Edge Intelligence

P Sushma

25MML0050

Task 1:

- Analysing an image dataset MNIST, and applying basic processing operations.
- Applied Convolution neural network on the following dataset.
- Saving the model using pickle model.

[2]:
```
import tensorflow as tf
from tensorflow.keras import layers, models
import pickle

(train_images, train_labels), (test_images, test_labels) = tf.keras.datasets.mnist.load_data()

x_train, y_train = train_images[:600], train_labels[:600]
x_test, y_test = test_images[:400], test_labels[:400]
x_train = x_train.reshape((600, 28, 28, 1)).astype('float32') / 255.0
x_test = x_test.reshape((400, 28, 28, 1)).astype('float32') / 255.0
```

[3]:
```
model = models.Sequential([
    layers.Conv2D(32, (3, 3), activation='relu', input_shape=(28, 28, 1)),
    layers.MaxPooling2D((2, 2)),

    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D((2, 2)),

    layers.Flatten(),
    layers.Dense(64, activation='relu'),
    layers.Dense(10, activation='softmax')
])
```

```
C:\Users\sushm\anaconda3\envs\ml_lab\lib\site-packages\keras\src\layers\convolutional\base_conv.py:113: UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer. When using Sequential models, prefer using an `Input(shape)` object as the first layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
```

[4]:
```
model.compile(optimizer='adam',
              loss='sparse_categorical_crossentropy',
              metrics=['accuracy'])

print("Training CNN on 600 images...")
model.fit(x_train, y_train, epochs=10, batch_size=32)
```

```
Training CNN on 600 images...
Epoch 1/10
19/19 ━━━━━━━━━━━━━━━━━━━━ 3s 17ms/step - accuracy: 0.2834 - loss: 2.1807
Epoch 2/10
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 11ms/step - accuracy: 0.6867 - loss: 1.2433
Epoch 3/10
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 14ms/step - accuracy: 0.7901 - loss: 0.6412
Epoch 4/10
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 20ms/step - accuracy: 0.8976 - loss: 0.3552
Epoch 5/10
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 15ms/step - accuracy: 0.9333 - loss: 0.2504
Epoch 6/10
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - accuracy: 0.9436 - loss: 0.1910
Epoch 7/10
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 15ms/step - accuracy: 0.9583 - loss: 0.1598
Epoch 8/10
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 14ms/step - accuracy: 0.9804 - loss: 0.0949
Epoch 9/10
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 10ms/step - accuracy: 0.9836 - loss: 0.0825
Epoch 10/10
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 13ms/step - accuracy: 0.9826 - loss: 0.0591
```

[4]:
```
<keras.src.callbacks.history.History at 0x2b90c2adac0>
```

[5]:
```
print("\nEvaluating CNN on 400 images...")
test_loss, test_acc = model.evaluate(x_test, y_test, verbose=1)
print(f"CNN Test Accuracy: {test_acc * 100:.2f}%")
```

```
Evaluating CNN on 400 images...
13/13 ━━━━━━━━━━━━━━━━━━━━ 0s 12ms/step - accuracy: 0.9174 - loss: 0.2569
CNN Test Accuracy: 90.50%
```

[7]:
```
with open('mnist_cnn_model.pkl', 'wb') as f:
    pickle.dump(model, f)
print("\nCNN model saved to mnist_cnn_model.pkl")
```

```
CNN model saved to mnist_cnn_model.pkl
```

[8]:
```
import pickle
import os
import tensorflow as tf
import numpy as np
with open('mnist_cnn_model.pkl', 'rb') as f:
    loaded_model = pickle.load(f)

print("--- Model Loaded Successfully ---")
```

```
--- Model Loaded Successfully ---
```

[9]:
```
first_layer = loaded_model.layers[0]
weights, biases = first_layer.get_weights()

print(f"Layer Name: {first_layer.name}")
print(f"Weights Shape (Filter H, Filter W, Channels, Num Filters): {weights.shape}")
print(f"Biases Shape: {biases.shape}")
print("\nSample Weights (first 3x3 filter kernel):\n", weights[:, :, 0, 0])
```

```
Layer Name: conv2d
Weights Shape (Filter H, Filter W, Channels, Num Filters): (3, 3, 1, 32)
Biases Shape: (32,)

Sample Weights (first 3x3 filter kernel):
 [[-0.15950842 -0.04815693 -0.06375122]
 [-0.08455123 -0.01896419 -0.0415653 ]
 [ 0.17882669 -0.01614539  0.18866213]]
```

[11]:
```
(train_images, train_labels), (test_images, test_labels) = tf.keras.datasets.mnist.load_data()
x_train = train_images[:600].reshape((600, 28, 28, 1)).astype('float32') / 255.0
y_train = train_labels[:600]
x_test = test_images[:400].reshape((400, 28, 28, 1)).astype('float32') / 255.0
y_test = test_labels[:400]
```

[12]:
```
print("\n--- Resuming Training for 2 More Epochs ---")
loaded_model.fit(x_train, y_train, epochs=2, batch_size=32)

print("\n--- Running Evaluation ---")
test_loss, test_acc = loaded_model.evaluate(x_test, y_test, verbose=1)
print(f"Test Accuracy: {test_acc * 100:.2f}%")

file_size = os.path.getsize('mnist_cnn_model.pkl') / (1024 * 1024) # Convert to MB
print(f"\nModel File Size on Disk: {file_size:.2f} MB")
```

```
--- Resuming Training for 2 More Epochs ---
Epoch 1/2
19/19 ━━━━━━━━━━━━━━━━━━━━ 2s 20ms/step - accuracy: 0.9830 - loss: 0.0552
Epoch 2/2
19/19 ━━━━━━━━━━━━━━━━━━━━ 0s 16ms/step - accuracy: 0.9992 - loss: 0.0223

--- Running Evaluation ---
13/13 ━━━━━━━━━━━━━━━━━━━━ 1s 12ms/step - accuracy: 0.9589 - loss: 0.1439
Test Accuracy: 93.50%

Model File Size on Disk: 1.43 MB
```

```
[13]: model.summary()
```

```
Model: "sequential"
```

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 26, 26, 32)	320
max_pooling2d (MaxPooling2D)	(None, 13, 13, 32)	0
conv2d_1 (Conv2D)	(None, 11, 11, 64)	18,496
max_pooling2d_1 (MaxPooling2D)	(None, 5, 5, 64)	0
flatten (Flatten)	(None, 1600)	0
dense (Dense)	(None, 64)	102,464
dense_1 (Dense)	(None, 10)	650

```
Total params: 365,792 (1.40 MB)

Trainable params: 121,930 (476.29 KB)

Non-trainable params: 0 (0.00 B)

Optimizer params: 243,862 (952.59 KB)
```

Task 2:

The following steps for gathering and organizing data using the Edge Impulse platform:

1. Account Setup

Begin by visiting the Edge Impulse website to create a new user account. Once registered and logged into the Edge Impulse Studio, you will have access to the dashboard where you can manage projects, datasets, and training configurations.

2. Accessing Data Acquisition

Navigate to the Data Acquisition tab within your project dashboard. This section serves as the central hub for collecting and uploading the raw inputs—such as images, audio, or sensor readings—required to train your model.

3. Connecting a Device

Click on the "Connect Data" option. This feature allows you to link external hardware (such as a development board, computer, or smartphone) directly to the platform for real-time data streaming.

4. Smartphone Integration via QR Code

Edge Impulse will generate a unique QR code on your screen. Scan this code with your smartphone to bridge your mobile device with the project. This turns your phone into a remote sensor, allowing you to capture and upload images directly to the cloud.

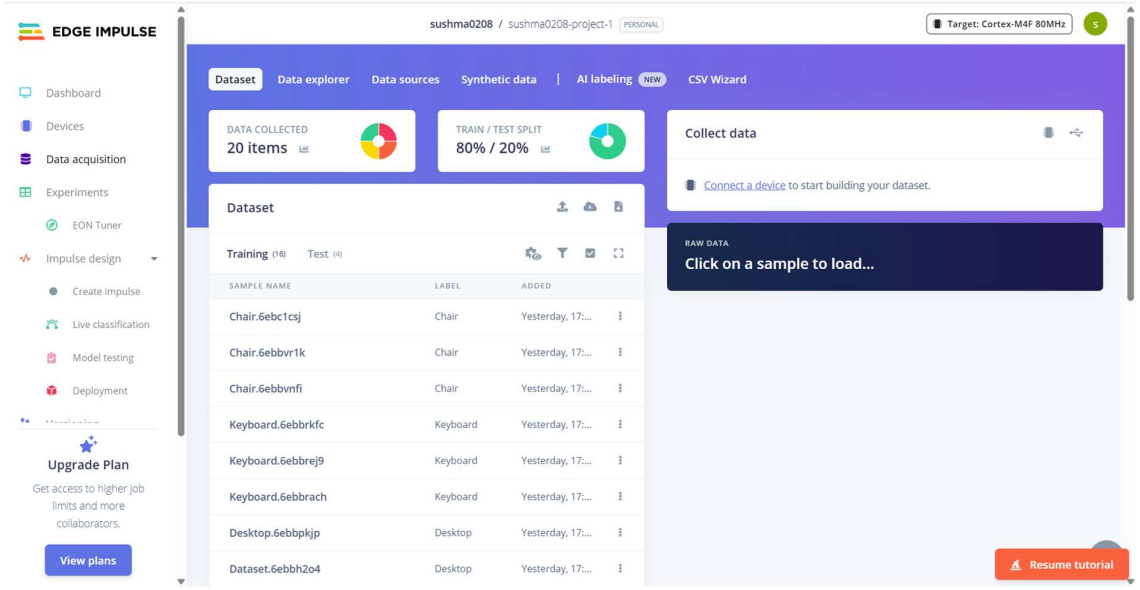
5. Labeling Data for Supervised Learning

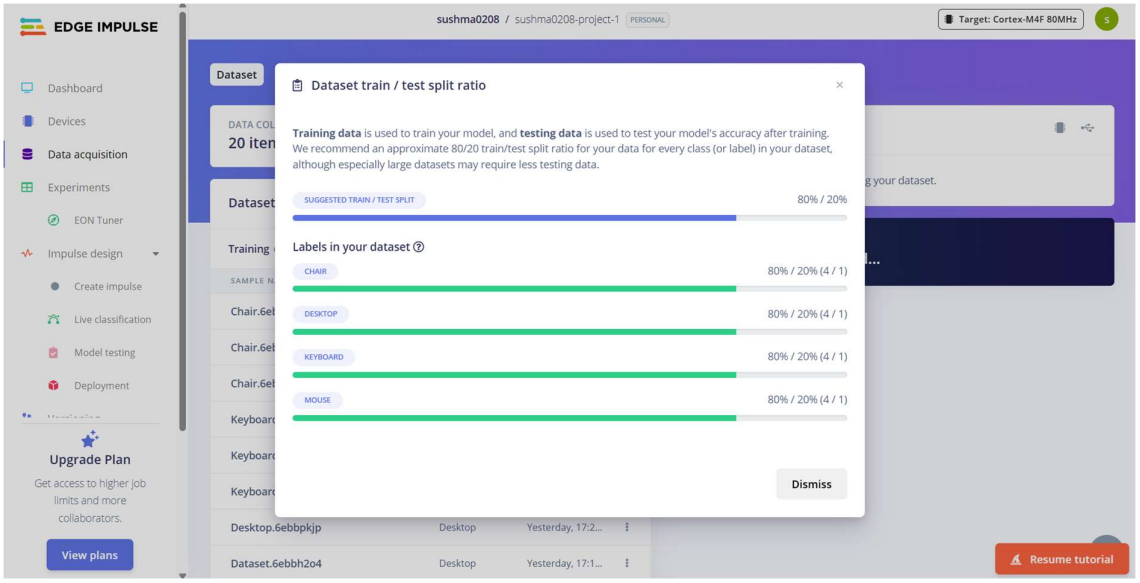
Before capturing an image, it is vital to assign a Label (e.g., "cat," "dog," or "car"). Accurate labeling at the source ensures the machine learning model can correctly categorize the features of each image during the training process.

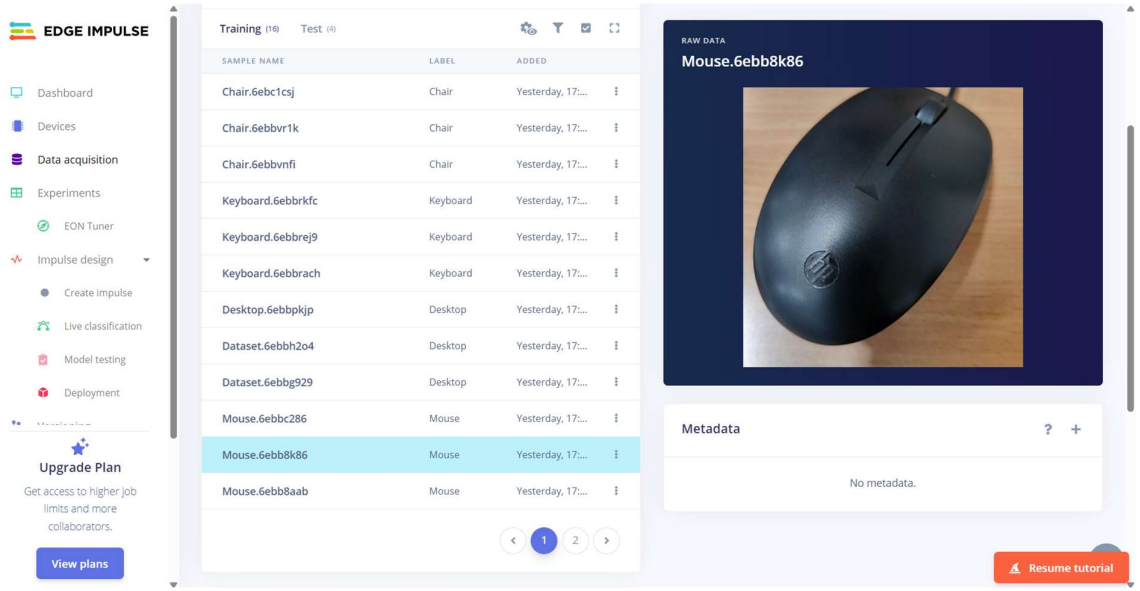
6. Dataset Partitioning (Train/Test Split)

After collecting the images, the dataset must be divided to evaluate model performance fairly. In this task, a total of 20 images were collected and split using an 80:20 ratio:

- Training Set: 16 images (used to teach the model).
- Testing Set: 4 images (used to validate accuracy).




EDGE IMPULSE
sushma0208 / sushma0208-project-1
PERSONAL
Target: Cortex-M4F 80MHz
Dashboard
Devices
Data acquisition
Experiments
EON Tuner
Impulse design
Create impulse
Live classification
Model testing
Deployment
Upgrade Plan
Get access to higher job limits and more collaborators.
View plans
Dataset
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
CHAIR
80% / 20% (4 / 1)
DESKTOP
80% / 20% (4 / 1)
KEYBOARD
80% / 20% (4 / 1)
MOUSE
80% / 20% (4 / 1)
Dismiss
Desktop.6ebbpkjp
Desktop
Dataset.6ebbh2o4
Desktop
Resume tutorial


EDGE IMPULSE
Dashboard
Devices
Data acquisition
Experiments
EON Tuner
Impulse design
Create impulse
Live classification
Model testing
Deployment
Upgrade Plan
Get access to higher job limits and more collaborators.
View plans
Training (16)
Test (4)
SAMPLE NAME
LABEL
ADDED
Chair.6ebc1csj
Chair.6ebbvr1k
Chair.6ebbvnfi
Keyboard.6ebbrkfc
Keyboard.6ebbrej9
Keyboard.6ebbrach
Desktop.6ebbpkjp
Dataset.6ebbh2o4
Dataset.6ebbg929
Mouse.6ebbc286
Mouse.6ebb8k86
Mouse.6ebb8aab
Yesterday, 17:...
RAW DATA
Mouse.6ebb8k86
Metadata
No metadata.
Resume tutorial