hpssjellis Update ictp2023presentation.md

1 contributor

ICTP Presentation July 2023 by Jeremy Ellis about tinyMLjs

Version 0.2.0-7

qrcode

Next Page

Finding the fun, while teaching machine learning with microcontrollers to the general population.

QR Code is for this link https://hpssjellis.github.io/tinyMLjs/public/acceleration/a00-best-acceleration.html To the main acceleration TinyMLjs webpage

TinyMLjs by Jeremy Ellis. My Github Profile at https://github.com/hpssjellis

394 lines (260 sloc) | 10.2 KB

Introduction

Next Page

1. When I heard of the ICTP Workshop on Widening Access to TinyML Network by Establishing Best Practices in Education, I was eager to

connect with like-minded educators who could collaborate on creating an engaging method for teaching machine learning on microcontrollers to a wider audience.

2. By "fun," I mean an approach that is open-source, powerful, fosters a passion for learning, enables building proof of concepts, affordable, fast, user-friendly, operates on the client-side, ensures security, covers the entire process, supports future edge devices, is hardware and internet/cloud independent.

About-Me

Next Page

1. I am Jeremy Ellis, known online as @rocksetta, jerteach, or hpssjellis.

As an unconventional learner, I am self-taught in machine learning but probably have no chance of a PhD. My strength lies in simplifying technology.

2. Around 2017 I made a machine learning curriculum based on TensorflowJS, but deprecated it when I found out about EdgeImpulse.com

3. My Robotics course is called Maker100 based on the Arduino PortentaH7 with LoRa vision Shield and the corresponding PortentaH7 library is called the Portenta Pro Community Solutions with over 100 of my examples relevant to my course.

4. With 48 years of computer programming experience, 35 years of teaching high school coding, 30 years of obsession with coding neural networks, and 8 years of teaching robotics and TensorflowJS on microcontrollers (the last 3 using EdgeImpulse.com), the only constant in my journey has been the deprecation of my work.

5. The only consistent thing about teaching coding for that many years is the amount of times all my work has been deprecated! It doesn't matter if the cloud platform has been sold: (Cloud9 to AWS) or the software has been updated: (Too many to mention), or the IDE has changed (Arduino IDE 1.8.19 to 2.10), the board has changed (too many to mention) the software has changed (Python 2 to Python 3), and each deprecation destroys any relevant lesson plans or videos.

6. One of the methods that has been reasonably stable is Javascript, mainly because it's script tag can be versioned

   <script src="https://cdn.jsdelivr.net/npm/@tensorflow/tfjs@4.8.0">
   </script>

7. However, converting TensorflowJS machine learning to microcontrollers has been a challenge. Although I managed to

accomplish it years ago here (though it may be deprecated now), it may no longer be necessary. If data from any hardware can be saved as a CSV file, it can later be loaded onto any cloud platform.

Progress

Next Page

1. At the start of June 2023 I had successfully implemented desktop webSerial/polyfill (also works on Android Pixel Phones) and achieved functionality for saving microcontroller data to CSV files. I also experimented with using coPilot, chatGPT, and BingChat.

2. In summary, I accomplished in a much shorter timeframe what I initially believed would take a year. On a webpage, we can now load raw CSV data or data directly from a connected microcontroller using the `println()` command to a desktop or laptop computer. We can clean the data if necessary, convert it to a tensor, train a machine learning model, load more data, clean and classify it, and finally send the classification results back to the microcontroller (e.g., turning on an LED, etc).

3. All this functionality is encapsulated in approximately 1000 lines of

code on a single webpage, allowing for quick testing of machine learning viability with any sensor data from any microcontroller.

4. Since CSV files can be saved, you can now load that data onto your favorite cloud platform, such as EdgeImpulse.com, for microcontroller programming.

tinyMLjs-csv

Next Page

Let's have a look at sections of the webpage:

Click `Choose Files` to select CSV files. Currently, the file name is important, and there are no column headings—just raw, cleaned data.

Version 0.36.7-154 Note: works on Chrome or Edge for Mac, Android (pixel Phones) or Windows, only on Edge for Linux

tinyMLjs

Upload from a raw CSV file or an Arduino style microcontroller using webSerial (Android Pixel phones also work) or your cell phone motion sensor. Keep the raw sensor data then Machine Learning train a tensorflowJS model for export or for live classification all on this single vanilla Javascript webpage!

Show: ---------- Hide: ---------- Choose Files No file chosen
Following is the list of actual labels used in the same order as uploaded (comma-separated) Note: expecting files to be named: "name-lable.csv" or "name-lable (1).csv" etc.

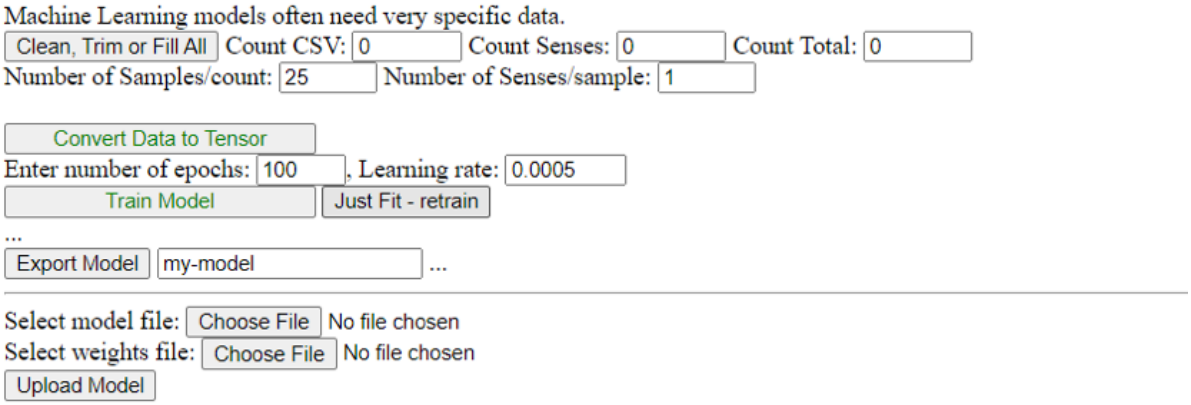
CSV Lables (careful):
Senses Labels (careful):

tinyMLjs-tensor

Next Page

Information here about number of samples, and sensors. Click `Convert Data to Tensor` then `Train Model`. View console ctrl-shift-i for any issues

Here we can save the model or upload a previously saved model. Note: Labels are not loaded with the model. This is a work in progress.



Vision-Model

Next Page

This section focuses on tuning a vision model.

```
// Define the model architecture
model = tf.sequential();
model.add(tf.layers.conv2d({                                    // vision
    inputShape: [minWidth, minHeight, 3],
    kernelSize: 3,
    filters: 16,
    activation: 'relu'
}));
model.add(tf.layers.maxPooling2d({poolSize: 2}));
model.add(tf.layers.conv2d({kernelSize: 3, filters: 32, activation: 'relu'}));
model.add(tf.layers.maxPooling2d({poolSize: 2}));
model.add(tf.layers.conv2d({kernelSize: 3, filters: 32, activation: 'relu'}));
model.add(tf.layers.maxPooling2d({poolSize: 2}));
model.add(tf.layers.conv2d({kernelSize: 3, filters: 32, activation: 'relu'}));
model.add(tf.layers.maxPooling2d({poolSize: 2}));
model.add(tf.layers.conv2d({kernelSize: 3, filters: 32, activation: 'relu'}));
model.add(tf.layers.flatten({}));
model.add(tf.layers.dense({units: 64, activation: 'relu'}));
model.add(tf.layers.dense({units: uniqueLabels.length, activation: 'softmax'}));

// Compile the model
const myRate = parseFloat(document.getElementById('myLearningRate').value)
model.compile({
    optimizer: tf.train.adam(myRate),
    loss: 'categoricalCrossentropy',
    metrics: ['accuracy']
});
```

Sound-Model

Next Page

Here, we can tune a sound model.

```
function buildModel() {
 model = tf.sequential();
 model.add(tf.layers.depthwiseConv2d({
   depthMultiplier: 8,
   kernelSize: [NUM_FRAMES, 3],
   activation: 'relu',
   inputShape: INPUT_SHAPE
 }));
 model.add(tf.layers.maxPooling2d({poolSize: [1, 2], strides: [2, 2]}));
 model.add(tf.layers.flatten());
 model.add(tf.layers.dense({units: 3, activation: 'softmax'}));
 const optimizer = tf.train.adam(0.01);
 model.compile({
   optimizer,
   loss: 'categoricalCrossentropy',
   metrics: ['accuracy']
 });
}
```

// Sound

Sensor-Model

Next Page

This section demonstrates the model for acceleration or any other sensor combination.

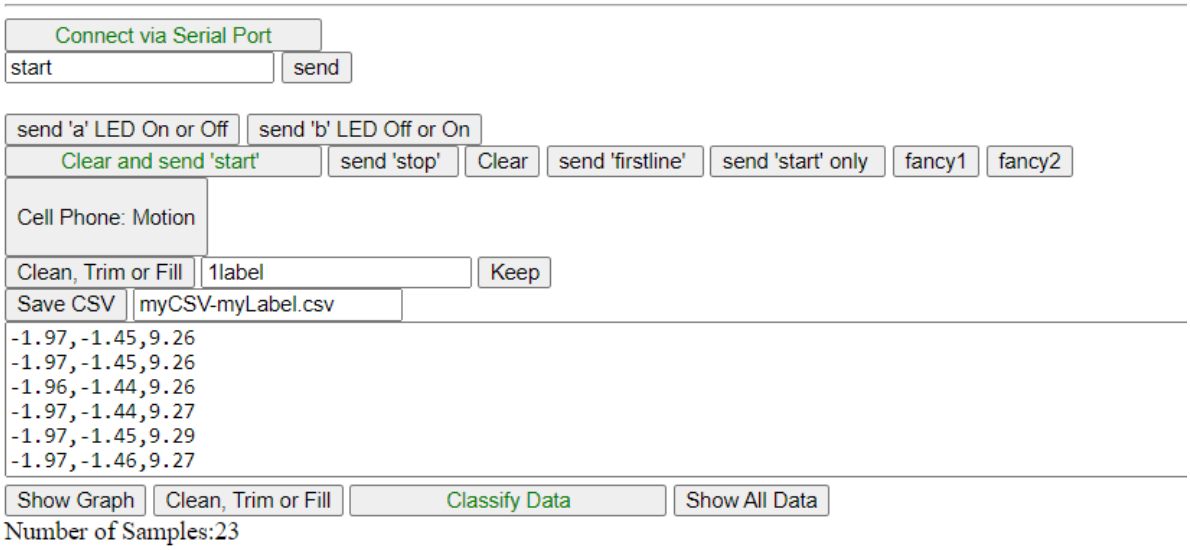
```
model = tf.sequential();                                    // acceleration or other sensors
model.add(tf.layers.lstm({units: 8, inputShape: [myModelSamples, myModelSenses] }));

//model.add(tf.layers.dense({ units: 30,  inputShape: [myModelSamples, myModelSenses] }) );
model.add(tf.layers.dense({ units: 30 }) );
model.add(tf.layers.dense({ units: 30 }) );
model.add(tf.layers.dense({units: uniqueLabels.length, activation: 'softmax'}));

// Compile the model
const myRate = parseFloat(document.getElementById('myLearningRate').value)
model.compile({
      optimizer: tf.train.adam(myRate),
      loss: 'categoricalCrossentropy',
      metrics: ['accuracy']
});
```

tinyMLjs-webSerial

Next Page

Here is where we connect a microcontroller `Connect via Serial Port`
Then `Clear and send Start`
If needed `Clean` the data and check the label name and `Keep` and/or `Save CSV` checking the file name.

When using more than two labels, you can return to the model training part of the webpage to train your model.



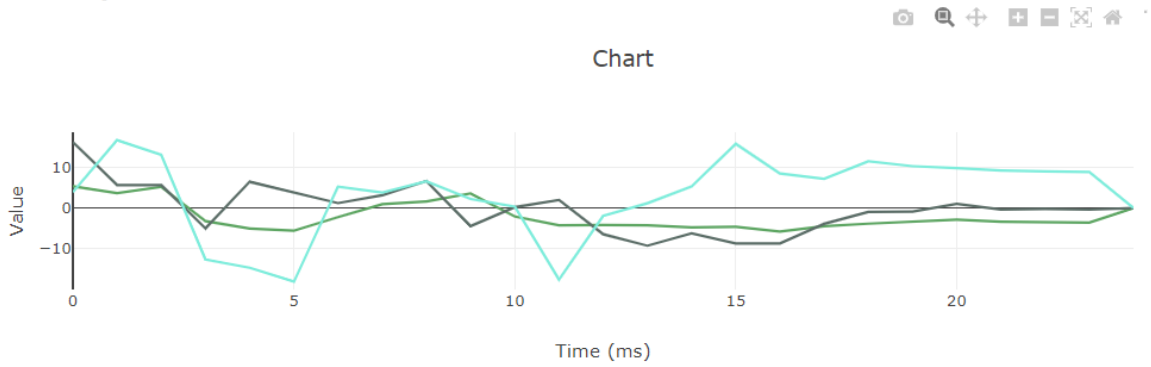
tinyMLjs-Chart

Next Page

Now it is time to test your model. Load more data, clean if needed and click `Classify Data`
Note that the code to be loaded onto the Nano33BleSense (Rev1) is displayed in the textarea for easy copying.

Show Graph | Clean, Trim or Fill | Classify Data | Show All Data

Number of Samples:25



Aruino Nano33BleSense webSerial code that can be adapted for other microcontrollers

copy

The "fancy" Arduino sketch is on the github here https://github.com/hpssjellis/tinyMLjs/blob/main/public/acceleration/a00-accell-nano33-fancy.txt

```
/*
 * webSerial for testing javascript connection with an arduino
 *
 * Latest work at   https://github.com/hpssjellis/webMLserial
 *
```

tinyMLjs-Gotchas

Next Page

Use at your own risk!
By Jeremy Ellis @rocksetta
Github at https://github.com/hpssjellis/tinyMLjs/tree/main
Demo's Index at https://hpssjellis.github.io/tinyMLjs/public/index.html
This page should be at https://hpssjellis.github.io/tinyMLjs/public/acceleration/a00-best-acceleration.html

A couple of gotchas (as of Jun 22, 2023):

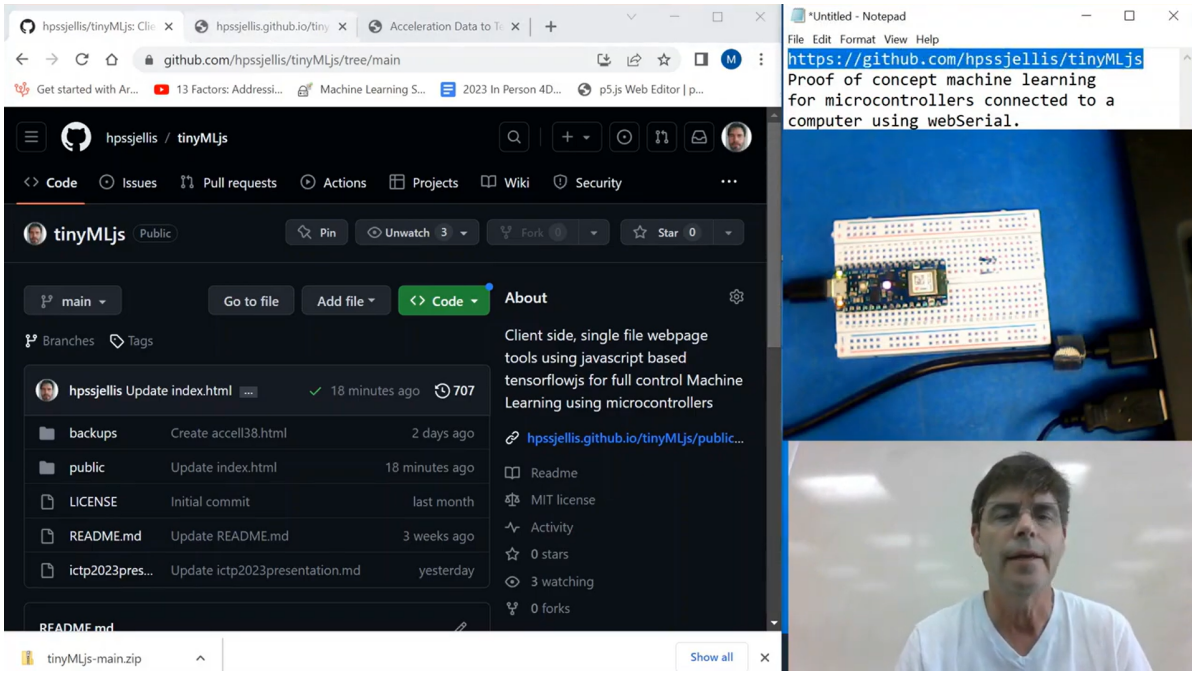
1. File names have to be in the format "name-lable.csv" or "name-lable (1).csv" or "name-lable (2).csv" etc. Unfortunately Android and iPhone don't auto make the numbering for you.
2. Android and Apple device have an opposite orientation, so I have made negative all the android motion data so when your phone is on a table z = -9.8 m/s^2 etc. When looking veritacally at your phone y = -9.8 m/s^2. The auto detect of this only works if an Android phone is in mobile format not "desktop site"
3. Real data has lots of rough data, machine learning models do not like missing data. If your results show "NaN" either your training data or classification data has errors. Note: If the loss is not changing your trained data probably has errors. The "clean,trim,fill" buttons might help.
4. Presently a CSV label upload bug happens sometimes. Easy to fix by entering the correct labels in the correct order. I will try to fix the issue when I figure out what is causing it.

Conclusion

Back to the top

Tutorial playlist Video here

Direct link https://youtu.be/3f4led32SL8



The github is at: https://github.com/hpssjellis/tinyMLjs

The index webpage is at https://hpssjellis.github.io/tinyMLjs/public/index.html

While this presentation represents a starting point, it demonstrates that powerful, proof of concept, end-to-end machine learning on edge devices does not have to rely on the cloud or specific hardware. It can be done in the field or in a classroom without internet access.

By Jeremy Ellis @rocksetta

Github Profile at https://github.com/hpssjellis

Back to the top