EE P 595A: TinyML
Lecture 8: American Sign Language Recognition

Dept. of Electrical and Computer Engineering
University of Washington

Instructor: Dinuka Sahabandu (sdinuka@uw.edu)

ELECTRICAL & COMPUTER ENGINEERING
UNIVERSITY of WASHINGTON

NSL

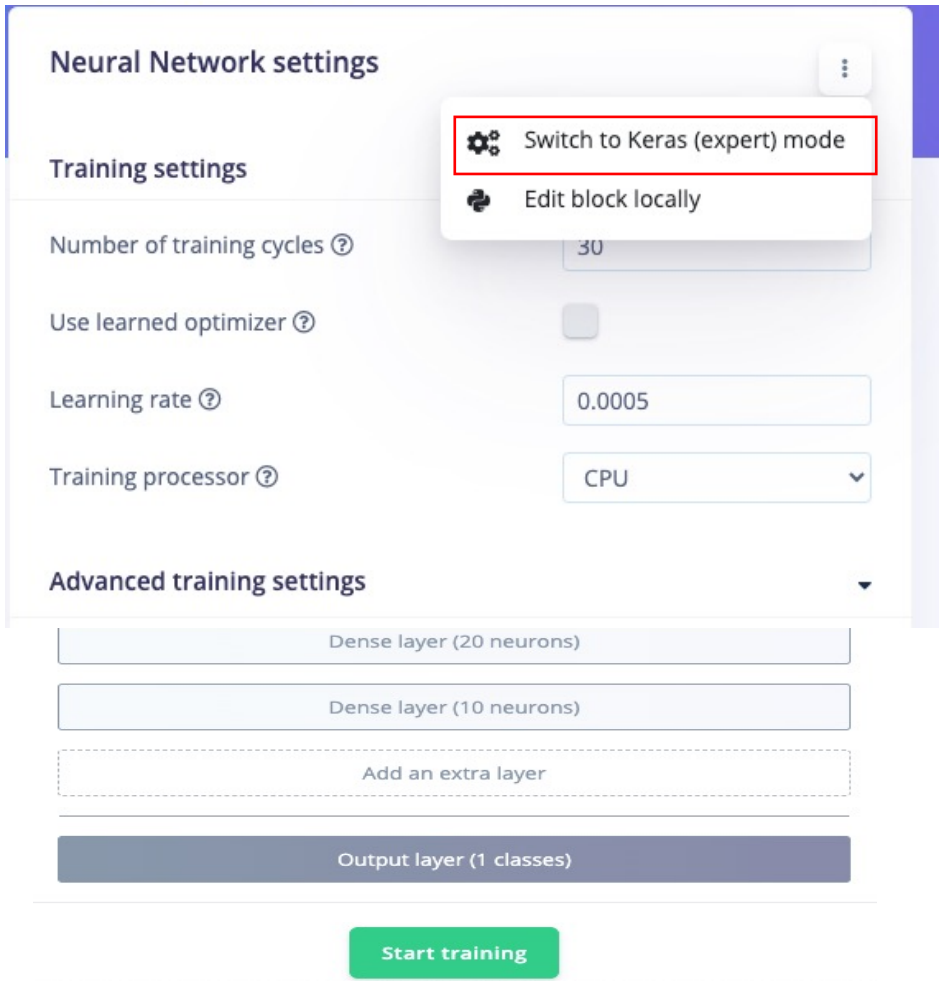
Topics Covered

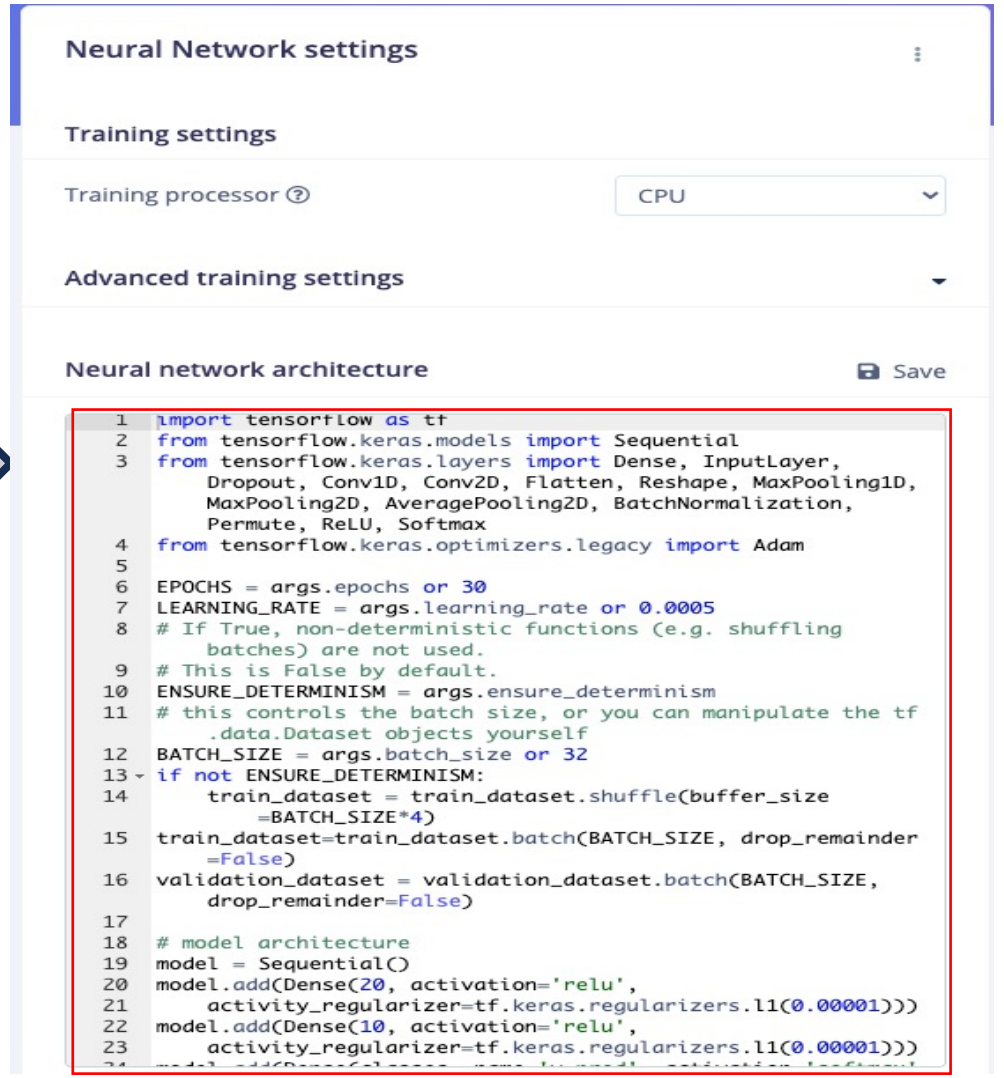
- Edge Impulse Tricks for Your Projects
- Homework #2 – Common Pitfalls and Workarounds
- Background on ASL and ASL Interpretation
- Gesture Based Dataset and Model
- TinyML Framework for ASL Interpretation
- Model Training, Evaluation Metrics, and Deployment

❖Relevant readings from the Textbook [Introduction to TinyML by Rohit Sharma] is Chapter 3

Edge Impulse Tricks for Your Projects

1. Access TensorFlow Code in Edge Impulse





Edge Impulse Tricks for Your Projects

2. Upload Your Sketch Without Arduino IDE

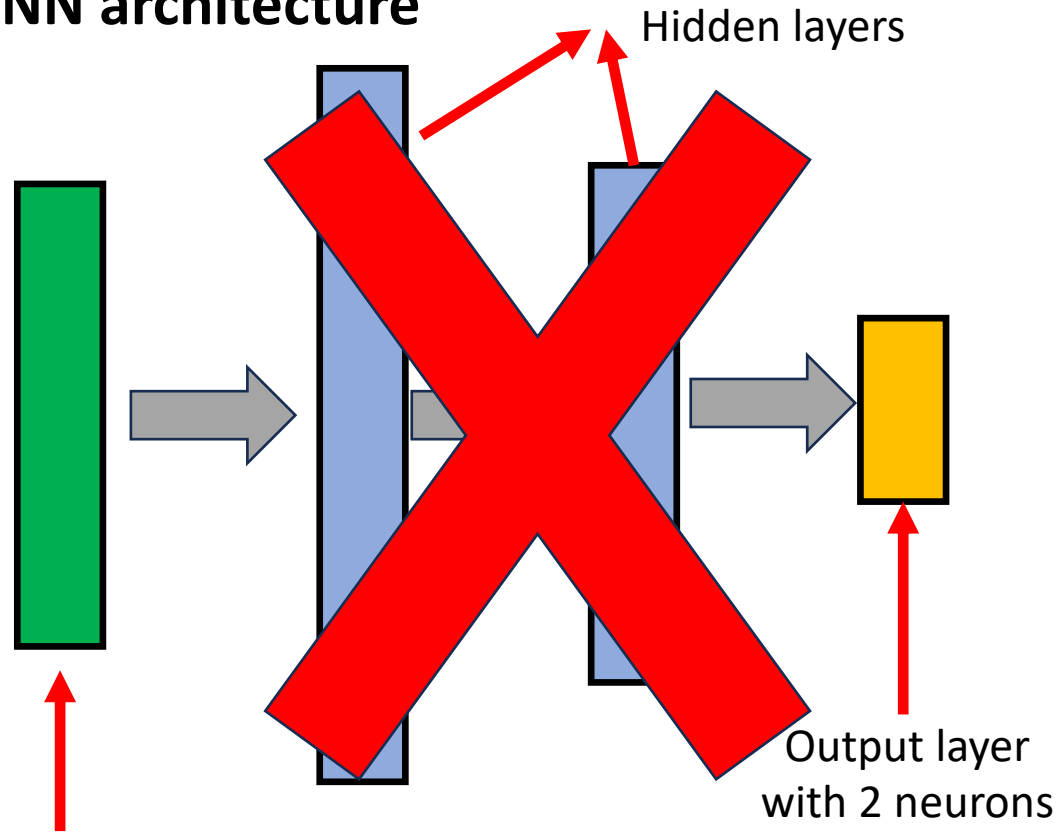
In **Deployment:**

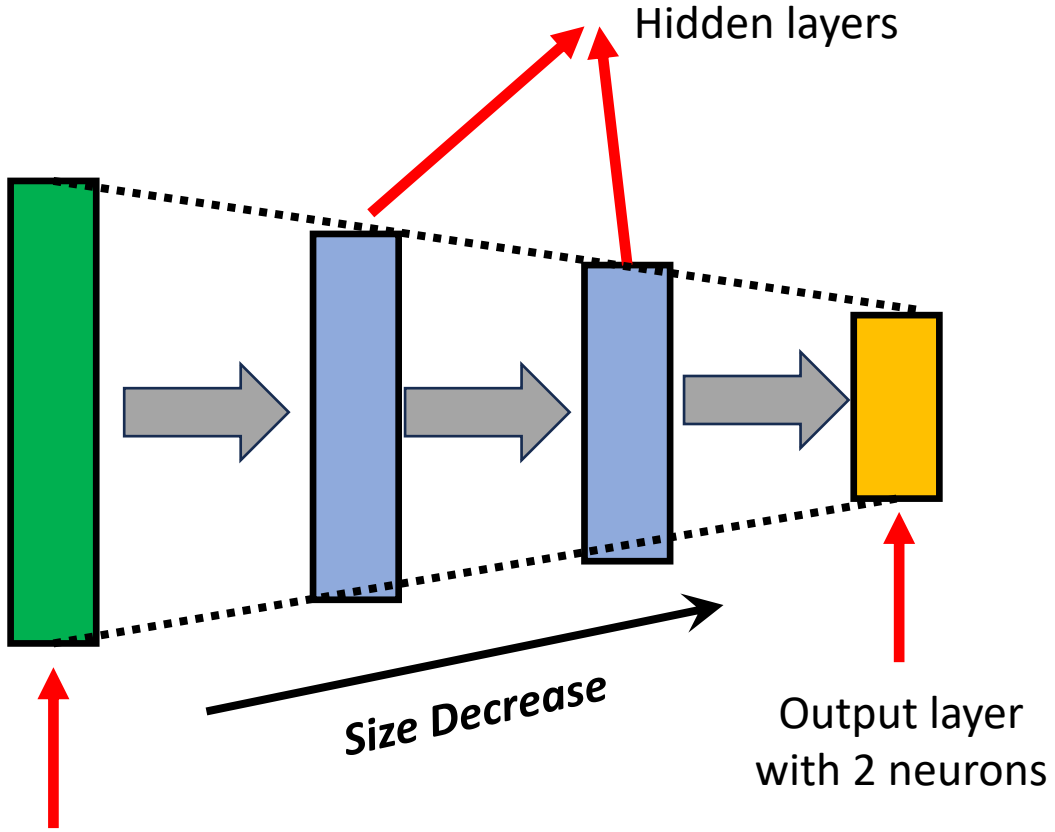
- Select “Arduino Nano BLE”
- Choose “EON Compiler”
- Build and download files
- Connect the Arduino Nano BLE and Flash
  (choose the appropriate flash file – mac, windows, or Linux)
- Open another terminal (with admin privileges in Windows) and execute
  “`edge-impulse-run-impulse`”

Note: You need to have Edge Impulse CLI and Arduino CLI Setup (from Lab 7) to do this

Homework #2 – Common Pitfalls and Workarounds

1. DNN architecture





Homework #2 – Common Pitfalls and Workarounds

2. TensorFlow Lite Library Issue --- For Windows

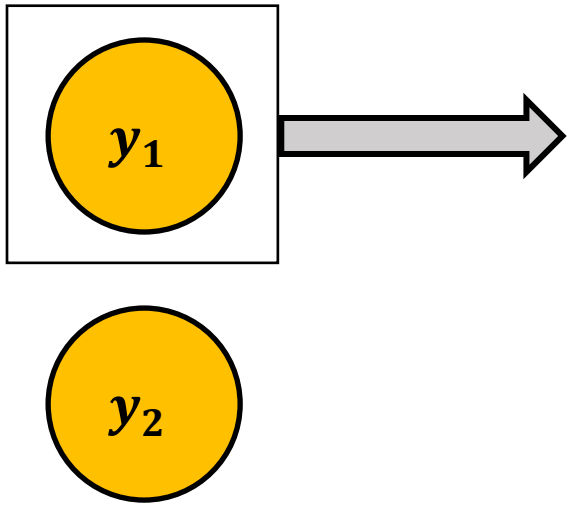
```
\4.1.3\\variants\\ARDUINO_NANO33BLE/cflags.txt" -DARDUINO_ARCH_NRF52840 -MMD -mcpu=cortex-m4 -mfloat-abi=softfp -mfpu=fpv4-sp-d16 -DARDUINO=10607 -DARDUINO_ARDUINO_NANO33BLE -DARDUINO_ARCH_MBED_NANO
\4.1.3\\variants\\ARDUINO_NANO33BLE/cflags.txt" -DARDUINO_ARCH_NRF52840 -MMD -mcpu=cortex-m4 -mfloat-abi=softfp -mfpu=fpv4-sp-d16 -DARDUINO=10607 -DARDUINO_ARDUINO_NANO33BLE -DARDUINO_ARCH_MBED_NANO
tensorflow\lite\micro\tools\make\downloads\cmsis\CMSIS\NN\Source\ConvolutionFunctions\arm_depthwise_separable_conv_HWC_q7_nonsquare.c.d: No such file or directory
```

*Find the "**arm_depthwise_separable_conv_HWC_q7_nonsquare.c**" in your above file path and **DELETE** it*

In general, it is not Advised to remove library files, but in this case, it gets the job done

Homework #2 – Common Pitfalls and Workarounds

3. How to write prediction class logic with one output variable in binary classification task?



Output layer
with 2 neurons

If $y_1 >$ threshold:
Predicted class = 1
else:
Predicted class = 0

In general

Threshold = 0.5

Note for HW #2: Label binarizer assigns "1" to "normal" and "0" to "attack" --> Change your prediction logic accordingly!

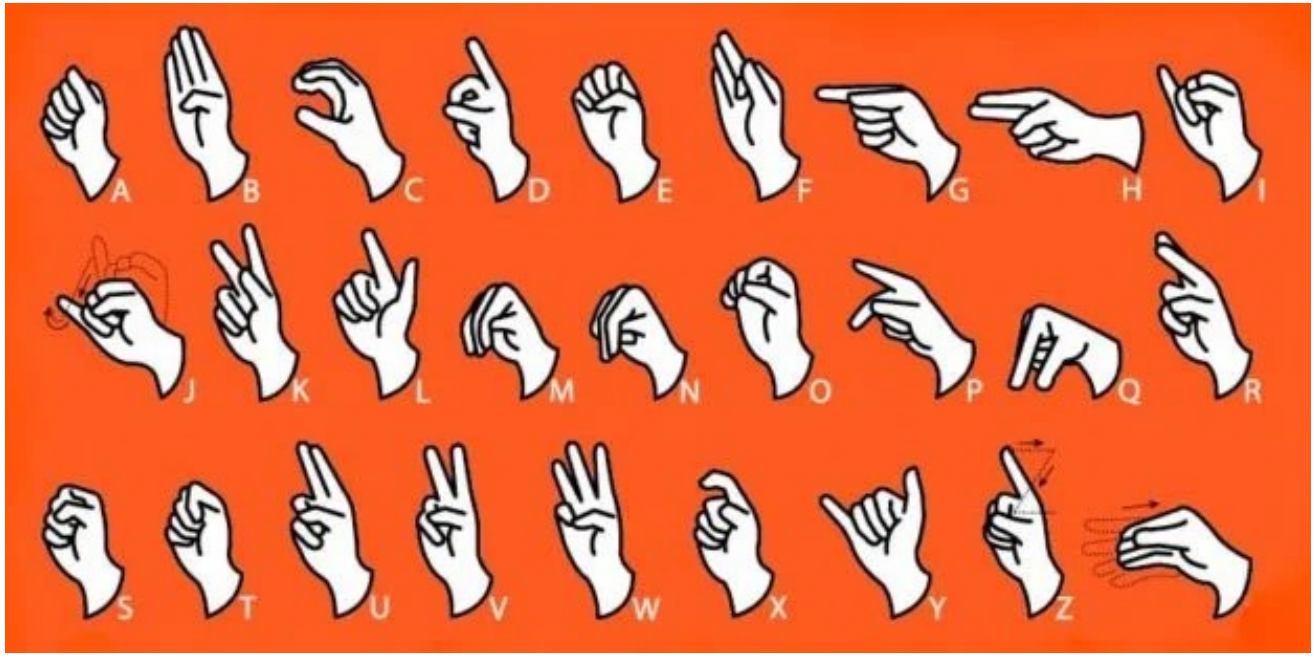
Introduction to American Sign Language

Sign Language is a visual mode of communication.
Sign language is the **third most used language** in the world!

Introduction to American Sign Language

American Sign Language (ASL) is a natural language that serves as the predominant sign language of Deaf communities in the United States of America.

ASL is expressed by movements of the hands and face.



Introduction to American Sign Language

History of Sign Language

Sign Language originated in France the 1700's

In 1816 Thomas Galluadet, an American minister who wanted to find ways to educate deaf children met Abbe' Sicard who was trained by Abbe'de l'Epee in French sign language.

Laurent Clerc and Thomas Galluadet taught and advocated for the deaf in America for over 40 years.

In 1989, the U.S. Supreme Court recognizes American Sign Language as a standard independent language.

Source: https://www.startasl.com/history-of-sign-language/

Introduction to American Sign Language

Source: https://learnhowtosign.org

Impact of ASL

Over 70 million deaf people use sign languages as their preferred communication form.

The number of deaf and hard-of-hearing individuals in the United States is about 3.6% of the population (~11.8 million).

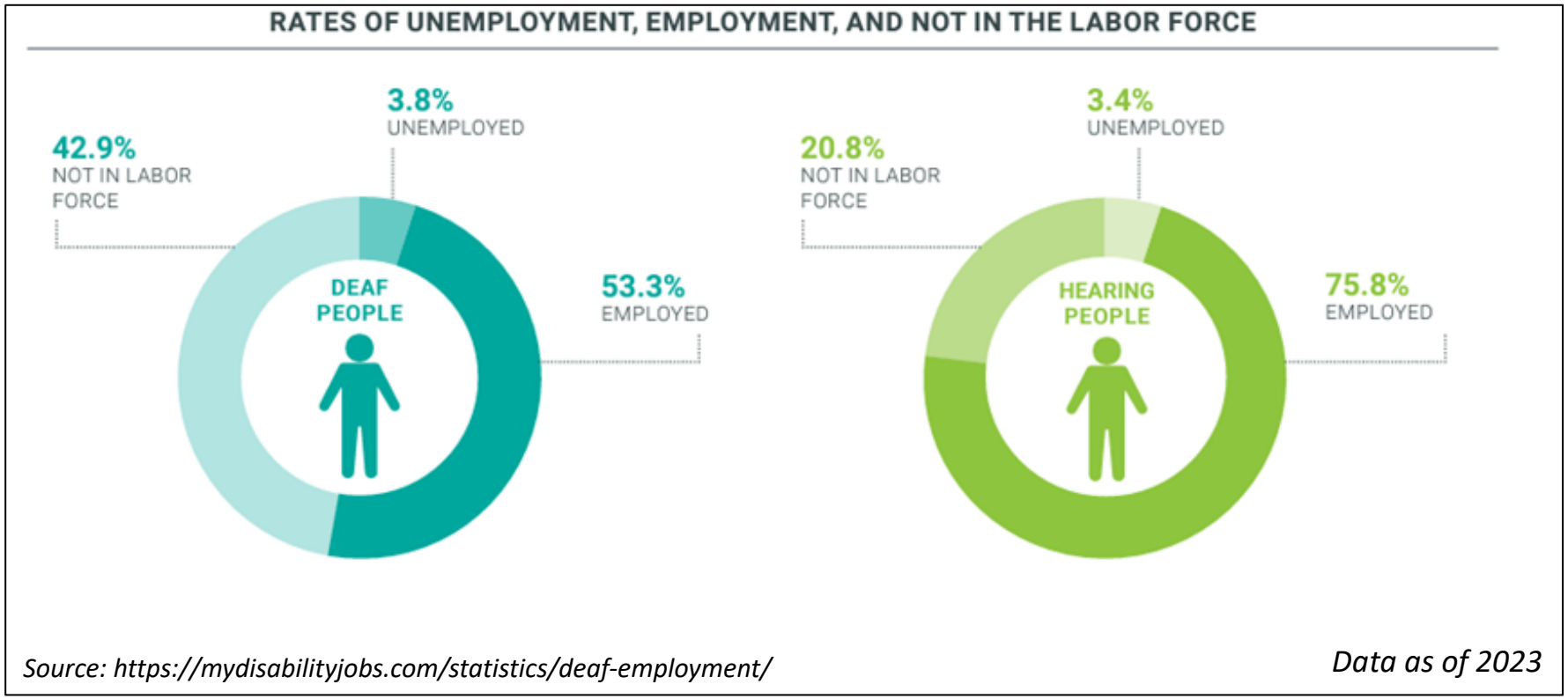
There are only 2,300 sign language interpreters in the USA.

There are 300 different sign languages and 130 that are recognized worldwide.

Source: https://www.gracesigns.org/

Impact of ASL on the Society

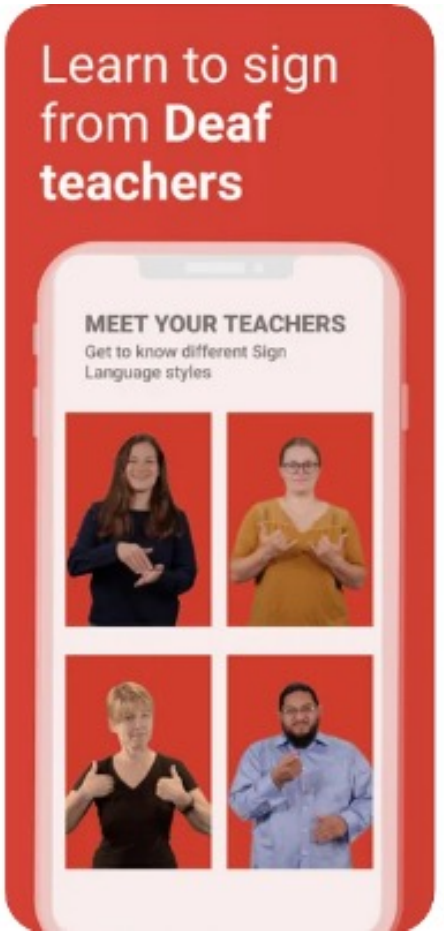
- For the deaf community, sign language equals rights.
- Engaging and learning sign language is crucial to being inclusive.
- ASL has enabled increased employment rates in people with a hearing disability.



Source: https://mydisabilityjobs.com/statistics/deaf-employment/

Data as of 2023

ASL Interpreter Apps



Top Universities for Deaf Students

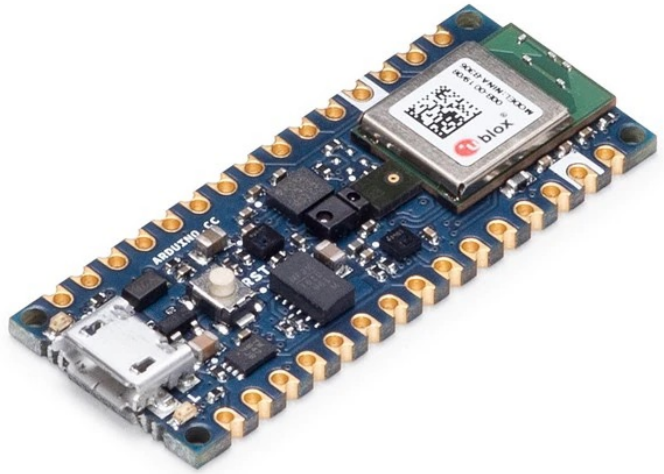
World leader in Deaf Education

RIT | National Technical Institute for the Deaf

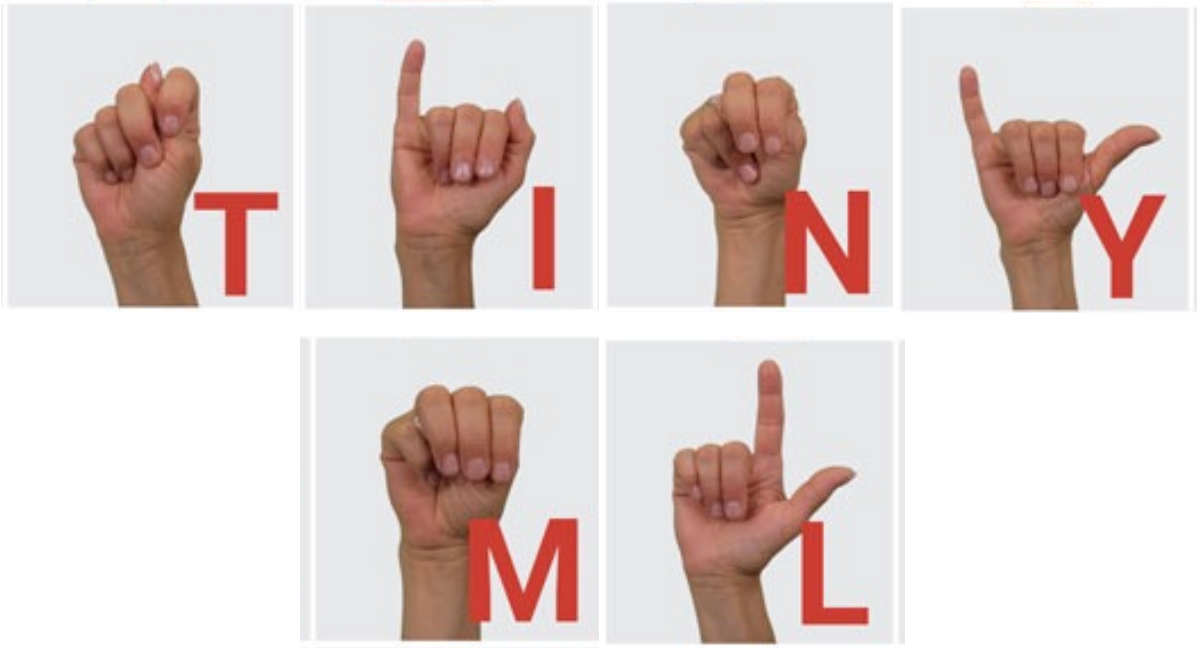
Source: https://www.healthyhearing.com/report/52682-Top-universities-for-deaf-students

ASL and TinyML

- How can we use machine learning for ASL?
- Why do we need TinyML for ASL recognition? What are the advantages that TinyML provides over other ASL recognition models?
- What can be possible input data and processing techniques?



+



ASL Recognition Techniques

Vision-based Methods

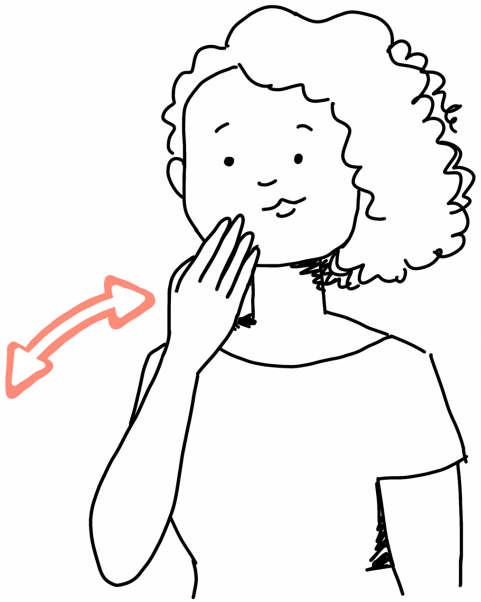
Camera image capture
Motion based Gesture recognition
Template Matching

Sensor-based Methods

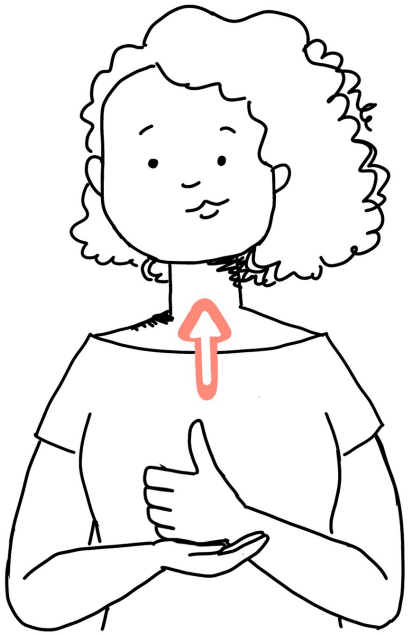
Data Gloves
Accelerometers and gyroscopes
Electromyography (EMG) sensors

Sensing-based Method –Input Data

ASL Gestures

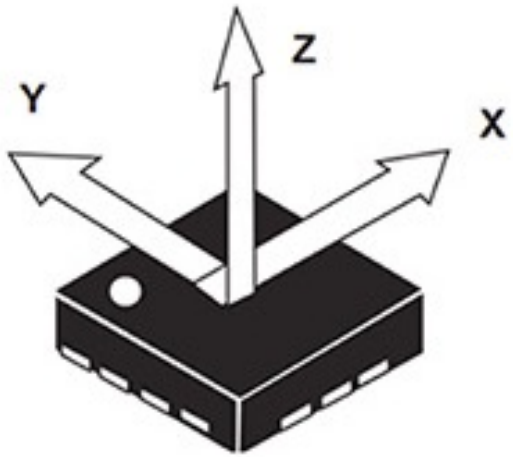


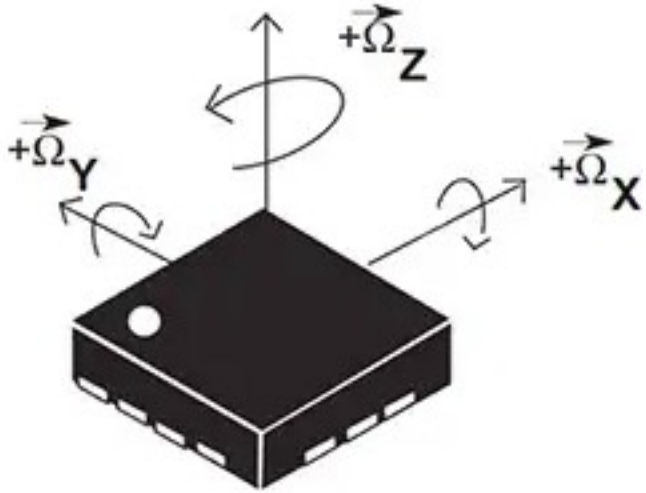
Thank you



Help

Accelerometer v. Gyroscope



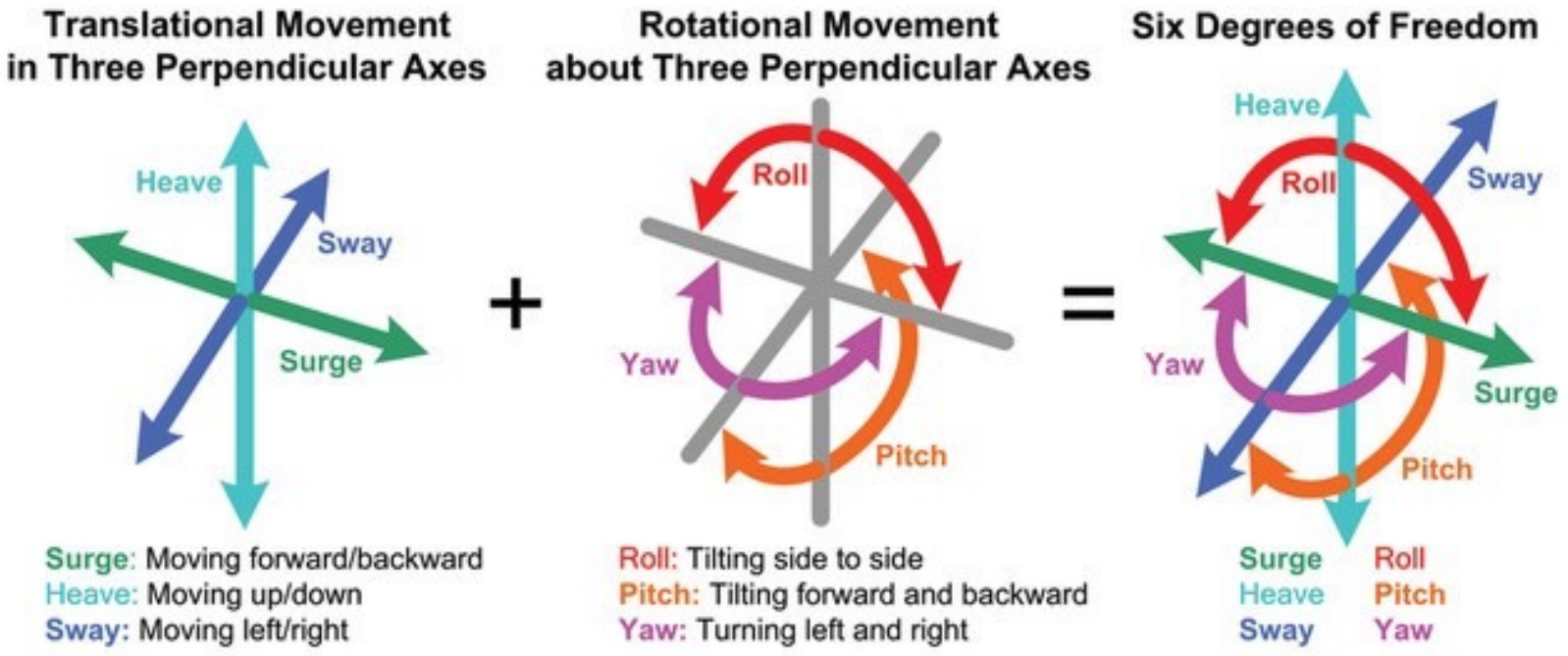


- The accelerometer measures linear acceleration along the x, y, and z axis.
- Senses axis orientation

- The gyroscope measures the rate of change of angular velocity over time (angular changes).
- Senses angular orientation

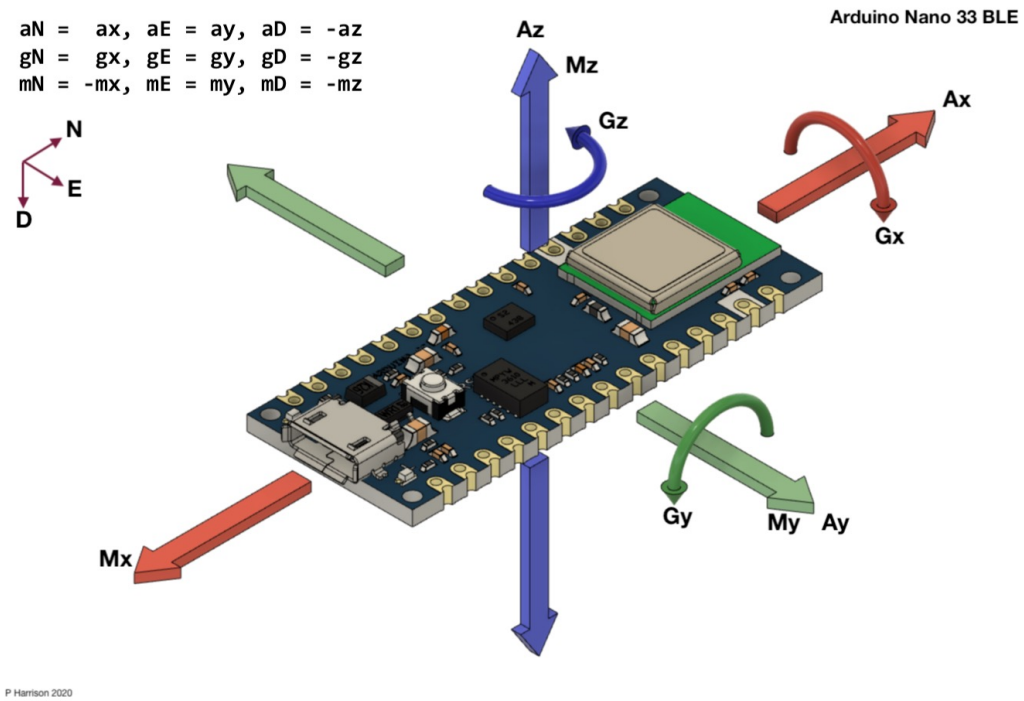
Source: https://github.com/billiyz/tinyml-on-the-edge

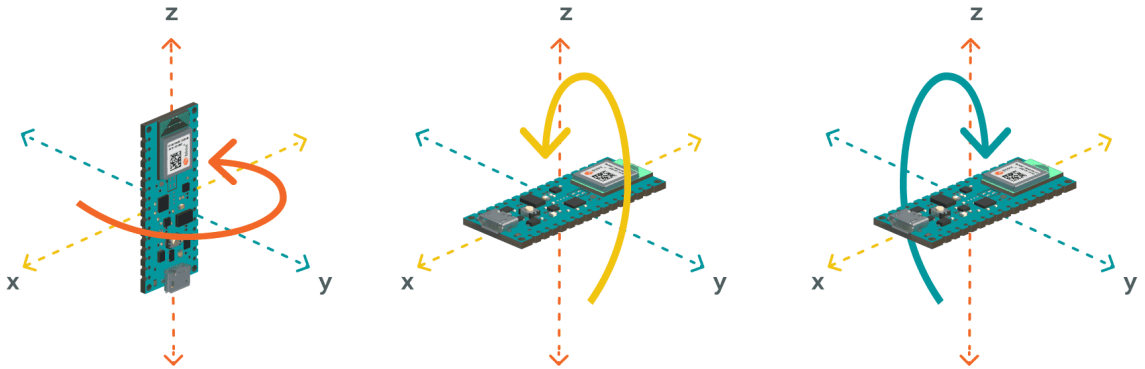
Degrees of Freedom

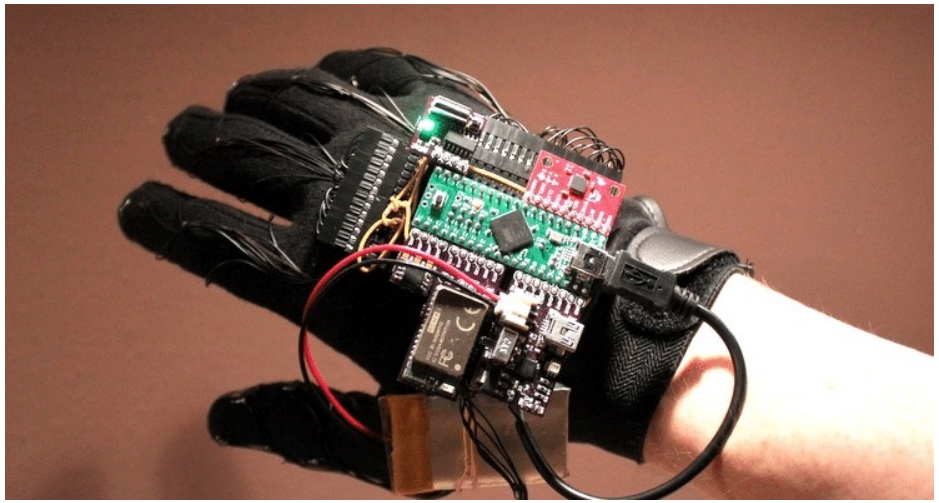


Source: https://www.scalawilliam.com/1612/limit-degrees-of-freedom/

Gyroscope on Arduino Nano 33 BLE



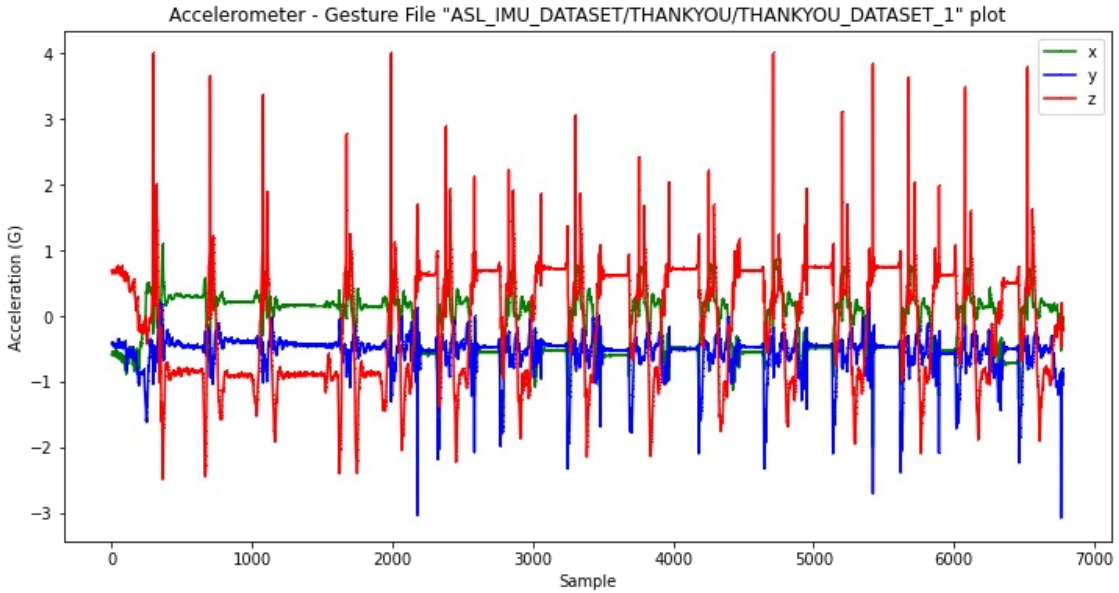




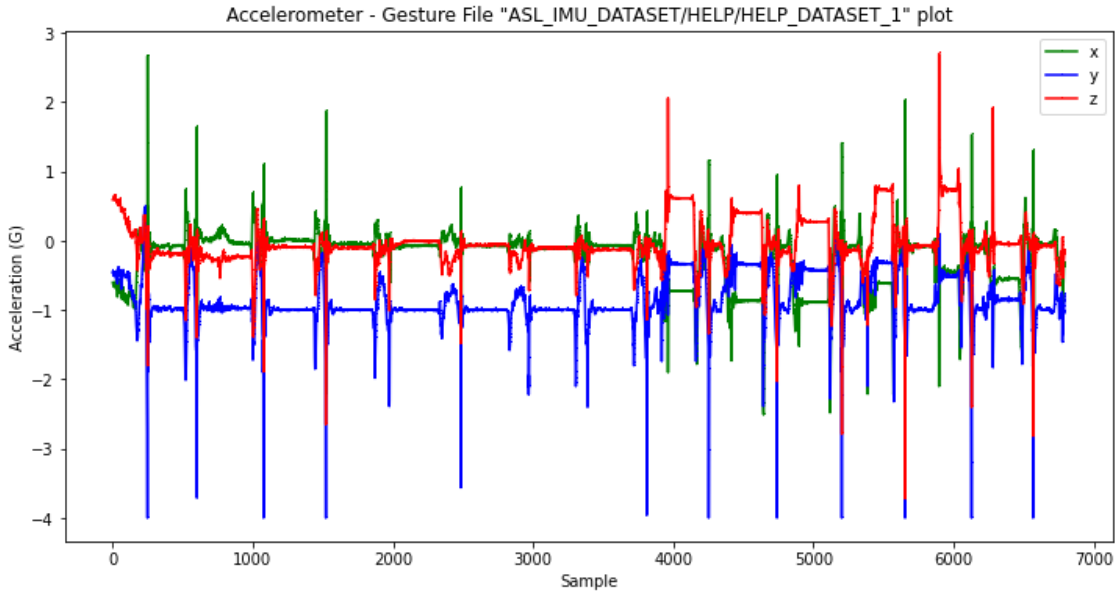
Source: https://docs.arduino.cc/tutorials/nano-33-ble/imu-gyroscope

Sensing-based Method - Dataset and Features

ASL IMU Dataset - Accelerometer



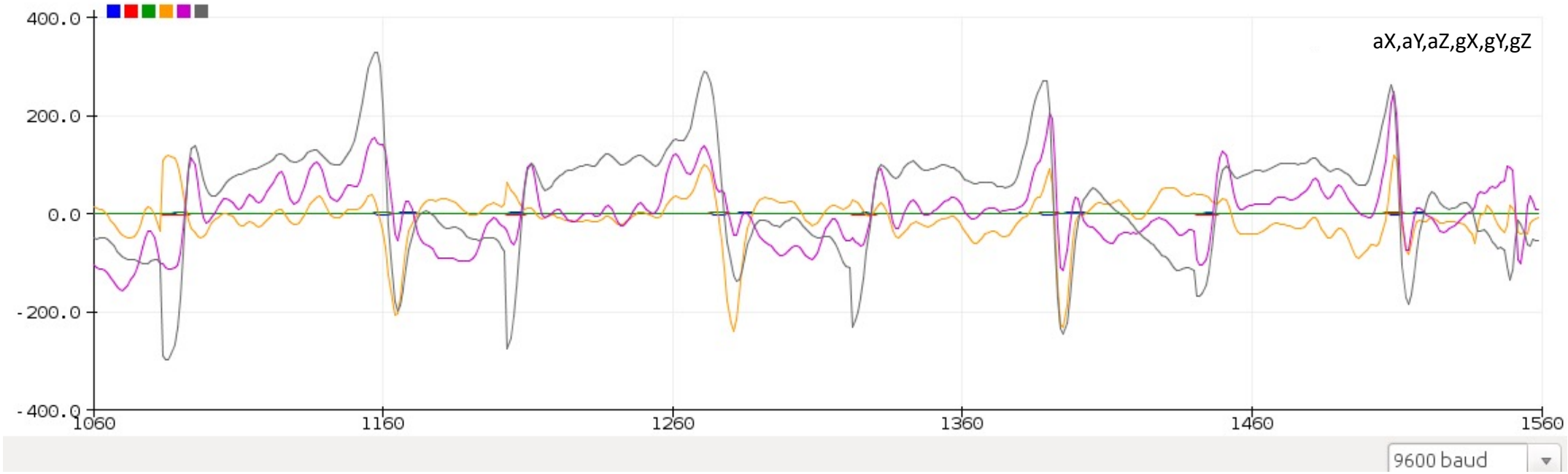
Thank You



Help

Sensing-based Method - Dataset and Features

ASL IMU Dataset – Accelerometer and Gyroscope



Thank You

Vision-based Method – Dataset and Features

Sign Language MNIST

- No cases for the letters J & Z (Reason: J & Z require motion)
- Grayscale Images
- Pixel Values ranging from 0 to 255
- Each image contains 784 Pixels

Source: Sign Language MNIST | Kaggle

Data Pre-Processing

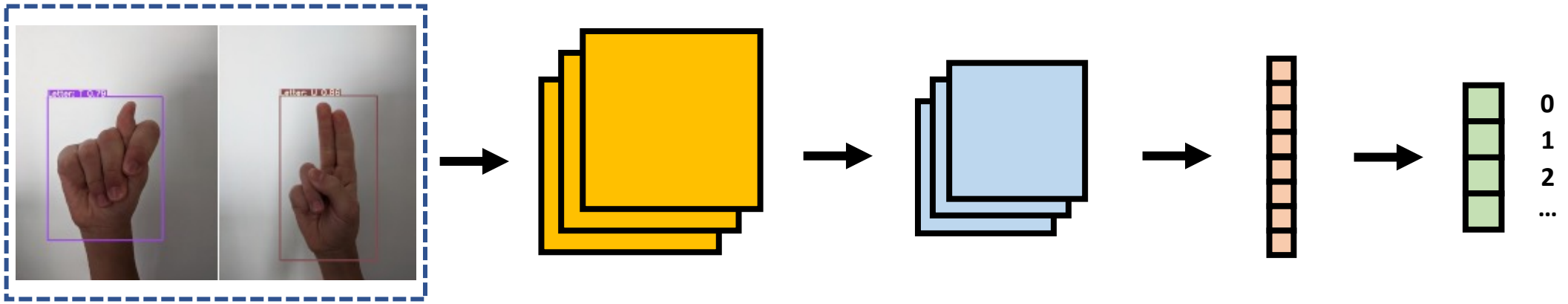
- ASL model must be able to work with real time data considerations:
  - Human hand need not always be straight while using gestures
  - There will be background images and disturbances

Dataset with and without a clear background

Data Augmentation to avoid overfitting

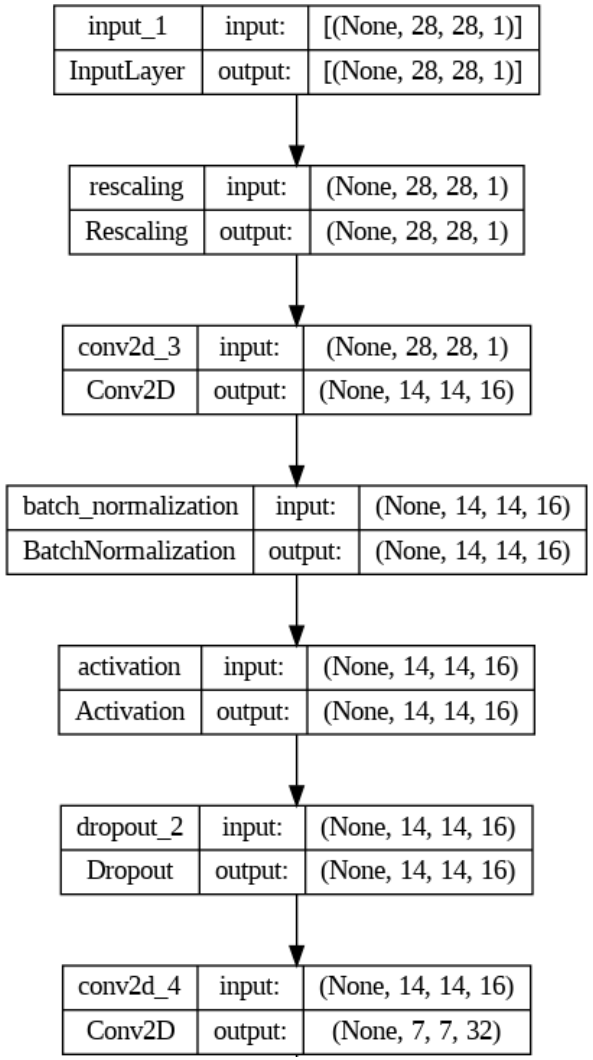
CNN for ASL

Convolutional Neural Networks (CNNs)



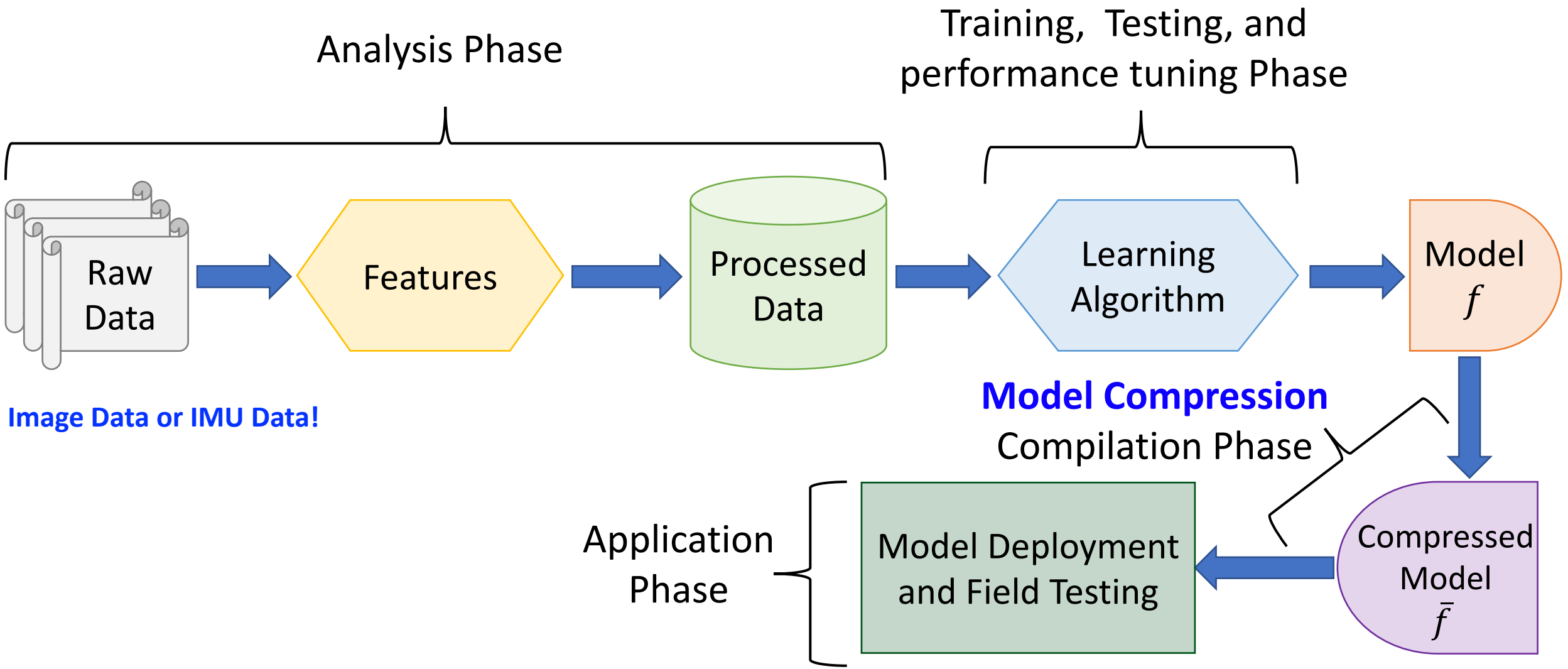
Our CNN Model

- Our CNN model classifies ASL alphabets
- Input image size: **28 x 28**
- Output classes: **26**



Note: The Arduino Nano 33 BLE has a camera module which takes input images in 32 x 32 so necessary data pre-processing must be done before running our model.

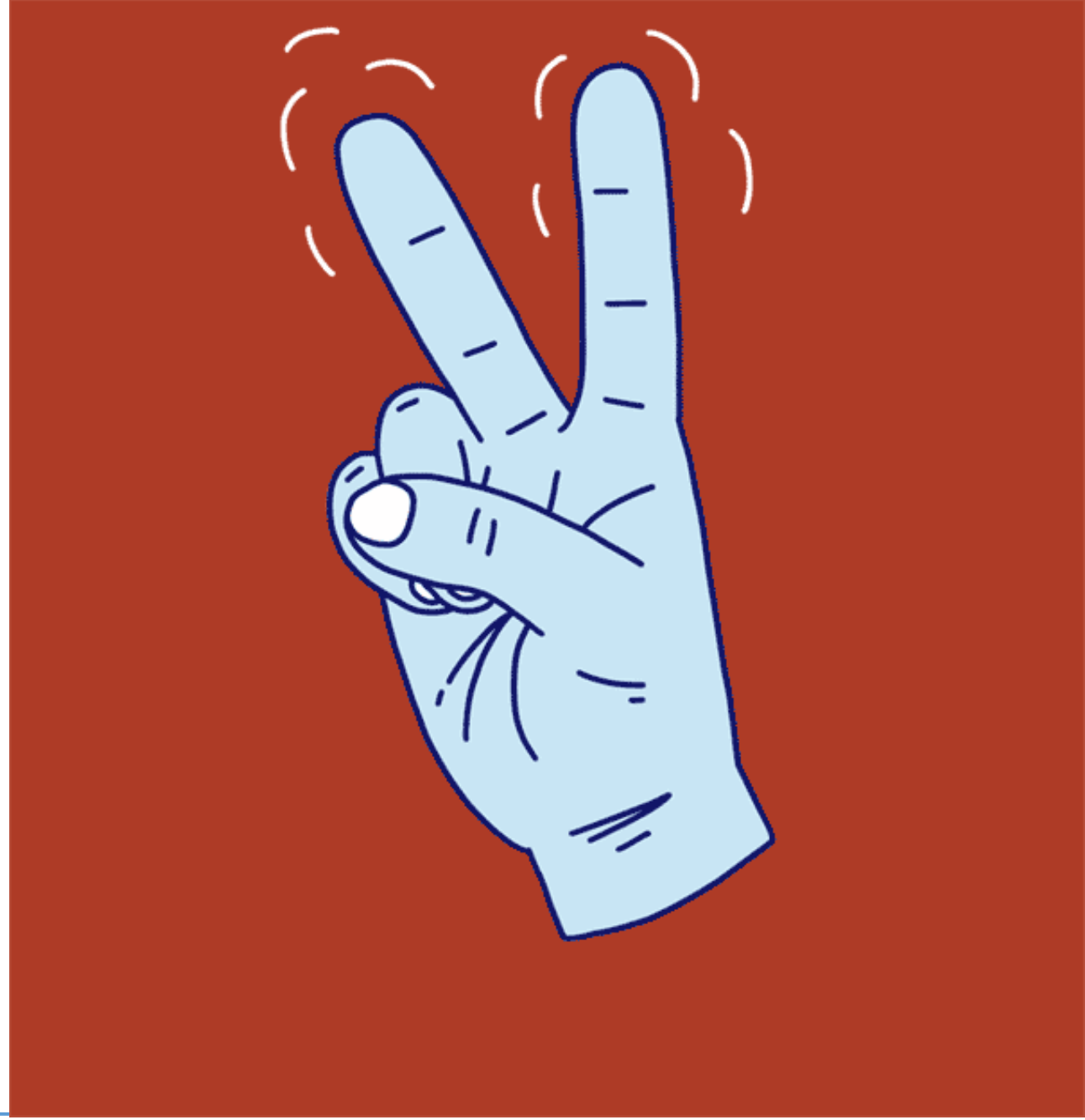
A Schematic View of TinyML Pipeline



5 min Break

Today's Lab

- Both Software and Hardware Lab!
- Training models to recognize the alphabet
- Quantize the model for TinyML
- Deploy the model to the Arduino board



Lab 8 - Software - Road Map

1. Gesture Based ASL recognition
2. Model training
3. Model Evaluation
4. Covert the TensorFlow model to TensorFlow Lite Micro model
5. Compress the model

Lab 8 - Hardware - Road Map

1. Connect Arduino Boards to your computer

2. Implement ASL recognition using compressed ML model

3. Deploy the model on to board

4. Open Serial Monitor and test the model

What you need

- Mac, Linux or Windows laptop
- Arduino Nano BLE Sense 33
- MicroUSB cable
- Possibly a USB C to USB A converter, if you're on a USB C laptop
- Optional: Wearable Glove and Sticky tape

Source: https://towardsdatascience.com/k-means-a-complete-introduction

Training and Quantize the Model

Open EEP595-TinyML-Lab8-Vision.ipynb (Gesture Based ASL) in Colab

Let’s train our model!

Source: https://towardsdatascience.com/k-means-a-complete-introduction

Training and Quantize the Model

Open EEP595-TinyML-Lab8-Gesture.ipynb (Gesture Based ASL) in Colab

Let’s train our model!

You will be able to generate a .cc file which is your compressed model.

Deploy the Model

1. Use a USB cable to connect the Arduino Nano 33 BLE Sense to your machine. You should see the green LED power indicator come on when the board first receives power.
2. Open the lab8-classifier.ino sketch, which you can find in the Slack channel.

Deploy the Model

3. As always, use the Tools drop-down menu to select the appropriate Port and Board, and use the rightward arrow to upload / flash the code.

4. Now, open the serial monitor and test out your custom model. As a reminder, the serial monitor will output two words as well as a confidence score for each of the gestures. The confidence score indicates how strongly the model believes you performed the gesture. Do note that every time you move the board and then stop, a new gesture will be processed, so don't be surprised to get some odd results as you move the board to prepare for a gesture.