9.66				
19	-0.06	-0.47	9.65				
20	-0.05	-0.47	9.66				
21	-0.05	-0.47	9.65				
22	-0.05	-0.47	9.66				
23	-0.05	-0.48	9.66				
24	-0.05	-0.48	9.66				
25	-0.06	-0.48	9.67				
26	-0.05	-0.48	9.66				



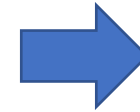
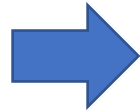
5. (.CSV) PLX-DAQ (Windows)



fritzing

5. Data Ingestion using PLX-DAQ (Windows) => Final Format: .csv

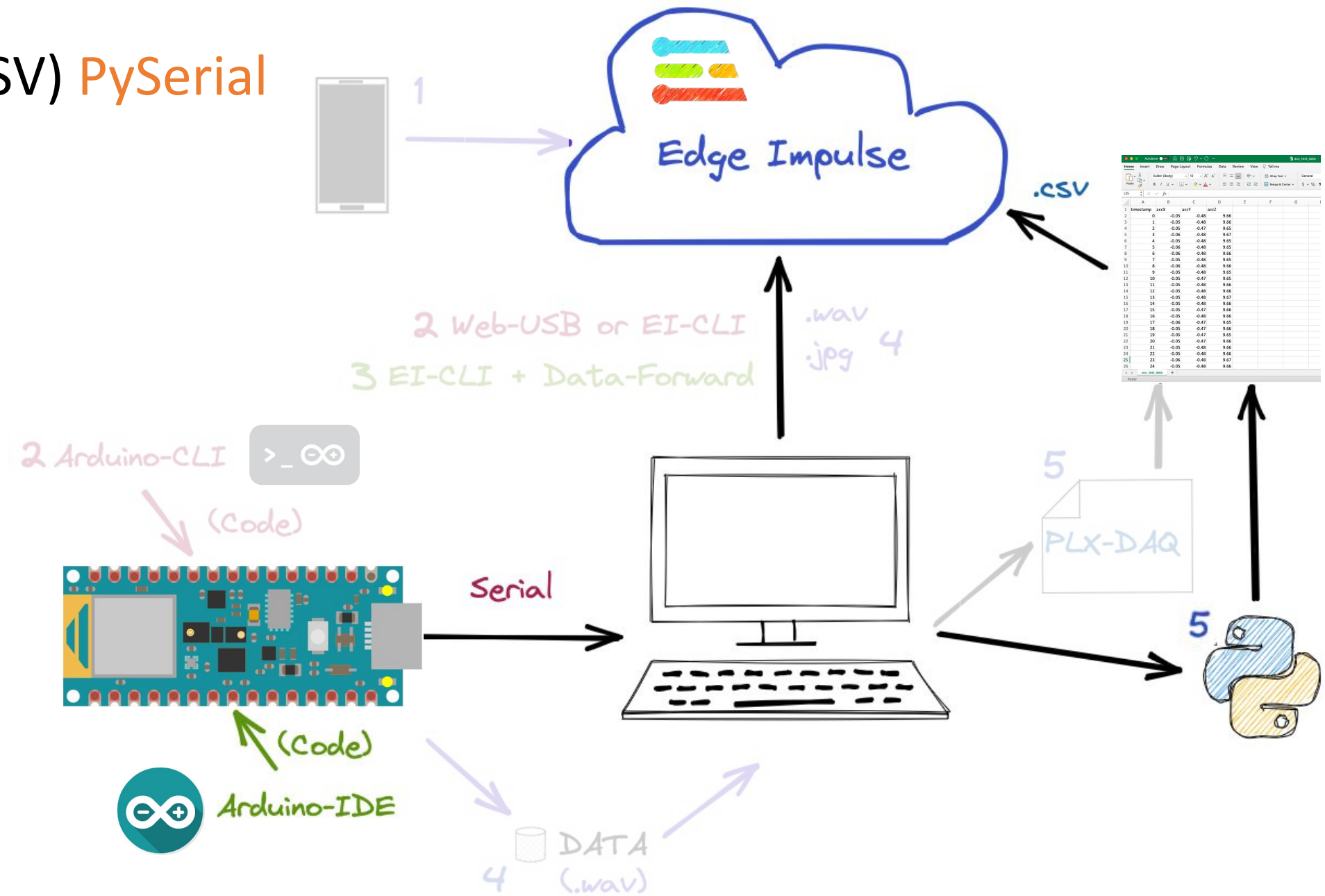
```
Capture_Ardu33_Sense_IMU_Acc
1 #include <Arduino_LSM9DS1.h>
2
3 #define CONVERT_G_TO_MS2 9.80665f
4 #define FREQUENCY_HZ 50
5 #define INTERVAL_MS (1000 / (FREQUENCY_HZ + 1))
6
7 void setup() {
8     Serial.begin(115200);
9     while (!Serial);
10    Serial.println("Started");
11
12    if (!IMU.begin()) {
13        Serial.println("Failed to initialize IMU!");
14        while (1);
15    }
16 }
17
18 void loop() {
19     static unsigned long last_interval_ms = 0;
20     float x, y, z;
21
22     if (millis() > last_interval_ms + INTERVAL_MS) {
23         last_interval_ms = millis();
24
25         IMU.readAcceleration(x, y, z);
26
27         Serial.print(x * CONVERT_G_TO_MS2);
28         Serial.print(',');
29         Serial.print(y * CONVERT_G_TO_MS2);
30         Serial.print(',');
31         Serial.println(z * CONVERT_G_TO_MS2);
32     }
33 }
```



	A	B	C	D	E	F	G	I
1	timestamp	accX	accY	accZ				
2	0	-0.05	-0.48	9.66				
3	1	-0.05	-0.48	9.66				
4	2	-0.05	-0.47	9.65				
5	3	-0.06	-0.48	9.67				
6	4	-0.05	-0.48	9.65				
7	5	-0.06	-0.48	9.65				
8	6	-0.06	-0.48	9.66				
9	7	-0.05	-0.48	9.65				
10	8	-0.06	-0.48	9.66				
11	9	-0.05	-0.48	9.65				
12	10	-0.05	-0.47	9.65				
13	11	-0.05	-0.48	9.66				
14	12	-0.05	-0.48	9.66				
15	13	-0.05	-0.48	9.67				
16	14	-0.05	-0.48	9.66				
17	15	-0.05	-0.47	9.66				
18	16	-0.05	-0.48	9.66				
19	17	-0.06	-0.47	9.65				
20	18	-0.05	-0.47	9.66				
21	19	-0.05	-0.47	9.65				
22	20	-0.05	-0.47	9.66				
23	21	-0.05	-0.48	9.66				
24	22	-0.05	-0.48	9.66				
25	23	-0.06	-0.48	9.67				
26	24	-0.05	-0.48	9.66				

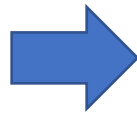
<https://www.youtube.com/watch?v=BwbmNle2CZo>

5. (.CSV) PySerial



5. Data Ingestion using Python (PySerial) => Final Format: .csv

```
Capture_Ardu33_Sense_IMU_Acc
1 #include <Arduino_LSM9DS1.h>
2
3 #define CONVERT_G_TO_MS2 9.80665f
4 #define FREQUENCY_HZ 50
5 #define INTERVAL_MS (1000 / (FREQUENCY_HZ + 1))
6
7 void setup() {
8     Serial.begin(115200);
9     while (!Serial);
10    Serial.println("Started");
11
12    if (!IMU.begin()) {
13        Serial.println("Failed to initialize IMU!");
14        while (1);
15    }
16 }
17
18 void loop() {
19     static unsigned long last_interval_ms = 0;
20     float x, y, z;
21
22     if (millis() > last_interval_ms + INTERVAL_MS) {
23         last_interval_ms = millis();
24
25         IMU.readAcceleration(x, y, z);
26
27         Serial.print(x * CONVERT_G_TO_MS2);
28         Serial.print(',');
29         Serial.print(y * CONVERT_G_TO_MS2);
30         Serial.print(',');
31         Serial.println(z * CONVERT_G_TO_MS2);
32     }
33 }
```

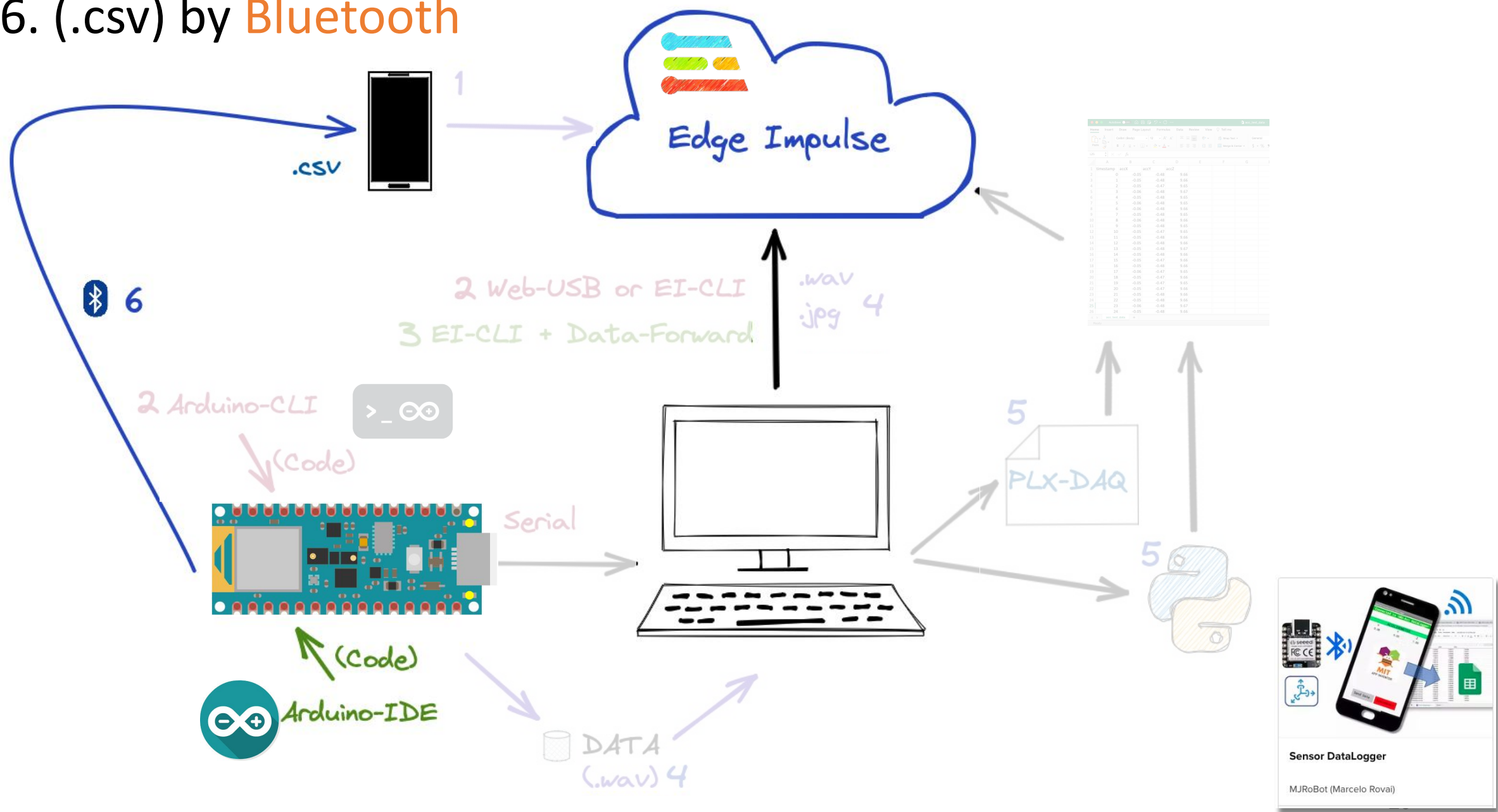


```
1 # Sensor data Logger (CSV)
2 # by Marcelo Rovai @ 13July21
3
4 import serial
5
6 arduino_port = '/dev/tty.usbmodem144301'
7 baud_rate = 115200
8 ser = serial.Serial(port=arduino_port, baudrate=baud_rate)
9
10 fileName = "acc_test_data.csv" # name of the CSV file generated
11
12 first_line = 'timestamp,accX,accY,accZ'
13 file = open(fileName, "w")
14 file.write(first_line + "\n") # write data with a newline
15 file.close()
16
17 Freq_hz = 50
18 num_seconds = 10 # number of seconds collecting data
19 samples = num_seconds * Freq_hz # number of samples to collect
20
21 sample = 0
22 while sample <= samples:
23     getData = str(ser.readline())
24     data = getData[2:][:5]
25     print(data)
26
27     file = open(fileName, "a")
28     file.write(str(sample) + "," + data + "\n")
29     sample = sample+1
30 print("Data collection complete!")
31 file.close()
```

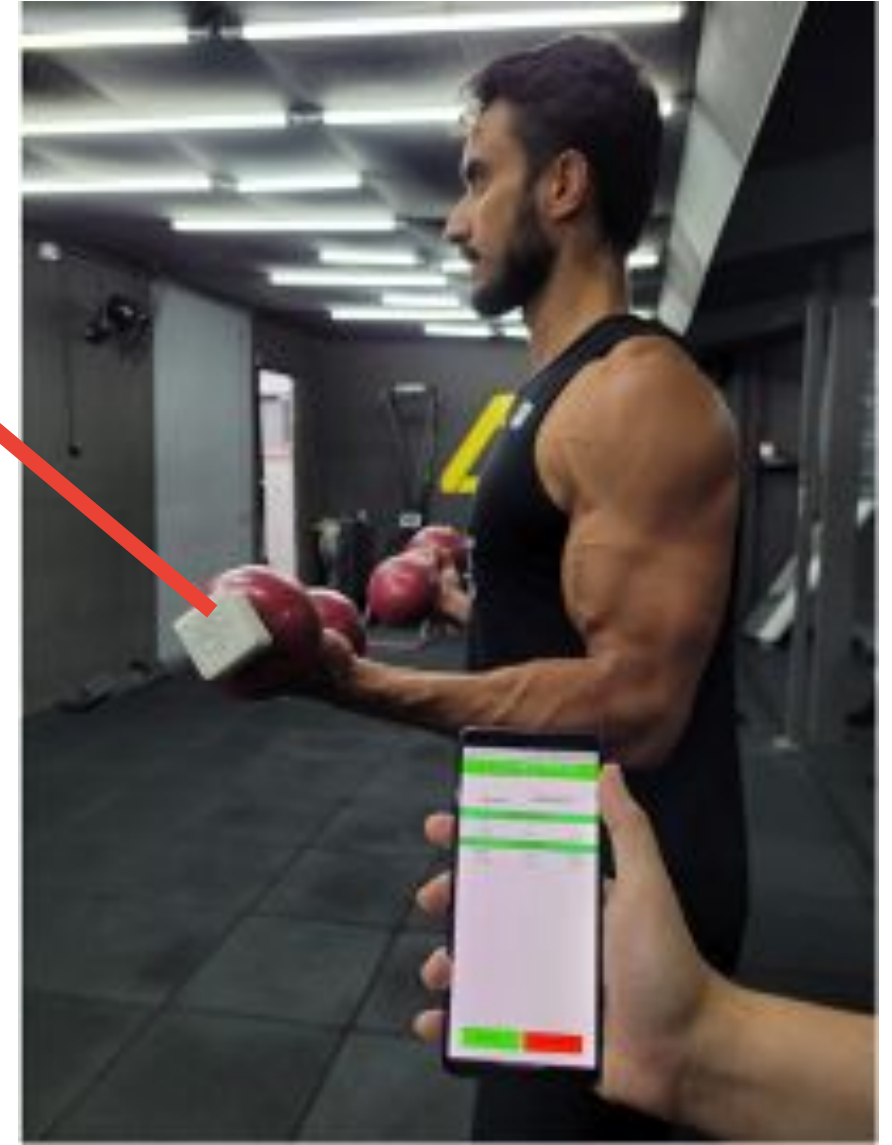
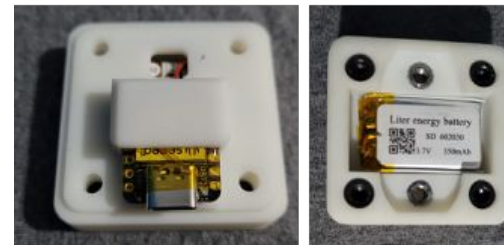


	A	B	C	D	E	F	G	H	I
1	timestamp	accX	accY	accZ					
2	0	-0.05	-0.48	9.66					
3	1	-0.05	-0.48	9.66					
4	2	-0.05	-0.47	9.65					
5	3	-0.06	-0.48	9.67					
6	4	-0.05	-0.48	9.65					
7	5	-0.06	-0.48	9.65					
8	6	-0.06	-0.48	9.66					
9	7	-0.05	-0.48	9.65					
10	8	-0.06	-0.48	9.66					
11	9	-0.05	-0.48	9.65					
12	10	-0.05	-0.47	9.65					
13	11	-0.05	-0.48	9.66					
14	12	-0.05	-0.48	9.66					
15	13	-0.05	-0.48	9.67					
16	14	-0.05	-0.48	9.66					
17	15	-0.05	-0.47	9.66					
18	16	-0.05	-0.48	9.66					
19	17	-0.06	-0.47	9.65					
20	18	-0.05	-0.47	9.66					
21	19	-0.05	-0.47	9.65					
22	20	-0.05	-0.47	9.66					
23	21	-0.05	-0.48	9.66					
24	22	-0.05	-0.48	9.66					
25	23	-0.06	-0.48	9.67					
26	24	-0.05	-0.48	9.66					

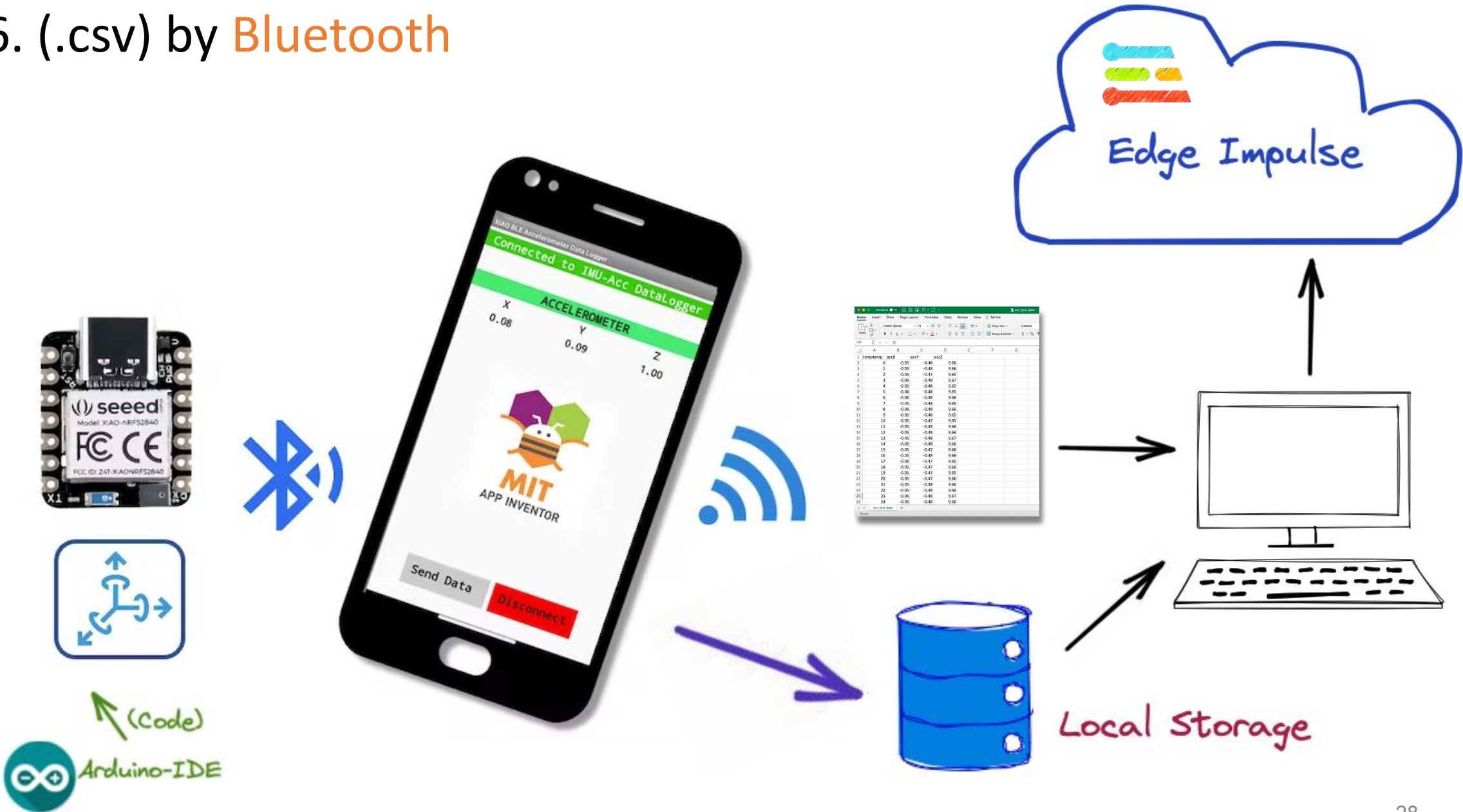
6. (.csv) by Bluetooth



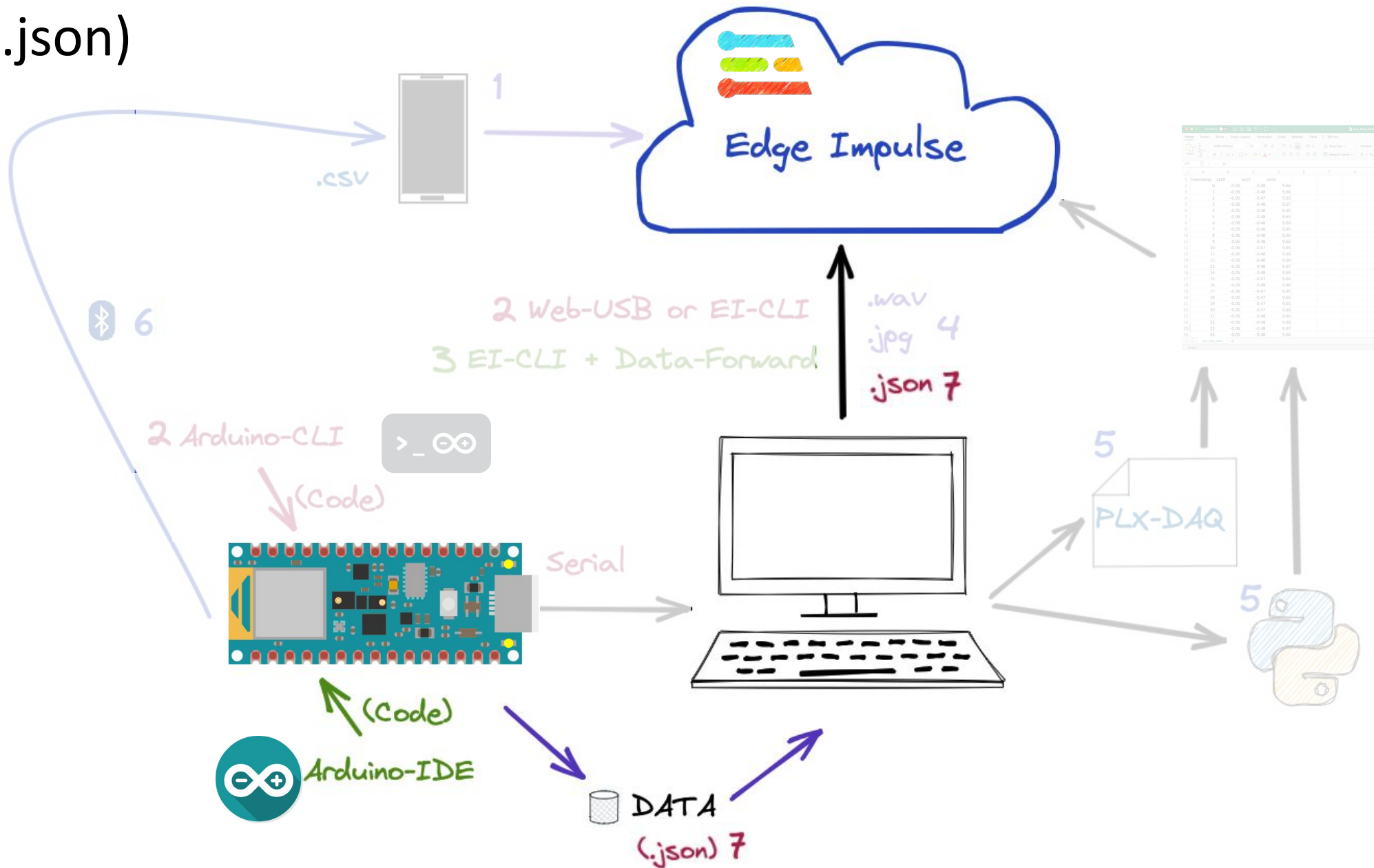
6. (.csv) by Bluetooth



6. (.csv) by Bluetooth



7. (.json)



7. Raw Uploader (.json files)

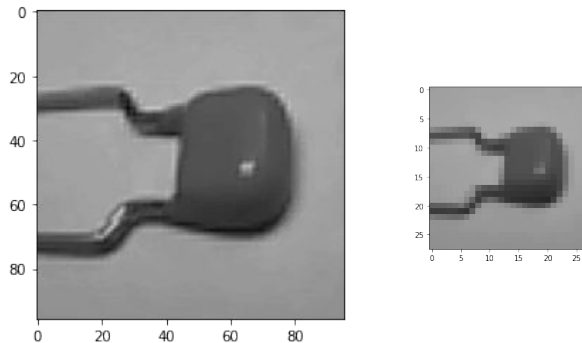
Image Classification: Raw Uploader



Run this notebook to convert images to a single row of raw, normalized values (between 0 and 1) and upload them to Edge Impulse as raw samples. Note that pixel values will be normalized to be between 0 and 1.

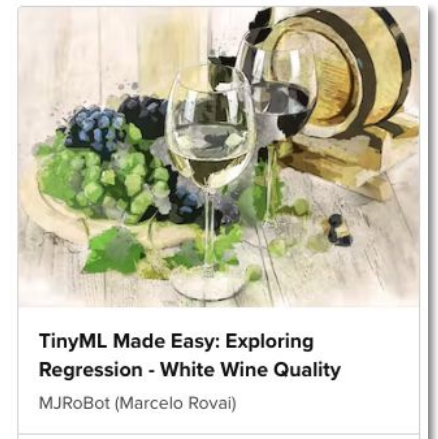
Create a folder named "dataset" in the /content directory and upload your images there. The images should be divided into their respective classes, where each class has its own folder with the name of the class. For example:

```
/content
|- dataset
  |- background
  |- capacitor
  |- diode
  |- led
  |- resistor
```

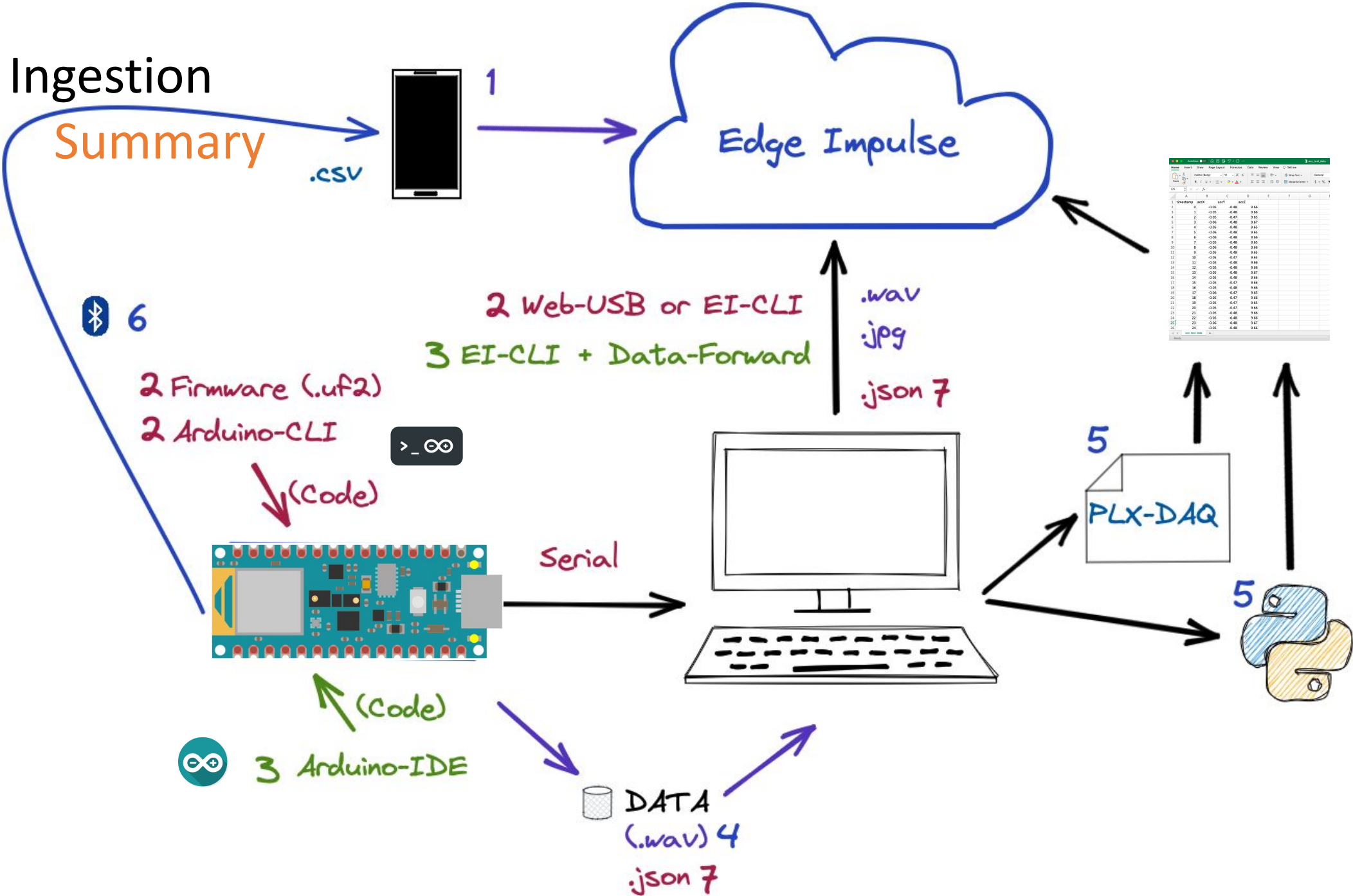


 **EDGE IMPULSE**
(Training as DNN)

Author: EdgeImpulse, Inc.
Date: June 6, 2021
License: [Apache-2.0](#)



Data Ingestion



Temperature Dependence Psychoacoustics

Simple **TinyML** Proof-of-concept



<https://www.hackster.io/mjrobot/listening-temperature-with-tinyml-7e1325>



Audio Engineering Society Convention e-Brief 473

Presented at the 145th Convention
2018 October 17 – 20, New York, NY, USA

This Engineering Brief was selected on the basis of a submitted synopsis. The author is solely responsible for its presentation, and the AES takes no responsibility for the contents. All rights reserved. Reproduction of this paper, or any portion thereof, is not permitted without direct permission from the Audio Engineering Society.

Why can you hear a difference between pouring hot and cold water? An investigation of temperature dependence in psychoacoustics.

He Peng¹ and Joshua D. Reiss²

¹Tianjin University

²Queen Mary University of London

Correspondence should be addressed to He Peng, Joshua Reiss (hepeng2018@hotmail.com, joshua.reiss@qmul.ac.uk)

[http://www.eecs.qmul.ac.uk/~josh/
documents/2018/19737.pdf](http://www.eecs.qmul.ac.uk/~josh/documents/2018/19737.pdf)

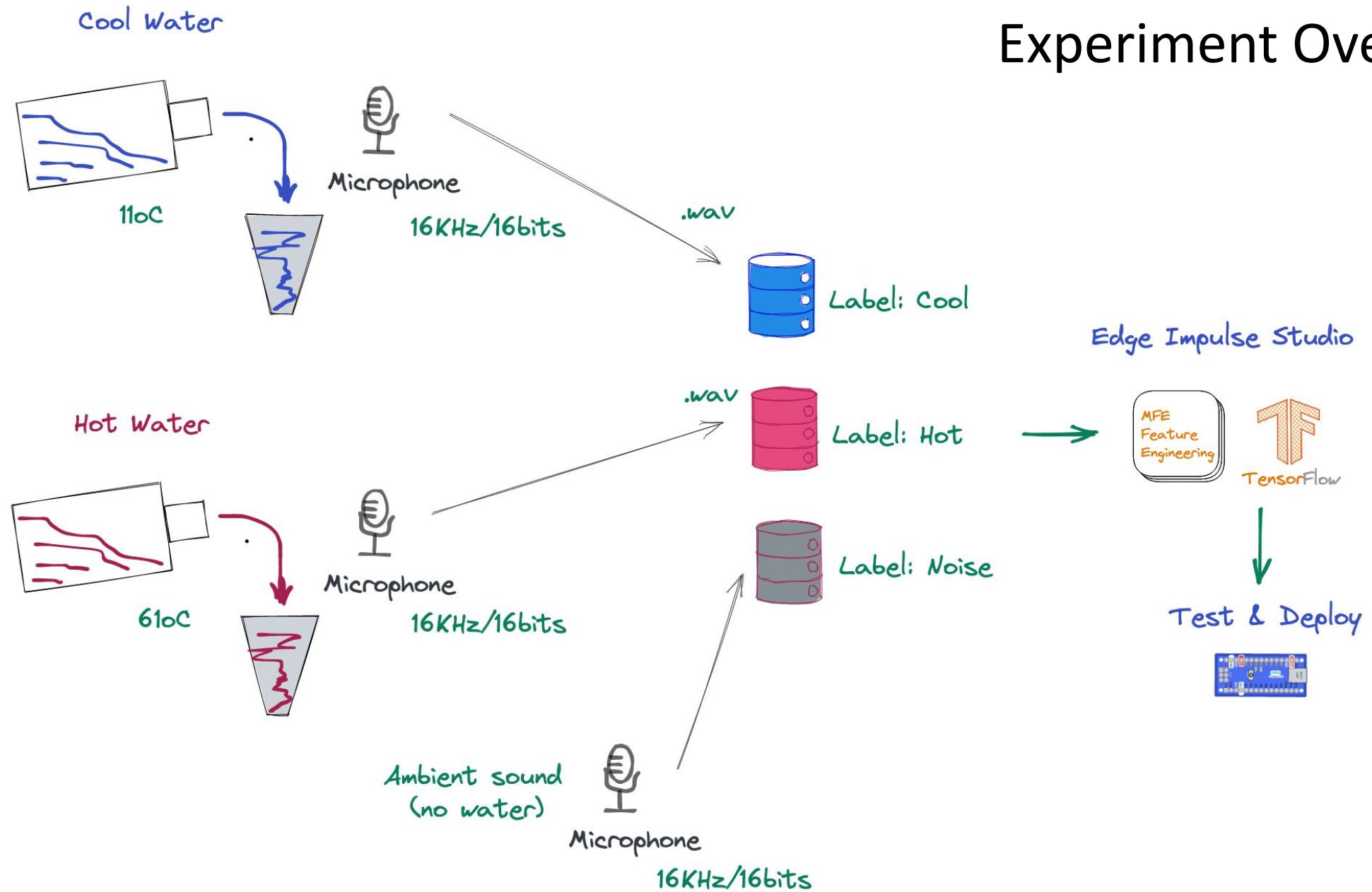


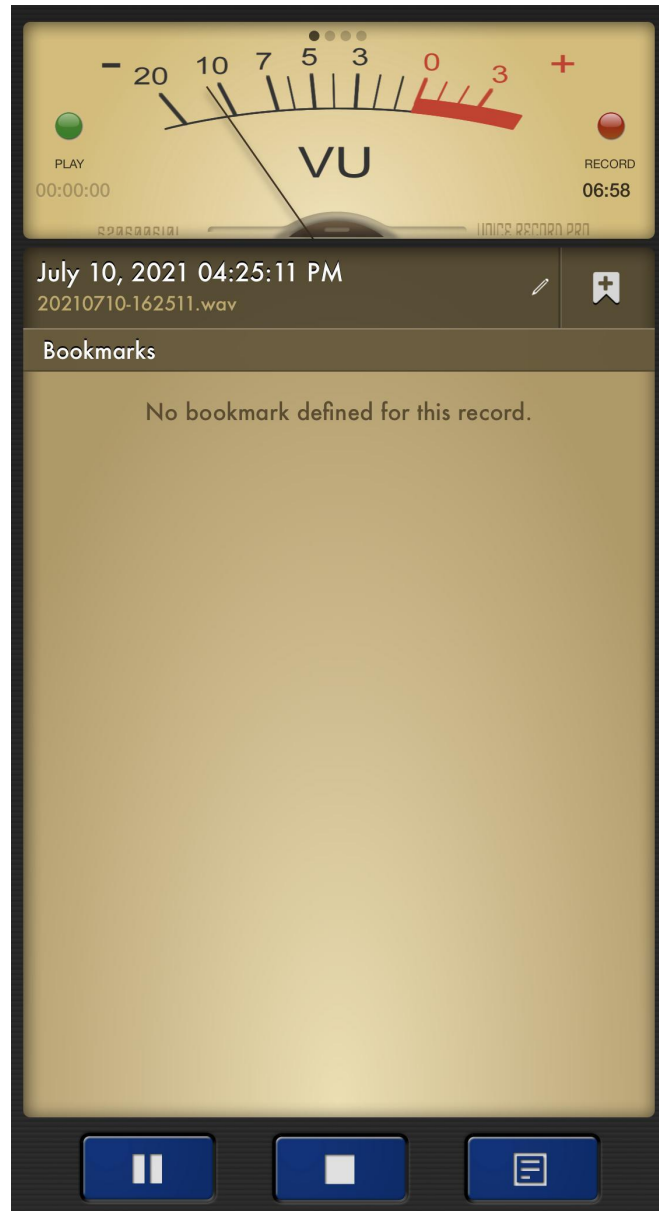
You Can Hear The Difference Between Hot and Cold W...

Tom Scott

https://www.youtube.com/watch?v=Ri_4dDvcZeM
(min: 0.17 => min 2:37)

Experiment Overview





Voice Recorder



Sample Sound:

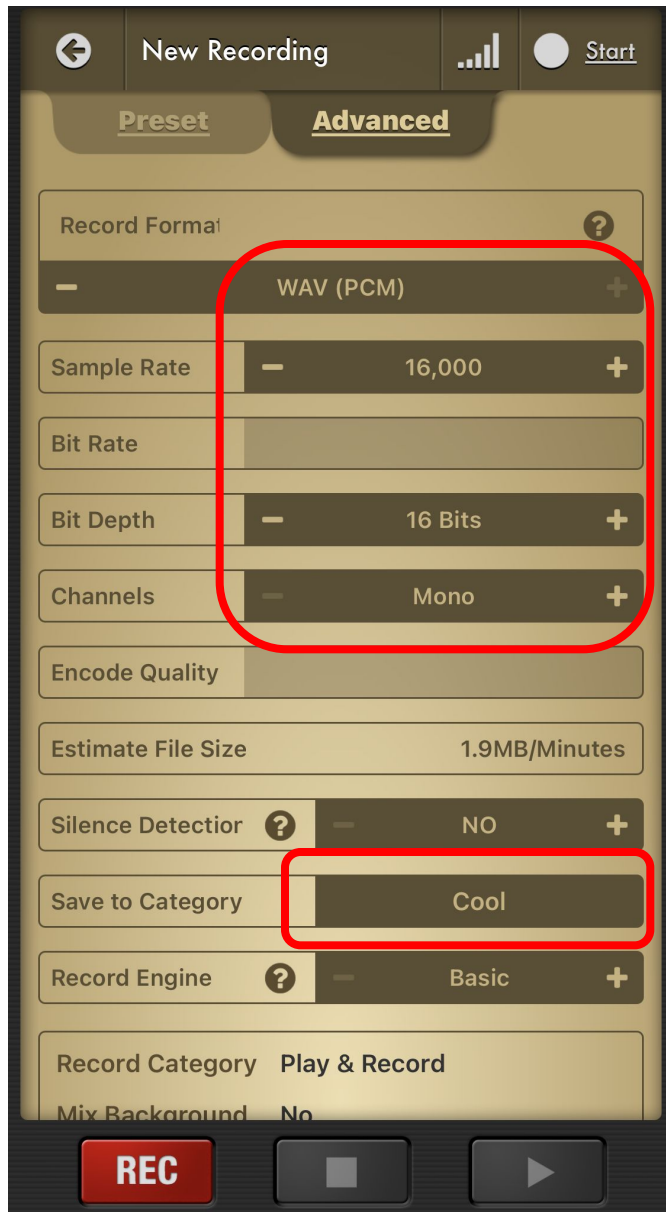
- 16KHz
- PCM – 16bits
- Mono

Classes:

- Hot
- Cool
- Noise



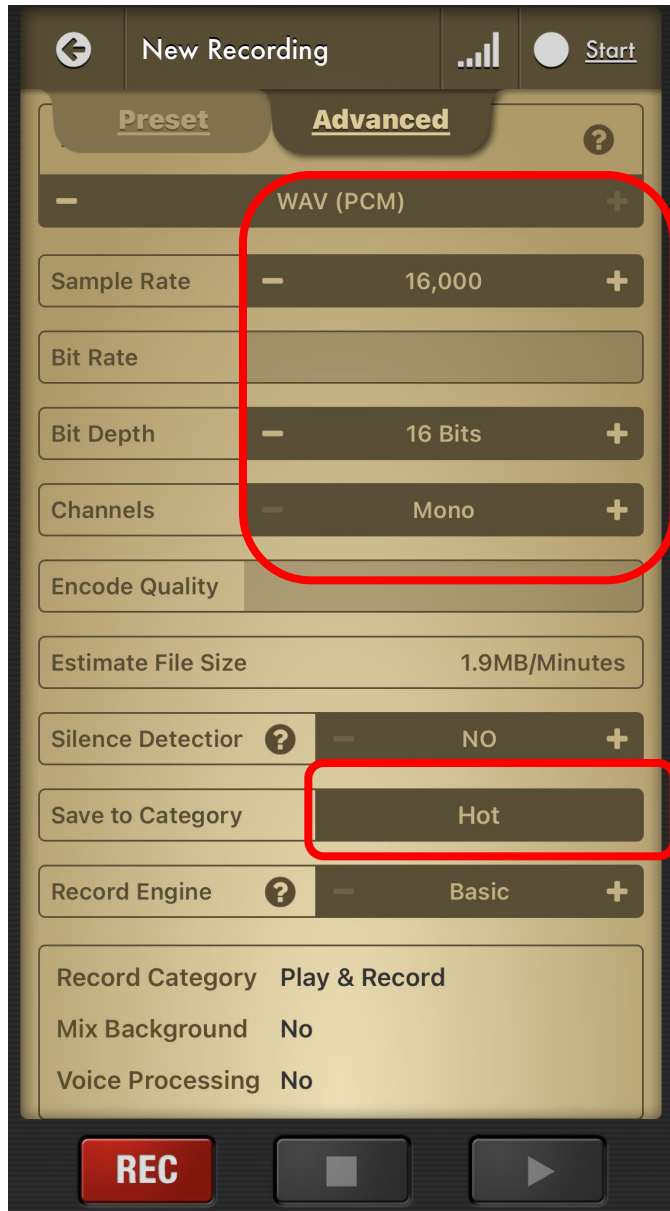
Ambient Temperature: 19°C



Class: Cool



Cool Water Temperature: 11°C



Class: Hot



Hot Water Temperature: 61°C

data

cool

hot

20210710-125854.wav

20210710-125930.wav

20210710-125956.wav

20210710-130010.wav

20210710-130041.wav

20210710-130100.wav

20210710-130119.wav

20210710-130129.wav

20210710-130142.wav

20210710-130200.wav

20210710-130212.wav

20210710-130224.wav

20210710-130236.wav

20210710-130246.wav

20210710-130256.wav

20210710-130304.wav

20210710-130315.wav

20210710-130329.wav

20210710-130348.wav

20210710-130358.wav

20210710-130408.wav

20210710-130416.wav



20210710-125854.wav

Waveform audio - 640 KB

Information

Show Less

Created

Today 13:58

Modified

Today 13:58

Duration

00:20

Audio channels

Mono

Sample rate

16 kHz

Bits per sample

16

Data captured using app Voice Recorder and uploaded to Computer



Upload existing data

You can upload existing data to your project in the [Data Acquisition Format](#) (CBOR, JSON, CSV), or as WAV, JPG or PNG files.

Select files

No file chosen

Upload into category

- ☐ Automatically split between training and testing ?
- ☒ Training
- ☐ Testing

Label

- ☐ Infer from filename ?
- ☒ Enter label:

hot

[Begin upload](#)

Upload output

Uploading 14 files...

```
[ 1/14] Uploading 20210710-130535.wav OK
[ 2/14] Uploading 20210710-130603.wav OK
[ 3/14] Uploading 20210710-130544.wav OK
[ 4/14] Uploading 20210710-130553.wav OK
[ 5/14] Uploading 20210710-130738.wav OK
[ 6/14] Uploading 20210710-130718.wav OK
[ 7/14] Uploading 20210710-130649.wav OK
[ 8/14] Uploading 20210710-130700.wav OK
[ 9/14] Uploading 20210710-130630.wav OK
[10/14] Uploading 20210710-130621.wav OK
[11/14] Uploading 20210710-130709.wav OK
[12/14] Uploading 20210710-130611.wav OK
[13/14] Uploading 20210710-130728.wav OK
[14/14] Uploading 20210710-130639.wav OK
```

Done. Files uploaded successful: 14. Files that failed to upload: 0.

Job completed

Raw Data uploaded to Edge Impulse Studio as .wav

Training data

Test data



Did you know? You can capture data from any device or development board, or upload your existing datasets - [Show options](#)



DATA COLLECTED

2m 49s



LABELS

2



Collected data



SAMPLE NAME	LABEL	ADDED	LENGTH	
20210710-130621.wav.2a5e0...	hot	Today, 14:02:55	4s	...
20210710-130630.wav.2a5e0...	hot	Today, 14:02:54	3s	...
20210710-130700.wav.2a5e0...	hot	Today, 14:02:52	4s	...
20210710-130649.wav.2a5e0...	hot	Today, 14:02:52	5s	...
20210710-130718.wav.2a5e0...	hot	Today, 14:02:52	5s	...
20210710-130738.wav.2a5e0...	hot	Today, 14:02:51	5s	...
20210710-130553.wav.2a5e0...	hot	Today, 14:02:51	4s	...
20210710-130544.wav.2a5e0...	hot	Today, 14:02:51	4s	...
20210710-130603.wav.2a5e0...	hot	Today, 14:02:51	3s	...
20210710-130535.wav.2a5e0...	hot	Today, 14:02:48	5s	...
20210710-130416.wav.2a5dv...	cool	Today, 14:02:12	4s	...
20210710-130408.wav.2a5dv...	cool	Today, 14:02:11	4s	...

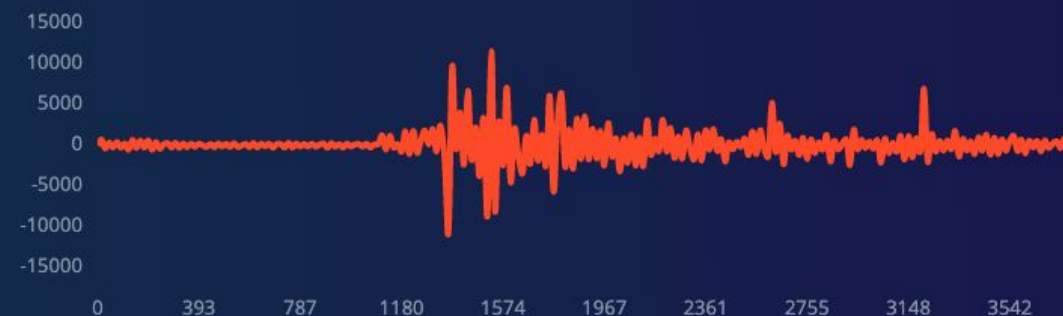
Record new data

[Connect using WebUSB](#)

No devices connected to the remote management API.

RAW DATA

20210710-130621.wav.2a5e0r33



audio

▶ 0:03 / 0:03



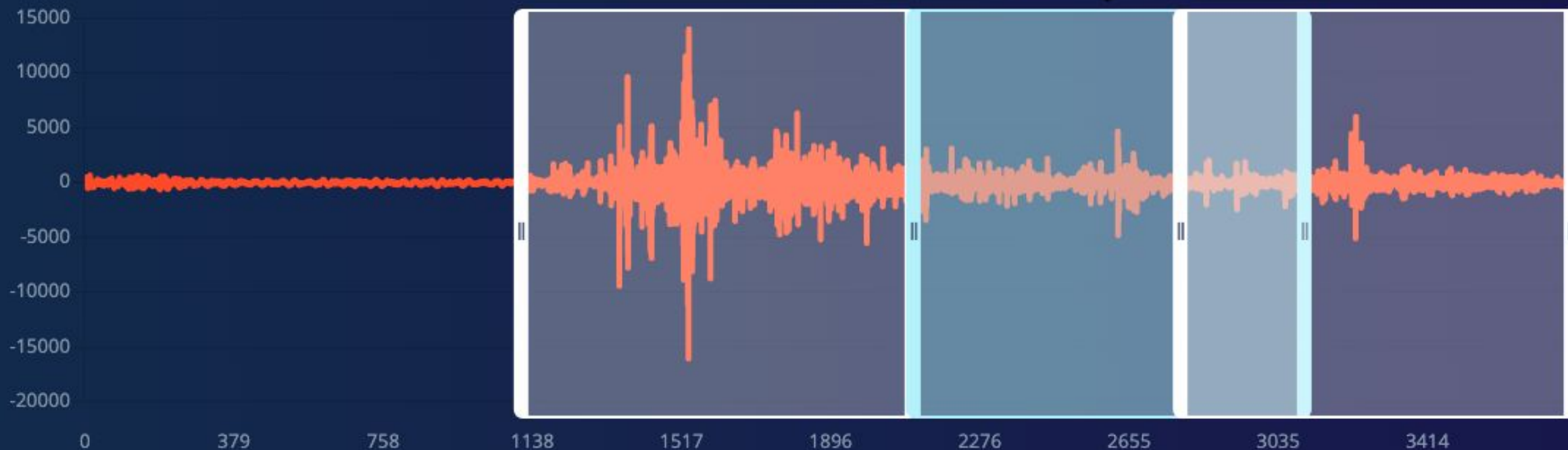
Raw Data cleaned as split in 1 second samples

+ Add Segment

Remove segment

(s.): 1000

Apply



0:01 / 0:01



Cancel

Raw Data cleaned as split in 1 second samples



Shift samples ?

Split



 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

audio

Window size



1000 ms.

Window increase



500 ms.

Zero-pad data



Audio (MFE)

Name

MFE

Input axes

☒ audio

Neural Network (Keras)

Name

NN Classifier

Input features

☒ MFE

Output features

3 (cool, hot, noise)

Output features

3 (cool, hot, noise)

Save Impulse



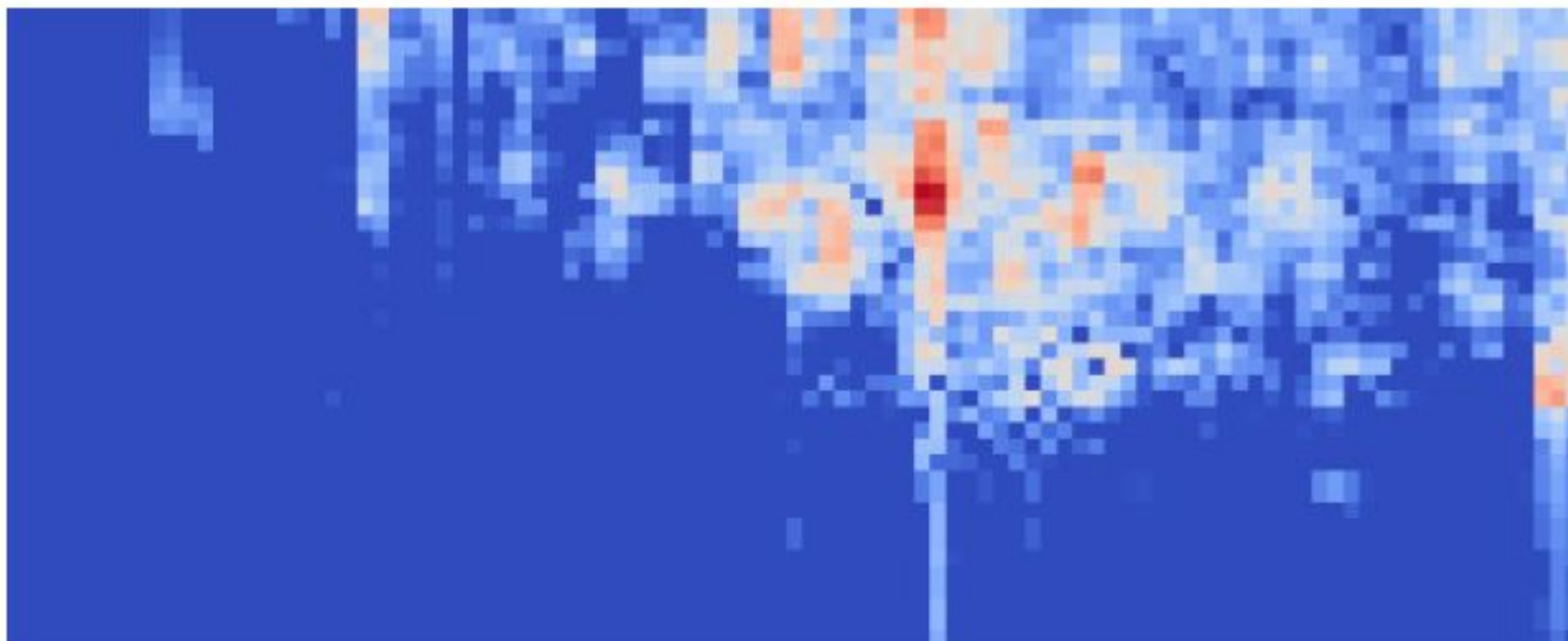
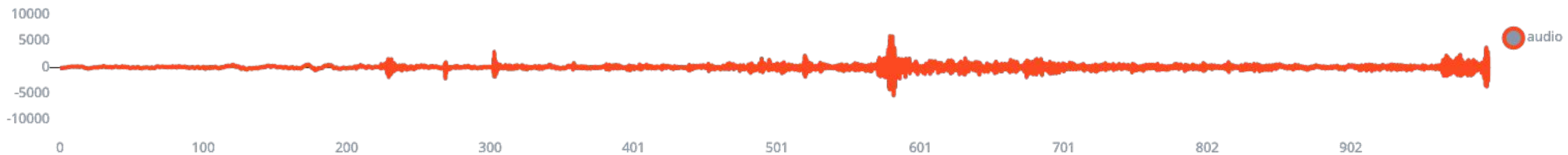
Add a processing block



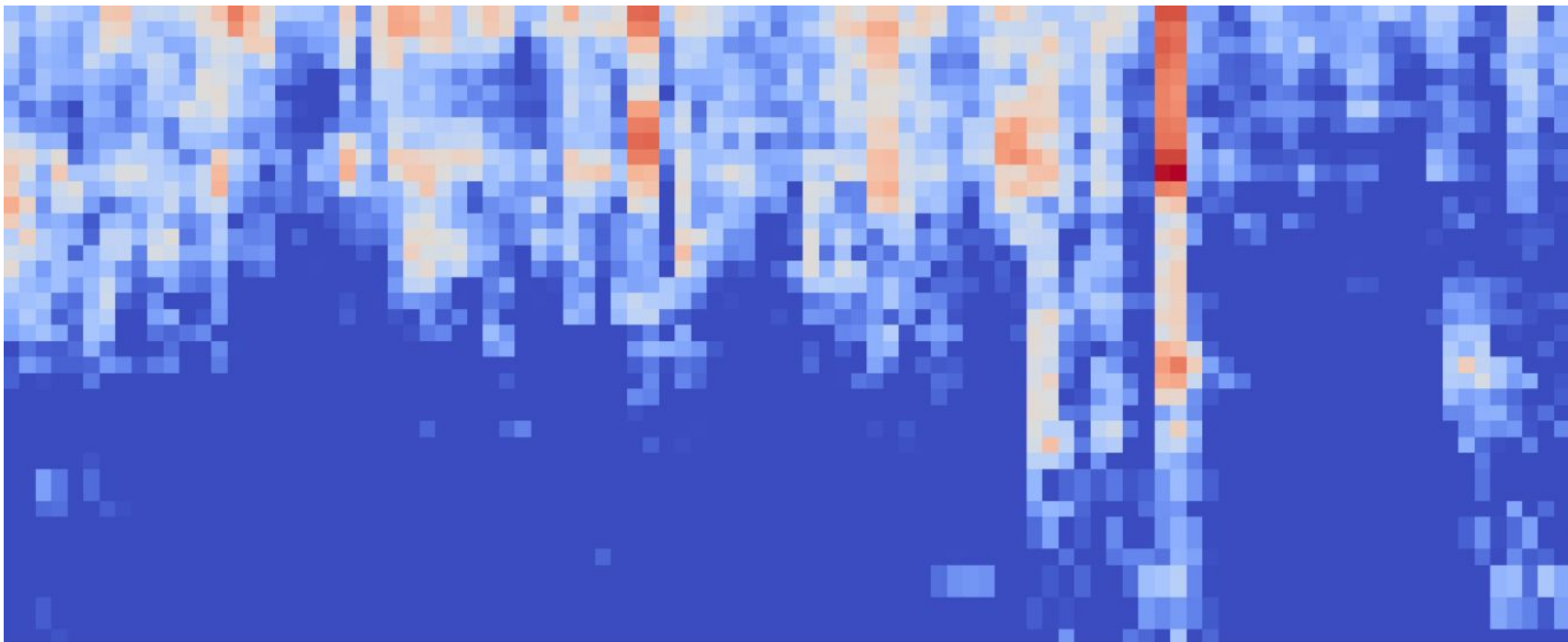
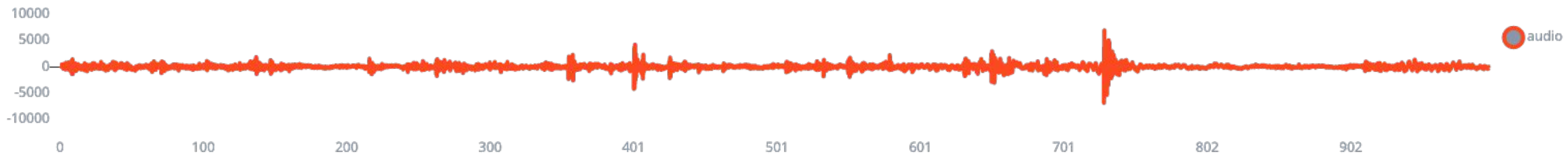
Add a learning block

Audio (MFE)
Extracts a spectrogram from audio signals using **Mel-filterbank energy features**, great for non-voice audio.

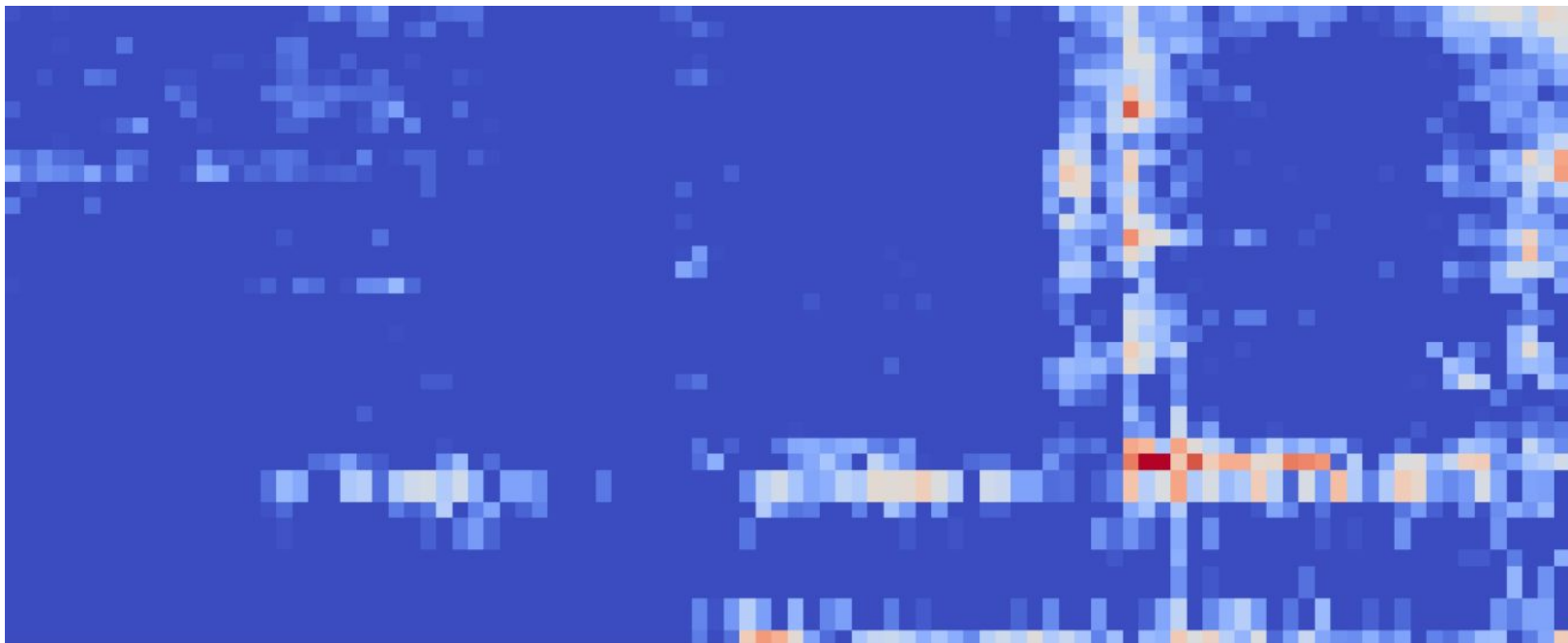
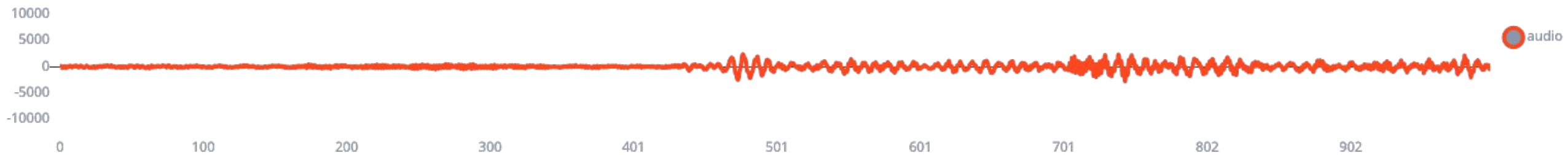
Cool Water 1 second sample



Hot Water 1 second sample



Noise 1 second sample





Parameters

Generate features

Training set

Data in training set	2m 35s
Classes	3 (cool, hot, noise)
Window length	1000 ms.
Window increase	500 ms.
Training windows	155

Generate features

Feature explorer (155 samples)



X Axis

Visualization layer 1

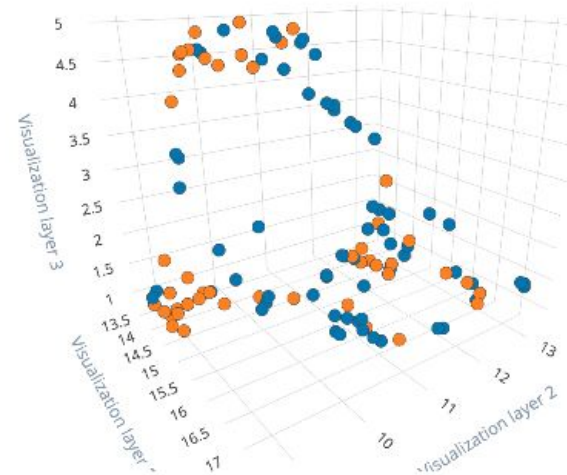
Y Axis

Visualization layer 2

Z Axis

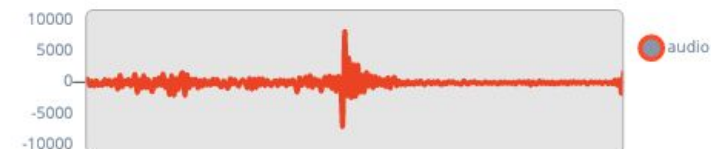
Visualization layer 3

- cool
- hot
- noise



20210710-125930.wav.2a5dv05j.s2

Label: cool

[View sample](#)[View features](#)

0:00 / 0:01



On-device performance ?



PROCESSING TIME

250 ms.



PEAK RAM USAGE

25 KB



#1 [Click to set a description for this version](#)

Neural Network settings

Training settings

Number of training cycles [?](#)

100

Learning rate [?](#)

0.005

Minimum confidence rating [?](#)

0.60

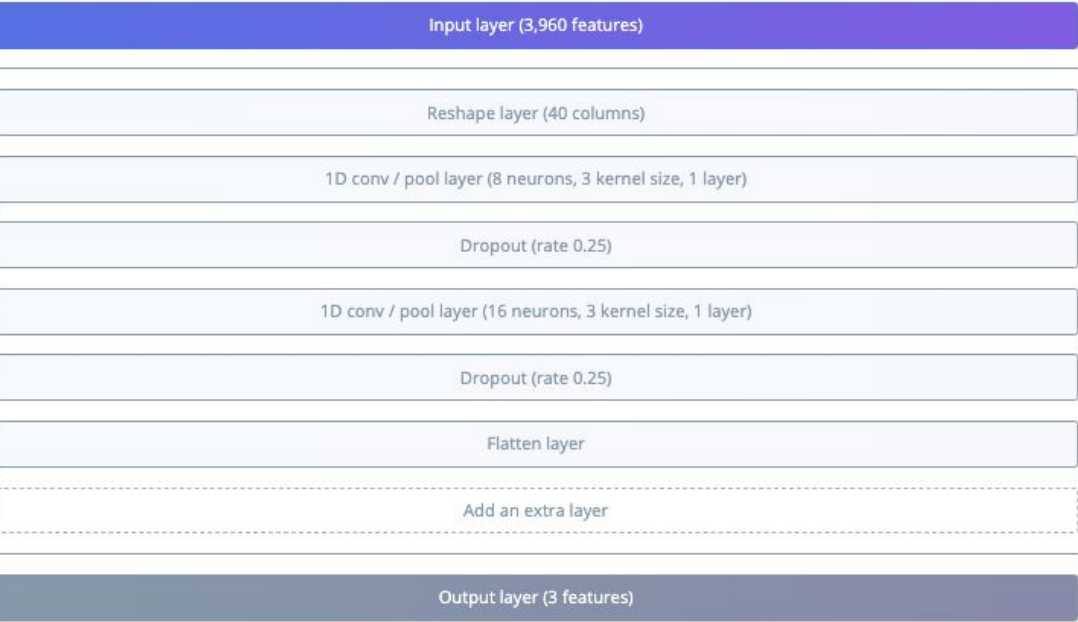
Audio training options

Data augmentation [?](#)

☐

Neural network architecture

Architecture presets [?](#) 1D Convolutional (Default) 2D Convolutional



Training output

Model

Model version: [?](#) [Quantized \(int8\)](#)

Last training performance (validation set)

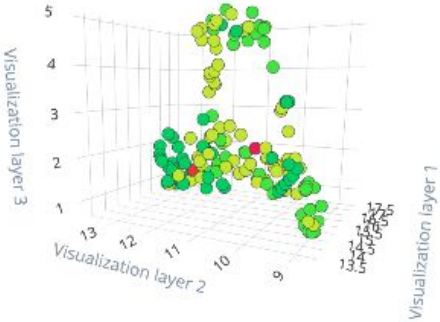


Confusion matrix (validation set)

	COOL	HOT	NOISE
COOL	92.9%	7.1%	0%
HOT	16.7%	83.3%	0%
NOISE	0%	0%	100%
F1 SCORE	0.93	0.83	1.00

Feature explorer (full training set) [?](#)

- cool - correct
- hot - correct
- noise - correct
- cool - incorrect
- hot - incorrect



On-device performance [?](#)





This lists all test data. You can manage this data through [Data acquisition](#).

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	
20210710-130728.wa...	hot	1s	100%	1 hot	⋮
20210710-130639.wa...	hot	1s	100%	1 hot	⋮
20210710-130553.wa...	hot	1s	100%	1 hot	⋮
20210710-130535.wa...	hot	1s	100%	1 hot	⋮
20210710-130535.wa...	hot	1s	100%	1 hot	⋮
20210710-130224.wa...	cool	1s	0%	1 noise	⋮
20210710-130304.wa...	cool	1s	100%	1 cool	⋮
20210710-130236.wa...	cool	1s	100%	1 cool	⋮
20210710-130256.wa...	cool	1s	100%	1 cool	⋮
20210710-130224.wa...	cool	1s	100%	1 cool	⋮
20210710-130142.wa...	cool	1s	0%	1 noise	⋮
20210710-130100.wa...	noise	1s	100%	1 noise	⋮
20210710-130041.wa...	noise	1s	100%	1 noise	⋮
20210710-125854.wa...	noise	1s	100%	1 noise	⋮
20210710-125854.wa...	noise	1s	100%	1 noise	⋮

Model testing output

Model testing results

ACCURACY

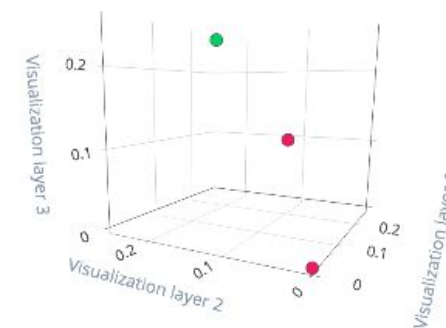
86.67%

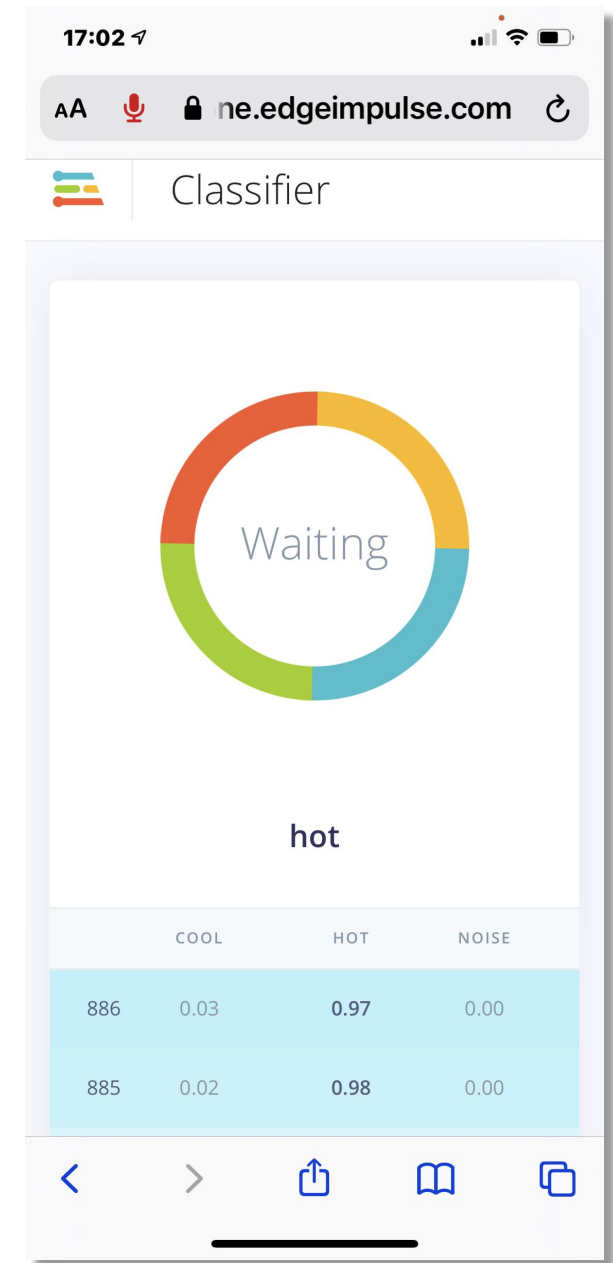
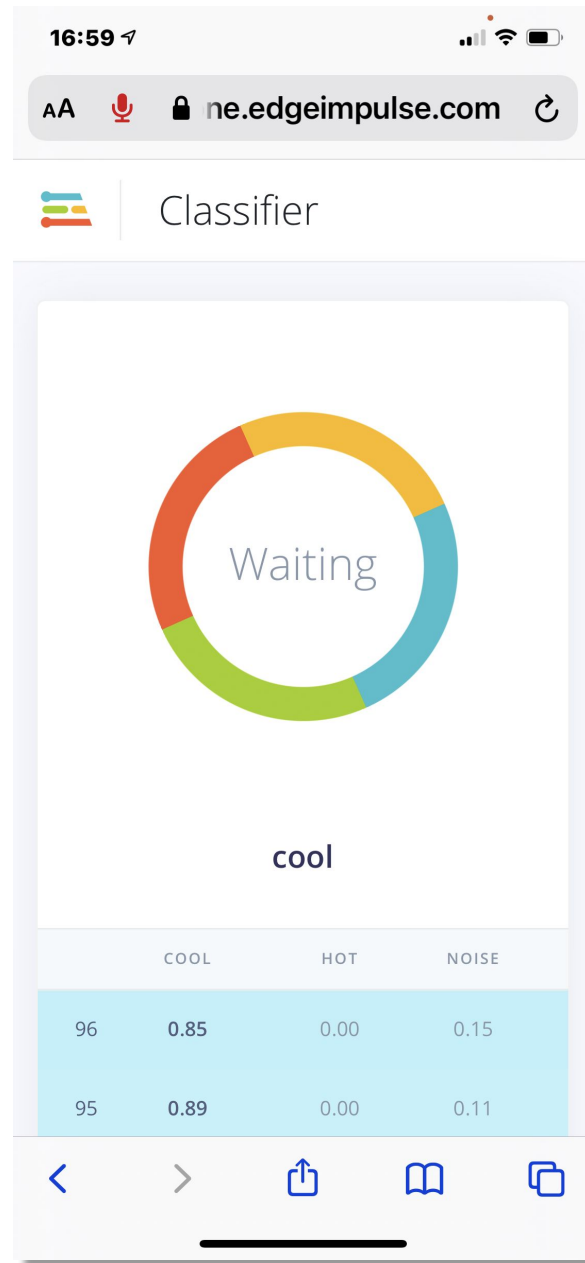
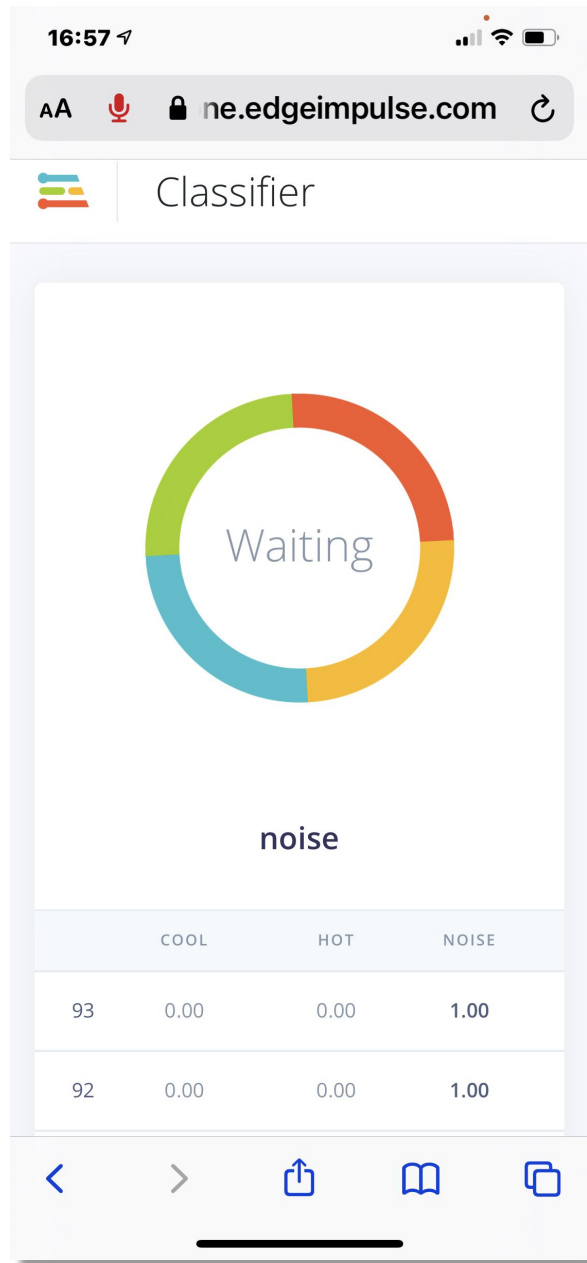


	COOL	HOT	NOISE	UNCERTAIN
COOL	66.7%	0%	33.3%	0%
HOT	0%	100%	0%	0%
NOISE	0%	0%	100%	0%

Feature explorer

- cool - correct
- hot - correct
- noise - correct
- cool - incorrect





Live Classifier (Off line) using iphone

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) ★ Currently selected This optimization is recommended for best performance.	RAM USAGE 10.9K FLASH USAGE 31.4K	LATENCY 17 ms ACCURACY 86.67%	CONFUSION MATRIX ? <table><tr><td>66.7</td><td>0</td><td>33.3</td><td>0</td></tr><tr><td>0</td><td>100</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>100</td><td>0</td></tr></table>	66.7	0	33.3	0	0	100	0	0	0	0	100	0
66.7	0	33.3	0												
0	100	0	0												
0	0	100	0												
Unoptimized (float32) Click to select	RAM USAGE 33.9K FLASH USAGE 38.0K	LATENCY 78 ms ACCURACY 86.67%	CONFUSION MATRIX ? <table><tr><td>66.7</td><td>0</td><td>33.3</td><td>0</td></tr><tr><td>0</td><td>100</td><td>0</td><td>0</td></tr><tr><td>0</td><td>0</td><td>100</td><td>0</td></tr></table>	66.7	0	33.3	0	0	100	0	0	0	0	100	0
66.7	0	33.3	0												
0	100	0	0												
0	0	100	0												

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

/dev/cu.usbmodem144301

Send

ICTP - PSYCOACOUSTICS TEMPERATURE Project

Inferencing settings:

Interval: 0.06 ms.

Frame size: 16000

Sample length: 1000 ms.

No. of classes: 3

Predictions (DSP: 126 ms., Classification: 21 ms., Anomaly: 0 ms.):

:

PREDICTION: ==> noise with probability 1.00

:

Predictions (DSP: 126 ms., Classification: 21 ms., Anomaly: 0 ms.):

:

PREDICTION: ==> noise with probability 1.00

:

Predictions (DSP: 126 ms., Classification: 20 ms., Anomaly: 0 ms.):

:

PREDICTION: ==> noise with probability 1.00

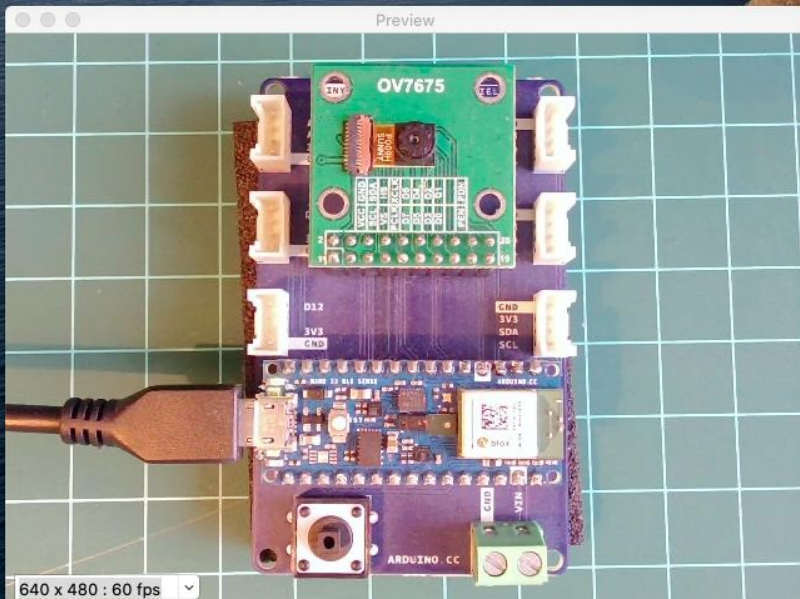
☐ Autoscroll ☐ Show timestamp

Both NL & CR

115200 baud

Clear output

Preview



640 x 480 : 60 fps

nano_ble33_sense_microphone_continuous_LED

```

88 // **
89 * @brief      Special Postprocess function for RGB LEDs
90 * /
91
92 void turn_off_leds(){
93     digitalWrite(LED_R, HIGH);
94     digitalWrite(LED_G, HIGH);
95     digitalWrite(LED_B, HIGH);
96 }
97
98 // *
99 * cool:  [0] ==> Blue ON
100 * hot:   [1] ==> Red ON
101 * noise: [2] ==> ALL OFF
102 * /
103
104 void turn_on_leds(int pred_index) {
105     switch (pred_index)
106     {
107     case 0:
108         turn_off_leds();
109         digitalWrite(LED_B, LOW);
110         break;
111
112     case 1:
113         turn_off_leds();
114         digitalWrite(LED_R, LOW);
115         break;
116
117     case 2:
118         turn_off_leds();
119         break;
120     }
121 }
122

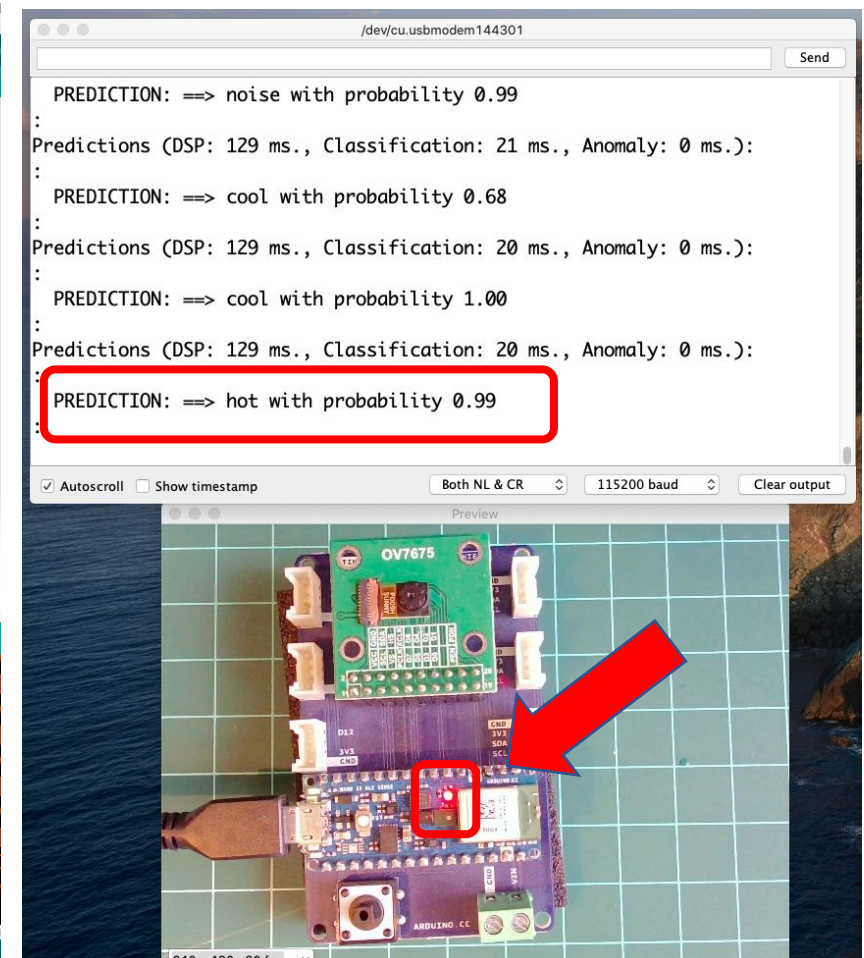
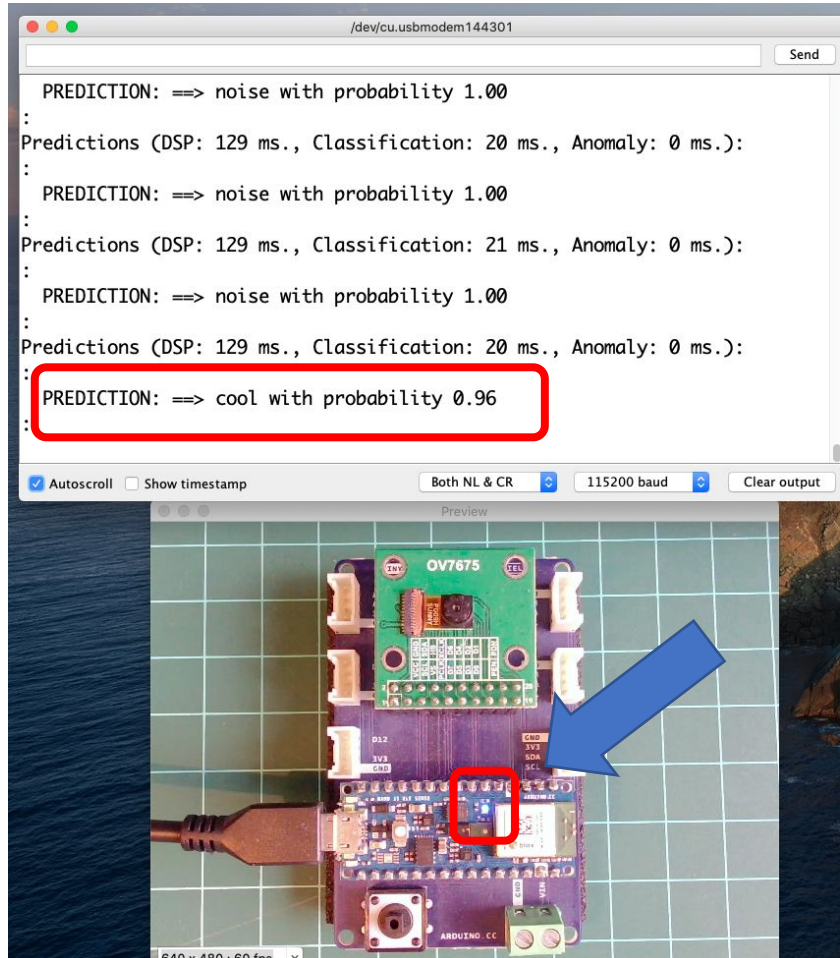
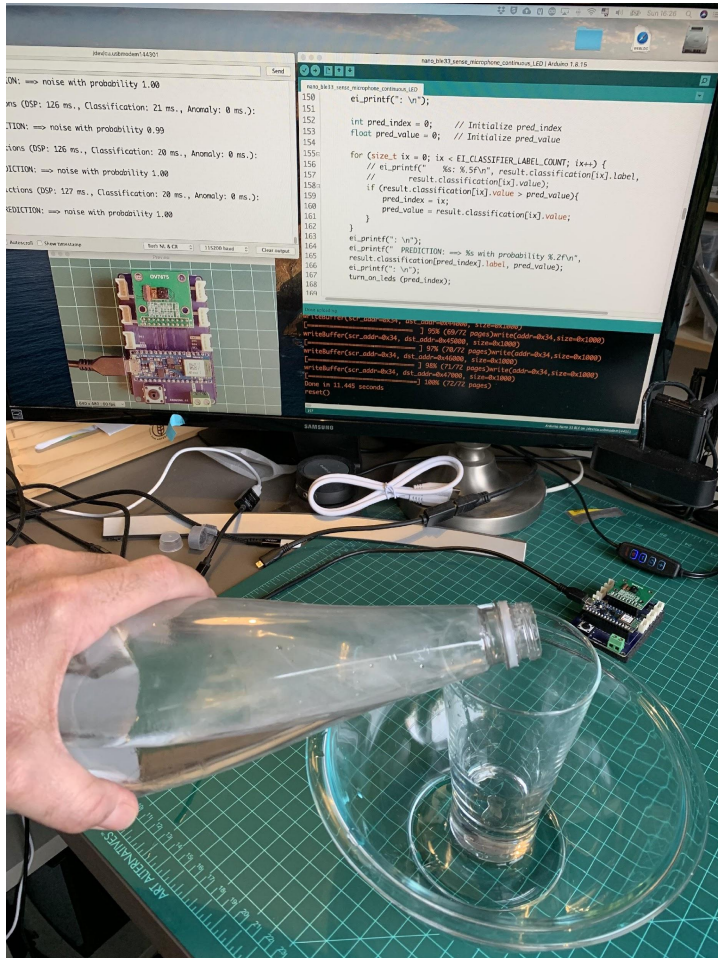
```

Done uploading.

```

writeBuffer(scr_addr=0x34, dst_addr=0x44000, size=0x1000)
[===== ] 95% (69/72 pages)write(addr=0x34,size=0x1000)
writeBuffer(scr_addr=0x34, dst_addr=0x45000, size=0x1000)
[===== ] 97% (70/72 pages)write(addr=0x34,size=0x1000)

```



Reading Material

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/Edge Impulse](#)
- [Computer Vision with Embedded Machine Learning - Coursera/Edge Impulse](#)
- Fundamentals textbook: [“Deep Learning with Python” by François Chollet](#)
- Applications & Deploy textbook: [“TinyML” by Pete Warden, Daniel Situnayake](#)
- Deploy textbook [“TinyML Cookbook” by Gian Marco Iodice](#)

I want to thank **Shawn Hymel** and Edge Impulse, **Pete Warden** and **Laurence Moroney** from Google, Professor **Vijay Janapa Reddi** and **Brian Plancher** from Harvard, and the rest of the **TinyMLedu** team 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



UNIFEI