CS249r Fall 2023 Assignment 3: Magic Nicla Wand

Assignment Due: Monday, Nov 6, 2023 at 11:59pm on Canvas

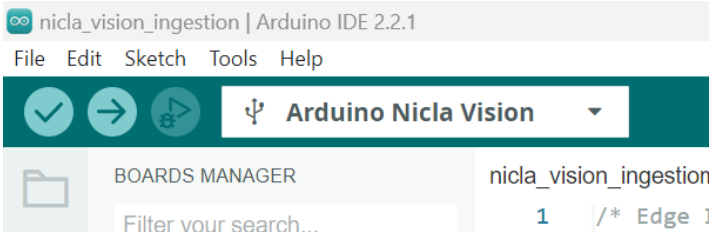
Assignment Overview

In this assignment, you'll dive into the Tensorflow ecosystem, specifically exploring the Tensorflow, Tensorflow Lite, and Tensorflow Lite Micro frameworks using IMU data from the Arduino Nicla Vision.

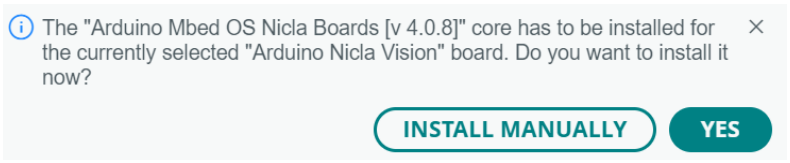
- **What**: You will work with IMU data gathered from the Arduino Nicla Vision. The tasks encompass building a Tensorflow model, optimizing it using Tensorflow Lite's quantization methods, applying pruning techniques, and finally converting it for compatibility with Tensorflow Lite Micro.
- **Why**: The ability to transition and optimize models between Tensorflow, Tensorflow Lite, and Tensorflow Lite Micro is a fundamental skill in tiny machine learning. This ensures models are efficient, effective, and deployable on a wide range of devices.
- **How**: You will learn to construct convolutional neural networks with various architectures and analyze the size of the model post-training. We will be utilizing IMU data which is unique due to its time series nature. Additionally, you will examine the effects of applying quantization and pruning on the model.

Set up

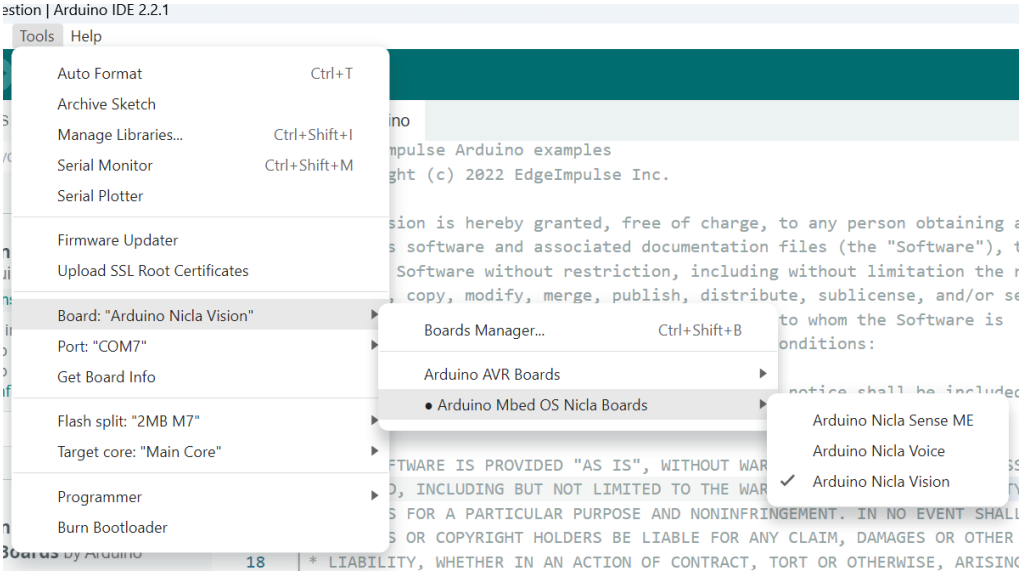
Open the Arduino IDE, connect your Arduino Nicla Vision to your computer, and choose this option from the drop down menu at the top of your IDE.



A notification will pop up at the button of your IDE. Please click ‘Yes’ to install. If you have already connected your Arduino Nicla Vision previously, you may skip this step.



You should now see that you've established a connection to the board.



Install Arduino Library dependencies

Open your Arduino IDE.

The `Arduino_LSM6DSOX` and `VL53L1X` Libraries provide support for your Nicla's 6-axis IMU sensor (LSM6DSOX). You need to install them both.

Go to Sketch > Include Library > Manage Libraries.
In the Library Manager, enter LSM6DSOX into the search box.
Find the Arduino_LSM6DSOX by Arduino library in the list and install it.

Go to Sketch > Include Library > Manage Libraries.
In the Library Manager, enter VL53L1X into the search box.
Find the library VL53L1X by Pololu in the list and install it.

We will also be using Bluetooth to communicate the Nicla Vision's IMU data to a webpage where you will be able to visualize the motion of your magic wand. Fancy! To do this, we must install the `ArduinoBLE` library.

Go to Sketch > Include Library > Manage Libraries.

In the Library Manager, enter ArduinoBLE into the search box.
Find the library ArduinoBLE by Arduino in the list and install it.

The code for collecting data and viewing your IMU readings are located in the Github Repository.

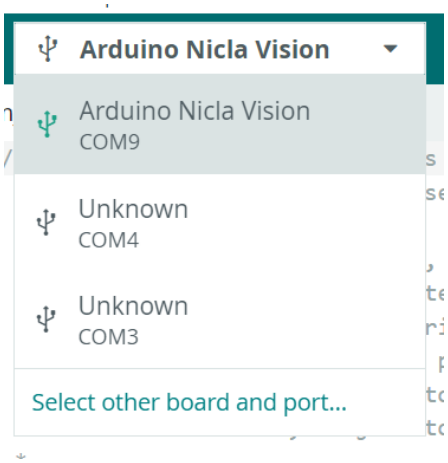
```
git clone https://github.com/jasonjabbour/CS249r_TinyML_Assignment3.git
```

View IMU Readings

Let's make sure that we can read the values from your IMU. Recall that an IMU has both a gyroscope and an accelerometer. The gyroscope measures the rate of rotation or angular velocity, represented by Gx, Gy, and Gz. Specifically, Gx indicates the rate of rotation around the X-axis (pitch), Gy around the Y-axis (roll), and Gz around the Z-axis (yaw). On the other hand, the accelerometer detects linear acceleration or changes in velocity, depicted by Ax, Ay, and Az. Ax corresponds to acceleration along the X-axis (left and right movement), Ay along the Y-axis (forward and backward movement), and Az along the Z-axis (upward and downward movement).

Open the `nicla_vision_ingestion.ino` file in your Arduino IDE. You can find this file within the `nicla_vision_ingestion directory` of the repository you cloned above. To open this file, you can simply click on the file and the Arduino IDE should open right up. Remember to connect your Nicla to your computer through a USB.

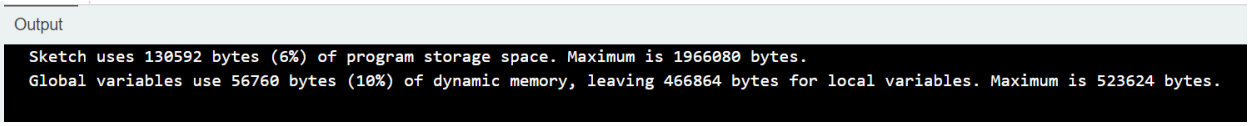
Select your Arduino Nicla Board Vision Board. Depending on the USB port you plugged your Nicla into, you might see a different COM port. No need to worry, that is okay.



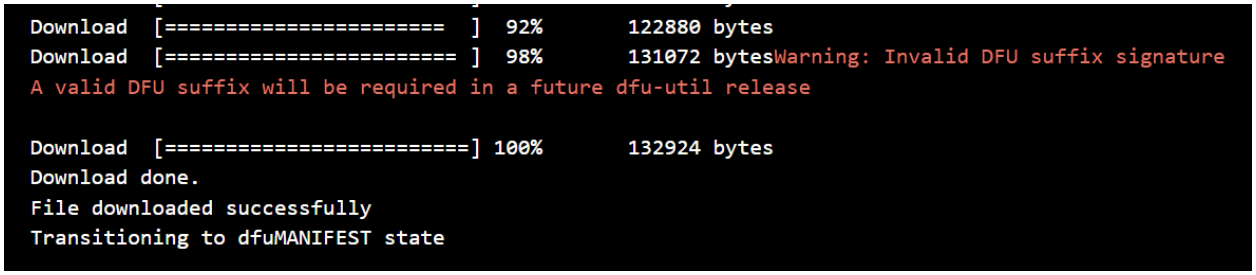
In the Arduino IDE, running your code involves two main steps. First, we 'Verify' or 'Compile' our code. This step checks for any errors. If there are no issues, then we proceed to the next step. Second, we 'Upload' our code to the device. Once uploaded, we can open a serial port to view our script in action on the Arduino.

Now that we're familiar with the Arduino workflow, let's get started.

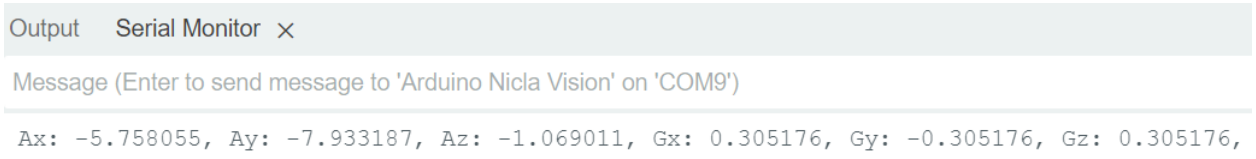
Click Sketch > Verify/Compile from the dropdown menu. If you haven't made any changes to the script, it should compile without errors. Upon successful compilation, your Arduino IDE terminal should display a message similar to the following:



Next, let's upload our script and double check that our IMU is working properly. Go to Sketch > Upload. You should see that your script has been successfully uploaded. (You may ignore the warning.)



Once Uploaded, you can open your serial monitor by going to Tools > Serial Monitor. You should now see your accelerometer and gyroscope readings. Go ahead and move your IMU around. Move it around slowly then move it around quickly. Can you see your readings changing reasonably?

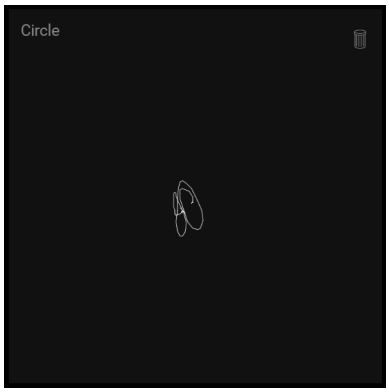


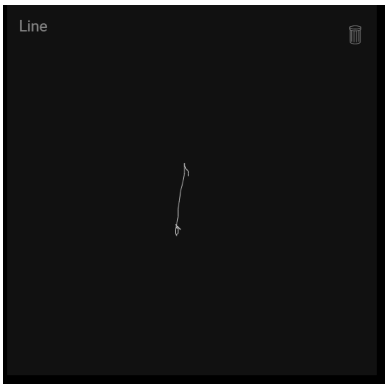
(Question) In your write-up, please expand your serial monitor and include a screenshot showing multiple lines of readings. This will help demonstrate the data output from your device.

Magic Wand Data

The objective of this assignment is to build a model that recognizes gestures made by waving your magic wand, using data from the accelerometer and gyroscope sensors of the IMU on your Nicla. You can either collect your own gesture data or use the provided .json file (CS249r_TinyML_Assignment3 > magic_nicla_wand > data > wanddata.json), which captures

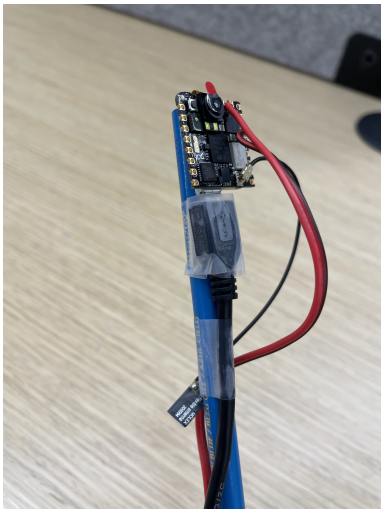
my wand making a clockwise circular motion and motions in a linear path. Whether you choose to classify a unique gesture or work with the provided dataset is entirely up to you.





(Question) Please make note of your decision in your write up.

(Optional) If you choose to collect your own wand gesture. Please follow the instructions below.



You can start by building the wand. The wand itself can be as simple as a stick or pencil, 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. 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. Please note that the orientation of your IMU **does** matter. If you decide not to collect your own data then matching the orientation of the IMU to the one in the figure is important. Use sticky tape or some other easy-to-remove method to attach the board to the wand, and hold the cable in place. The end result should look something like the one in the figure. Share a picture and show off your creation in your write up! (optional)

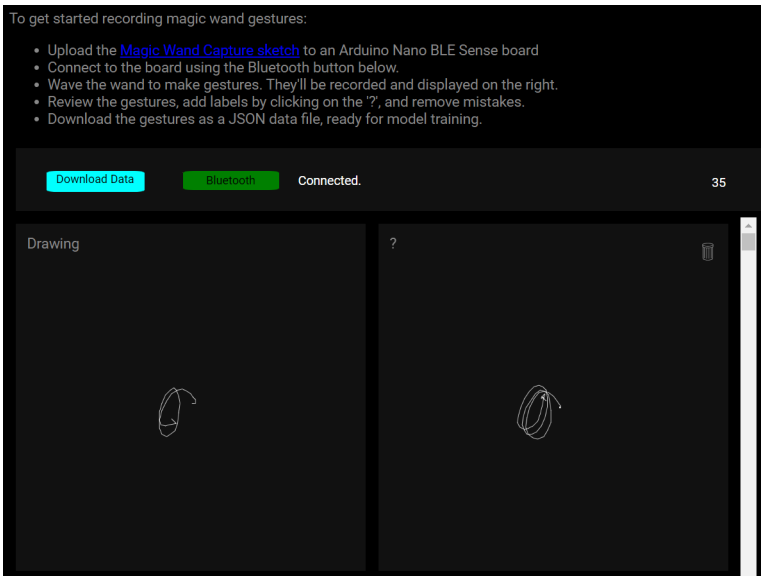
You can easily collect data by using the `magic_nicla_wand_data_capture.ino` script that we provided you with located inside the `magic_nicla_wand_data_capture` directory. To run this script we must follow the two steps mentioned in the section above. First, select Sketch > Verify/Compile. Once your script has compiled without any errors, select Sketch > Upload.

Next, open the `index.html` file in your browser. This file is located within the `magic_nicla_wand_data_capture/Website` directory. Once you open the webpage in the browser, please follow the instructions at the top of the page. You will simply connect to your Nicla through bluetooth then collect several samples of you waving your wand around in your

desired path. The motion you make will be displayed within this webpage. Once you have collected your motions, replace the '?' with the class of your motion then download the data.

The gestures are automatically split up 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. 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.

The figures below show what you should expect to see.



The website interface is not perfect as people have faced some bluetooth disconnecting issues. Unplugging and plugging in your Nicla will do the trick until it disconnects again. You might also want to use Chrome as your browser.

Model Training and Optimization

Please follow along with the provided Colab file, where we will convert the IMU data into images to be input into a convolutional neural network. Your responsibility will be to construct the TensorFlow architecture for a CNN, apply optimization techniques such as quantization and pruning, and ultimately produce a model compatible with Tensorflow Lite Micro. Much of the code has already been implemented for you, but please read through it carefully to get the most out of this assignment. Tasks and questions for you to address are located within the Colab file.

CS249R Assignment 3: Magic Nicla Wand Training

Assignment Deliverables

Coding Implementations

Section 2

1. Complete the CNN architecture in the `make_cnn_model` function.
2. Apply Float16 Quantization to your model weights.

Section 3:

1. Apply pruning to each Conv2D and Dense Layer within the `make_pruned_cnn_model` function.

Write up Questions

Section 0

1. Insert an image of your serial monitor IMU readings.
2. Specify if you used the dataset we provided you with or you collected your own. If you did collect your own dataset, please submit the .json file and describe the classes in your dataset.

Section 1

1. What is the total accuracy for this baseline CNN model on the test dataset?
2. How large are your Tensorflow, TensorFlow Lite, and Tensorflow Lite Quantized models for this baseline CNN? How many times smaller is the quantized model from the original tensorflow model?

Section 2

1. What is the total accuracy for the CNN model that you built on the test dataset?
2. Given the code snippet provided, which converts a TensorFlow model to TensorFlow Lite (TFLite) format both in its regular and quantized forms, could you explain the quantization method applied? Specifically, which parts of the model undergo quantization? To what values or types are they being quantized. Additionally, how does the representative_dataset function contribute to this process?
3. Please discuss the trade-offs between Float16 Quantization and the quantization method you applied in the previous cell.
4. How large are your Tensorflow, TensorFlow Lite, and Tensorflow Lite Quantized models for this CNN? How many times smaller is your quantized model from the original tensorflow model?

5. How many times smaller is your Float16 quantized model from the original tensorflow model? Why is this the case?

Section 3

1. What is the total accuracy for the pruned CNN model that you built on the test dataset?
2. Please describe what you learned about your pruned model from the summary table. (Hint: Compare it to the previous summary table)
3. Out of all the models you trained, what model would you choose? Please justify your answer.

Section 4

1. Download and open up your `magic_wand_model_data.cc`. Please explain what this file contains. (Hint: What does the array of hexadecimal values represent?)

Files to Submit

- Download your final `magic_wand_model_data.cc` and `quantized_model.tfl` you will need to submit these files.
- Download your Colab file as a `.ipynb` file (File > Download > .ipynb). You will need to submit this file.
- Submit a PDF containing your answers to the questions mentioned above. Structure your write-up according to the specified sections (Sections 1-4). Ensure that all questions are numbered in accordance with the numbering given in these instructions. Any questions not placed under their respective section and question numbers **will not be graded**. Do not include your coding implementations in the write-up; your code will be reviewed through the .ipynb file that you submit. Limit your write-up to a maximum of three pages. Content beyond the third page will **not be graded**.

Extra Credit

The Arduino Nicla Vision is a relatively new device. As of now, it hasn't been integrated into the TFLM ecosystem, making it a challenge to deploy a TFLM model on this device. If you're aiming for the extra credit, you'll need to deploy a CNN model for wand gesture recognition on the Arduino. You might find this repository useful, as it demonstrates deploying a wand model on the Arduino Nano 33 BLE Sense. However, you'll need to adjust the peripherals located here. This could simply involve removing certain files since the Nicla Vision might not require access to many peripherals like the microphone, button, or LED light. Note that some preprocessing is essential to first determine when a motion starts and then rasterize the data into an image suitable for the Arduino. I've provided a starter script for this purpose. Still, adjustments will be needed in the /micro directory to enable the TensorFlow Lite Micro calls within the INO script. Please remember that in order to run TFLM you need a serialized model. You will find the `magic_wand_model_data.cc` file that you downloaded from the Colab useful.

Alternatively, you might consider creating and running the model through Edge Impulse. This tutorial could be helpful. However, do ensure that you implement the preprocessing step that converts the IMU data into a rasterized image. Fortunately, Edge Impulse allows you to integrate preprocessing scripts.

Please submit a video or a link to a YouTube video with your model detecting gestures on board the Nicla Vision.

Please submit a zip file of your code.