

```

import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Flatten
from tensorflow.keras.optimizers import Adam
from tensorflow.keras.datasets import mnist

# Load and preprocess data
(x_train, y_train), (x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0 # Normalize pixel values
y_train, y_test = tf.keras.utils.to_categorical(y_train, 10), tf.keras.utils.to_categorical(y_test, 10)

# Define a compact deep learning model
def create_compact_model():
    model = Sequential([
        Flatten(input_shape=(28, 28)),
        Dense(128, activation='relu'),
        Dense(64, activation='relu'),
        Dense(10, activation='softmax')
    ])
    return model

# Compile and train the model
model = create_compact_model()
model.compile(optimizer=Adam(learning_rate=0.001), loss='categorical_crossentropy', metrics=['accuracy'])
model.fit(x_train, y_train, epochs=5, batch_size=32, validation_data=(x_test, y_test))

# Evaluate the model
loss, accuracy = model.evaluate(x_test, y_test, verbose=0)
print(f"Test Accuracy: {accuracy:.2f}")

# Convert the model to TensorFlow Lite format for deployment
converter = tf.lite.TFLiteConverter.from_keras_model(model)
tflite_model = converter.convert()


# Save the TensorFlow Lite model
with open("compact_model.tflite", "wb") as f:
    f.write(tflite_model)

print("Model successfully converted to TensorFlow Lite and saved.")

# Optionally, include quantization for even more efficiency
def convert_with_quantization():
    converter.optimizations = [tf.lite.Optimize.DEFAULT]
    quantized_model = converter.convert()
    with open("compact_model_quantized.tflite", "wb") as f:
        f.write(quantized_model)
    print("Quantized model saved.")

convert_with_quantization()

```

 Downloading data from <https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz>
 11490434/11490434 1s 0us/step
 /usr/local/lib/python3.10/dist-packages/keras/src/layers/reshaping/flatten.py:37: UserWarning: Do not pass an `input_shape`/`input_dim`
 super().__init__(**kwargs)
 Epoch 1/5
 1875/1875 23s 11ms/step - accuracy: 0.8728 - loss: 0.4326 - val_accuracy: 0.9642 - val_loss: 0.1162
 Epoch 2/5
 1875/1875 7s 4ms/step - accuracy: 0.9682 - loss: 0.1039 - val_accuracy: 0.9685 - val_loss: 0.1053
 Epoch 3/5
 1875/1875 11s 4ms/step - accuracy: 0.9786 - loss: 0.0682 - val_accuracy: 0.9728 - val_loss: 0.0876
 Epoch 4/5
 1875/1875 8s 4ms/step - accuracy: 0.9841 - loss: 0.0494 - val_accuracy: 0.9786 - val_loss: 0.0715
 Epoch 5/5
 1875/1875 7s 4ms/step - accuracy: 0.9883 - loss: 0.0363 - val_accuracy: 0.9743 - val_loss: 0.0847
 Test Accuracy: 0.97
 Saved artifact at '/tmp/tmp3izhc5xp'. The following endpoints are available:
 * Endpoint 'serve'
 args_0 (POSITIONAL_ONLY): TensorSpec(shape=(None, 28, 28), dtype=tf.float32, name='keras_tensor')
 Output Type:
 TensorSpec(shape=(None, 10), dtype=tf.float32, name=None)
 Captures:
 136949347782144: TensorSpec(shape=(), dtype=tf.resource, name=None)
 136949347787072: TensorSpec(shape=(), dtype=tf.resource, name=None)
 136949347778800: TensorSpec(shape=(), dtype=tf.resource, name=None)
 136949347788304: TensorSpec(shape=(), dtype=tf.resource, name=None)
 136949347785664: TensorSpec(shape=(), dtype=tf.resource, name=None)

```
136949347789360: TensorSpec(shape=(), dtype=tf.resource, name=None)
Model successfully converted to TensorFlow Lite and saved.
Saved artifact at '/tmp/tmpd_d1bke4'. The following endpoints are available:

* Endpoint 'serve'
  args_0 (POSITIONAL_ONLY): TensorSpec(shape=(None, 28, 28), dtype=tf.float32, name='keras_tensor')
Output Type:
  TensorSpec(shape=(None, 10), dtype=tf.float32, name=None)
Captures:
136949347782144: TensorSpec(shape=(), dtype=tf.resource, name=None)
136949347787072: TensorSpec(shape=(), dtype=tf.resource, name=None)
136949347778800: TensorSpec(shape=(), dtype=tf.resource, name=None)
136949347788304: TensorSpec(shape=(), dtype=tf.resource, name=None)
136949347785664: TensorSpec(shape=(), dtype=tf.resource, name=None)
136949347789360: TensorSpec(shape=(), dtype=tf.resource, name=None)
Quantized model saved.
```