

Edge Impulse : Object Detection

Raspberry Pi 4, Mobile Phone, JetsonNano

1.0.0

2021. 05. 03

SunBeen Moon

Contents

1. Edge Impulse 소개	4
2. 개발 환경 구축	6
2.1. Linux 기반 보드	6
2.1.1. RPI Initial Setting	6
2.1.2. Jetson Nano Initial Setting	6
2.1.3. Edge Impulse with RPI&JetsonNano	7
2.1.4. 관련 명령어 설명	11
2.2. Mobile Phone	13
2.3. Issues	16
3. 모델 구축 방법	17
4. RPI Tests and Results	21
4.1. 1 st Test	21
4.2. 2 nd Test	24
4.3. 3 rd Test	27
4.4. Issues	30
5. Jetson Nano Tests and Results	31
5.1. 1 st Test	31

5.2. 2 nd Test	<u>34</u>
5.3. Issues	<u>37</u>

1. Edge Impulse 소개

✓ Introduction

Edge Impulse 는 다양한 edge device 를 타겟으로 센서 데이터 수집에서 모델 학습, 모델 테스트, deployment 까지 모든 과정을 지원하는 end-to-end, edge AI solution 개발을 위한 IDE(Integrated Development Environment) tool 이다. 'tinyML Awards 2021, Best Products & Best Innovation'에서 edge impulse EON(Edge Optimized Neural) 컴파일러는 기존 TF lite 의 memory 비효율성을 개선한 점에서 주목받았다. 위 award 에서 edge impulse 을 소개하고 해당 tutorial 을 진행했다. TFLM 의 개발환경과 사용법을 기술한 도서 '초소형 머신러닝 tinyML'의 제 2 의 저자, Danial Situnayake 가 founding engineer 로 일하고 있다. 또한, edge impulse 사는 다양한 edge device 회사들과 파트너십을 맺고 있으며 그 중 STM 사와 Arm Cortex, Synopsys 사의 Himax 보드 등이 포함된다. 지속적으로 개발자와의 소통을 통해 기술을 발전시키고 있으며, 새로운 solution 이 지속적으로 업데이트되고 있다. 최근에는, NVIDIA Jetson Nano, Raspberry Pi 4 보드에서 linux 환경에서 object detection 을 지원한다.

(edge impulse 홈페이지) <https://www.edgeimpulse.com/>

(edge Impulse Performance metric 설명 페이지) <https://docs.edgeimpulse.com/docs/inference-performance-metrics>

✓ Edge Impulse Forum

깃허브보다는 edge impulse 자사에서 운영하는 Forum 활성화되어 있다. 깃허브에 질문을 남길 경우, 답변이 달리지 않으나, Forum 질문을 남길 경우 빠른 시간 내에 답변이 달린다. Forum 을 통해 update 소식과 bug fix 소식을 빠르게 접할 수 있다.

(edge impulse 포럼) <https://forum.edgeimpulse.com/>

(edge impulse 깃허브) <https://github.com/edgeimpulse>

✓ EON Compiler

TFLM 의 비효율적인 메모리 구조를 최적화해 주는 compiler 이다. 실제 뒤 example 에서 실행한 표를 참조하면 50%가량의 메모리 이득을 얻을 수 있다. 동일한 정확도를 유지하지만, TFLM 에 비해 효율적인 메모리 이득을 보인다. EON 컴파일러를 활용하여 TFLM 의 interpreter 를 거치지 않고, .C++로 컴파일된 코드를 활용한다.

(edge Impulse 사의 EON 컴파일러 소개 글) <https://www.edgeimpulse.com/blog/introducing-eon>

✓ Edge Impulse with RPI4 and Mobile Phone

Edge Impulse사에서 최근에 RPI4를 지원하면서 뒤에 정리할 object detection을 지원하기로 하였다. 기존에도 mobile phone을 이용하여 데이터를 수집, Live Configuration, 완성된 모델을 다운로드 받아 Classification하는 기능을 제공하였다.

(edge impulse사에서 소개하는 RPI보드 지원) <https://www.edgeimpulse.com/blog/ei-extends-the-edge-to-embedded-linux-with-official-support-for-raspberry-pi-4>

(edge impulse사에서 소개하는 Mobile Phone 지원) <https://www.edgeimpulse.com/blog/build-tinyml-models-using-your-mobile-phone>

✓ Object Detection

Edge Impulse사에서 최근에 MobileNetV2 SSD FPN-Lite 모델을 이용하여 object detection하는 모델을 지원하고 있다. 위 Mobilenet을 이용하여 transfer learning에 적용하고 있다. 3.4 절의 issue에서 한 번 더 언급할 예정이지만, 해당 모델은 속도가 빠른 대신 COCO Map가 낮기 때문에, 정확도가 다른 모델과 같은 환경의 데이터셋에 비해 정확도가 떨어지는 경향을 보인다. MobileNetV2 SSD FPN-Lite 모델과 edge impulse사의 Object Detection에 대해 더 알고 싶다면 아래 링크를 참조하길 바란다.

(Tensorflow 2 Object Detection Blog) <https://blog.tensorflow.org/2020/07/tensorflow-2-meets-object-detection-api.html>

(Tensorflow 2 Detection Model)

https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md

(edge Impulse에서 Object Detection을 소개하는 글) <https://docs.edgeimpulse.com/docs/object-detection>

(note) Edge Impulse 소개 자료는 이 전에 작성된 'Himax WE-1 : TFLM & Edge Impulse' 보고서에서 차용하였다. 더 상세한 내용을 알고 싶다면 해당 보고서의 30 페이지부터 참조하면 된다.

(Himax WE-1 : TFLM & Edge Impulse 보고서) <https://www.notion.so/notaai/Himax-WE-1-TFLM-Edge-Impulse-b0bf7df6bf664de5ab122363dfe88ff3>

2. 개발 환경 구축

2.1 Linux 기반 보드

2.1.1 RPI Initial Setting

✓ Raspberry Pi 4 를 초기 세팅

RPI 를 초기 세팅을 해주어야 한다. 이와 관련된 내용은 notion 에 상세히 기술해 놓았다. 아래 링크를 참조하길 바란다. IP 를 찾아, Mobaxterm(SSH) 혹은 VNC Viewer(VNC)에 연결하는 방법까지 기술되어 있다.

(RPI 4 초기 세팅 Tutorial) <https://www.notion.so/notaai/1-Rpi4-fbf3527d63294c41ac77b15ec3440cf1>

✓ Raspberry Pi 4 에 Edge Impulse 설치

Edge Impulse 를 RPI4 에서 구동하기 위해서 프로그램을 설치해야 한다. 1 번 과정을 충족했다면, Edge Impulse 사의 Document 2 번부터 진행하면 문제없이 구동할 수 있다.(SSH or VNC 연결을 한 후, 2 번부터 진행하면 된다) 아래 링크를 따라 진행하면 된다.

(RPI4 에 edge impulse 를 설치 tutorial document) <https://docs.edgeimpulse.com/docs/raspberry-pi-4>

2.1.2 JetsonNano Initial Setting

✓ JetsonNano 를 초기 세팅

RPI 를 초기 세팅을 해주어야 한다. 이와 관련된 내용은 notion 에 상세히 기술해 놓았다. 아래 링크를 참조하길 바란다.

(Jetson Nano 초기 세팅 Tutorial) <https://www.notion.so/notaai/5-JetsonNano-9c43858354e449fd8a51d0c516efcb3f>

✓ JetsonNano 에 Edge Impulse 설치

Edge Impulse 를 JetsonNano 에서 구동하기 위해서는 프로그램을 설치해야 한다. 1 번 과정을 충족했다면 Edge Impulse 사의 Document 2 번부터 진행하면 문제없이 구동할 수 있다. 아래 링크를 참조 바란다.

(JetsonNano 에 edge impulse 를 설치 tutorial document) <https://docs.edgeimpulse.com/docs/nvidia-jetson-nano#running-models-on-the-gpu>

(note) Edge Impulse 에서는 RPI 와 동일하게 JetsonNano 를 linux 버전으로 지원한다. TensorRT(GPU 사용)의 경우 아직 Object Detection 은 지원하지 않고 있다. Image Classification 은 지원함으로 3 절의 JetsonNano Example 에서 소개한다.

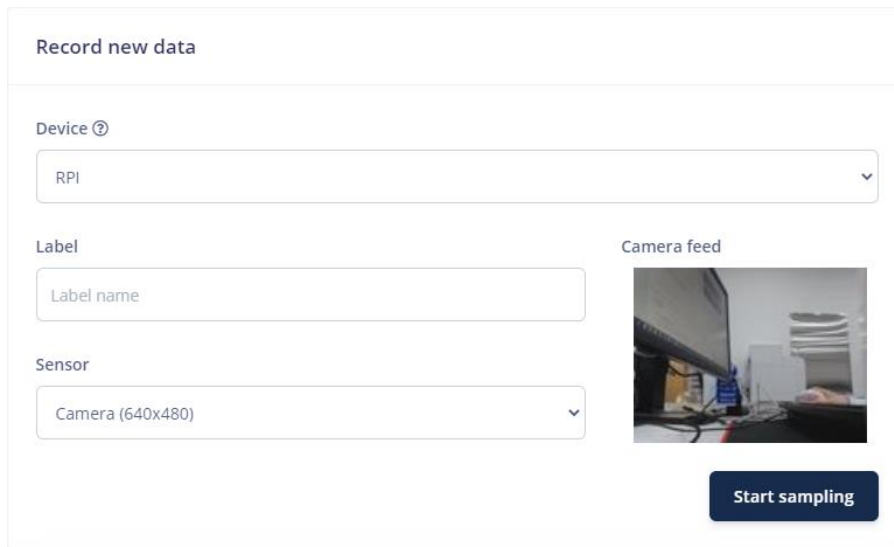
2.1.3 Edge Impulse with RPI&JetsonNano

(note) 아래 기술된 'Linux board data collect', 'Linux board Live Classification', 'Linux board Model Deployment'는 기존 소개되었던 edge impulse 모델 구상 중에서 추가적으로 다른 부분을 중점적으로 기술했다. 이전 소개된 edge impulse 내용 확인하고 싶다면, 이전 보고서('Himax WE-1_final report (ver1) 210420.pdf')에 35 페이지를 확인하면 된다.

(Himax WE-1 final report) <https://www.notion.so/notaai/Himax-WE-1-TFLM-Edge-Impulse-b0bf7df6bf664de5ab122363dfe88ff3>

✓ Linux board data collect

Linux 에 연결된 카메라를 통해 데이터를 얻는 창은 아래와 같다. Project 의 목차 중에서 'Data acquisition'에 들어간다. 이미 기기가 연결되어 있을 경우 Fig 1 과 같은 창이 뜬다. 'Start Sampling'을 눌러 data 를 모으면 된다.



The screenshot shows the 'Record new data' window in the Edge Impulse software. It contains the following elements:

- Device:** A dropdown menu currently showing 'RPI'.
- Label:** A text input field containing the placeholder text 'Label name'.
- Sensor:** A dropdown menu currently showing 'Camera (640x480)'.
- Camera feed:** A small video window displaying a live camera feed from the connected device.
- Start sampling:** A dark blue button located at the bottom right of the form.

Fig 1. Data Collect with RPI

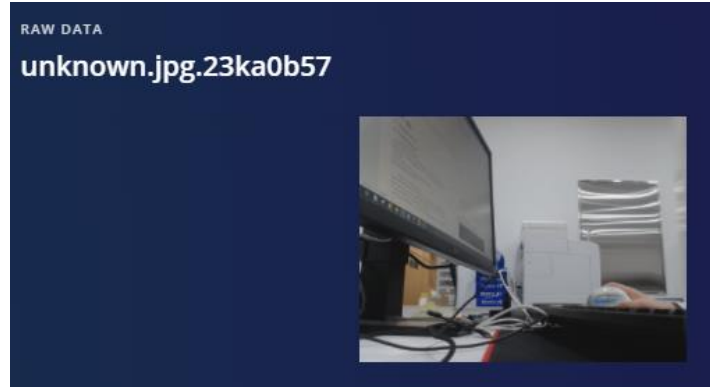


Fig 2. Data 을 취득한 후, 보여지는 화면

✓ Linux board Live Classification

모델을 device 에 deploy 하지 않고, live classification 을 통해 모델을 테스트할 수 있다. Project 목차 중에서 'Live Classification'에 들어가 연결된 기기를 선택한 후 'Start Sampling'을 눌러 데이터를 취득하면 된다.

Classify new data

Device ②

Sensor

Camera feed

Fig 3. RPI 를 이용하여 모델 테스트를 위한 sample 를 얻는 화면

데이터 취득 후, 자동적으로 모델 테스트가 이루어지며 테스트를 진행한 화면은 Fig 4 와 같다. Live Classification 을 하면서 취득한 데이터는 Test data 에 저장된다.

RAW DATA

testing.jpg.23liuqed

Raw features

0x9c9fa1, 0xa2a3e5, 0xa9a9ac, 0xa9a9ac, 0xa5a6a8, 0x9fa8a2, 0xa1a2a4, 0xa7a8a9, 0xa8a9ab, 0xa7a8a9, 0xa8a9ac...

Classification result

Summary

Name

CATEGORY	COUNT
----------	-------

Fig 4. Sample 를 얻은 후 모델 테스트를 진행한 화면

✓ Linux board Model Deployment

모델을 linux 기반의 device 에 deploy 하기 위해서는 기존 소개되었던 deploy 방식과 다른 방식으로 진행된다. 먼저, Project 목차 중 'Deployment'를 들어간 후, 'Build Firmware'에서 'Linux boards'(Fig 5)를 선택한다.

Build firmware

Or get a ready-to-go binary for your development board that includes your impulse.

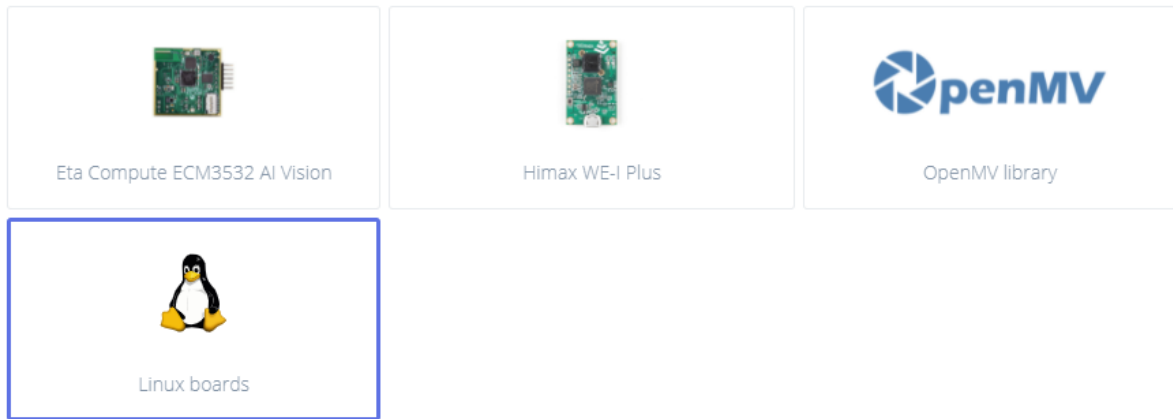


Fig 5. Deploy 하기 위해 Linux boards 를 선택한 화면

'Build'를 진행하면 edge impulse 의 장점인 EON compiler 가 작동하지 않는다. 빌드가 완료된 후 화면은 Fig 6 과 같으면 'ROM usage'만 표시될 뿐 이 외 RAM, LATENCY, ACCURACY 는 표시되지 않는다. 이전과 다른 차이 중 하나는, Unoptimized(float 32)로 모델을 deploy 할 수 있도록 빌드가 진행되지 않았다. EON 컴파일러의 메모리 사용량 감소를 보여주고자, Unoptimized(float32)가 보여지긴 하나 선택 후 edge device 에 deploy 하기 위해 build 되지 않았다. 하지만, linux 환경의 device 는 edge device 보다 메모리(RAM, ROM)이 크기 때문에, Unoptimized 버전도 선택하면 deploy 할 수 있다.

Available optimizations for Object detection

Quantized (int8) ★ Currently selected This optimization is recommended for best performance.	RAM USAGE N/A	LATENCY -
	ROM USAGE 3.6M	ACCURACY -
Unoptimized (float32) Click to select	RAM USAGE N/A	LATENCY -
	ROM USAGE 10.9M	ACCURACY -

Fig 6. Build 가 완료된 후의 화면

Fig 6 처럼 build 가 완료된 후에는, 2.1.3 절에서 설명한 'edge-impulse-linux-runner' 명령어를 이용하여 device 에 deploy 한다.

(note) quantization 이 진행되지 않은 'Unoptimized(float32)'을 사용하여 RPI 에 테스트를 진행하였을 때, quantized(int8)보다 좋은 정확도를 보인다. 관련 테스트 결과는 3 절에서 확인하면 된다.

(note) EON compiler 가 작동하지 않는다.

✓ RPI 모델이 저장된 위치

2.1.4 절에 보다 상세히 기술되어 있으나, '--clean' 명령어를 이용하여 새롭게 모델을 다운로드 받지 않는 이상 기존의 모델은 '/home/pi/.ei-linux-runner/models/숫자'에 저장된다. 숫자의 의미는 edge impulse 에서 생성된 프로젝트 넘버이다. Fig 7 에서 확인할 수 있다. 각 프로젝트 별로 update 하고 달리한 버전이 있다가 Fig 8 처럼 각 버전별(다운로드한 순서대로 번호가 매겨진다)로 저장된다. 파일 형식은 '.eim'이다.



Fig 7. RPI 프로젝트 넘버별로 저장된 화면

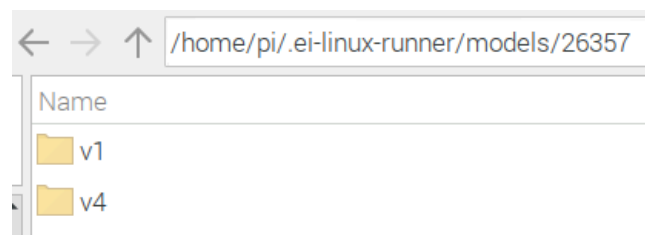


Fig 8. RPI 다운로드 된 모델 버전

✓ JetsonNano 모델일 저장된 위치

RPI 모델과 동일한 위치에 저장된다. 관련 설명은 바로 위 'RPI 모델이 저장된 위치' 섹션을 확인하면 된다.

```
nota@nota-desktop:~/ei-linux-runner/models$ ls -al
total 16
drwxrwxr-x 4 nota nota 4096 5월 3 10:11 .
drwxrwxr-x 3 nota nota 4096 4월 30 16:12 ..
drwxrwxr-x 3 nota nota 4096 4월 30 16:12 29514
drwxrwxr-x 4 nota nota 4096 5월 3 11:36 29779
```

Fig 9. JetsonNano 프로젝트 넘버별로 저장된 화면

```
nota@nota-desktop:~/ei-linux-runner/models/29779$ ls -al
total 16
drwxrwxr-x 4 nota nota 4096 5월 3 11:36 .
drwxrwxr-x 4 nota nota 4096 5월 3 10:11 ..
drwxrwxr-x 2 nota nota 4096 5월 3 10:11 v1
drwxrwxr-x 2 nota nota 4096 5월 3 11:36 v3
```

Fig 10. JetsonNano 다운로드 된 모델 버전 정보

2.1.4 관련 명령어 설명

Linux 환경의 device(RPI, Jetson Nano 등)에서 적용할 수 있는 명령어는 2 가지이다. 각 명령어 뒤에 '-clean' 옵션이 붙을 수 있다.

✓ edge-impulse-linux

처음 edge impulse 에 RPI 를 연결하기 위한 명령어이다. 명령을 입력 후, Fig 1 처럼 연결이 되면, 해당 project 에서 data 를 모을 수 있고, live classification 을 진행할 수 있다.

```
pi@raspberrypi:~$ edge-impulse-linux
Edge Impulse Linux client v1.2.1

? Select a microphone (or run this command with --disable-microphone to skip selection) USB-Audio - HD Pro Webcam C920
[SER] Using microphone hw:1,0
? Select a camera (or run this command with --disable-camera to skip selection) HD Pro Webcam C920
[SER] Using camera HD Pro Webcam C920 starting...
[SER] Connected to camera
[WS ] Connecting to wss://remote-mgmt.edgeimpulse.com
[WS ] Connected to wss://remote-mgmt.edgeimpulse.com
? What name do you want to give this device? RPI
[WS ] Device "RPI" is now connected to project "RPI3"
[WS ] Go to https://studio.edgeimpulse.com/studio/28004/acquisition/training to build your machine learning model!
```

Fig 11. Edge-impulse-linux 명령어 실행 prompt

✓ edge-impulse-linux-runner

프로젝트에서 빌드가 완료된 모델을 RPI 에 deploy 하기 위한 명령어이다. 위 명령어를 입력하면, 모델이 deploy 이 되지 않았을 경우 자동으로 deploy 되며, 이미 deploy 가 되어 있는 경우 deploy 된 모델을 곧바로 실행한다.

```
pi@raspberrypi:~ $ edge-impulse-linux-runner --clean
Edge Impulse Linux runner v1.2.1
? What is your user name or e-mail address (edgeimpulse.com)? SunBeenMoon
? What is your password? [hidden]

? From which project do you want to load the model?
SunBeenMoon / dogcat_final
SunBeenMoon / mountain_sea
SunBeenMoon / ei_temperature_sensor
> SunBeenMoon / RPI
SunBeenMoon / RPI2
```

Fig 12. edge-impulse-linux-runner --clean 입력 후, 프로젝트를 선택하는 창

```
? From which project do you want to load the model? SunBeenMoon / RPI
[RUN] Already have model /home/pi/.ei-linux-runner/models/26357/v4/model.eim not
downloading...
[RUN] Starting the image classifier for SunBeenMoon / RPI (v4)
[RUN] Parameters image size 320x320 px (3 channels) classes [ 'lip', 'scrunch' ]
[RUN] Using camera mmal service 16.1 starting...
[RUN] Connected to camera
```

Fig 13. 프로젝트를 선택한 후, Edge Impulse 프로젝트에 연결된 모습

Fig 9 를 실행하면, RPI prompt 에서는 각 모델이 돌아가는 frame 마다 결과 값을 Fig 10 처럼 적어준다.

```
boundingBoxes 473ms. [{"height":288,"label":"handcream","value":0.5002697706222534,"width":108,"x":28,"y":10}]
boundingBoxes 468ms. [{"height":288,"label":"handcream","value":0.5002697706222534,"width":108,"x":28,"y":10}]
boundingBoxes 486ms. [{"height":288,"label":"handcream","value":0.5002697706222534,"width":108,"x":28,"y":10}]
boundingBoxes 481ms. [{"height":288,"label":"handcream","value":0.5002697706222534,"width":108,"x":28,"y":10}]
boundingBoxes 484ms. [{"height":288,"label":"handcream","value":0.5002697706222534,"width":108,"x":28,"y":10}]
boundingBoxes 486ms. [{"height":288,"label":"handcream","value":0.5002697706222534,"width":108,"x":28,"y":10}]
boundingBoxes 485ms. [{"height":288,"label":"handcream","value":0.5002697706222534,"width":108,"x":28,"y":10}]
boundingBoxes 453ms. [{"height":152,"label":"handcream","value":0.5167295336723328,"width":64,"x":46,"y":21}]
boundingBoxes 486ms. [{"height":152,"label":"handcream","value":0.5167295336723328,"width":64,"x":46,"y":21}]
boundingBoxes 504ms. [{"height":300,"label":"handcream","value":0.5568788647651672,"width":96,"x":223,"y":19}]
boundingBoxes 467ms. [{"height":300,"label":"handcream","value":0.5568788647651672,"width":96,"x":223,"y":19}]
boundingBoxes 486ms. [{"height":300,"label":"handcream","value":0.5568788647651672,"width":96,"x":223,"y":19}]
boundingBoxes 458ms. [{"height":320,"label":"handcream","value":0.5389894843101501,"width":85,"x":234,"y":0}]
```

Fig 14. RPI 실시간 검출되는 prompt

검출되는 label 를 확인할 수 있도록, API 를 제공하는데, Fig 11 에서 볼 수 있다 싶이, Fig 11 가 실행된 이후 창에서 확인할 수 있다. (<http://Board SSH NUM:4912>)

```
[BLD] Building binary OK
[RUN] Downloading model OK
[RUN] Stored model version in /home/pi/.ei-linux-runner/models/26917/v6/model.eim
[RUN] Starting the image classifier for SunBeenMoon / RPI2 (v6)
[RUN] Parameters image size 320x320 px (3 channels) classes [ 'handcream', 'omega-3' ]
[RUN] Using camera HD Pro Webcam C920 starting...
[RUN] Connected to camera

Want to see a feed of the camera and live classification in your browser? Go to http://192.168.1.74:4912
```

Fig 15. RPI 와 연결된 Web 주소

웹에 주소를 입력하고 들어가면, Fig 12 처럼 카메라 창을 확인할 수 있다. 밑에서 Inference Time 을 알 수 있다.

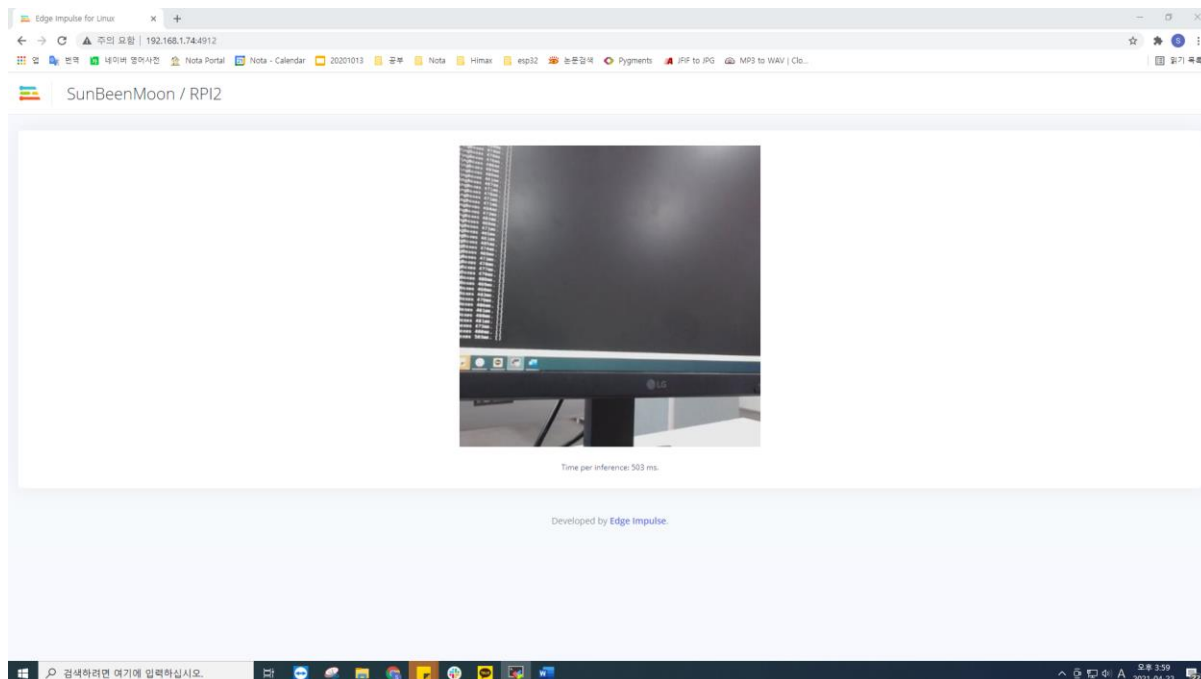


Fig 16. 연결된 웹페이지에서 확인하는 RPI 카메라 모듈 창

✓ --clean 옵션

Clean 옵션은 프로젝트를 재 선택하고 싶을 때 활용하는 옵션이다.

예를 들어, 'edge-impulse-linux --clean'을 입력하면 연결하고 싶은 project 를 재 선택할 수 있다.

'edge-impulse-linux-runner --clean'을 입력하면 Fig 2 에서 확인할 수 있듯이 새로운 project 를 선택하여 모델을 다운로드 받을 수 있다.

2.2 Mobile Phone

Edge Impulse 는 Mobile Phone 을 해당 프로젝트에 연결하여, 데이터를 모으고 model classification 을 진행할 수 있다. 해당 튜토리얼은 아래 링크를 참고하면 된다.

(mobile phone 설치 tutorial document) <https://docs.edgeimpulse.com/docs/using-your-mobile-phone>

✓ Mobile phone data Collect

'show option'을 선택한 이후, 'use your mobile phone'에서 'Show QR code'를 선택한다. QR code 를 인식하면, 해당 웹사이트가 뜬다. 'smartphone.edgeimpulse.com' 웹사이트에 연결된 것을 확인할 수 있다.

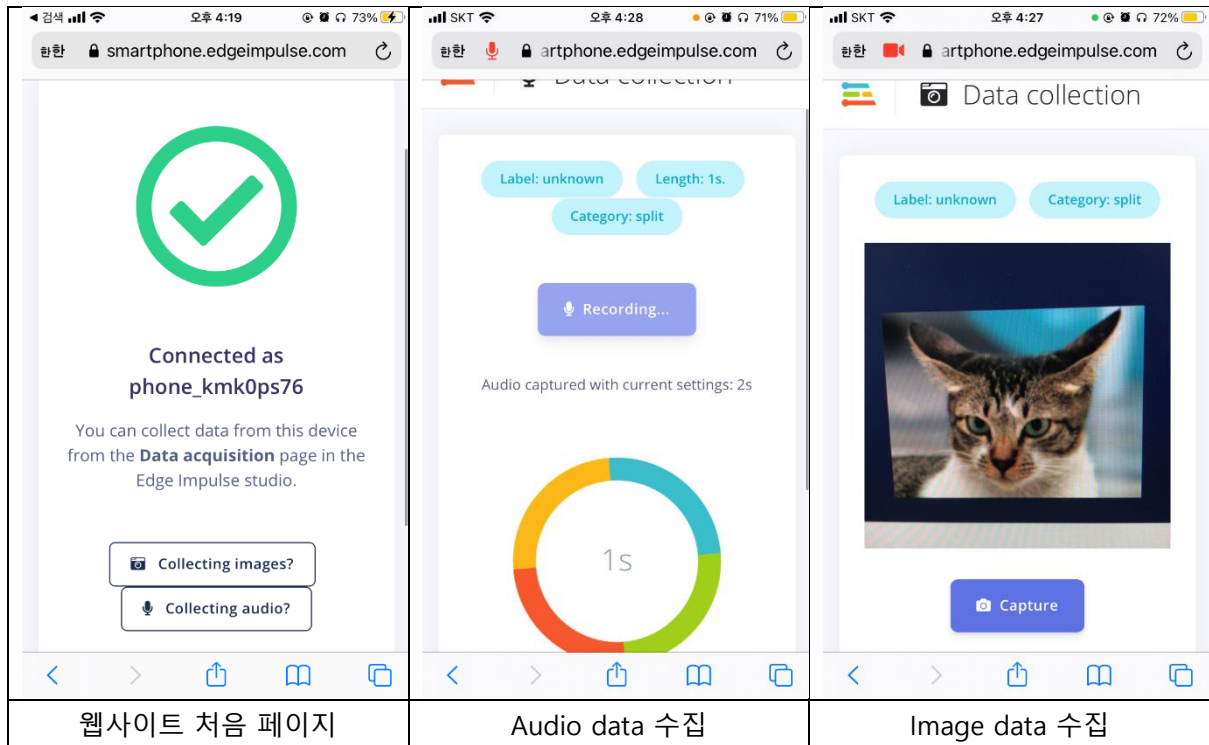


Table 1. Mobile Phone Data Collect Option

✓ Mobile phone Live Classification

앞서 설명한 QR 코드를 통해 mobile phone 에 연결한다. 연결한 웹사이트 맨 아래 쪽에서 'Switch to data collection mode', 'Switch to classification mode' option 중 classification mode 를 선택한다.

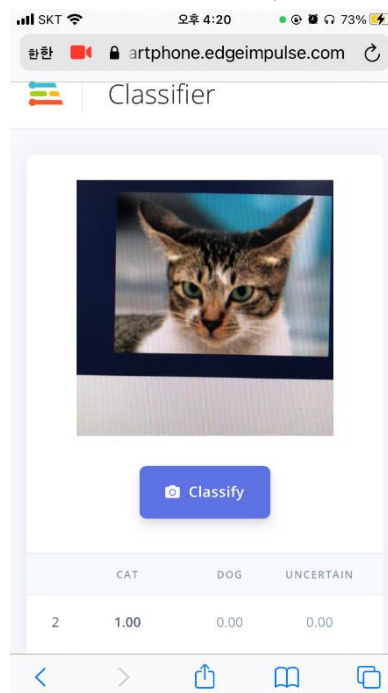


Fig 17. Live Classification via Mobile Phone Output

Mobile phone 에서 직접 모델을 load 하여 결과를 확인할 수 있으며, audio 와 image 두 가지 option 을 제공한다

(note) 실상, Live Classification 은 build 된 모델을 다운로드 받아 test 진행하는 것과 동일하다.
3 절에서 진행한 mobile test 에서 동일하게 진행되었다.

2.3 Issues

<Issue 1> Mobile Phone 에서 모델이 저장되지 않고, 모델을 구동하고 싶을 때마다 새롭게 다운로드 받아야 한다. 해당 웹페이지를 새로 고침 할 때마다 새롭게 다운로드 된다.

<Issue 2> RPI 에서도 모델이 저장되긴 하지만, 원하는 모델을 지정하여서 작동시킬 수 없다. 예를 들어 모델 1, 모델 2 를 선택하여서 구동시킬 수 없고 무조건 마지막에 다운로드 받아 구동시킨 모델이 작동된다. 다른 모델을 구동하고 싶을 경우 'edge-impulse-linux-runner -clean'을 통해 새롭게 다운로드 받아야 한다.

<Issue 3> Mobile Phone 과 RPI 에서 모두 모델을 deploy 한 후, 모델 구동할 때 classification 한 상자가 겹치는 이슈가 있다. 이러한 경우, RPI 는 다시 모델을 구동하고 Mobile phone 은 모델을 재 deploy 해야 한다.

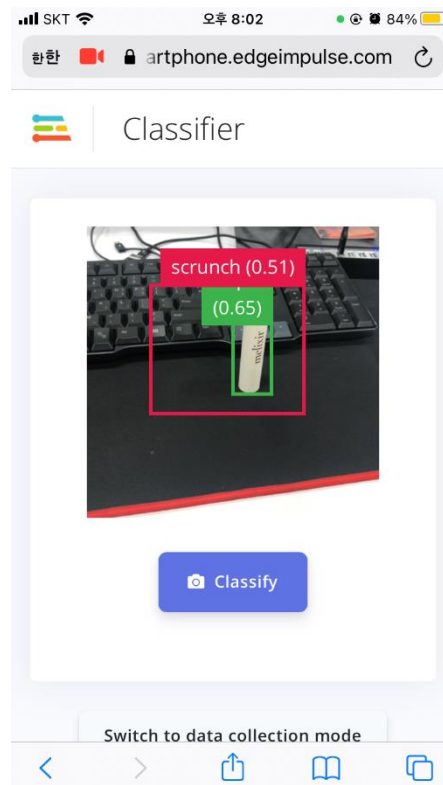


Fig 18. Issue 3 오류 화면

3. 모델 구축 방법

Object Detection 모델을 구성하여 테스트하는 과정을 뒤에 진행할 3 가지 Test 모두 동일하다. 공통적인 Object Detection 모델 구성 플로우 '3.Examples'에서 설명하며, 차이점인 데이터셋에 대한 설명은 각 Test 에서 설명한다. 아래 링크는 전반적인 RPI 를 이용한 edge impulse 모델 생성에 대해서 설명하고 있다.

(edge impulse 에서 RPI 를 이용한 모델생성 과정을 상세히 설명한 링크)

<https://www.raspberrypi.org/blog/edge-impulse-and-tinymml-on-raspberry-pi/>

(note) 보다 상세한 edge impulse 모델 구상 내용은 이전 보고서('Himax WE-1_final report (ver1) 210420.pdf')에 35 페이지를 확인하면 된다.

(Himax WE-1 final report) <https://www.notion.so/notaai/Himax-WE-1-TFLM-Edge-Impulse-b0bf7df6bf664de5ab122363dfe88ff3>

✓ Generate Project

Edge Impulse 가 업데이트되면서 project 를 만들고 처음 시작할 때, Fig 15, 16 과 같이 선택하는 창이 추가되었다. Object Detection 을 실행하기 위해서 아래 순서대로 선택한다. 순서대로 선택하면 'Data Acquisition'에 'labeling'을 할 수 있는 창이 생긴다.

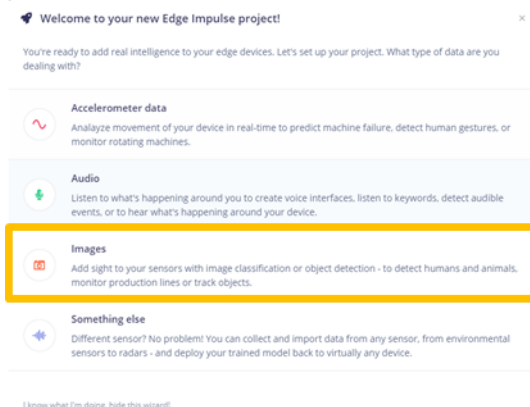


Fig 19. 'Images' 선택하는 첫 번째 창

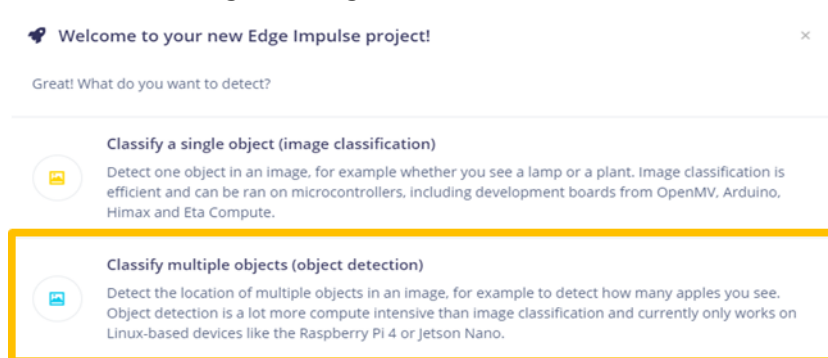


Fig 20. 'Classify multiple objects(object detection)' 선택하는 창

✓ Data Acquisition

앞서 설명한 대로 project 를 생성하고 나면 'Data acquisition'에 Fig 21 와 같은 옵션이 생긴다. Object Detection 의 경우 대부분 Data 를 업로드하기 보다는 Device 를 통해 직접 데이터를 가공하는 경우가 많다. 따라서, 데이터를 얻은 후, 순서대로 'Labeling queue'에 업로드 되며, 괄호 속에는 아직 labeling 이 되지 않은 데이터의 개수를 알려준다.

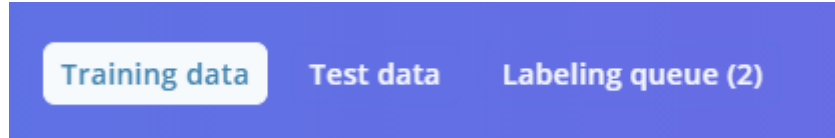


Fig 21. 'Labeling queue'가 생성된 모습

'Labeling queue'창에 들어가면, 아직 labeling 이 되지 않은 데이터들이 순차적으로 있다. 이 데이터에 labeling 을 하기 위해서는 아래 Fig 22 에서처럼 직접 하나하나 box 를 쳐서 labeling 을 해 주어야 한다.

Use your mouse to drag a box around an object to add a label. Then click **Save labels** to advance to the next item.

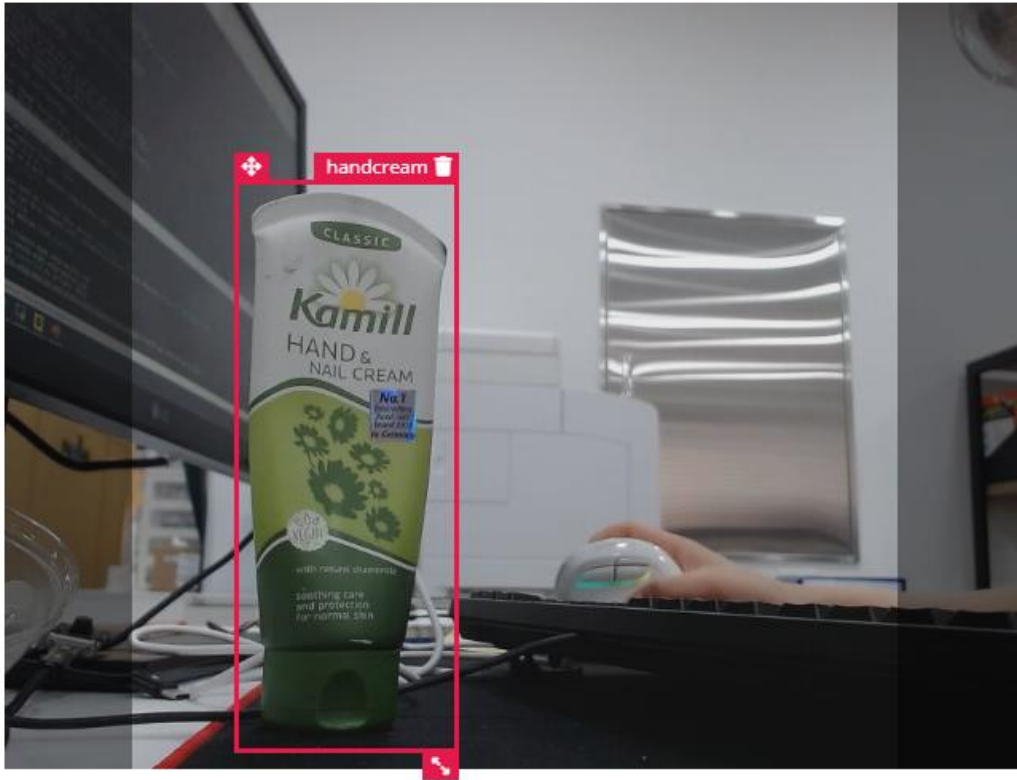


Fig 22. Set Labeling

Fig 22 에서처럼 labeling 의 완료한 후, 'Save label'을 눌러 저장한 label 을 수정하기 위해서는 Fig 23 처럼 데이터를 일일이 들어가서 수정해 줘야 한다. 'Edit labels'로 들어가 직접 라벨을 수정한다. 'Edit Labels'를 누르면 Fig 22 와 같이 label 을 수정하는 칸이 재생성된다.

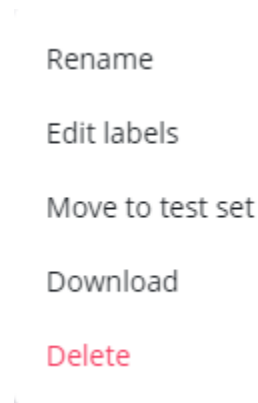


Fig 23. Data 수정 options

✓ Impulse Design

'Impulse Design'에 들어가서 아래와 Fig 24 과 같이 impulse 를 디자인하면 된다.

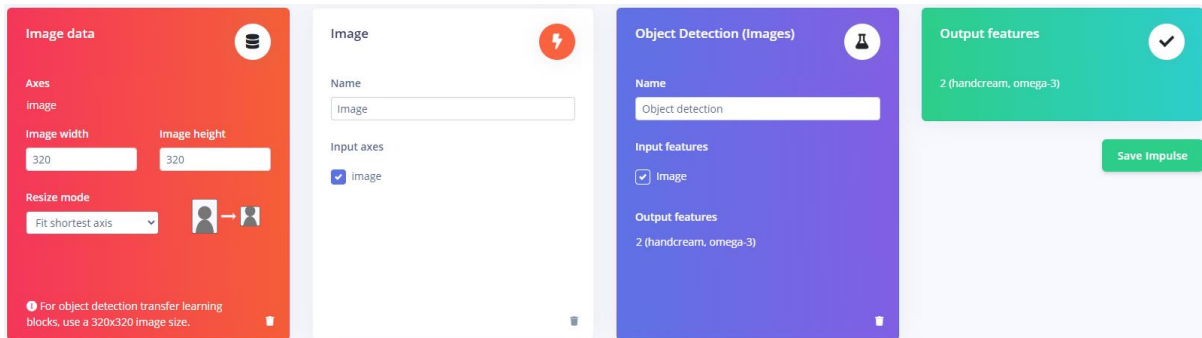


Fig 24. Object detection impulse 디자인

순서대로 'Input data → Images → Object Detection (Images) → Output features' 순서대로 진행된다. 각 block 에 대한 간략한 설명은 아래와 같다.

- Input data : Object Detection 모델이 320*320 input Image 로 픽스임으로 image 사이즈는 320*320 으로 고정해야 한다. Image 는 input 이미지보다 클 경우 crop 하는 형식으로 설정되어 있다.
- Image : 이미지를 전처리 해주는 block 으로, Object Detection 은 Color Depth 를 RGB 로 설정해준다.
- Object Detection : Fig 21 에서 확인할 수 있듯이, object detection block 에서 사용자가 변경해줄 수 있는 option(visual simple mode 기준)은, Training Cycle, Learning Rate, Score Threshold 이다. 그 중 Score threshold 은 이 기준을 넘으면 그 object 라고 detect 하는 기준이다. 지원되는 모델은 'MobileNetV2 SSD FPN-Lite 320*320' 한 가지이다.
- Output features : impulse 디자인에 마지막에 위치하는 block 으로 default 이다.

Neural Network settings

Training settings


Number of training cycles ②25

Learning rate ②0.15

Score threshold ②0.50

Neural network architecture

Input layer (307,200 features)



MobileNetV2 SSD FPN-Lite 320x320

Choose a different model

Output layer (2 features)

Fig 25. Object Detection Block 세부 사항

Fig 26 는 Object Detection block 에서 학습이 끝난 후, 정확도와 ROM 사용량이 뜨는 표이다. 다른 task 들과 달리, RAM 사용량과 Loss 가 나오지 않고 ROM 사용량과 정확도만 나온다.





Model	Model version: ②	Quantized (int8) ▾
Last training performance (validation set)		
	PRECISION SCORE	90.2%
On-device performance ②		
	ROM USAGE	3.6M
Model	Model version: ②	Unoptimized (float32) ▾
Last training performance (validation set)		
	PRECISION SCORE	96.5%
On-device performance ②		
	ROM USAGE	10.9M

Fig 26. Output of Model Validation Quatized/Unoptimized

4. RPI Tests and Results

4.1 1st Test

<Purpose>

Edge Impulse Object Detection Tutorial 동영상에서 제시한 대로 동일하게 진행했다. 적은 수 데이터셋(30 개 가량)으로 높은 정확도를 보일 수 있는 지 확인하기 위한 테스트이다.

<Data>

Data Label	Training data	Test data
Lip	23	10
Scrunch	25	10

Table 2. RPI 1st Test Data

Information about Data : 데이터는 mobile phone 으로 취득하였으며, 같은 장소에서 배경과 해당 object 가 한 면만 보이도록(다양한 면을 캡처하지 않고 한 면만 보이도록 설정하였다) 데이터를 직접 촬영하였다.

<Result>


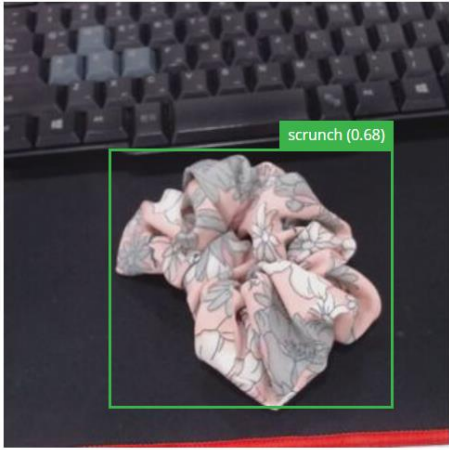
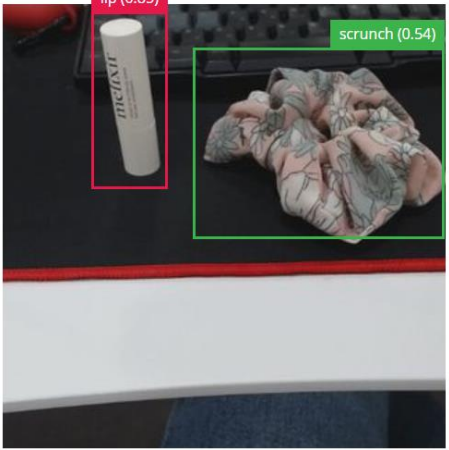
		Ver 1	Ver 2	Ver 3	Ver 4
Training cycles		25	30	25	30
Learning rate		0.15	0.15	0.1	0.1
Model Test(only Quantized)		0%	41.67%	50.00%	8.33%
Quantized	Validation	19.3%	47.5%	38.6%	5.2%
	ROM	3.6M	3.6M	3.6M	3.6M
Unoptimized	Validation	69.7%	73.2%	63.00%	68.1%
	ROM	10.9M	10.9M	10.9M	10.9M

Table 3. RPI 1st Test Result

<Analysis>

전체적으로 accuracy 가 낮은 것을 볼 수 있다. 또한, Quantized 와 Unoptimized 된 모델의 성능 차이가 크다. Training cycle 과 Learning rate 의 큰 성능 차이가 Unoptimized 모델에서는 보이지 않았으나, Quantized 모델에서는 보였다. 실제 모델을 Device 에 올려 보았을 때, 상대적으로 scrunch 가 lip 보다 검출이 잘 되었다. 두 가지 사물을 동시에 검출할 때, 모델 자체의 낮은 성능으로 인해 lip 만 검출되는 경우가 많았다.

<On Device Test – RPI>

Result	Inference Time	Confidence
 <p>Time per inference: 494 ms.</p>	494[ms]	0.84
 <p>Time per inference: 477 ms.</p>	477[ms]	0.68
 <p>Time per inference: 478 ms.</p>	478[ms]	0.85(lip) 0.54(scrunch)

*Quantized(int8) 모델로 테스트 진행

Table 4. 1st Test RPI Result

<On Device Test – Mobile Phone>

Result	Confidence
	0.76
	0.63
	0.68(Lip) 0.71(Scrunch)

Table 5. RPI 1st Test Mobile Phone Test

4.2 2nd Test

<Purpose>

1st Test 에서 Accuracy 와 Tutorial 동영상에서 보인 정확도(90% 가량)보다 현저히 낮은 원인을 적은 데이터 수라고 추정했다. 따라서, 데이터 수를 늘렸을 경우, 높은 정확도를 보일 수 있는 지 확인하기 위한 테스트이다.

<Data>

Data Label	Training data	Test data
Handcream	127	41
Omega-3	136	31

Table 6. RPI 2nd Test Data

Information about Data : 데이터는 mobile phone 으로 취득하였으며, 같은 장소에서 배경을 계속해서 변화를 주었으며 해당 object 가 여러 면이 보이도록 데이터를 직접 촬영하였다.

(note) 총 training data 의 개수가 200 개가 넘어가면 Developer 옵션 제한으로 모델이 끝까지 생성되지 않고 멈춘다. Dashboard 에서 .tflite 파일로 모델을 다운로드 받을 수는 있으나 유효한 정보인지는 체크 해 보아야 한다.

(note) 뒤 analysis 에서 언급하겠지만, 데이터의 수가 accuracy 에 가장 주요한 요인으로 작동한다. 데이터 수를 100 개이상~(검출하는 object 개수 training cycle 등)으로 설정하는 편이 좋다.

<Result>

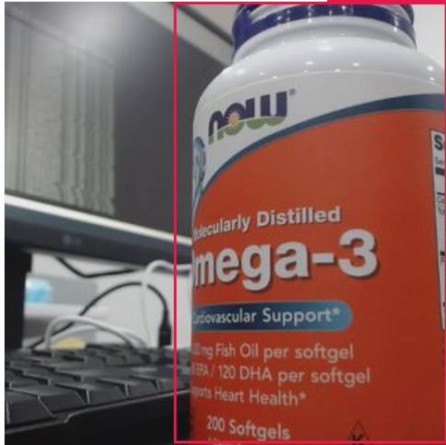
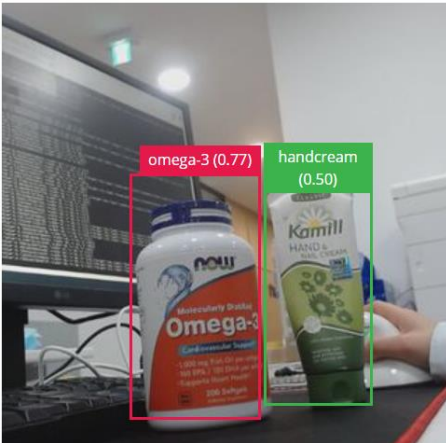
		Ver 1	Ver 2	Ver 3	Ver 4
Training cycles		25	30	25	30
Learning rate		0.15	0.15	0.1	0.1
Model Test(only Quantized)		82.46%	82.46%	78.95%	80.70%
Quantized	Validation	90.2%	90.2%	83.9%	86.9%
	ROM	3.6M	3.6M	3.6M	3.6M
Unoptimized	Validation	96.5%	96.5%	91.6%	93.5%
	ROM	10.9M	10.9M	10.9M	10.9M

Table 7. RPI 2nd Test Result

<Analysis>

1st Test 보다 데이터 수를 늘렸을 경우에 더 좋은 성능을 얻을 수 있었다. Quantized, Unoptimized 둘 다 높은 accuracy 를 보였다. 실제, device 에서 올렸을 경우에도 좋은 성능을 보였으며, Unoptimized 버전을 올렸을 경우에는 overfitting 의 양상도 어느 정도 보였다.

<On Device Test – RPI>

Result	Inference Time	Confidence
 <p>Time per inference: 476 ms.</p>	476[ms]	0.59
 <p>Time per inference: 487 ms.</p>	487[ms]	0.73
 <p>Time per inference: 493 ms.</p>	476[ms]	0.77(omega-3) 0.50(handcream)

*Quantized(int8) 모델로 테스트 진행

Table 8. RPI 3rd Test RPI Result

<On Device Test – Mobile Phone>

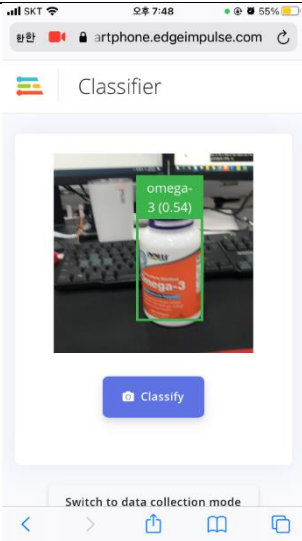
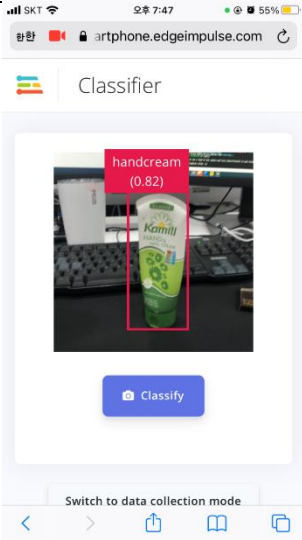
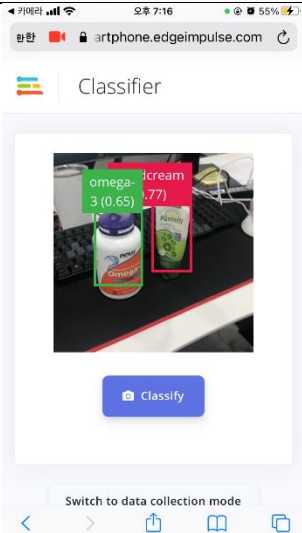
Result	Confidence
	0.54
	0.82
	0.65(omega-3) 0.77(handcream)

Table 9. RPI 3rd Test Mobile Phone Result

4.3 3rd Test

<Purpose>

1st Test 에서 Accuracy 와 Tutorial 동영상에서 보인 정확도(90% 가량)보다 현저히 낮은 원인을 RGB 로 전처리 될 때 해상도 차이라고 추정했다. 따라서, 해상도를 고르게 분포했을 경우, 높은 정확도를 보일 수 있는 지 확인하기 위한 테스트이다.

<Data>

Data Label	Training data	Test data
Airpot	31	6
Hairpin	33	4

Table 10. RPI 3rd Test Data

Information about Data : 데이터는 mobile phone 으로 취득하였으며, 같은 장소에서 배경을 변화를 주지 않고, 조명의 밝기를 조정하여서 가능한 한 사물의 색감이 고르게 분포되도록 직접 촬영하였다. 또한, 사물의 다양한 면을 담기 보다는 한 면만 촬영하여 다양성을 최소화하였다.

<Result>

		Ver 1	Ver 2	Ver 3	Ver 4
Training cycles		25	30	25	30
Learning rate		0.15	0.15	0.1	0.1
Model Test(only Quantized)		66.7%	66.67%	50.00%	66.67%
Quantized	Validation	53.6%	49.8%	47.0%	75.9%
	ROM	3.6M	3.6M	3.6M	3.6M
Unoptimized	Validation	90.8%	98%	85.2%	82%
	ROM	10.9M	10.9M	10.9M	10.9M

Table 11. RPI 3rd Test Result

<Analysis>

1st Test 보다 높은 성능을 보이지만, 2nd Test 보다 낮은 성능을 보였다. 모델 성능을 높일 수 있는 데 가장 크게 기여하는 요인은 아닌 것으로 보인다. 마찬가지로 Quantized 모델과 Unoptimized 모델의 성능 차이가 크게 나타난다. 실제 device 에서 테스트해 보았을 때, hairpin 이 airpot 보다 더 검출이 잘 됐다.

<On Device Test – RPI>

Result	Inference Time	Confidence
 <p>Time per inference: 463 ms.</p>	463[ms]	0.70
 <p>Time per inference: 456 ms.</p>	456[ms]	0.73
 <p>Time per inference: 471 ms.</p>	471[ms]	0.58(hairpin) 0.57(airpot)

*Quantized(int8) 모델로 테스트 진행

Table 12. RPI 3rd Test RPI Result

<On Device Test – Mobile Phone>

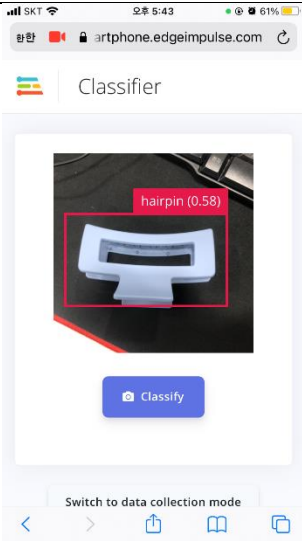
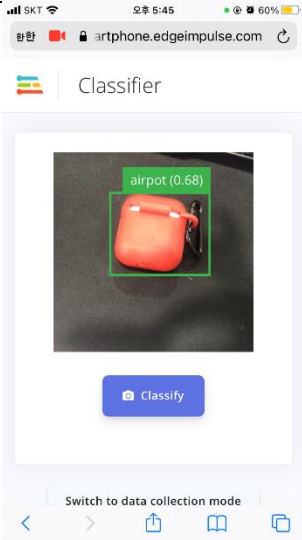
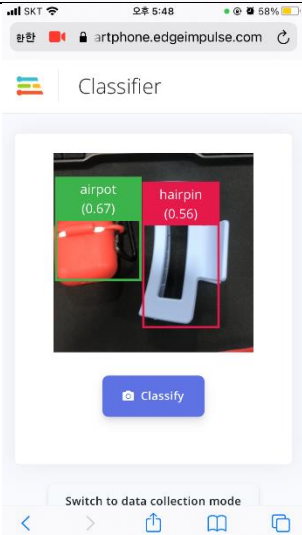
Result	Confidence
	0.58
	0.68
	0.56(hairpin) 0.67(airpot)

Table 13. RPI 3rd Test Mobile Phone Result

4.4 Issues

<Issue 1> Quantized 된 모델과 Unoptimized 된 모델의 정확도 차이가 많이 난다. Edge device 의 경우, 메모리의 제한으로 Unoptimized(float32)모델이 deploy 되지 않았으나, linux 기반의 device 의 경우에는 deploy 가 가능하다. Quantized(int8) 모델의 성능이 더 낮고, inference time 의 차이가 크지 않다면 굳이 사용할 이유가 없어 보인다.

실제 device 위에서 테스트를 진행해 보았을 때, 다운로드 받는 모델의 크기 차이로 Unoptimized(float32) 모델이 보다 느리게 다운 받아졌으나, inference time 의 차이는 없었다.

Quantized(int8) 모델의 경우 inference time 이 모델에 따라 상이하지만 450~490[ms]가량의 소요됐으며, Unoptimized(float32) 모델 또한 마찬가지였다.

<Issue 2> Quantized 모델의 경우, accuracy 가 낮아 사물 검출율이 낮을 수 있고 Unoptimized 모델의 경우, accuracy 가 높아 overfitting 이 발생하는 경우가 있다.

5. Jetson Nano Tests and Results

5.1 1st Test

<Purpose>

RPI 에서 Test 에서 가장 좋은 성능을 보였던, 데이터 수를 늘리는 방법을 택했다.

<Data>

Data Label	Training data	Test data
1(airpot)	88	1
2(lip)	92	1

Table 14. JetsonNano 1st Test Data

Information about Data : 데이터는 mobile phone 으로 취득하였으며, 다양한 배경에서 변화를 주면서 직접 촬영하였다. Label 을 간략하게 숫자로 설정한 이유는, input 데이터를 전처리 과정(box 를 치는 과정)에서 더 정교하게 box 를 치기 위해서 이다.

(note) Test data 를 아예 '0'으로 설정하면, 보드 위에 deploy 되지 않는다. Test data 를 꼭 넣어주어야 한다.

<Result>



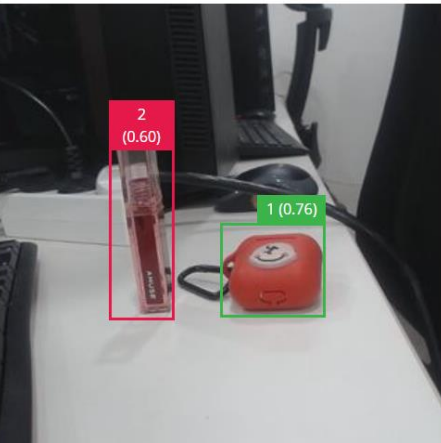
		Ver 1	Ver 2	Ver 3	Ver 4
Training cycles		25	30	25	30
Learning rate		0.15	0.15	0.1	0.1
Quantized	Validation	83.6%	84.9%	78.1%	81.4%
	ROM	3.6M	3.6M	3.6M	3.6M
Unoptimized	Validation	88.3%	89.7	84.9%	85.9%
	ROM	10.9M	10.9M	10.9M	10.9M

Table 14. JetsonNano 1st Test Result

<Analysis>

Edge Impulse tutorial 동영상에서 보인 데이터의 수보다 많은 수의 데이터를 input 으로 넣었을 때, 좋은 결과를 보였다. 모델을 보드와 mobile phone 에 deploy 했을 때도 좋은 결과를 보였다. 두 물체 모두 차이 없이 검출이 잘 되었다.

On Device Test – JetsonNano>

Result	Inference Time	Confidence
 <p>Time per inference: 233 ms.</p>	233[ms]	0.65
 <p>Time per inference: 230 ms.</p>	230[ms]	0.77
 <p>Time per inference: 226 ms.</p>	226[ms]	0.76(1) 0.60(2)

*Unoptimized(float32) 모델로 테스트 진행

Table 15. 1st Test JetsonNano Result

<On Device Test – Mobile Phone>

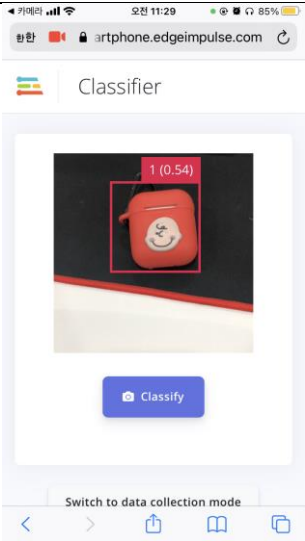
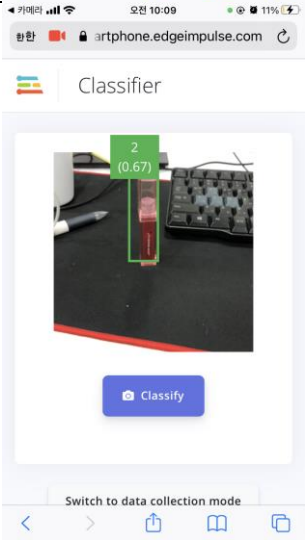
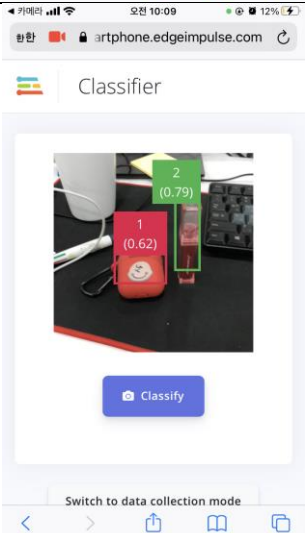
Result	Confidence
	0.54
	0.67
	0.62(1) 0.79(2)

Table 16. JetsonNano 1st Test Mobile Phone Result

5.2 2nd Test

<Purpose>

1st Test 에서 원하는 정확도를 보였으므로, Edge Impulse Tutorial 동영상에서 소개한 대로 적은 데이터 수로 높은 모델 퍼포먼스를 보이게 했다. 적은 데이터로 training cycle 과 Learning rate 를 조절해서 높은 정확도를 보이는 지 확인하기 위한 테스트이다.

<Data>

	Training data	Test data
1(brush)	29	1
2(board)	29	1

Table 17. JetsonNano 2nd Test Data

Information about Data : 데이터는 mobile phone 으로 취득하였으며, 가능한한 적은 배경 변화를 주었다. 2 가지 배경에서 촬영을 하였으며, 대신 배경에는 다양한 물체들을 포함하였다. 물체의 위치 방향 순서 또한 2 가지 정도의 변화만 주고 촬영하였다.

<Result>

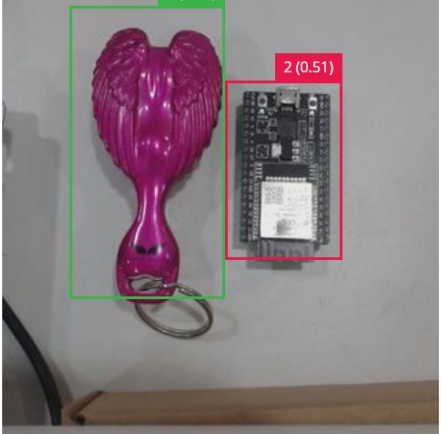
		Ver 1	Ver 2	Ver 3	Ver 4
Training cycles		100	100	200	150
Learning rate		0.15	0.1	0.15	0.15
Quantized	Validation	67.5%	64.5%	X	76.5%
	ROM	3.6M	3.6M		3.6M
Unoptimized	Validation	81.4%	78.5%		81.4%
	ROM	10.9M	10.9M		10.9M

Table 18. JetsonNano 2nd Test Result

<Analysis>

Edge Impulse Tutorial 에서 만큼 좋은 결과를 얻을 수 없었다. 하지만, training cycle 을 늘렸을 때, 높은 정확도를 보이는 것을 확인할 수 있었다. Quantized(int8)과 Unoptimized(float32)의 모델 성능 차이가 크다. Jetson Nano 는 아주 적은 memory 를 가진 타 edge device 에 비해 큰 memory access 가 가능함으로 Unoptimized(float32) 버전을 선택하는 편이 좋다. 실제 테스트를 진행하였을 때, board 보다 brush 를 더 잘 검출하였다.

On Device Test – JetsonNano>

Result	Inference Time	Confidence
 <p>Time per inference: 225 ms.</p>	225[ms]	0.80
 <p>Time per inference: 227 ms.</p>	227[ms]	0.81
 <p>Time per inference: 243 ms.</p>	243[ms]	0.66(1) 0.51(2)

*Unoptimized(float32) 모델로 테스트 진행

Table 19. JetsonNano 2nd Test JetsonNano Result

<On Device Test – Mobile Phone>

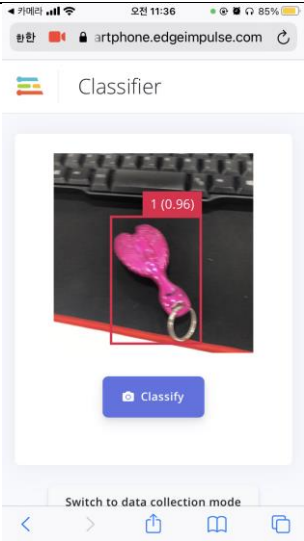
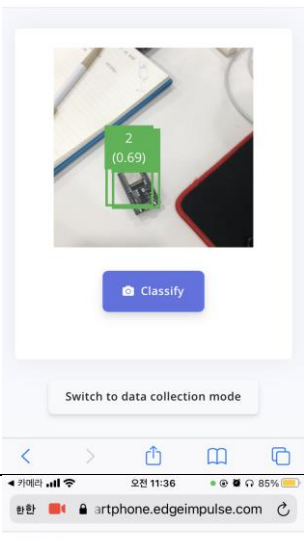
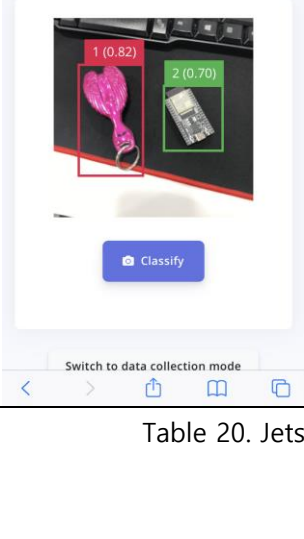
Result	Confidence
	0.96
	0.69
	0.82(1) 0.70(2)

Table 20. JetsonNano 2nd Test Mobile Phone Result

5.3 Issues

<Issue 1> TensorRT 를 이용한 Live Classification 모델이 다운로드 받아졌으나, build 되는 과정에서 문제가 생겨 해결 중에 있다.

```
ash -lfft2d_fftsig -lfft2d_fftsig2d -lruby -lXNNPACK -lpthread
source/custom.o: In function 'run_inference':
/home/nota/edgeimpulse/.edge-impulse-sdk/classifier/ei_run_classifier.h:886: undefined reference to `libeitrtrt::create_EiTrt(char const*, bool)'
/home/nota/edgeimpulse/.edge-impulse-sdk/classifier/ei_run_classifier.h:891: undefined reference to `libeitrtrt::infer(EiTrt*, float*, float*, int)'
clang: error: linker command failed with exit code 1 (use -v to see invocation)
Makefile:94: recipe for target 'runner' failed
make: *** [runner] Error 1
nota@nota-desktop:~/edgeimpulse$
```

Fig 27. TensorRT Build 에러 모습