MACSE604 – Edge Intelligence

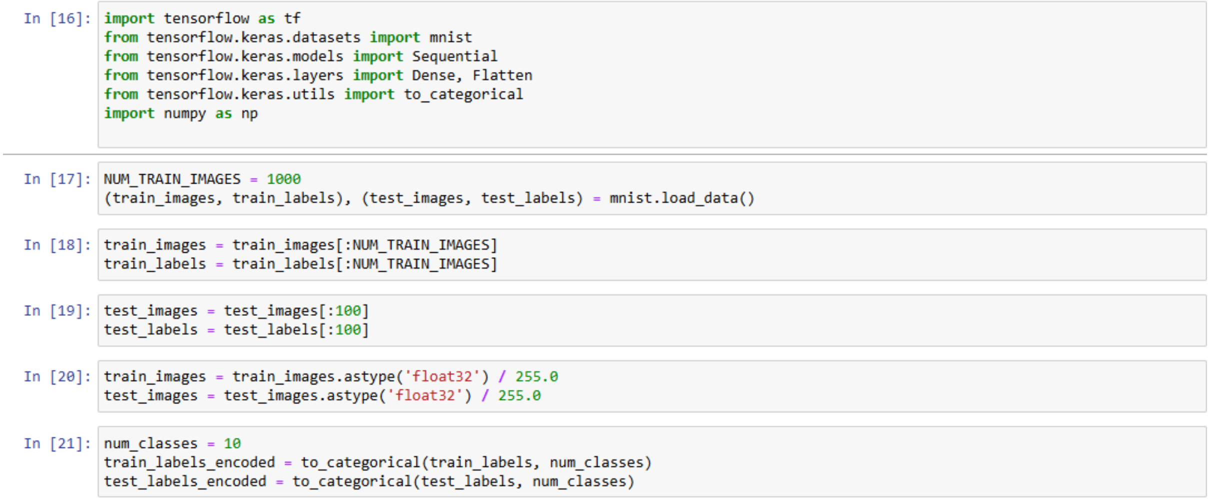
Lab Task 2

MNIST training and Edge Impulse

Name: Rohit P

Roll No: 25MML0001

1. MNIST Training



```
In [16]: import tensorflow as tf
         from tensorflow.keras.datasets import mnist
         from tensorflow.keras.models import Sequential
         from tensorflow.keras.layers import Dense, Flatten
         from tensorflow.keras.utils import to_categorical
         import numpy as np
```

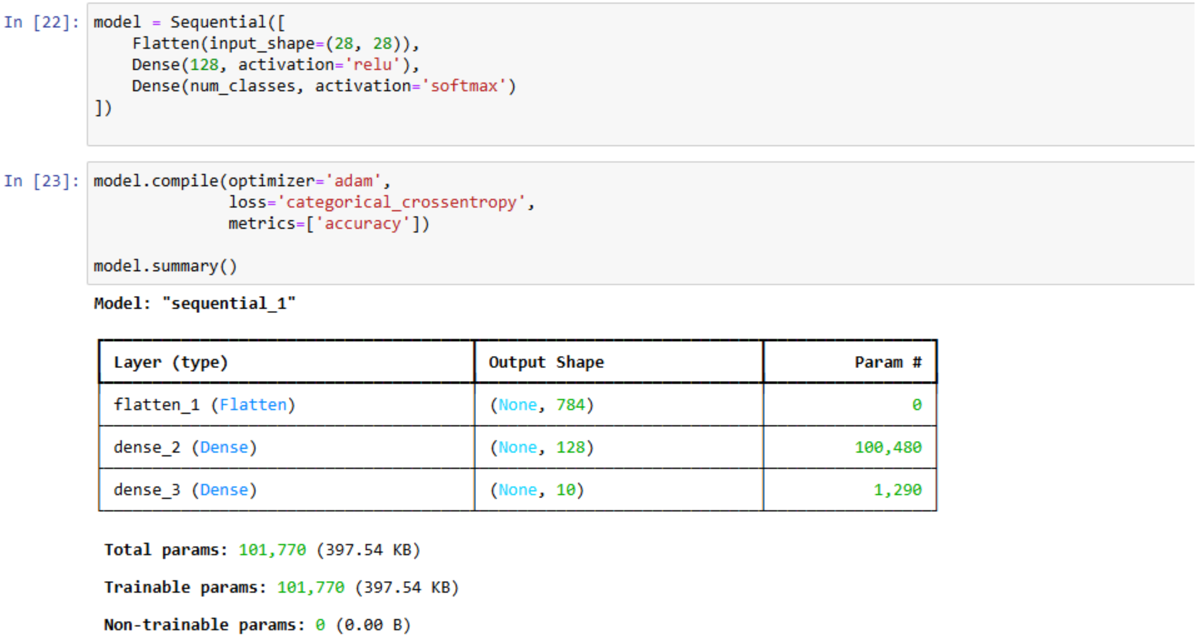
```
In [17]: NUM_TRAIN_IMAGES = 1000
         (train_images, train_labels), (test_images, test_labels) = mnist.load_data()
```

```
In [18]: train_images = train_images[:NUM_TRAIN_IMAGES]
         train_labels = train_labels[:NUM_TRAIN_IMAGES]
```

```
In [19]: test_images = test_images[:100]
         test_labels = test_labels[:100]
```

```
In [20]: train_images = train_images.astype('float32') / 255.0
         test_images = test_images.astype('float32') / 255.0
```

```
In [21]: num_classes = 10
         train_labels_encoded = to_categorical(train_labels, num_classes)
         test_labels_encoded = to_categorical(test_labels, num_classes)
```



```
In [22]: model = Sequential([
             Flatten(input_shape=(28, 28)),
             Dense(128, activation='relu'),
             Dense(num_classes, activation='softmax')
         ])
```

```
In [23]: model.compile(optimizer='adam',
                       loss='categorical_crossentropy',
                       metrics=['accuracy'])

         model.summary()
```

Model: "sequential_1"

Layer (type)	Output Shape	Param #
flatten_1 (Flatten)	(None, 784)	0
dense_2 (Dense)	(None, 128)	100,480
dense_3 (Dense)	(None, 10)	1,290

Total params: 101,770 (397.54 KB)

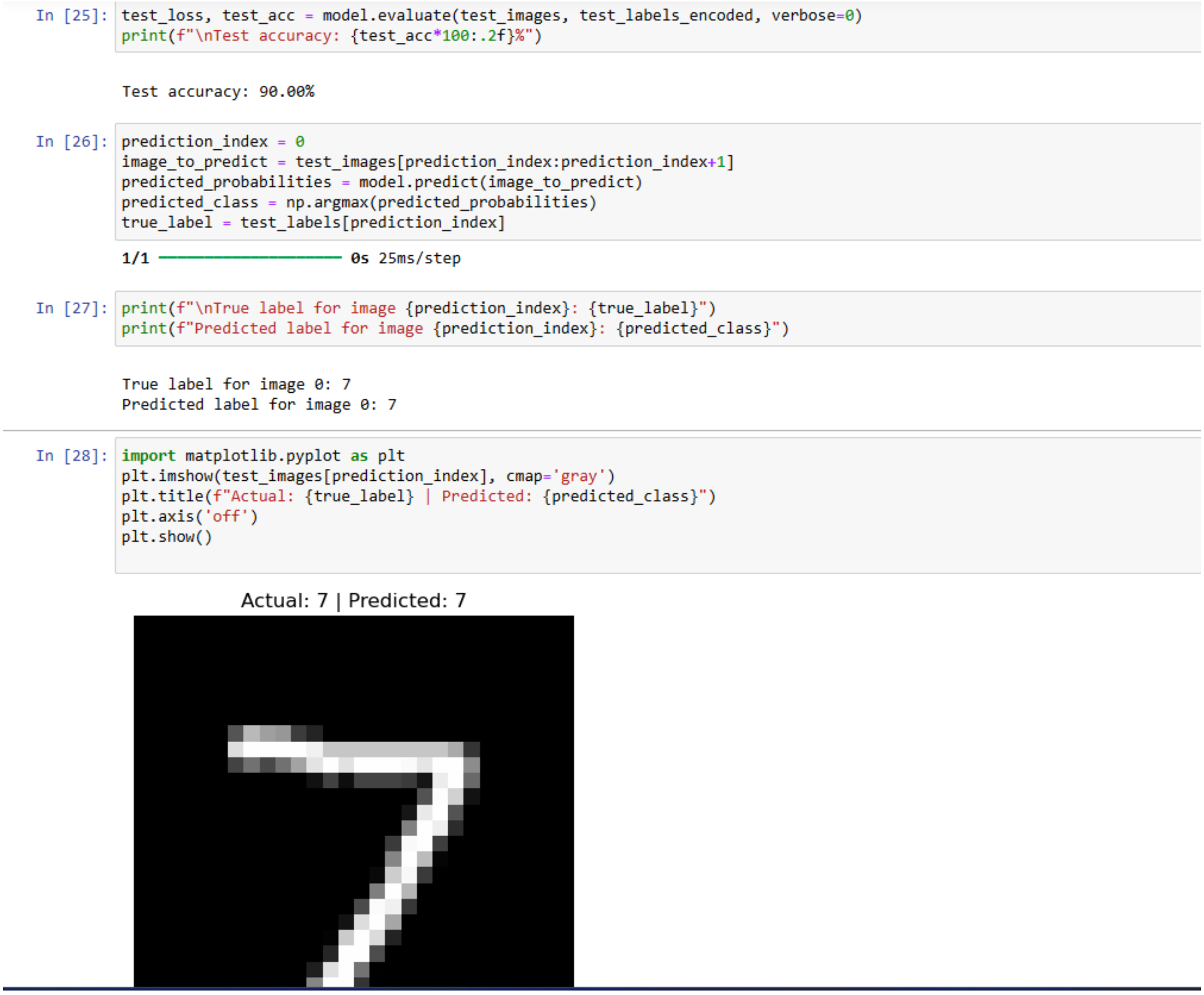
Trainable params: 101,770 (397.54 KB)

Non-trainable params: 0 (0.00 B)

In [24]:

```python
history = model.fit(train_images, train_labels_encoded,
                    epochs=20,
                    batch_size=32,
                    validation_split=0.2,
                    verbose=1)
```

```
Epoch 1/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 5ms/step - accuracy: 0.3497 - loss: 1.9986 - val_accuracy: 0.7600 - val_loss: 1.0575
Epoch 2/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.8294 - loss: 0.8072 - val_accuracy: 0.8250 - val_loss: 0.6681
Epoch 3/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.8933 - loss: 0.4842 - val_accuracy: 0.8400 - val_loss: 0.5720
Epoch 4/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9058 - loss: 0.3572 - val_accuracy: 0.8450 - val_loss: 0.5366
Epoch 5/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9342 - loss: 0.2939 - val_accuracy: 0.8650 - val_loss: 0.5142
Epoch 6/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9651 - loss: 0.1938 - val_accuracy: 0.8500 - val_loss: 0.4852
Epoch 7/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9774 - loss: 0.1571 - val_accuracy: 0.8650 - val_loss: 0.4963
Epoch 8/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9873 - loss: 0.1271 - val_accuracy: 0.8650 - val_loss: 0.5026
Epoch 9/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9802 - loss: 0.1250 - val_accuracy: 0.8750 - val_loss: 0.5010
Epoch 10/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9909 - loss: 0.1020 - val_accuracy: 0.8800 - val_loss: 0.4954
Epoch 11/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9937 - loss: 0.0807 - val_accuracy: 0.8700 - val_loss: 0.4993
Epoch 12/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 0.9919 - loss: 0.0722 - val_accuracy: 0.8700 - val_loss: 0.5085
Epoch 13/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 1.0000 - loss: 0.0572 - val_accuracy: 0.8750 - val_loss: 0.5024
Epoch 14/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 1.0000 - loss: 0.0461 - val_accuracy: 0.8750 - val_loss: 0.5061
Epoch 15/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 1.0000 - loss: 0.0369 - val_accuracy: 0.8800 - val_loss: 0.5027
Epoch 16/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 1.0000 - loss: 0.0344 - val_accuracy: 0.8750 - val_loss: 0.5131
Epoch 17/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 1.0000 - loss: 0.0320 - val_accuracy: 0.8800 - val_loss: 0.5107
Epoch 18/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 1.0000 - loss: 0.0266 - val_accuracy: 0.8800 - val_loss: 0.5232
Epoch 19/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 1.0000 - loss: 0.0253 - val_accuracy: 0.8800 - val_loss: 0.5225
Epoch 20/20
25/25 ━━━━━━━━━━━━━━━━━━━━ 0s 2ms/step - accuracy: 1.0000 - loss: 0.0232 - val_accuracy: 0.8800 - val_loss: 0.5293
```



```
In [25]: test_loss, test_acc = model.evaluate(test_images, test_labels_encoded, verbose=0)
         print(f"\nTest accuracy: {test_acc*100:.2f}%")

         Test accuracy: 90.00%

In [26]: prediction_index = 0
         image_to_predict = test_images[prediction_index:prediction_index+1]
         predicted_probabilities = model.predict(image_to_predict)
         predicted_class = np.argmax(predicted_probabilities)
         true_label = test_labels[prediction_index]

         1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 25ms/step

In [27]: print(f"\nTrue label for image {prediction_index}: {true_label}")
         print(f"Predicted label for image {prediction_index}: {predicted_class}")

         True label for image 0: 7
         Predicted label for image 0: 7

In [28]: import matplotlib.pyplot as plt
         plt.imshow(test_images[prediction_index], cmap='gray')
         plt.title(f"Actual: {true_label} | Predicted: {predicted_class}")
         plt.axis('off')
         plt.show()
```

In [30]:
```
import pickle
```

In [31]:
```
filename = 'finalized_model.pkl'

with open(filename, 'wb') as file:
    pickle.dump(model, file)

print(f"Model saved as {filename}")
```

```
Model saved as finalized_model.pkl
```

In [32]:
```
loaded_model = None
with open(filename, 'rb') as file:
    loaded_model = pickle.load(file)

print("Model loaded successfully.")
```

```
Model loaded successfully.
```

In [35]:
```
model.save('model.keras')
```

In [41]:
```
predicted_probabilities_2 = model.predict(test_images[88:88+1])
predicted_class_2 = np.argmax(predicted_probabilities_2)
true_label_2 = test_labels[88]
```

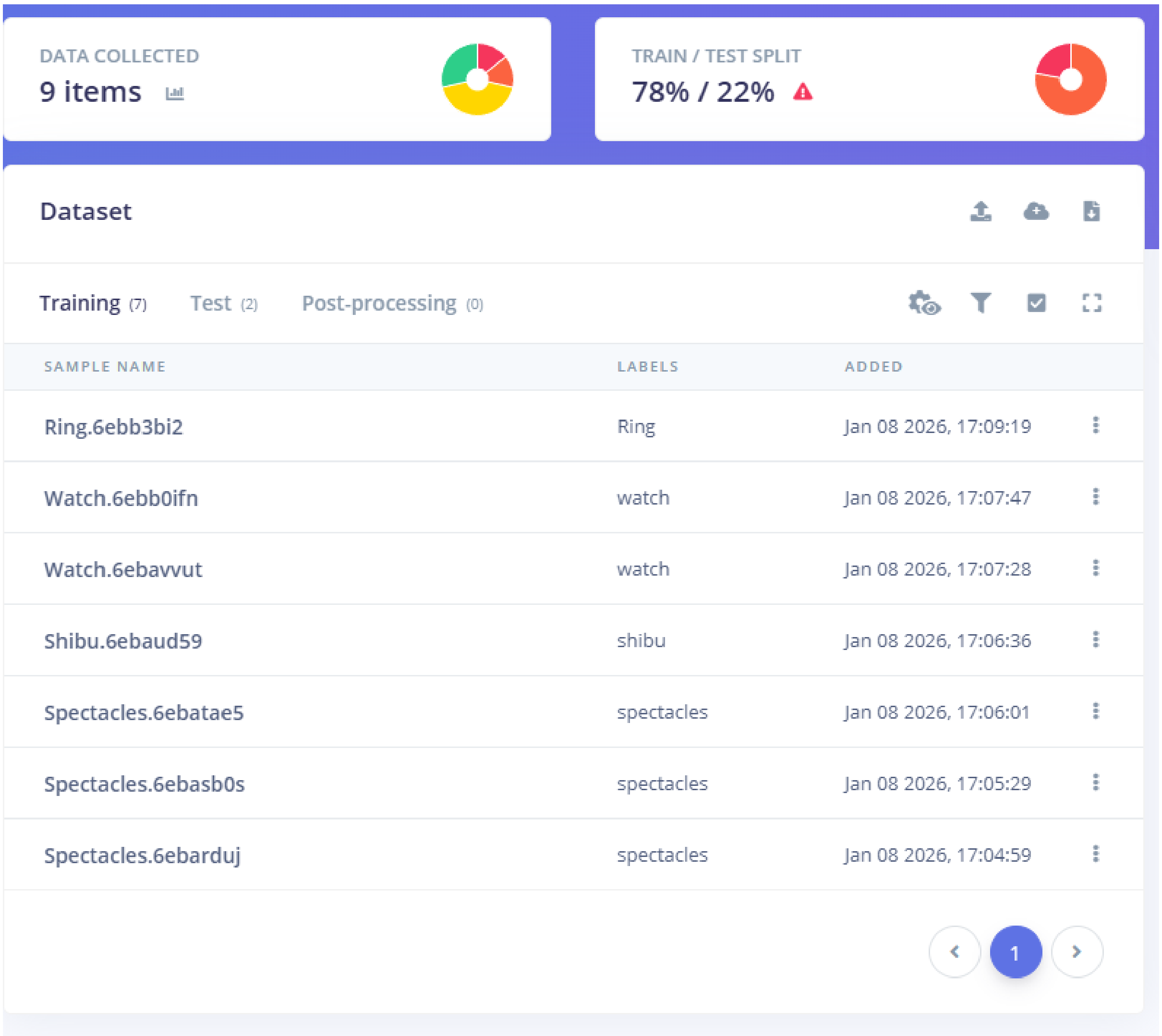
```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 11ms/step
```

In [43]:
```
print(f"\nTrue label for image : {true_label_2}")
print(f"Predicted label for image : {predicted_class_2}")
```

```

True label for image : 6
Predicted label for image : 6
```

2. Edge Impulse



DATA COLLECTED

9 items

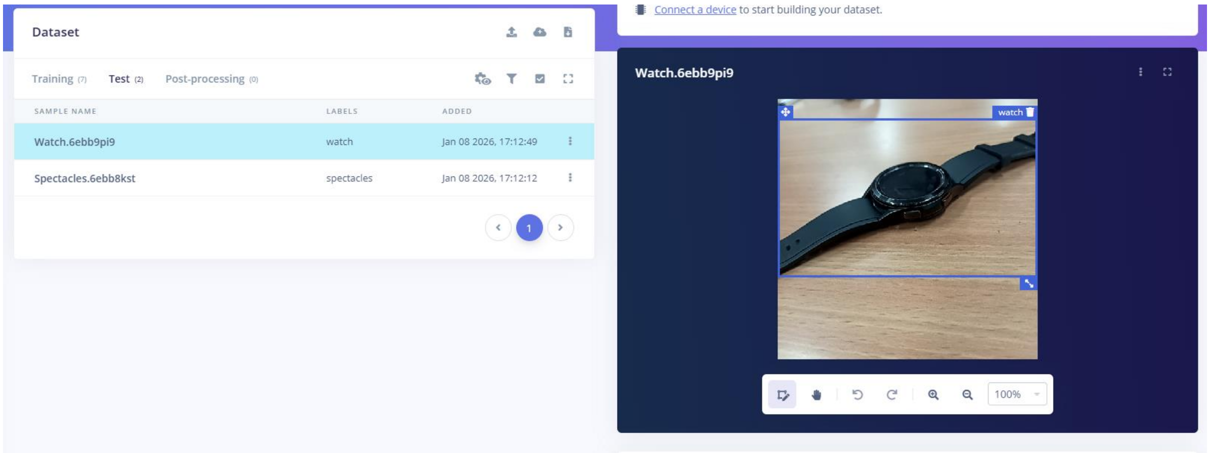
TRAIN / TEST SPLIT

78% / 22%

Dataset

Training (7) Test (2) Post-processing (0)

SAMPLE NAME	LABELS	ADDED
Ring.6ebb3bi2	Ring	Jan 08 2026, 17:09:19
Watch.6ebb0ifn	watch	Jan 08 2026, 17:07:47
Watch.6ebavvut	watch	Jan 08 2026, 17:07:28
Shibu.6ebaud59	shibu	Jan 08 2026, 17:06:36
Spectacles.6ebatae5	spectacles	Jan 08 2026, 17:06:01
Spectacles.6ebasb0s	spectacles	Jan 08 2026, 17:05:29
Spectacles.6ebarduj	spectacles	Jan 08 2026, 17:04:59



Dataset

Training (7) Test (2) Post-processing (0)

SAMPLE NAME	LABELS	ADDED
Watch.6ebb9pi9	watch	Jan 08 2026, 17:12:49
Spectacles.6ebb8kst	spectacles	Jan 08 2026, 17:12:12

Connect a device to start building your dataset.

Watch.6ebb9pi9