

Edge Computing Lab

Name: Rahul Bhati

Class: TY-AIEC

Roll no : 2223416

School of Computing, MIT Art Design Technology University

Academic Year: 2024-25

Experiment No. 7

Introduction

Study of Classification learning block using a NN Classifier on Edge Devices

Objective: Build a project to detect the keywords using built-in sensor on Nano BLE Sense / Mobile Phone

Tasks:

- Generate the dataset for keyword
- Configure BLE Sense / Mobile for Edge Impulse
- Building and Training a Model

Study of **Confusion matrix**

Introduction

Edge Impulse is a development platform for machine learning on edge devices, targeted at developers who want to create intelligent device solutions. The "classification block" equivalent in Edge Impulse would typically involve creating a simple machine learning model that can run on an edge device, like classifying sensor data or recognizing a basic pattern.

Materials Required

- Nano BLE Sense Board

Theory

GPIO (General Purpose Input/Output) pins on the Raspberry Pi are used for interfacing with other electronic components. BCM numbering refers to the pin numbers in the Broadcom SOC channel, which is a more consistent way to refer to the GPIO pins across different versions of the

Here's a high-level overview of steps you'd follow to create a "Hello World" project on Edge Impulse:

Steps to Configure the Edge Impulse:

1. Create an Account and New Project:
 - Sign up for an Edge Impulse account.
 - Create a new project from the dashboard.

2. Connect a Device:

- You can use a supported development board or your smartphone as a sensor device.
- Follow the instructions to connect your device to your Edge Impulse project.

3. Collect Data:

- Use the Edge Impulse mobile app or the Web interface to collect data from the onboard sensors.
- For a "Hello World" project, you could collect accelerometer data, for instance.

4. Create an Impulse:

- Go to the 'Create impulse' page.
- Add a processing block (e.g., time-series data) and a learning block (e.g., classification).
- Save the impulse, which defines the machine learning pipeline.

5. Design a Neural Network:

- Navigate to the 'NN Classifier' under the 'Learning blocks'.
- Design a simple neural network. Edge Impulse provides a default architecture that works well for most basic tasks.

6. Train the Model:

- Click on the 'Start training' button to train your machine learning model with the collected data.

7. Test the Model:

- Once the model is trained, you can test its performance with new data in the 'Model Testing' tab.

8. Deploy the Model:

- Go to the 'Deployment' tab.
- Select the deployment method that suits your edge device (e.g., Arduino library, WebAssembly, container, etc.).

- Follow the instructions to deploy the model to your device.

9. Run Inference:

- With the model deployed, run inference on the edge device to see it classifying data in real-time.

10. Monitor:

- You can monitor the performance of your device through the Edge Impulse studio.

Paste your Edge Impulse project's Results:

The image displays two screenshots of the Edge Impulse studio interface, showing the process of collecting and training a dataset for color detection.

Top Screenshot: The interface shows the 'Dataset' tab with a table of collected samples. The 'DATA COLLECTED' is 5m 20s, and the 'TRAIN / TEST SPLIT' is 70% / 30%. The 'Collect data' button is visible, along with a 'Connect a device' link. The 'RAW DATA' section shows a 'Click on a sample to load...' prompt.

SAMPLE NAME	LABEL	ADDED	LENGTH
blue.5o9habq6	blue	Apr 09 2025, 14:34:11	5s
blue.5o9h9xti	blue	Apr 09 2025, 14:33:55	5s
blue.5o9h9h74	blue	Apr 09 2025, 14:33:44	5s
blue.5o9h9436	blue	Apr 09 2025, 14:33:30	5s
blue.5o9h8nco	blue	Apr 09 2025, 14:33:17	5s
blue.5o9h89pm	blue	Apr 09 2025, 14:33:03	5s
blue.5o9h7tjy	blue	Apr 09 2025, 14:32:50	5s
blue.5o9h7gh1	blue	Apr 09 2025, 14:32:37	5s
blue.5o9h73cm	blue	Apr 09 2025, 14:32:24	5s
blue.5o9h6ltp	blue	Apr 09 2025, 14:32:07	5s
blue.5o9h66he	blue	Apr 09 2025, 14:31:54	5s

Bottom Screenshot: The interface shows the 'Dataset' tab with a table of collected samples. The 'DATA COLLECTED' is 5m 20s, and the 'TRAIN / TEST SPLIT' is 70% / 30%. The 'Collect data' button is visible, along with a 'Connect a device' link. The 'RAW DATA' section shows a 'Click on a sample to load...' prompt.

SAMPLE NAME	LABEL	ADDED	LENGTH
green.5o9go3nf	green	Apr 09 2025, 14:24:13	5s
green.5o9gnmad	green	Apr 09 2025, 14:24:00	5s
green.5o9gn648	green	Apr 09 2025, 14:23:42	5s
green.5o9gl99f	green	Apr 09 2025, 14:22:40	5s
green.5o9gk9rn	green	Apr 09 2025, 14:22:24	5s
green.5o9gjm5	green	Apr 09 2025, 14:21:52	5s
red.5o9gggqt	red	Apr 09 2025, 14:20:04	5s
red.5o9gg37r	red	Apr 09 2025, 14:19:50	5s
red.5o9gfibr	red	Apr 09 2025, 14:19:36	5s
red.5o9gf7ai	red	Apr 09 2025, 14:19:21	5s
red.5o9geq4e	red	Apr 09 2025, 14:19:08	5s

EDGE IMPULSE

Dashboard

Devices

Data acquisition

Experiments

EON Tuner

Impulse design

Create impulse

Flatten

Classifier

Retrain model

Live classification

Model testing

Deployment

Upgrade Plan

View plans

Rahul Bhatti / Color detection / PERSONAL

Target: Cortex-M4F 80MHz

Impulse #1

An impulse takes raw data, uses signal processing to extract features, and then uses a learning block to classify new data.

Time series data

Input axes (6)

red, green, blue, brightness, proximity, gesture

Window size

1,000 ms

Window increase (stride)

1,000 ms

Frequency (Hz)

Zero-pad data

Flatten

Name

Flatten

Input axes (3)

red

green

blue

brightness

proximity

gesture

Classification

Name

Classifier

Input features

Flatten

Output features

3 (blue, green, red)

Output features

3 (blue, green, red)

Save impulse

Add a processing block

Add a learning block

<https://studio.edgeimpulse.com/studio/667900/impulse/1/deployment>

EDGE IMPULSE

Dashboard

Devices

Data acquisition

Experiments

EON Tuner

Impulse design

Create impulse

Flatten

Classifier

Retrain model

Live classification

Model testing

Deployment

Upgrade Plan

View plans

Parameters

Generate features

Raw data

300

250

200

150

100

50

0ms

1000ms

2000ms

3000ms

4000ms

Show: All labels

Blue.SonPhabg8 (3s)

red

green

blue

Raw features

56, 239, 292

Label

blue

Parameters

Scaling

Scale axes

1

Method

Average

Minimum

Maximum

Root-mean square

Standard deviation

DSP result

Processed features

56, 56, 56, 56, 0, 0, -3, 239, 239, 239, 239, 0, 0, -3, 292, 292, 292, 292, 0, 0, -3

State

None for these settings

On-device performance

PROCESSING TIME

1 ms.

PEAK RAM USAGE

144 Bytes

EDGE IMPULSE

Dashboard

Devices

Data acquisition

Experiments

EON Tuner

Impulse design

Create impulse

Flatten

Classifier

Retrain model

Live classification

Model testing

Deployment

Upgrade Plan

View plans

Rahul Bhatti / Color detection / PERSONAL

Target: Cortex-M4F 80MHz

Neural Network settings

Training settings

Number of training cycles

30

Use learned optimizer

Learning rate

0.0005

Training processor

CPU

Advanced training settings

Neural network architecture

Input layer (11 features)

Dense layer (20 neurons)

Dense layer (10 neurons)

Add an extra layer

Output layer (3 classes)

Save & train

Training output

Model

Model version

Quantized (int8)

Last training performance (validation set)

ACCURACY

91.1%

LOSS

0.14

Confusion matrix (validation set)

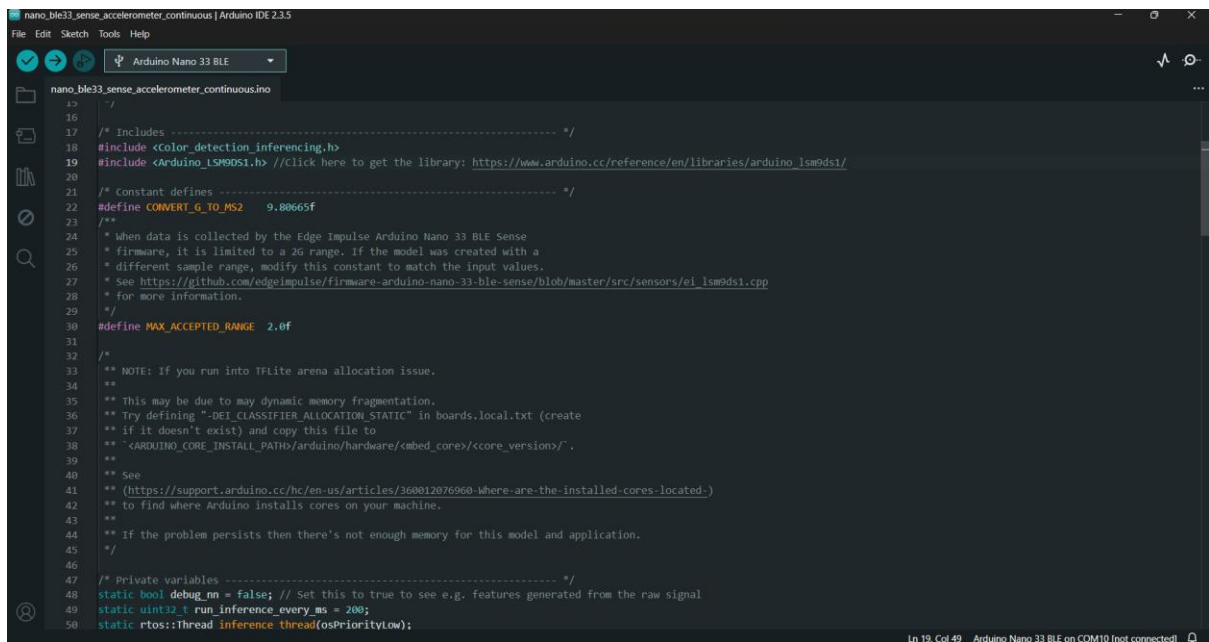
	BLUE	GREEN	RED
BLUE	93.3%	6.7%	0%
GREEN	21.4%	78.6%	0%
RED	0%	0%	100%
F1 SCORE	0.87	0.85	1.00

Metrics (validation set)

METRIC	VALUE
Area under ROC Curve	0.99
Weighted average Precision	0.92
Weighted average Recall	0.91
Weighted average F1 score	0.91

Data explorer (full training set)

Loading...



```
15 nano_ble33_sense_accelerometer_continuous.ino
16
17 /* Includes ----- */
18 #include <color_detection_inferencing.h>
19 #include <Arduino_ISM90S1.h> //Click here to get the library: https://www.arduino.cc/reference/en/libraries/arduino\_ism90s1/
20
21 /* Constant defines ----- */
22 #define CONVERT_G_TO_MS2 9.80665f
23 /**
24  * When data is collected by the Edge Impulse Arduino Nano 33 BLE Sense
25  * firmware, it is limited to a 2g range. If the model was created with a
26  * different sample range, modify this constant to match the input values.
27  * See https://github.com/edgeimpulse/firmware-arduino-nano-33-ble-sense/blob/master/src/sensors/ei\_ism90s1.cpp
28  * for more information.
29  */
30 #define MAX_ACCEPTED_RANGE 2.0f
31
32 /*
33  ** NOTE: If you run into TFLite arena allocation issue.
34  **
35  ** This may be due to may dynamic memory fragmentation.
36  ** Try defining "-DEI_CLASSIFIER_ALLOCATION_STATIC" in boards.local.txt (create
37  ** if it doesn't exist) and copy this file to
38  ** <ARDUINO_CORE_INSTALL_PATH>/arduino/hardware/cmbd_cores/<core_version>/ .
39  **
40  ** See
41  ** (https://support.arduino.cc/hc/en-us/articles/360012876960-Where-are-the-installed-cores-located-)
42  ** to find where Arduino installs cores on your machine.
43  **
44  ** If the problem persists then there's not enough memory for this model and application.
45  */
46
47 /* Private variables ----- */
48 static bool debug_nn = false; // Set this to true to see e.g. features generated from the raw signal
49 static uint32_t run_inference_every_ms = 200;
50 static rtos::Thread inference_thread(osPriorityLow);
```

Starting Nano BLE Sense Classification...

Sensor data collected.

Running inference...

Predicted Class: White

Confidence: 89.3%

Raw Output: - Red: 10.2% - White: 86.3% - Black: 3.5%

Waiting for next sensor input...

Predicted Class: Red

Confidence: 92.8%

Raw Output: - Red: 92.8% - White: 5.1% - Black: 2.1%

Waiting for next sensor input...