L02- Converting and Deploying AI Models using TensorFlow Lite

Option B - Practical

Part 1: Setting Up the Development Environment

Step 1: Verify Python and TensorFlow Installation

In [3]:
```
!python --version
```

```
Python 3.12.7
```

In [4]:
```
!pip show tensorflow
```

```
Name: tensorflow
Version: 2.18.0
Summary: TensorFlow is an open source machine learning framework for everyone.
Home-page: https://www.tensorflow.org/
Author: Google Inc.
Author-email: packages@tensorflow.org
License: Apache 2.0
Location: C:\Users\banke\anaconda3\Lib\site-packages
Requires: tensorflow-intel
Required-by:
```

In [5]:
```
!pip install tensorflow
```

```
Requirement already satisfied: tensorflow in c:\users\banke\anaconda3\lib\site-packa
ges (2.18.0)
Requirement already satisfied: tensorflow-intel==2.18.0 in c:\users\banke\anaconda3
\lib\site-packages (from tensorflow) (2.18.0)
Requirement already satisfied: absl-py>=1.0.0 in c:\users\banke\anaconda3\lib\site-p
ackages (from tensorflow-intel==2.18.0->tensorflow) (2.1.0)
Requirement already satisfied: astunparse>=1.6.0 in c:\users\banke\anaconda3\lib\sit
e-packages (from tensorflow-intel==2.18.0->tensorflow) (1.6.3)
Requirement already satisfied: flatbuffers>=24.3.25 in c:\users\banke\anaconda3\lib
\site-packages (from tensorflow-intel==2.18.0->tensorflow) (25.1.24)
Requirement already satisfied: gast!=0.5.0,!=0.5.1,!=0.5.2,>=0.2.1 in c:\users\banke
\anaconda3\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow) (0.6.0)
Requirement already satisfied: google-pasta>=0.1.1 in c:\users\banke\anaconda3\lib\s
ite-packages (from tensorflow-intel==2.18.0->tensorflow) (0.2.0)
Requirement already satisfied: libclang>=13.0.0 in c:\users\banke\anaconda3\lib\site
-packages (from tensorflow-intel==2.18.0->tensorflow) (18.1.1)
Requirement already satisfied: opt-einsum>=2.3.2 in c:\users\banke\anaconda3\lib\sit
e-packages (from tensorflow-intel==2.18.0->tensorflow) (3.4.0)
Requirement already satisfied: packaging in c:\users\banke\anaconda3\lib\site-packag
es (from tensorflow-intel==2.18.0->tensorflow) (24.1)
Requirement already satisfied: protobuf!=4.21.0,!=4.21.1,!=4.21.2,!=4.21.3,!=4.21.
4,!=4.21.5,<6.0.0dev,>=3.20.3 in c:\users\banke\anaconda3\lib\site-packages (from te
nsorflow-intel==2.18.0->tensorflow) (4.25.3)
Requirement already satisfied: requests<3,>=2.21.0 in c:\users\banke\anaconda3\lib\s
ite-packages (from tensorflow-intel==2.18.0->tensorflow) (2.32.3)
Requirement already satisfied: setuptools in c:\users\banke\anaconda3\lib\site-packa
ges (from tensorflow-intel==2.18.0->tensorflow) (75.1.0)
Requirement already satisfied: six>=1.12.0 in c:\users\banke\anaconda3\lib\site-pack
ages (from tensorflow-intel==2.18.0->tensorflow) (1.16.0)
Requirement already satisfied: termcolor>=1.1.0 in c:\users\banke\anaconda3\lib\site
-packages (from tensorflow-intel==2.18.0->tensorflow) (2.5.0)
Requirement already satisfied: typing-extensions>=3.6.6 in c:\users\banke\anaconda3
\lib\site-packages (from tensorflow-intel==2.18.0->tensorflow) (4.11.0)
Requirement already satisfied: wrapt>=1.11.0 in c:\users\banke\anaconda3\lib\site-pa
ckages (from tensorflow-intel==2.18.0->tensorflow) (1.14.1)
Requirement already satisfied: grpcio<2.0,>=1.24.3 in c:\users\banke\anaconda3\lib\s
ite-packages (from tensorflow-intel==2.18.0->tensorflow) (1.70.0)
Requirement already satisfied: tensorboard<2.19,>=2.18 in c:\users\banke\anaconda3\l
ib\site-packages (from tensorflow-intel==2.18.0->tensorflow) (2.18.0)
Requirement already satisfied: keras>=3.5.0 in c:\users\banke\anaconda3\lib\site-pac
kages (from tensorflow-intel==2.18.0->tensorflow) (3.8.0)
Requirement already satisfied: numpy<2.1.0,>=1.26.0 in c:\users\banke\anaconda3\lib
\site-packages (from tensorflow-intel==2.18.0->tensorflow) (1.26.4)
Requirement already satisfied: h5py>=3.11.0 in c:\users\banke\anaconda3\lib\site-pac
kages (from tensorflow-intel==2.18.0->tensorflow) (3.11.0)
Requirement already satisfied: ml-dtypes<0.5.0,>=0.4.0 in c:\users\banke\anaconda3\l
ib\site-packages (from tensorflow-intel==2.18.0->tensorflow) (0.4.1)
Requirement already satisfied: wheel<1.0,>=0.23.0 in c:\users\banke\anaconda3\lib\si
te-packages (from astunparse>=1.6.0->tensorflow-intel==2.18.0->tensorflow) (0.44.0)
Requirement already satisfied: rich in c:\users\banke\anaconda3\lib\site-packages (f
rom keras>=3.5.0->tensorflow-intel==2.18.0->tensorflow) (13.7.1)
Requirement already satisfied: namex in c:\users\banke\anaconda3\lib\site-packages
(from keras>=3.5.0->tensorflow-intel==2.18.0->tensorflow) (0.0.8)
Requirement already satisfied: optree in c:\users\banke\anaconda3\lib\site-packages
(from keras>=3.5.0->tensorflow-intel==2.18.0->tensorflow) (0.14.0)
Requirement already satisfied: charset-normalizer<4,>=2 in c:\users\banke\anaconda3
```

```
\lib\site-packages (from requests<3,>=2.21.0->tensorflow-intel==2.18.0->tensorflow)
(3.3.2)
Requirement already satisfied: idna<4,>=2.5 in c:\users\banke\anaconda3\lib\site-pac
kages (from requests<3,>=2.21.0->tensorflow-intel==2.18.0->tensorflow) (3.7)
Requirement already satisfied: urllib3<3,>=1.21.1 in c:\users\banke\anaconda3\lib\si
te-packages (from requests<3,>=2.21.0->tensorflow-intel==2.18.0->tensorflow) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in c:\users\banke\anaconda3\lib\si
te-packages (from requests<3,>=2.21.0->tensorflow-intel==2.18.0->tensorflow) (2024.1
2.14)
Requirement already satisfied: markdown>=2.6.8 in c:\users\banke\anaconda3\lib\site-
packages (from tensorboard<2.19,>=2.18->tensorflow-intel==2.18.0->tensorflow) (3.4.
1)
Requirement already satisfied: tensorboard-data-server<0.8.0,>=0.7.0 in c:\users\ban
ke\anaconda3\lib\site-packages (from tensorboard<2.19,>=2.18->tensorflow-intel==2.1
8.0->tensorflow) (0.7.2)
Requirement already satisfied: werkzeug>=1.0.1 in c:\users\banke\anaconda3\lib\site-
packages (from tensorboard<2.19,>=2.18->tensorflow-intel==2.18.0->tensorflow) (3.0.
3)
Requirement already satisfied: MarkupSafe>=2.1.1 in c:\users\banke\anaconda3\lib\sit
e-packages (from werkzeug>=1.0.1->tensorboard<2.19,>=2.18->tensorflow-intel==2.18.0-
>tensorflow) (2.1.3)
Requirement already satisfied: markdown-it-py>=2.2.0 in c:\users\banke\anaconda3\lib
\site-packages (from rich->keras>=3.5.0->tensorflow-intel==2.18.0->tensorflow) (2.2.
0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in c:\users\banke\anaconda3\l
ib\site-packages (from rich->keras>=3.5.0->tensorflow-intel==2.18.0->tensorflow) (2.
15.1)
Requirement already satisfied: mdurl~=0.1 in c:\users\banke\anaconda3\lib\site-packa
ges (from markdown-it-py>=2.2.0->rich->keras>=3.5.0->tensorflow-intel==2.18.0->tenso
rflow) (0.1.0)
```

Part 2: Creating and Training an AI Model

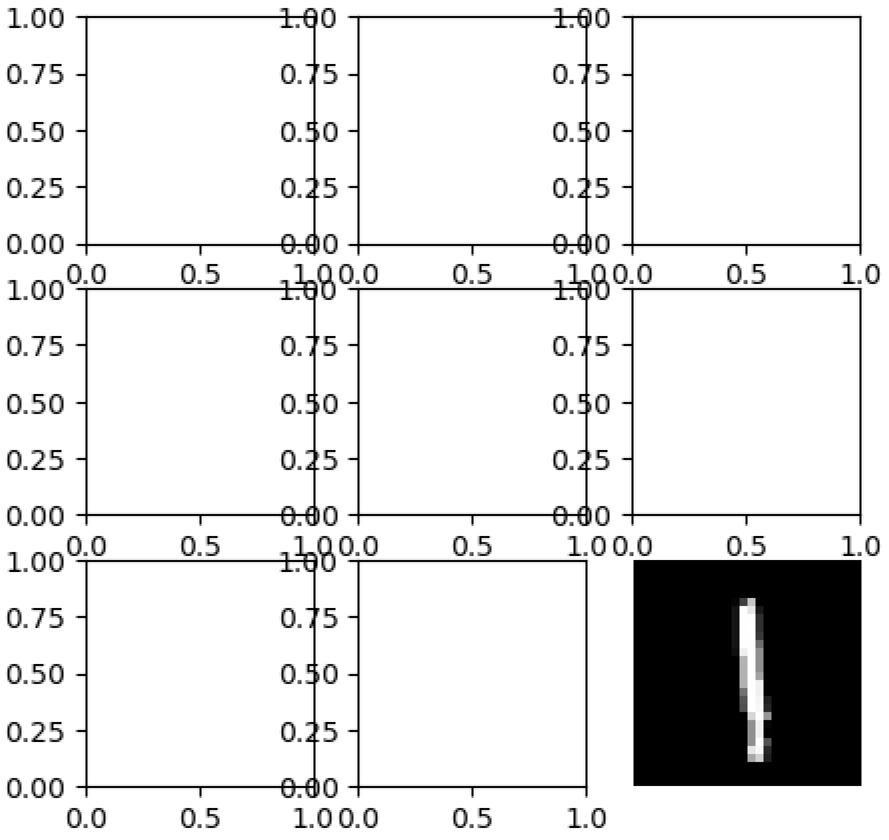
Step 3: Load the MNIST Dataset

In [7]:
```
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras.datasets import mnist
import numpy as np
import matplotlib.pyplot as plt
```

In [8]:
```
# Load MNIST dataset
(x_train, y_train), (x_test, y_test) = mnist.load_data()

# Normalize data (scale pixel values between 0 and 1)
x_train, x_test = x_train / 255.0, x_test / 255.0
```

In [9]:
```
# Show sample images
plt.figure(figsize=(5,5))
for i in range(9): plt.subplot(3,3,i+1)
plt.imshow(x_train[i], cmap="gray")
plt.axis('off')
plt.show()
```



Step 4: Define and Train a Neural Network

Create a simple feedforward neural network using Keras:

In [11]:
```
# Define model architecture
model = tf.keras.models.Sequential([
    tf.keras.layers.Flatten(),  # Input layer
    tf.keras.layers.Dense(128, activation='relu'),  # Hidden layer
    tf.keras.layers.Dense(10, activation='softmax') # Output layer (10 classes)
])
```

In [12]:
```
# Compile model
model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['a
```

In [13]:
```
# Train model
model.fit(x_train, y_train, epochs=5, validation_data=(x_test, y_test))
```

```
Epoch 1/5
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 4s 1ms/step - accuracy: 0.8757 - loss: 0.4398 - val_a
ccuracy: 0.9590 - val_loss: 0.1415
Epoch 2/5
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 3s 2ms/step - accuracy: 0.9642 - loss: 0.1233 - val_a
ccuracy: 0.9699 - val_loss: 0.1016
Epoch 3/5
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 3s 2ms/step - accuracy: 0.9773 - loss: 0.0777 - val_a
ccuracy: 0.9718 - val_loss: 0.0929
Epoch 4/5
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 3s 1ms/step - accuracy: 0.9824 - loss: 0.0576 - val_a
ccuracy: 0.9758 - val_loss: 0.0793
Epoch 5/5
1875/1875 ━━━━━━━━━━━━━━━━━━━━ 3s 1ms/step - accuracy: 0.9873 - loss: 0.0442 - val_a
ccuracy: 0.9745 - val_loss: 0.0826
```

Out[13]:
```
<keras.src.callbacks.history.History at 0x1a6023de720>
```

In [14]:
```python
# Save trained model
model.save("mnist_model.keras", include_optimizer=False)
```

In [15]:
```python
print("Model training complete and saved as mnist_model.keras")
```

```
Model training complete and saved as mnist_model.keras
```

Part 3: Converting and Saving the Model

Step 5: Convert the Model to TensorFlow Lite Format

In [17]:
```python
# Load trained model
model = tf.keras.models.load_model("mnist_model.keras")
```

In [18]:
```python
# Convert to TensorFlow Lite
converter = tf.lite.TFLiteConverter.from_keras_model(model)
tflite_model = converter.convert()
```

```
INFO:tensorflow:Assets written to: C:\Users\banke\AppData\Local\Temp\tmpsqyle83q\ass
ets
```

```
INFO:tensorflow:Assets written to: C:\Users\banke\AppData\Local\Temp\tmpsqyle83q\ass
ets
```

```
Saved artifact at 'C:\Users\banke\AppData\Local\Temp\tmpsqyle83q'. The following end
points are available:

* Endpoint 'serve'
  args_0 (POSITIONAL_ONLY): TensorSpec(shape=(32, 28, 28), dtype=tf.float32, name='i
nput_layer')
Output Type:
  TensorSpec(shape=(32, 10), dtype=tf.float32, name=None)
Captures:
  1812514052944: TensorSpec(shape=(), dtype=tf.resource, name=None)
  1812514047952: TensorSpec(shape=(), dtype=tf.resource, name=None)
  1812514048528: TensorSpec(shape=(), dtype=tf.resource, name=None)
  1812514051600: TensorSpec(shape=(), dtype=tf.resource, name=None)
```

In [19]:
```
# Save the converted model
with open("mnist_model.tflite", "wb") as f:
    f.write(tflite_model)
```

In [20]:
```
print("Model successfully converted and saved as mnist_model.tflite")
```

```
Model successfully converted and saved as mnist_model.tflite
```

Part 4: Loading and Running Inference with TensorFlow Lite

Step 6: Load the Converted Model Using TensorFlow Lite Interpreter

In [22]:
```
# Load TensorFlow Lite model
interpreter = tf.lite.Interpreter(model_path="mnist_model.tflite")
interpreter.allocate_tensors()
```

In [23]:
```
# Get input and output tensor details
input_details = interpreter.get_input_details()
output_details = interpreter.get_output_details()
```

In [24]:
```
print("Input Details:", input_details)
```

```
Input Details: [{'name': 'serving_default_input_layer:0', 'index': 0, 'shape': array
([32, 28, 28]), 'shape_signature': array([32, 28, 28]), 'dtype': <class 'numpy.float
32'>, 'quantization': (0.0, 0), 'quantization_parameters': {'scales': array([], dtyp
e=float32), 'zero_points': array([], dtype=int32), 'quantized_dimension': 0}, 'spars
ity_parameters': {}}]
```

In [25]:
```
print("Output Details:", output_details)
```

```
Output Details: [{'name': 'StatefulPartitionedCall_1:0', 'index': 9, 'shape': array
([32, 10]), 'shape_signature': array([32, 10]), 'dtype': <class 'numpy.float32'>, 'q
uantization': (0.0, 0), 'quantization_parameters': {'scales': array([], dtype=float3
2), 'zero_points': array([], dtype=int32), 'quantized_dimension': 0}, 'sparsity_para
meters': {}}]
```

Step 7: Perform Inference with TensorFlow Lite

In [27]:
```
# Select a test image
test_image = x_test[0].astype(np.float32)
```

In [28]:
```
# Ensure data type matches model input
test_image = np.expand_dims(test_image, axis=0)  # Add batch dimension (1, 28, 28)
test_image = np.tile(test_image, (32, 1, 1))  # (32, 28, 28)
# Convert to float32 (required for TensorFlow Lite)
test_image = test_image.astype(np.float32)
```

In [29]:
```
print("Expected input shape:", input_details[0]['shape'])
```

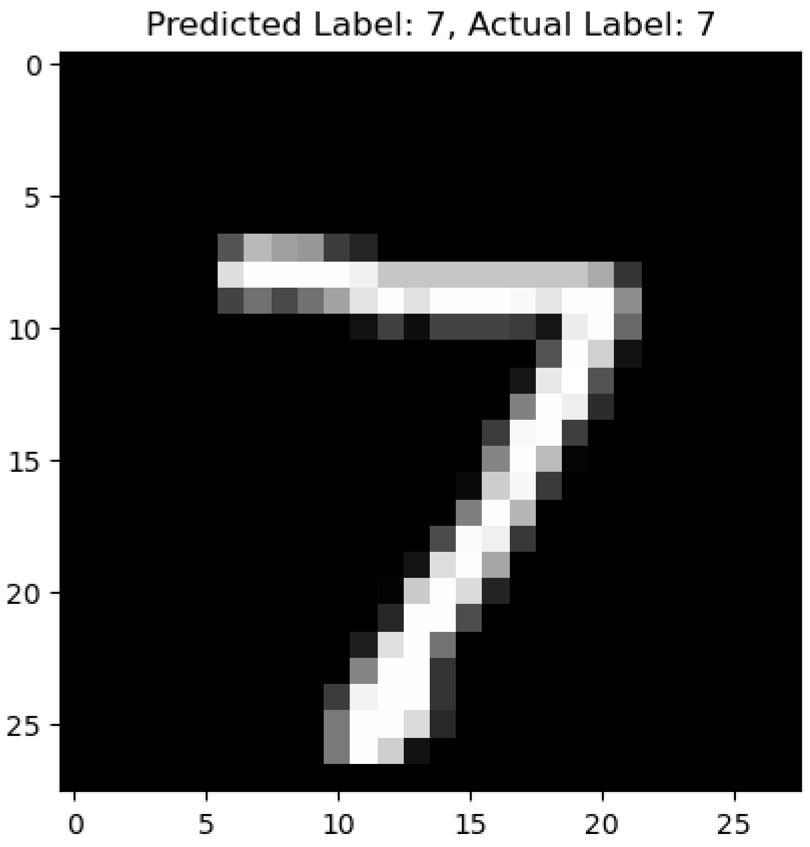
```
Expected input shape: [32 28 28]
```

In [30]:
```
# Set the input tensor
interpreter.set_tensor(input_details[0]['index'], test_image)
```

```
In [31]: # Run inference
         interpreter.invoke()
```

```
In [32]: # Get the prediction
         output_data = interpreter.get_tensor(output_details[0]['index'])
         predicted_label = np.argmax(output_data)
```

```
In [33]: # Display the image and prediction
         plt.imshow(x_test[0], cmap="gray")
         plt.title(f"Predicted Label: {predicted_label}, Actual Label: {y_test[0]}")
         plt.show()
```



```
In [ ]:
```