

Edge Computing Laboratory

Lab Assignment 10

Name: Janhavi Khune

Class: TY AIEC Batch C

Enrollment No: MITU22BTCS0348

Roll No: 2223862

Title

Study of Transfer Learning (Images) on Edge Computing Devices

Objective: Build a project to apply Transfer Learning of MobileNetV1 & V2 architectures trained on an ImageNet dataset

Tasks:

- Understand Transfer learning
- Understanding of MobileNetV1 & V2 Architectures
- Configure Edge Impulse for Object Detection
- Apply a pre-trained network for you to fine-tune your specific application
- Building and Training a Model
- Deploy on Edge Computing Devices

Introduction

Edge Impulse is a development platform for machine learning on edge devices, targeted at developers who want to create intelligent device solutions. The "Camera" sensor reading 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:

40. Create an Account and New Project:

- Sign up for an Edge Impulse account.
- Create a new project from the dashboard.

41. 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.

42. 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.

43. 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.

44. 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.

45. Train the Model:

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

46. Test the Model:

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

47. 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.

48. Run Inference:

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

49. Monitor:

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

Screenshots:

Dataset image

The screenshot displays the Edge Impulse web interface. The top navigation bar includes the user name 'Gaurav Gadekar / Transfer Learning' and a target device 'Target: Cortex-M4F 80MHz'. The left sidebar contains a navigation menu with options like Dashboard, Devices, Data acquisition, Experiments, EON Tuner, Impulse design, and Upgrade Plan. The main content area is divided into three sections: 'Dataset' (showing 64 items collected), 'Train / Test Split', and 'Collect data'. The 'Dataset' section features a table with columns for Sample Name, Labels, and Added. The table lists several samples, with 'Headphone.5rehc1df' highlighted. The 'Collect data' section includes a button to 'Connect a device' to start building the dataset. A 'RAW DATA' preview on the right shows a photograph of a pair of headphones.

SAMPLE NAME	LABELS	ADDED
Headphone.5rehc7mg	-	Today, 19:58:30
Headphone.5rehc4qf	-	Today, 19:58:27
Headphone.5rehc1df	-	Today, 19:58:23

The screenshot displays the Edge Impulse web interface, similar to the first one, but with a different dataset. The 'Dataset' section shows 64 items collected. The table lists several mobile phone samples, with 'Mobile.5reimggr' highlighted. The 'RAW DATA' preview on the right shows a photograph of a red smartphone.

SAMPLE NAME	LABELS	ADDED
Mobile.5reimsae	-	Today, 20:21:47
Mobile.5reimq18	-	Today, 20:21:45
Mobile.5reimo9l	-	Today, 20:21:43
Mobile.5reimggr	-	Today, 20:21:35
Mobile.5reimdii	-	Today, 20:21:32
Mobile.5reimbq7	-	Today, 20:21:30
Mobile.5reim9bb	-	Today, 20:21:28
Mobile.5reim749	-	Today, 20:21:25

Feature extraction - Image

The screenshot shows the Edge Impulse IDE interface. On the left is a sidebar with navigation links: Dashboard, Devices, Data acquisition, Experiments, EON Tuner, Impulse design (selected), Upgrade Plan, and View plans. The main workspace is titled 'Impulse #1' and contains four panels: 'Image data' (red), 'Image' (white), 'Object Detection (Images)' (blue), and 'Output features' (green). The 'Image data' panel has input axes for 'image' with width and height set to 96, and a 'Resize mode' dropdown set to 'Fit shortest axis'. The 'Image' panel has a name 'Image' and input axes set to 'Image'. The 'Object Detection (Images)' panel has a name 'Object detection' and input features checked for 'Image'. The 'Output features' panel shows '0 ()' and a 'Save Impulse' button. At the top right, it indicates 'Target: Cortex-M4F 80MHz' and a user profile 'JK'.

Accuracy / Loss image

The screenshot shows the Edge Impulse IDE interface for an impulse. At the top is a header with a dark blue background and a small image of a red smartphone. Below the header are four main sections: 'Raw features', 'Parameters', 'DSP result', and 'On-device performance'. The 'Raw features' section displays a hex string: '0x9a9a98, 0x9c9c9a, 0x9d9db, 0x9b9b9a, 0x979794, 0x9b9c96, 0x9d9f9a, 0x9e9e9a, 0xa0a00b, 0xa...'. The 'Parameters' section has a dropdown for 'Image' set to 'Color depth' with a value of 'RGB' and a 'Save parameters' button. The 'DSP result' section shows a small image of the red smartphone. The 'On-device performance' section displays two metrics: 'PROCESSING TIME' at '7 ms.' and 'PEAK RAM USAGE' at '4 KB'.

Validation Result – Image

Neural Network settings

Training settings

Number of training cycles 60

Use learned optimizer ☐

Learning rate 0.001

Training processor CPU

Data augmentation ☒

Advanced training settings

Validation set size 20 %

Split train/validation set on metadata key

Batch size 32

Profile int8 model ☒

Training output

Calculating inferencing time OK
Calculating float32 accuracy...
INFO: Created TensorFlow Lite XNNPACK delegate for CPU.

Calculating int8 accuracy...

Model training complete

Model training complete

Job completed (success)

Model

Model version: Quantized (int8)

Last training performance (validation set)

F1 SCORE 94.7%

Confusion matrix (validation set)

- **Conclusion:** Understood of MobileNetV1 & V2 Architectures and custom training on new dataset for edge devices.