Embedded Online Conference

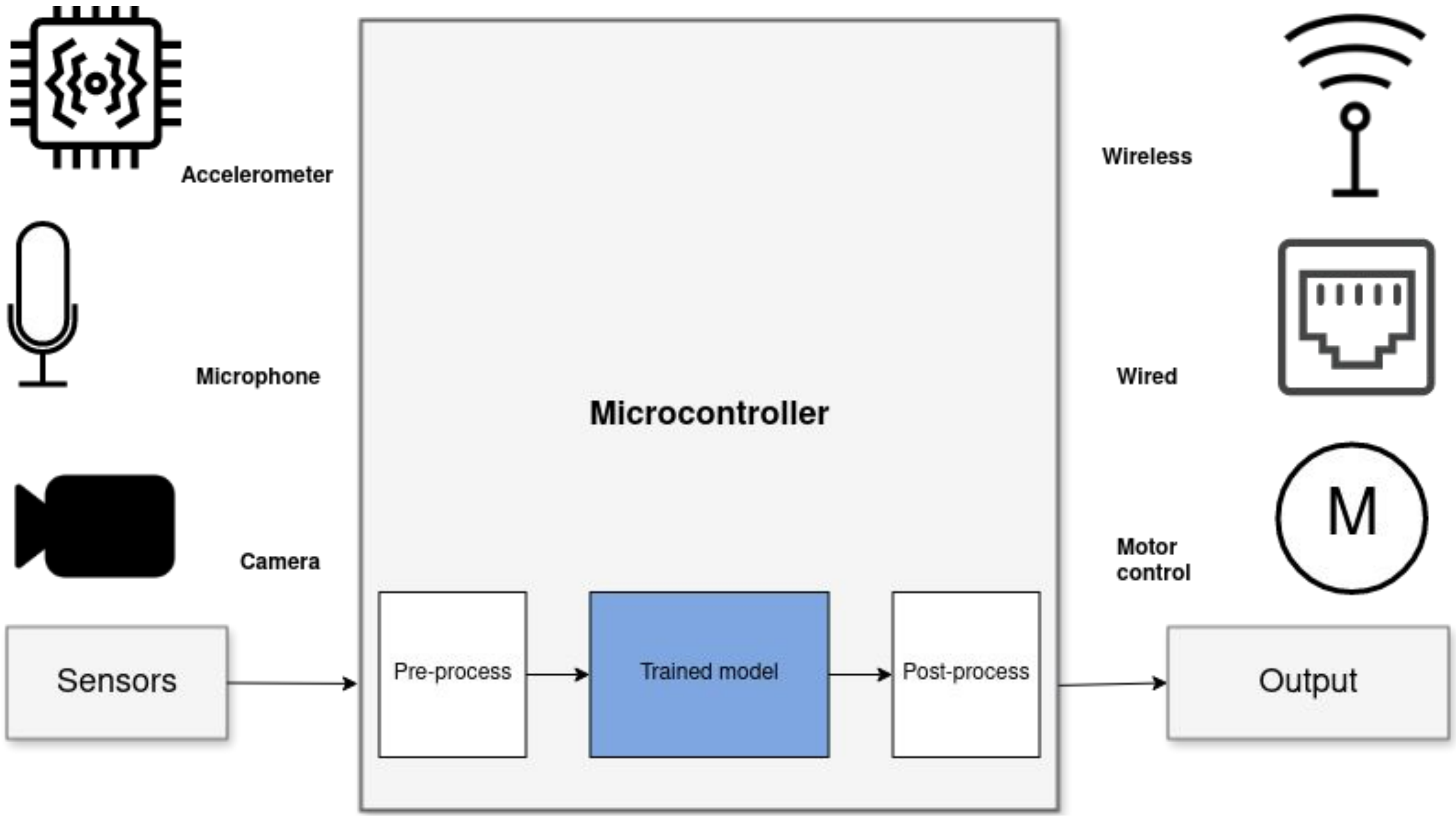
www.embeddedonlineconference.com

emlearn

Machine Learning for Microcontrollers

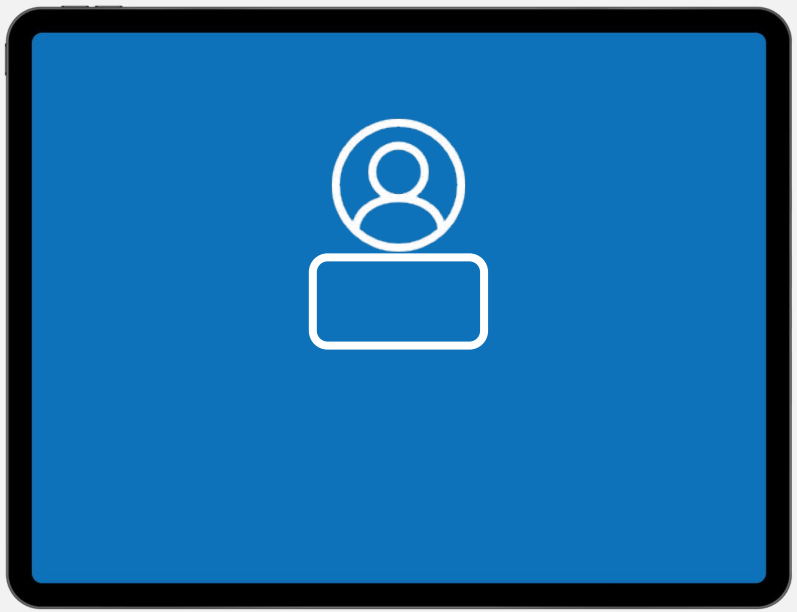
Jon Nordby, Soundsensing

Sensor data analysis with Machine Learning



Benefits

- Stand-alone
- Low latency
- Power efficiency
- Privacy
- Easy to consume



Example project

Cattle activity tracking using accelerometer

Activity data used to **detect abnormal behavior**, can indicate **health issues** or other problems

Classified using a **Decision Tree**

lying/walking/standing/grazing/ruminating

Activity is transmitted using **LoRaWAN**

Running ML on-sensor: **under 1 mW**
50 times lower power than sending raw data

Power Efficient Wireless Sensor Node through Edge Intelligence
Abhishek Damle (2022)

emlearn - Machine Learning for microcontrollers

https://github.com/emlearn/emlearn/

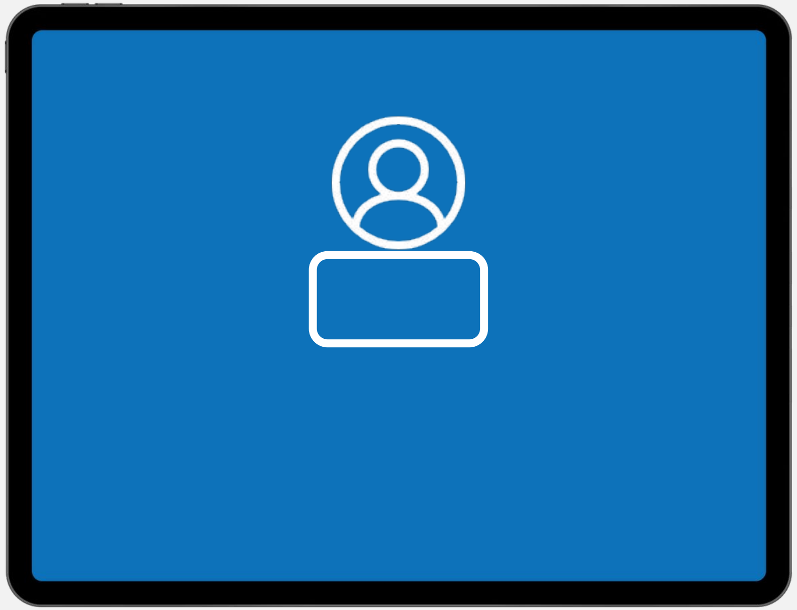
Convenient training

- Model creation **in Python**
- **Use standard libraries**
  a. scikit-learn
  b. Keras
- One-line to **export to C**
- Verification tools included

Embedded Friendly

- **Portable C99** code
- No dynamic allocations
- **Header-only**
- High **test coverage**
- Integer/**fixed-point** math *
- **libc optional** *
- **Small**. 1 kB+ FLASH

* Only some models



Supported tasks & models

Supports the most common tasks

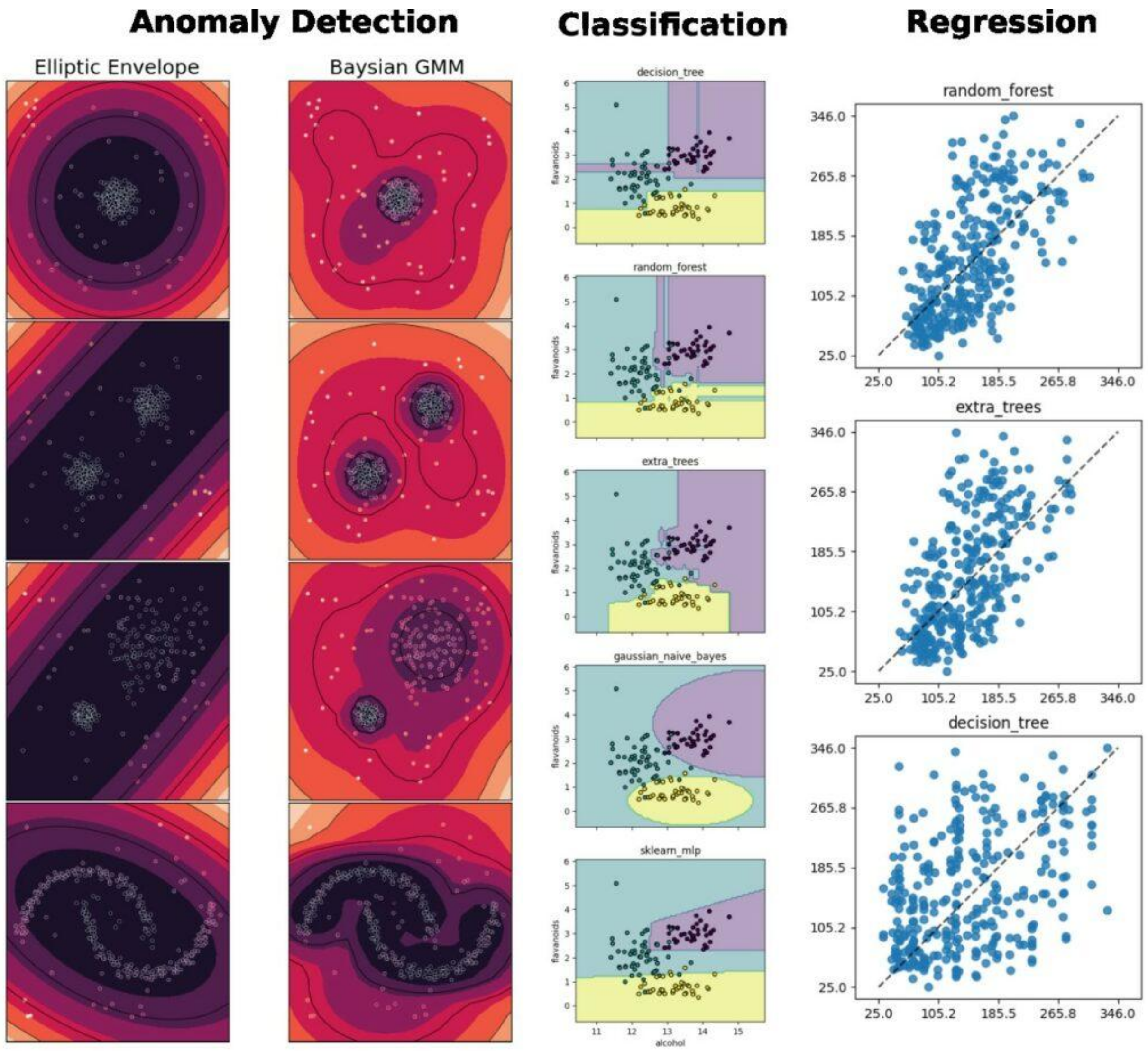
Selection of simple & effective embedded-friendly models

"Classic" ML

- Decision Trees (DT)
- Random Forest (RF)
- K Nearest Neighbors (KNN)
- Gaussian Mixture Models (GMM)

Neural networks

- Multi-Layer-Perceptron (ML)



How to use emlearn

0. Install the library
1. Train model in Python
2. Convert model to C code
3. Use the C code

Install as Python package (PyPI)

pip install emlearn

Install as git submodule

git submodule add https://github.com/emlearn/emlearn/

Training a model

Using standard Python ML libraries.

```
from keras import …

model = Sequential([
    Dense(16, input_dim=n_features, activation='relu'),
    Dense(8, activation='relu'),
    Dense(1, activation='sigmoid'),
])
model.compile(....)

model.fit(X_train, Y_train, epochs=1, batch_size=10)
```

A) keras neural network

```
from sklearn.ensemble import RandomForestClassifier
model = RandomForestClassifier(n_estimators=10, max_depth=10)

model.fit(X_train, Y_train)
```

B) scikit-learn Random Forest

```
from sklearn.neural_network import MLPClassifier
model = MLPClassifier(hidden_layer_sizes=(100,50,25))

model.fit(X_train, Y_train)
```

C) scikit-learn neural network

Convert model to C

Using **emlearn.convert()** and **.save()**

Takes the trained **model** from previous step

```
import emlearn

cmodel = emlearn.convert(model, method='inline')

cmodel.save(file=mynet_model.h', name='mynet')
```

```
#include <eml_net.h>

static const float mynet_layer_0_biases[8] = { -0.015587f, -0.005395f, -0.010957f, 0.015883f ....
static const float mynet_layer_0_weights[24] = { -0.256981f, 0.041887f, 0.063659f, 0.011013f, ...
static const float mynet_layer_1_biases[4] = { 0.001242f, 0.010440f, -0.005309f, -0.006540f };
static const float mynet_layer_1_weights[32] = { -0.577215f, -0.674633f, -0.376140f, 0.646900f, ...
static float mynet_buf1[8];
static float mynet_buf2[8];
static const EmlNetLayer mynet_layers[2] = {
{ 8, 3, mynet_layer_0_weights, mynet_layer_0_biases, EmlNetActivationRelu },
{ 4, 8, mynet_layer_1_weights, mynet_layer_1_biases, EmlNetActivationSoftmax }
};
static EmlNet mynet = { 2, mynet_layers, mynet_buf1, mynet_buf2, 8 };

   int32_t
   mynet_predict(const float *features, int32_t n_features)
   {
      return eml_net_predict(&mynet, features, n_features);
   }
.........
```

Example of generated code (neural network)

Using the C code

#include and call predict()

```
// Include the generated model code
#include "mynet_model.h"

// index for the class we are detecting
#define MYNET_VOICE 1

// Buffers for input data
#define N_FEATURES 6
float features[N_FEATURES];

#define DATA_LENGTH 128
int16_t sensor_data[DATA_LENGTH];
```

setup

```
// Get data and pre-process it
read_microphone(sensor_data,
DATA_LENGTH);
preprocess_data(sensor_data, features);

// Run the model
out = mynet_predict(features, N_FEATURES);

// Do something with results
if (out == MYNET_VOICE) {
    set_display("voice detected");
} else {
    set_display("");
}
```

loop

Summary

1. Machine Learning is useful on microcontrollers
Ex: extract information from sensors

2. emlearn is an open-source project with
embedded-friendly ML algorithms

3. Makes it easy to deploy
by generating portable C code

More information: https://emlearn.org

THANK YOU

Embedded
Online
Conference

www.embeddedonlineconference.com

Embedded Online Conference

www.embeddedonlineconference.com

1 dollar TinyML system

Goal: Sound and accelerometer sensing using ML for under 1 USD in total component cost (BOM)
https://hackaday.io/project/194511-1-dollar-tinyml
Challenge: 3 kB RAM / 32 kB FLASH