Data Pre-Processing for Hands-on Keyword Spotting

Brian Plancher
Harvard John A. Paulson School of Engineering and Applied Sciences
Barnard College, Columbia University
brianplancher.com
Quick Disclaimer:

Today will be **both too fast** and **too slow**!
Quick Disclaimer:

Today will be *both* too fast and *too slow*!
Feedback from yesterday:

The depth of material covered today was
15 responses

- 73.3% Just right
- 20% Too high level
- 6.7% Too detailed
Feedback from yesterday:

The pace of the lab today was
15 responses

- 60% Too fast
- 40% Just right
- Too slow
Feedback from yesterday:

The pace of the lab today was
15 responses

Please add more information about how to use your cell phone!
If we pick a simple task to only identifying a few key words we can then use a small model and train it with little data and fit it onto an embedded device.
We will explore the science behind KWS and collect data and train our own custom model to recognize “yes” vs. “no” using Edge Impulse.
Today’s Agenda

- A Quick Review of What We’ve Learned
- Data Engineering for KWS
- Hands-on KWS Data Collection with Edge Impulse
- Training our Model using Transfer Learning
- Deploying our Model onto our Arduino
- Summary
Today’s Agenda

- A Quick Review of What We’ve Learned
- Data Engineering for KWS
- Hands-on KWS Data Collection with Edge Impulse
- (Hands-on) Data Preprocessing for KWS
- Deploying our Model onto our Arduino
- Summary
Machine Learning

We provide answers, aka the labels of the data.

Inputs → The computer learns → Rules
Deep Learning with **Neural Networks**
Features can be found with **Convolutions**
Convolutional Neural Networks
The **TinyML Workflow**

1. **Collect Data**
2. **Preprocess Data**
3. **Design a Model**
4. **Train a Model**
5. **Evaluate/ Optimize**
6. **Convert Model**
7. **Deploy Model**
8. **Make Inferences**

**Dataset** → **Impulse** → **Test** → **Deploy**

**Camera feed**
Starting inferencing in 2 seconds...
Taking photo...
Predictions (DSP): 9 ms., Classification:
car: 0.07812
truck: 0.92188
Today’s Agenda

- A Quick Review of What We’ve Learned
- **Data Engineering for KWS**
- Hands-on KWS Data Collection with Edge Impulse
- (Hands-on) Data Preprocessing for KWS
- Deploying our Model onto our Arduino
- Summary
Data Engineering for KWS
(How to collect good data)
Data Engineering for KWS
(How to collect good data)

Who will use your ML model?

- What **languages** will they speak?
- What **accents** will they have?
- Will they use **slang** or formal diction?
Data Engineering for KWS
(How to collect good data)

**Who** will use your ML model?
- What **languages** will they speak?
- What **accents** will they have?
- Will they use **slang** or formal diction?

**Where** will your ML model be used?
- Will there be **background noise**?
- **How far** will users be from the microphone?
- Will there be **echos**?
Data Engineering for KWS
(How to collect good data)

Who will use your ML model?

- What languages will they speak?
- What accents will they have?
- Will they use slang or formal diction?

Where will your ML model be used?

- Will there be background noise?
- How far will users be from the microphone?
- Will there be echos?

Why will your ML model be used?

- What tone of voice will be used?
- Are your keywords commonly used? (aka will you get a lot of false positives)
- What about false negatives?

Why those Keywords?
There are a lot more things to consider to **eliminate bias** and **protect privacy** when collecting data that we will talk about in future sessions!
Tips and Tricks for Custom KWS

- Pick **uncommon words** for Keywords
- Record lots of “**other words**”
- Record in the **location** you are going to be **deploying**
- Get **your end users** to help you build a dataset
- Record with the same **hardware** you will **deploy**
- Always **test** and then **improve** your dataset and model
Tips and Tricks for Custom KWS

Today we are just working on a demo so to give our demo the best chance of working we will:

1. **Stay in one spot** (we’re cheating)
2. **Only record ourselves**
3. **Use common words (yes, no)**
4. **Only test ourselves**
Today’s Agenda

- A Quick Review of What We’ve Learned
- Data Engineering for KWS
- Hands-on KWS Data Collection with Edge Impulse
- (Hands-on) Data Preprocessing for KWS
- Deploying our Model onto our Arduino
- Summary
Today we’ll also collect all of our data using Edge Impulse... ...and deploy to your cell phone as well.
Edge Impulse Project Dashboard

- Collect Data
- Preprocess Data
- Design a Model
- Train a Model
- Evaluate
- Optimize
- Convert Model
- Deploy Model
- Make Inferences

Dataset → Impulse → Test → Deploy
Create an Edge Impulse Account

1. Create an Edge Impulse account: https://studio.edgeimpulse.com/signup

2. Validate your email by clicking the link in the email sent to your account’s email address
Activity: Create a Keyword Spotting Dataset

Collect ~30 samples each of the following classes of data:

• **Keyword #1 “yes”** (label: yes) (length: 1 seconds)

• **Keyword #2 “no”** (label: no) (length: 1 seconds)

• **“Unknown” words** that are not the keyword and **background noise** (label: unknown) (length: 1 seconds)
Activity: Create a Keyword Spotting Dataset

Collect ~30 samples each of the following classes of data:

- **Keyword #1 “yes”** (label: yes) (length: 1 seconds)
- **Keyword #2 “no”** (label: no) (length: 1 seconds)
- **“Unknown” words** that are not the keyword and background noise

I’ve pre-loaded in a bunch of background noise and unknown words!

[https://docs.edgeimpulse.com/docs/pre-built-datasets/keyword-spotting](https://docs.edgeimpulse.com/docs/pre-built-datasets/keyword-spotting)
Clone succeeded
You're now ready to build your next embedded Machine Learning project!

Clone progress

```
[1/100/3004] Restoring files...
[2/100/3004] Restoring files...
[3/100/3004] Restoring files...
[4/100/3004] Restoring files...
[5/100/3004] Restoring files...
[6/100/3004] Restoring files...
[7/100/3004] Restoring files...
[8/100/3004] Restoring files...
[9/100/3004] Restoring files...
[10/100/3004] Restoring files...
```

Job completed
I’ve pre-loaded in a bunch of noise and unknown words!
You can collect data from development boards, from your own devices, or by uploading an existing dataset.

**Connect a fully supported development board**
Get started with real hardware from a wide range of silicon vendors - fully supported by Edge Impulse.

**Use your mobile phone**
Use your mobile phone to capture movement, audio or images, and even run your trained model locally. No app required.

**Use your computer**
Capture audio or images from your webcam or microphone, or from an external audio device.

**Data from any device with the data forwarder**
Capture data from any device or development board over a serial connection, in 10 lines of code.

**Upload data**
Already have data? You can upload your existing datasets directly in WAV, JPG, PNG, CBOR, CSV or JSON format.
Collect data

You can collect data from development boards, from your own devices, or by uploading an existing dataset.

Connect a fully supported development board

Get started with real hardware from a wide range of silicon vendors - fully supported by Edge Impulse.

Show QR code

Collect data

Point your phone camera at the QR code and open the link!

Capture data from any device or development board over a serial connection, in 10 lines of code.

Upload data

Already have data? You can upload your existing datasets directly in WAV, JPG, PNG, CBOR, CSV or JSON format.
Connected as phone_kunh8zjd

You can collect data from this device from the Data acquisition page in the Edge Impulse studio.

- Collecting images?
- Collecting audio?
- Collecting motion?
Connected as phone_kunh8zjd

You can collect data from this device from the Data acquisition page in the Edge Impulse studio.

Collecting audio?
Connected as phone_kunh8zjd

You can collect data from this device from the Data acquisition page in the Edge Impulse studio.

- 📈 Collecting images?
- 🔊 Collecting audio?
- 🎧 Collecting motion?

Data collection

Label: goodbye
Length: 3s.
Category: split

Start recording

Audio captured with current settings: 0s
Connected as phone_kunh8zjd

You can collect data from this device from the Data acquisition page in the Edge Impulse studio.

Collecting images?
Collecting audio?
Collecting motion?
Connected as phone_kunh8zjd

You can collect data from this device from the Data acquisition page in the Edge Impulse studio.

https://docs.edgeimpulse.com/docs/using-your-mobile-phone
You may need to re-flash the EI Firmware!

1. Double tap RESET to enter bootloader mode


3. Run the flash script for your operating system
   (flash_windows.bat, flash_mac.command or flash_linux.sh).

4. Wait until flashing is complete, and press the RESET button once to launch the new firmware.
edgeimpulse.com wants to connect to a serial port

Nano 33 BLE (ttyACM0) - Paired

- ttyS0
- ttyS1
- ttyS10
- ttyS11
- ttyS12
- ttyS13
- ttyS14

[Button] Connect
Did you know? You can capture data from any device or development board, or upload your existing datasets - Show options

**Record new data**

- **Device**: [Dropdown menu]
- **Label**: `yes`
- **Sample length (ms.)**: `10000`

**Collected data**

<table>
<thead>
<tr>
<th>SAMPLE_NAME</th>
<th>LABEL</th>
<th>ADDED</th>
<th>LENGTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>noise.orig_train.Hallway_1.wav.70...</td>
<td>noise</td>
<td>Today, 11:22:57</td>
<td>1s</td>
</tr>
<tr>
<td>noise.orig_train.Metro_1.wav.2970...</td>
<td>noise</td>
<td>Today, 11:22:57</td>
<td>1s</td>
</tr>
<tr>
<td>noise.orig_train.CafeTeria_1.wav.2.wav...</td>
<td>noise</td>
<td>Today, 11:22:57</td>
<td>1s</td>
</tr>
<tr>
<td>noise.orig_train.AirportAnnounce...</td>
<td>noise</td>
<td>Today, 11:22:57</td>
<td>1s</td>
</tr>
<tr>
<td>noise.running_tap.wav.25000</td>
<td>noise</td>
<td>Today, 11:22:57</td>
<td>1s</td>
</tr>
<tr>
<td>noise.orig_train.Station_1.wav.2.wav.2013...</td>
<td>noise</td>
<td>Today, 11:22:57</td>
<td>1s</td>
</tr>
<tr>
<td>noise.orig_train.AirConditioner_9...</td>
<td>noise</td>
<td>Today, 11:22:57</td>
<td>1s</td>
</tr>
<tr>
<td>Sample Name</td>
<td>Label</td>
<td>Added</td>
<td>Length</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-------</td>
<td>----------------</td>
<td>--------</td>
</tr>
<tr>
<td>yes.30u5okgq</td>
<td>yes</td>
<td>Today, 14:24:58</td>
<td>10s</td>
</tr>
<tr>
<td>noise.orig_train.Hallway_1.wav.7...</td>
<td>noise</td>
<td>Today, 11:22:57</td>
<td></td>
</tr>
<tr>
<td>noise.orig_train.Metro_1.wav.297...</td>
<td>noise</td>
<td>Today, 11:22:57</td>
<td></td>
</tr>
<tr>
<td>noise.orig_train.AirportAnnounc...</td>
<td>noise</td>
<td>Today, 11:22:57</td>
<td></td>
</tr>
<tr>
<td>noise.running_tap.wav.29000</td>
<td>noise</td>
<td>Today, 11:22:57</td>
<td></td>
</tr>
<tr>
<td>noise.orig_train.Station_1.wav.30</td>
<td>noise</td>
<td>Today, 11:22:57</td>
<td></td>
</tr>
</tbody>
</table>

- **Split sample** option highlighted.
### Collected data

<table>
<thead>
<tr>
<th>SAMPLE_NAME</th>
<th>LABEL</th>
<th>ADDED</th>
<th>LENGTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>yes.3u5okgq.s5</td>
<td>yes</td>
<td>Today, 14:31:19</td>
<td>1s</td>
</tr>
<tr>
<td>yes.3u5okgg.s4</td>
<td>yes</td>
<td>Today, 14:31:19</td>
<td>1s</td>
</tr>
<tr>
<td>yes.3u5okgg.s3</td>
<td>yes</td>
<td>Today, 14:31:19</td>
<td>1s</td>
</tr>
<tr>
<td>yes.3u5okgg.s2</td>
<td>yes</td>
<td>Today, 14:31:19</td>
<td>1s</td>
</tr>
<tr>
<td>yes.3u5okgg.s1</td>
<td>yes</td>
<td>Today, 14:31:19</td>
<td>1s</td>
</tr>
<tr>
<td>noise.orig_train.Metro_1.wav.297...</td>
<td>noise</td>
<td>Today, 11:22:37</td>
<td>1s</td>
</tr>
<tr>
<td>noise.orig_train.AirportAnnounc...</td>
<td>noise</td>
<td>Today, 11:22:37</td>
<td>1s</td>
</tr>
<tr>
<td>noise.running_tap.wav.29000</td>
<td>noise</td>
<td>Today, 11:22:37</td>
<td>1s</td>
</tr>
<tr>
<td>noise.orig_train.Station_1.wav.20...</td>
<td>noise</td>
<td>Today, 11:22:37</td>
<td>1s</td>
</tr>
</tbody>
</table>

### Record new data

- **Device**: SF-RE4RE-1123
- **Label**: yes
- **Sample length (ms.)**: 10000
- **Sensor**: Built-in microphone
- **Frequency**: 16000Hz

**Start sampling**
Activity: Create a Keyword Spotting Dataset

Collect ~30 samples each of the following classes of data:

• Keyword #1 “yes” (label: yes) (length: 1 seconds)
• Keyword #2 “no” (label: no) (length: 1 seconds)

We’ll resume in 10 minutes!
Dashboard

Data acquisition

Impulse design

Create impulse

EON Tuner

Retrain model

Live classification

Model testing

Versioning

Deployment

GETTING STARTED

Documentation

Forums

Scroll Down to the Bottom

Danger zone

Perform train / test split

Launch getting started wizard

Transfer ownership

Delete this project

Delete all data in this project
**Data Collected**

- **Duration**: 40m 29s

**Train/Test Split**

- **80%** / **20%**

**Collected Data**

<table>
<thead>
<tr>
<th>Sample Name</th>
<th>Label</th>
<th>Added</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>no.30u6lbcns5</td>
<td>no</td>
<td>Today, 14:40:45</td>
<td>1s</td>
</tr>
<tr>
<td>no.30u6lbcns4</td>
<td>no</td>
<td>Today, 14:40:45</td>
<td>1s</td>
</tr>
<tr>
<td>no.30u6lbcns3</td>
<td>no</td>
<td>Today, 14:40:45</td>
<td>1s</td>
</tr>
<tr>
<td>no.30u6lbcns2</td>
<td>no</td>
<td>Today, 14:40:45</td>
<td>1s</td>
</tr>
<tr>
<td>no.30u6lbcns1</td>
<td>no</td>
<td>Today, 14:40:45</td>
<td>1s</td>
</tr>
<tr>
<td>no.30u6k9u9s5</td>
<td>no</td>
<td>Today, 14:40:13</td>
<td>1s</td>
</tr>
<tr>
<td>no.30u6k9u9s4</td>
<td>no</td>
<td>Today, 14:40:13</td>
<td>1s</td>
</tr>
</tbody>
</table>
Did you know? You can capture data from any device and upload your existing data. Click to learn how to rebalance your dataset.

**Collected data**

<table>
<thead>
<tr>
<th>Sample Name</th>
<th>Label</th>
<th>Added</th>
<th>Length</th>
</tr>
</thead>
<tbody>
<tr>
<td>noise.orig_train.Metro_1</td>
<td>noise</td>
<td>Today, 11:22:57</td>
<td>1s</td>
</tr>
<tr>
<td>noise.orig_train.CafeTeri_1</td>
<td>noise</td>
<td>Today, 11:22:57</td>
<td>1s</td>
</tr>
<tr>
<td>noise.orig_train.AirCond_1</td>
<td>noise</td>
<td>Today, 11:22:57</td>
<td>1s</td>
</tr>
</tbody>
</table>
Dataset train / test split ratio

**Training data** is used to train your model, and **testing data** is used to test your model's accuracy after training. We recommend an approximate 80/20 train/test split ratio for your data for every class (or label) in your dataset, although especially large datasets may require less testing data.

**Suggested train / test split**

80% / 20%

**Labels in your dataset**

The 'no' class has a poor train/test split ratio. To fix this, add or move samples to the training or testing data.

**NO**

100% / 0% (27s / 0s)

**NOISE**

80% / 20% (20m 22s / 5m 13s)

**UNKNOWN**

80% / 20% (19m 52s / 5m 7s)

**YES**

81% / 19% (22s / 5s)

**Perform train / test split**

Use this option to rebalance your data, automatically splitting items between training and testing datasets.

**Warning: this action cannot be undone.**
<table>
<thead>
<tr>
<th>SAMPLE NAME</th>
<th>LABEL</th>
<th>ADDED</th>
<th>LENGTH</th>
</tr>
</thead>
<tbody>
<tr>
<td>no.30u8qcwv.s1</td>
<td>no</td>
<td>Today, 15:22:58</td>
<td>1s</td>
</tr>
<tr>
<td>no.30u6k9u9.s5</td>
<td>no</td>
<td>Today, 15:22:58</td>
<td></td>
</tr>
<tr>
<td>no.30u6k9u9.s1</td>
<td>no</td>
<td>Today, 15:22:58</td>
<td></td>
</tr>
<tr>
<td>no.30u8qcwv.s9</td>
<td>no</td>
<td>Today, 15:22:49</td>
<td></td>
</tr>
<tr>
<td>no.30u8qcwv.s7</td>
<td>no</td>
<td>Today, 15:22:44</td>
<td></td>
</tr>
<tr>
<td>yes.30u8rq7l.s8</td>
<td>yes</td>
<td>Today, 15:20:19</td>
<td></td>
</tr>
<tr>
<td>yes.30u8rq7l.s7</td>
<td>yes</td>
<td>Today, 15:20:19</td>
<td></td>
</tr>
</tbody>
</table>
Dataset train / test split ratio

Training data is used to train your model, and testing data is used to test your model's accuracy after training. We recommend an approximate 80/20 train/test split ratio for your data for every class (or label) in your dataset, although especially large datasets may require less testing data.

Labels in your dataset

The 'no' class has a poor train/test split ratio. To fix this, add or move samples to the training or testing data.
Today’s Agenda

- A Quick Review of What We’ve Learned
- Data Engineering for KWS
- Hands-on KWS Data Collection with Edge Impulse
- (Hands-on) Data Preprocessing for KWS
- Deploying our Model onto our Arduino
- Summary
Edge Impulse Project Dashboard

- Collect Data
- Preprocess Data
- Design a Model
- Train a Model
- Evaluate
- Optimize
- Convert Model
- Deploy Model
- Make Inferences

Dataset -> Impulse -> Test -> Deploy

- Dashboard
- Devices
- Data acquisition
- Impulse design
- Create impulse
- EON Tuner
- Retrain model
- Live classification
- Model testing
- Versioning
- Deployment
Why might we want to **preprocess** data and not send the raw data to the neural network?
Can you tell these two signals apart?

“Yes” *(spoken loudly)*

“No” *(spoken loudly)*
Signal **Components**?
Signal **Components**?

? + ? = "No" (spoken loudly)
Fast Fourier Transform:
extract the frequencies from a signal
Fast Fourier Transform
Building a **Spectrogram** using FFTs
Building a **Spectrogram** using FFTs
Building a **Spectrogram** using FFTs
Building a **Spectrogram** using FFTs

Essentially if you stack up all the FFTs in a row then you get the **Spectrogram** (time vs. frequency with color indicating intensity)
Spectrograms help differentiate the data
Spectrograms help differentiate the data
Spectrograms help differentiate the data
A spectrogram is also effectively an image that we can use as an input to a CNN!
Can we do better than a spectrogram?

Can we take domain knowledge into account?
Mel Filterbanks
Spectrograms v. MFCCs
Spectrograms v. MFCCs
Additional **Feature Engineering**

- **Normalization:** remove volume differences
- **Denoise:** reduce background noise for clarity
WARNING: Whatever preprocessing you do on the computer in python for training you need to do in C++ on the microcontroller!
Today’s Agenda

- Deep ML Background
- Hands-on Computer Vision: Thing Translator
- The Tiny Machine Learning Workflow
- Keyword Spotting (KWS) Data Collection
- KWS Preprocessing and Training
  - Preprocessing (for KWS)
  - Hands-on Preprocessing and Training with Edge Impulse
- Deployment Challenges and Opportunities for Embedded ML
- Summary
An impulse takes raw data, uses signal processing to extract features, and then uses a learning block to classify new data.
An impulse takes raw data, uses signal processing to extract features, and then uses a learning block to classify new data.
We’ll keep things simple today and just add an MFCC but/and in future projects you can:

- create your own blocks
- use multiple blocks

https://docs.edgeimpulse.com/docs/custom-blocks
An impulse takes raw data, uses signal processing to extract features, and then uses a learning block to classify new data.
Add a learning block

Some learning blocks have been hidden based on the data in your project.

<table>
<thead>
<tr>
<th>DESCRIPTION</th>
<th>AUTHOR</th>
<th>RECOMMENDED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classification (Keras)</td>
<td>Edgelapse Inc.</td>
<td></td>
</tr>
<tr>
<td>Learn patterns from data, and can apply these to new data. Great for categorizing movement or recognizing audio.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Regression (Keras)</td>
<td>Edgelapse Inc.</td>
<td></td>
</tr>
<tr>
<td>Learn patterns from data, and can apply these to new data. Great for predicting numeric continuous values.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Add

Cancel
### Training set

<table>
<thead>
<tr>
<th>Data in training set</th>
<th>40m 29s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classes</td>
<td>4 (no, noise, unknown, yes)</td>
</tr>
<tr>
<td>Training windows</td>
<td>2,419</td>
</tr>
</tbody>
</table>
## Training set

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data in training set</td>
<td>40m 29s</td>
</tr>
<tr>
<td>Classes</td>
<td>4 (no, noise, unknown, yes)</td>
</tr>
<tr>
<td>Training windows</td>
<td>2,429</td>
</tr>
</tbody>
</table>

Generating features...

## Feature generation output

Creating job... OK (ID: 2596742)
Scheduling job in cluster...
Job started
Creating windows from 2429 files...
[2/3] Pre-caching files...
[3/3] Pre-caching files...
Pre-caching files OK
[ 1/2429] creating windows from files...
If you can visually see the clustering of the data then it is easier for the ML model to learn! (But its not required and provides no guarantees)
Neural Network settings

Training settings
Number of training cycles: 100
Learning rate: 0.005
Validation set size: 20%
Auto-balance dataset: Off

Audio training options
Data augmentation: Off

Neural network architecture
Architecture presets: 1D Convolutional (Default), 2D Convolutional

Input layer (550 features)
Reshape layer (13 columns)
1D conv / pool layer (8 neurons, 3 kernel size, 1 layer)
Model Design with Edge Impulse

Pre-made neural network “blocks” that you can add!
“Expert” mode to write your own TensorFlow code
```python
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, InputLayer, Dropout, Conv1D, Conv2D,
                              Flatten, Reshape, MaxPooling1D, MaxPooling2D, BatchNormalization,
                              TimeDistributed
from tensorflow.keras.optimizers import Adam

# model architecture

model.add(Reshape((int(input_length / 13), 13), input_shape=(input_length, )))
model.add(Conv1D(8, kernel_size=3, activation='relu', padding='same'))
model.add(MaxPooling1D(pool_size=2, strides=2, padding='same'))

model.add(Conv1D(16, kernel_size=3, activation='relu', padding='same'))
model.add(MaxPooling1D(pool_size=2, strides=2, padding='same'))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(classes, activation='softmax', name='y_pred'))

# this controls the learning rate
opt = Adam(lr=0.005, beta_1=0.9, beta_2=0.999)
# this controls the batch size, or you can manipulate the tf.data.Dataset objects yourself!
BATCH_SIZE = 32
train_dataset = train_dataset.batch(BATCH_SIZE, drop_remainder=False)
validation_dataset = validation_dataset.batch(BATCH_SIZE, drop_remainder=False)
callbacks = [BatchLoggerCallback(BATCH_SIZE, train_sample_count)]

# train the neural network
model.compile(loss='categorical_crossentropy', optimizer=opt, metrics=['accuracy'])
model.fit(train_dataset, epochs=100, validation_data=validation_dataset, verbose=2, callbacks=callbacks)
```
For now just stick with the defaults but/and you can easily design any model you want and use any optimizer you want using TensorFlow!
For now just stick with the defaults but you can easily design any model you want and use any optimizer you want using TensorFlow!

**WARNING:** if you want to deploy to a microcontroller make sure you only use Ops supported by TensorFlow Lite Micro!

[GitHub Link](https://github.com/tensorflow/tflite-micro/blob/main/tensorflow/lite/micro/all_ops_resolver.cc#L22)
For now just stick with the defaults but/and you can easily design any model you want and use any optimizer you want using TensorFlow!
Training output

Epoch 95/100
4/4 - 0s - loss: 0.1044 - accuracy: 0.9500 - val_loss: 0.2934 - val_accuracy: 0.9231
Epoch 96/100
4/4 - 0s - loss: 0.0256 - accuracy: 1.0000 - val_loss: 0.3830 - val_accuracy: 0.8846
Epoch 97/100
4/4 - 0s - loss: 0.0523 - accuracy: 0.9800 - val_loss: 0.4366 - val_accuracy: 0.8462
Epoch 98/100
4/4 - 0s - loss: 0.0451 - accuracy: 0.9800 - val_loss: 0.4265 - val_accuracy: 0.8846
Epoch 99/100
4/4 - 0s - loss: 0.0514 - accuracy: 0.9900 - val_loss: 0.3926 - val_accuracy: 0.8846
Epoch 100/100
4/4 - 0s - loss: 0.0348 - accuracy: 0.9900 - val_loss: 0.3571 - val_accuracy: 0.9231
Finished training
Final Accuracy
Final Accuracy

Accuracy Breakdown
## Confusion Matrix

<table>
<thead>
<tr>
<th></th>
<th>Actual Output = Yes</th>
<th>Actual Output = No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Predicted Output = Yes</td>
<td># of True Positive</td>
<td># of False Positive</td>
</tr>
<tr>
<td>Predicted Output = No</td>
<td># of False Negative</td>
<td># of True Negative</td>
</tr>
</tbody>
</table>

**Type 1 Error**

**Type 2 Error**
Final Accuracy
Accuracy Breakdown
Feature Explorer
Individual Data Points

Model

Last training performance (validation set)

- ACCURACY: 96.6%
- LOSE: 0.09

Confusion matrix (validation set)

<table>
<thead>
<tr>
<th></th>
<th>NO</th>
<th>NOISE</th>
<th>UNKNOWN</th>
<th>YES</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO</td>
<td>100%</td>
<td>0%</td>
<td>0%</td>
<td>0%</td>
</tr>
<tr>
<td>NOISE</td>
<td>0%</td>
<td>96.0%</td>
<td>0.0%</td>
<td>0%</td>
</tr>
<tr>
<td>UNKNOWN</td>
<td>0%</td>
<td>0.0%</td>
<td>97.6%</td>
<td>0.8%</td>
</tr>
<tr>
<td>YES</td>
<td>0%</td>
<td>0%</td>
<td>20%</td>
<td>80%</td>
</tr>
<tr>
<td>F1 SCORE</td>
<td>1.00</td>
<td>0.97</td>
<td>0.95</td>
<td>0.90</td>
</tr>
</tbody>
</table>

Feature explorer (full training set)

- noise - correct
- noise - incorrect
- unknown - correct
- unknown - incorrect
- yes - correct
- yes - incorrect

On-device performance

- INFERENCING TIME: 11 ms.
- PEAK RAM USAGE: 5.0K
- FLASH USAGE: 34.8K

113
Final Accuracy

Accuracy Breakdown

Feature Explorer

Individual Data Points

Expected runtime/memory
Today’s Agenda

- A Quick Review of What We’ve Learned
- Data Engineering for KWS
- Hands-on KWS Data Collection with Edge Impulse
- (Hands-on) Data Preprocessing for KWS
- Deploying our Model onto our Arduino
- Summary
Edge Impulse Project Dashboard

[Diagram showing the project lifecycle: Collect Data, Preprocess Data, Design a Model, Train a Model, Evaluate Model, Optimize Model, Convert Model, Deploy Model, Make Inferences, Test, Deploy]

Edge Impulse Project Dashboard
Deploy your impulse

You can deploy your impulse to any device. This makes the model run without an internet connection, minimizes latency, and runs with minimal power consumption. Read more.

Create library

Turn your impulse into optimized source code for your development environment.

Build firmware

Get a ready-to-go binary for your development board that includes your impulse.
Select optimizations (optional)

Model optimizations can increase on-device performance but may reduce accuracy. Click below to analyze optimizations and see the recommended choices for your target. Or, just click Build to use the currently selected options.

Enable EON™ Compiler
Same accuracy, up to 50% less memory. Open source.

Available optimizations for Transfer learning

<table>
<thead>
<tr>
<th>Variant</th>
<th>RAM USAGE</th>
<th>LATENCY</th>
<th>FLASH USAGE</th>
<th>ACCURACY</th>
</tr>
</thead>
<tbody>
<tr>
<td>Quantized (int8)</td>
<td>66.1K</td>
<td>58 ms</td>
<td>108.1K</td>
<td>-</td>
</tr>
<tr>
<td>Unoptimized (float32)</td>
<td>155.6K</td>
<td>43 ms</td>
<td>193.8K</td>
<td>-</td>
</tr>
</tbody>
</table>

Click to select

Build
You can collect data from development boards, from your own devices, or by uploading an existing dataset.

**Connect a fully supported development board**
- Get started with real hardware from a wide range of silicon vendors fully supported by Edge Impulse.

**Use your mobile phone**
- Use your mobile phone to capture movement, audio or images, and even run your trained model locally. No app required.

Connected as phone_kunh8zjd
You can collect data from this device from the Data acquisition page in the Edge Impulse studio.

- Collecting images?
- Collecting audio?
- Collecting motion?

Switch to classification mode
Deploy and Test your Model

Shows the score for (confidence that the current sounds is) each of the various keywords and unknown and bolds the highest score.
Built Arduino library

Add this library through the Arduino IDE via:

**Sketch > Include Library > Add .ZIP Library...**

Examples can then be found under:

**File > Examples > SciTinyML22-KWS-TestClone_inferencing**
Built Arduino library

Add this library through the Arduino IDE via:

Sketch > Include Library > Add .ZIP Library...

Examples can then be found under:

File > Examples > SciTinyML22-KWS-TestClone_inferencing
nano_ble33_sense_microphone

/* Edge Impulse Arduino examples
 * Copyright (c) 2021 EdgeImpulse Inc.
 *
 * Permission is hereby granted, free of charge, to ar
 * of this software and associated documentation files
 * in the Software without restriction, including with
 * to use, copy, modify, merge, publish, distribute, s
 * copies of the Software, and to permit persons to wr
 * furnished to do so, subject to the following condit
 * *
 * The above copyright notice and this permission noti
 * all copies or substantial portions of the Software.
 *
 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY
 * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIE
 * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT

Compiling sketch...

Arduino Nano 33 BLE on /dev/ttyACM0
nano_ble33_sense_microphone

* Edge Impulse Arduino examples
* Copyright (c) 2021 EdgeImpulse Inc.

* Permission is hereby granted, free of charge, to any person obt
of this software and associated documentation files (the "Soft
in the Software without restriction, including without limitat
* to use, copy, modify, merge, publish, distribute, sublicense, * copies of the Software, and to permit persons to whom the Soft
* furnished to do so, subject to the following conditions:
*
* The above copyright notice and this permission notice shall be * all copies or substantial portions of the Software.
*
* THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, * IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCIA
* FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EV
* HOWEVER CAUSED, UNDER ANY LAW RELATING TO ANY CLAIM, DAMAGE

Done uploading.

Locked: none
Security: false
Erase flash

Done in 0.000 seconds
Write 173792 bytes to flash (43 pages)
[==================================] 169% (43/43 pages)
Done in 6.694 seconds

Arduino Nano 33 BLE on /dev/ttyACM0
An error occurred while uploading the sketch

/home/plancher/Arduino/libraries/test_image_2_inferencing/src/edge-impulse-sdk/CMSIS/NN/Source/PoolingFunctions/arm_pool_q7_HW0

*/SIMD32(pCnt)++ = __QADD16(v02, in);

/home/plancher/Arduino/libraries/test_image_2_inferencing/src/edge-impulse-sdk/tensorflow/lite/core/api/op_resolver.cpp: In function
/home/plancher/Arduino/libraries/test_image_2_inferencing/src/edge-impulse-sdk/tensorflow/lite/core/api/op_resolver.cpp:33:20:

builtin_code < BuiltInOperator_MIN) { ____________^____________________________

Sketch uses 224024 bytes (22%) of program storage space. Maximum is 983640 bytes.
Global variables use 58672 bytes (22%) of dynamic memory, leaving 203472 bytes for local variables. Maximum is 262144 bytes.
An error occurred while uploading the sketch

Device unsupported

Double Tap Reset for Bootloader Mode!
Tools > Serial Monitor
Confidence that the audio is the given class (0-1 scale)
Today’s Agenda

- A Quick Review of What We’ve Learned
- Data Engineering for KWS
- Hands-on KWS Data Collection with Edge Impulse
- (Hands-on) Data Preprocessing for KWS
- Deploying our Model onto our Arduino
- Summary
Machine Learning

We provide answers, aka the labels of the data.

Inputs -> Computer learns -> Rules
Deep Learning with Neural Networks
Features can be found with **Convolutions**
The **TinyML Workflow**

- Collect Data
- Preprocess Data
- Design a Model
- Train a Model
- Evaluate
- Optimize
- Convert Model
- Deploy Model
- Make Inferences

**Dataset** → **Impulse** → **Test** → **Deploy**
The TinyML Workflow

Who will use your ML model?

Where will your ML model be used?

Why will your ML model be used?

Why those Keywords?

Collect Data

Preprocess Data

Design a Model

Train a Model

Evaluate
Optimize

Convert
Model

Deploy
Model

Make
Inferences

Training Set

Validation Set

Test Set
The TinyML Workflow

- Collect Data
- Preprocess Data
- Design a Model
- Train a Model
- Evaluate
- Optimize
- Convert Model
- Deploy Model
- Make Inferences

FFT, Spectrogram, MFCC
The TinyML Workflow

Confusion Matrix

<table>
<thead>
<tr>
<th>Predicted Output = Yes</th>
<th>Actual Output = Yes</th>
<th>Actual Output = No</th>
</tr>
</thead>
<tbody>
<tr>
<td># of True Positive</td>
<td># of False Positive</td>
<td></td>
</tr>
</tbody>
</table>

- Type 1 Error
- Type 2 Error

<table>
<thead>
<tr>
<th>Predicted Output = No</th>
<th># of False Negative</th>
<th># of True Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type 2 Error</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The TinyML Workflow

- Collect Data
- Preprocess Data
- Design a Model
- Train a Model
- Evaluate
- Optimize
- Convert Model
- Deploy Model
- Make Inferences

Quantization

Reduces the precision of numbers used in a model which results in:

- smaller model size
- faster computation
The TinyML Workflow

Collect Data → Preprocess Data → Design a Model → Train a Model → Evaluate → Optimize → Convert Model → Deploy Model → Make Inferences

Edge Impulse Simplifies Deployment
Better Data = Better Models!
Data Pre-Processing for Hands-on Keyword Spotting

Brian Plancher
Harvard John A. Paulson School of Engineering and Applied Sciences
Barnard College, Columbia University
brianplancher.com
1. Install the Arduino CLI
   a. On linux:
      ```
      ```
   b. On mac:
      ```
      brew update
      brew install arduino-cli
      ```
   c. Or view the link for binaries
2. Add to your .bashrc:
   ```
   # Arduino (CLI)
   export PATH="ARDUINO_INSTALL_LOCATION/bin:$PATH"
   ```
   Where ARDUINO_INSTALL_LOCATION is e.g.,: $HOME/Documents/arduino-1.8.19
1. Install the Edge Impulse CLI
   a. Install Node.js by following the link or on Linux:
      
      curl -sl https://deb.nodesource.com/setup_14.x | sudo -E bash -
      sudo apt-get install -y nodejs
   b. Run: npm install -g edge-impulse-cli --force
   c. Add to your .bashrc:
      
      # EI (CLI)
      export PATH="$HOME/.npm-global/bin:$PATH"

2. Run `edge-impulse-daemon --clean` to start the daemon and then follow the instructions in the terminal to add it to your current project using your edge impulse account!

https://docs.edgeimpulse.com/docs/edge-impulse-cli/cli-installation
It should then appear on your “Devices” tab in your project!

And then if you go to “Data Acquisition” you should be able to proceed as you would with the standard instructions!