Magic Wand

Magic Wand example for TensorFlow Lite Micro on the Arduino Nano 33 BLE Sense.

Table of contents

Introduction

This project shows you how to recognize gestures made by waving a magic wand, using machine learning to analyze accelerometer and gyroscope data. It demonstrates the three main stages of an end-to-end machine learning project:

- **Gathering Data**. Using a Bluetooth connection to a web page, you can capture gestures, label them, and download the results.
- **Training**. A Python notebook on the free Colab service shows how to use TensorFlow to train a model to recognize gestures from your data.
- **Deployment**. You can deploy your trained model to the Arduino board using TensorFlow Lite Micro and the Arduino IDE.

Hardware Requirements

You'll need the following:

- Arduino Nano 33 BLE Sense board. These are available as part of the TinyML Starter Kit, or separately from Arduino or resellers. Other Arduinos won't work unfortunately, because the Bluetooth and sensor code rely on accessing the particular hardware of the Nano 33 BLE Sense.
- MicroUSB cable. This is included in the TinyML Kit, but you'll need a USB-A adaptor too if your computer only has USB-C ports.
- Computer. The Arduino toolchain runs on Linux, Windows, and MacOS, so you should be able to use most

laptops, desktops, or even a Raspberry Pi. For the training process, you'll also need an up-to-date version of the Chrome web browser so you can use the Web Bluetooth APIs.

- Stick. We'll be attaching your Arduino to a 'wand', but this can be practically anything, as long as it's roughly a foot (30 centimeters) long.

Installing the Sketch

You'll need to ensure you can successfully connect and load sketches onto your Arduino board,
using the desktop IDE.
Once you've made sure you can load a simple sketch successfully, you'll follow these steps:

Arduino Desktop IDE

If you're running using the Arduino IDE application, you'll need to fetch the latest version of this sketch.
To install the TensorFlow Lite Micro for Arduino library, see the
how to install instructions.

Open up the magic_wand.ino file in the Arduino editor, and make sure the Arduino board is visible and connected to the right port. You'll need to search for the some libraries that the sketch depends on, using `Sketch->Include Library->Manage Libraries` from the main menu. The Arduino_LSM9DS1 lets us access the accelerometer and gyroscope readings from the board's IMU, and you need at least version 1.1.0. We'll be using Bluetooth to communicate with the web page, so you should also search for ArduinoBLE and make sure you've got version 1.1.3 or newer.

You should now be able to press the upload button to compile and install the sketch on your board.

Building the Wand

The 'wand' itself can be as simple as a stick, it doesn't need to do anything other than keep the board at its end as you hold the other end and wave it about. A cheap wand from an online retailer will work. A simple piece of wood or ruler works just as well.

You should place the board at the end of the wand, with the USB socket facing downwards, towards where you hold it, so that the cable can run down the handle. The sketch is designed to compensate for any rotation of the board around the wand's shaft, so as long as it's parallel to the wand's length the board's twist won't matter. Use sticky tape or some other easy-to-remove method to attach the board to the wand, and hold the cable in place along the shaft. The end result should look something like this:

If an ASCII-art diagram is more helpful, here's what you should aim for:

```
                 ____
                |    |<- Arduino board
                |    |
                | () |  <- Reset button
                |    |
                 -TT-   <- USB port
                  ||
                  ||<- Wand
                 ....
                  ||
                  ||
                  ()
```

Using the wand

The wand can be used with or without the Nano 33 BLE attached to the Tiny Machine Learning Shield. It is easier to use without, as your hand will not tire as quickly. The wand should be held as you would a pencil or pen, about 8 inches from the USB socket. Use your wrist to make strokes, not your arm. Strokes will need to be made somewhat quickly, without stopping during changes in direction.

Viewing Gestures in the Browser

To preview and record gestures, we'll be connecting the sketch you've just uploaded to a web page, using Bluetooth and Chrome's WebBLE API. The code for the page is in this repository, but it's all implemented using browser-side Javascript in a static HTML page, so you don't need to host it on your own server. Just dragging and dropping the `index.html` file into your browser should work.

If the sketch has uploaded successfully, the Arduino should be advertising itself through Bluetooth. On the web page, press the 'Bluetooth' button to connect, and you should see a dialog appear asking you to pair with a device. After a second or two, there should be an entry that looks something like "BLESense-2F00". Click on that to pair, and you should be returned to the web page.

If everything is working as expected, the Bluetooth button should turn blue, with "Connected" next to it. Now try moving the wand and look at the square below the button. As you gesture, you should see tracks appearing as lines in the web page in real time. Try doing small circles, or a 'Z' like Zorro!

Pretrained Model

The sketch comes with a model that's been trained to recognize the hand-drawn digits zero to nine. This is based on a small dataset recorded by Google, so your accuracy may vary, but if you bring up the Serial Monitor in the Arduino IDE you can see what the model predicts for each gesture you make, with a confidence score between 0% and 100%, as well as ASCII art of the gesture outline.

Recording Gestures

As you get familiar with the wand, so should notice that the gestures you have performed start to stack on the right side of the web page. This is where the data you'll eventually want to use for training is stored. When you leave or refresh the web page, these gestures will be lost, so make sure you use the "Download Data" link to save them locally if you've generated a number of them.

The gestures are automatically split up by times when the wand is kept still. These pauses act like spaces between words, and so when you've finished a gesture you should stop moving the wand so that it ends cleanly.

To get started, you should pick a couple of easy gestures to perform, like a 'Z' and an 'O'. As you make these gestures, you should see them appear in the right-hand stack of gestures. You can look at the shapes shown there to understand whether the gestures came out cleanly. A good rule of thumb is that if you can't tell what the gesture is by looking at it in the stack, then a model will have a hard time recognizing it too.

Once you have ten or so of each gesture, scroll through the stack to review them. If any don't seem very recognizable, or are too 'sloppy' (which is very subjective unfortunately), then you can press the trash can button on the top right of the image to remove it. If you removed any, try recording some more so you have at least ten of each gesture. If you are happy with a gesture, click on the label at the top left to type in the correct name for it (for example `O` or `Z`).

After you've reviewed and labeled all of your data, you can download it as a JSON text file that can be used for training.

Training

Once you have data, you should run the Python training notebook in Colab and follow the steps to create and export your own model.

Deployment

The Python training process should give you a `magic_wand_model_data.cc` file. Replace the file of the same name (but with a `.cpp` suffix) that's in the sketch you're using with this version. You'll also need to update the `labels` and `label_count` variables near the top of the `magic_wand.ino` to reflect any changes you made to the gestures you're trying to recognize.

Upload this modified sketch, and you should be able to perform gestures and see them recognized in the Serial Monitor of your Arduino editor.