**1. Introduction**

This report documents the process of completing Lab 03 for ITAI 3377, which involves training a Convolutional Neural Network (CNN) on the MNIST dataset, converting the trained model to TensorFlow Lite (TFLite) format, and preparing it for edge device deployment. The lab was executed using a combination of Google Colab for model training (if applicable) and Visual Studio Code (VS Code) for model conversion.

**2. Setup**

**2.1 Environment**

* **Training Environment**:
  + **Option Used:** Google Colab
  + **Details**:
    - **Google Colab**: Used for training due to limited local computational power. Colab provides pre-installed Python (version 3.x) and TensorFlow (version 2.x) with GPU support.
    - **VS Code (Local Machine)**: Used for both training and conversion. Configured with Python 3.10, TensorFlow 2.10, and the Jupyter extension in VS Code.
* **Conversion Environment**:
  + Software: Visual Studio Code (version 1.x)
  + Python Version: 3.10 (in a virtual environment)
  + TensorFlow Version: 2.10.0
  + Project Folder: C:\Users\qpsmith\Desktop\ITAI377\_Lab03

**2.2 Tools and Installation**

* **Python Installation**:
  + Installed Python 3.10 from [python.org](https://www.python.org/downloads/).
  + Added Python to PATH during installation.
* **VS Code Setup**:
  + Installed Visual Studio Code from [code.visualstudio.com](https://code.visualstudio.com/).
  + Installed the Jupyter extension (Microsoft) via the Extensions Marketplace.
* **Virtual Environment**:
  + Created a virtual environment in the project folder:

bash

Copy

python -m venv .venv

.venv\Scripts\activate

* + Upgraded pip:

bash

Copy

python -m pip install --upgrade pip

* + Installed TensorFlow:

bash

Copy

python -m pip install tensorflow

**Log** (example):

text

Copy

Collecting tensorflow

Downloading tensorflow-2.10.0-cp310-cp310-win\_amd64.whl (455.9 MB)

Successfully installed tensorflow-2.10.0

* **Jupyter Notebook**:
  + Created Lab03\_MNIST.ipynb for training (if done locally) and Lab03\_Conversion.ipynb for model conversion in VS Code.

**3. Deployment Process**

**3.1 Model Training**

The model was trained on the MNIST dataset using a CNN architecture. The steps are detailed below.

**3.1.1 Dataset Preparation**

* **Code**:

python

Copy

import tensorflow as tf

from tensorflow.keras.datasets import mnist

*# Load the MNIST dataset*

(x\_train, y\_train), (x\_test, y\_test) = mnist.load\_data()

*# Normalize pixel values*

x\_train, x\_test = x\_train / 255.0, x\_test / 255.0

*# Reshape for CNN input*

x\_train = x\_train.reshape((-1, 28, 28, 1))

x\_test = x\_test.reshape((-1, 28, 28, 1))

print("Training data shape:", x\_train.shape)

print("Test data shape:", x\_test.shape)

* **Output**:

text

Copy

Training data shape: (60000, 28, 28, 1)

Test data shape: (10000, 28, 28, 1)

**3.1.2 Model Definition and Training**

* **Code**:

python

Copy

*# Define CNN model*

model = tf.keras.Sequential([

tf.keras.layers.Conv2D(32, (3, 3), activation='relu', input\_shape=(28, 28, 1)),

tf.keras.layers.MaxPooling2D((2, 2)),

tf.keras.layers.Flatten(),

tf.keras.layers.Dense(128, activation='relu'),

tf.keras.layers.Dense(10, activation='softmax')

])

*# Compile model*

model.compile(optimizer='adam',

loss='sparse\_categorical\_crossentropy',

metrics=['accuracy'])

*# Display model summary*

model.summary()

*# Train model*

history = model.fit(x\_train, y\_train, epochs=5, validation\_data=(x\_test, y\_test))

* **Model Summary** (example):

text

Copy

Model: "sequential"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

conv2d (Conv2D) (None, 26, 26, 32) 320

max\_pooling2d (MaxPooling2D) (None, 13, 13, 32) 0

flatten (Flatten) (None, 5408) 0

dense (Dense) (None, 128) 692352

dense\_1 (Dense) (None, 10) 1290

=================================================================

Total params: 693,962

Trainable params: 693,962

Non-trainable params: 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

* **Training Log** (example):

text

Copy

Epoch 1/5

1875/1875 [==============================] - 10s 5ms/step - loss: 0.1467 - accuracy: 0.9567 - val\_loss: 0.0548 - val\_accuracy: 0.9820

...

Epoch 5/5

1875/1875 [==============================] - 10s 5ms/step - loss: 0.0178 - accuracy: 0.9944 - val\_loss: 0.0468 - val\_accuracy: 0.9860

**Screenshot**: Include a screenshot of the Jupyter Notebook showing the model summary and training output.

**3.1.3 Saving the Model**

* **Code** (if using Colab or local):

python

Copy

model.save('my\_model.h5')

print("Model saved as 'my\_model.h5'")

* **Output**:

text

Copy

Model saved as 'my\_model.h5'

**Note**: If trained in Colab, the my\_model.h5 file was downloaded and transferred to the VS Code project folder (C:\Users\qpsmith\Desktop\ITAI377\_Lab03).

**3.2 Model Conversion to TensorFlow Lite**

The trained model was converted to TFLite format for edge device compatibility.

* **Code**:

python

Copy

*# Load the model (if transferred from Colab)*

from tensorflow.keras.models import load\_model

model = load\_model('my\_model.h5')

print("Model loaded successfully!")

*# Convert to TFLite*

converter = tf.lite.TFLiteConverter.from\_keras\_model(model)

tflite\_model = converter.convert()

*# Save TFLite model*

with open('model.tflite', 'wb') as f:

f.write(tflite\_model)

print("Model has been converted to TFLite and saved as 'model.tflite'")

* **Output**:

text

Copy

Model loaded successfully!

Model has been converted to TFLite and saved as 'model.tflite'

**Screenshot**: Include a screenshot of the VS Code terminal or notebook output confirming the TFLite conversion and a file explorer view showing my\_model.h5 and model.tflite in the project folder.

**3.3 Edge Impulse Preparation**

* The model.tflite file was saved and is ready for evaluation in future labs.
* No Edge Impulse API key setup was required, as per the lab instructions.
* Reference: Watched the Edge Impulse video tutorial in the Canvas Media Gallery for context.

**4. Testing and Validation**

**4.1 Model Evaluation**

The trained model was evaluated on the MNIST test dataset.

* **Code**:

python

Copy

*# Evaluate the model*

test\_loss, test\_accuracy = model.evaluate(x\_test, y\_test)

print(f"Test accuracy: {test\_accuracy:.4f}")

* **Output** (example):

text

Copy

313/313 [==============================] - 1s 3ms/step - loss: 0.0468 - accuracy: 0.9860

Test accuracy: 0.9860

**4.2 Training and Validation Metrics**

* The model was trained for 5 epochs, with training and validation accuracy monitored.
* **Sample Metrics** (from training log):
  + Epoch 1: Training Accuracy = 95.67%, Validation Accuracy = 98.20%
  + Epoch 5: Training Accuracy = 99.44%, Validation Accuracy = 98.60%
* **Optional Visualization** (if included):

python

Copy

import matplotlib.pyplot as plt

plt.plot(history.history['accuracy'], label='Training Accuracy')

plt.plot(history.history['val\_accuracy'], label='Validation Accuracy')

plt.xlabel('Epoch')

plt.ylabel('Accuracy')

plt.legend()

plt.show()

**Screenshot**: Include a screenshot of the accuracy plot (if generated) or the evaluation output.

**4.3 TFLite Model Verification**

* Verified the TFLite model by loading it:

python

Copy

import tensorflow as tf

interpreter = tf.lite.Interpreter(model\_path='model.tflite')

interpreter.allocate\_tensors()

print("TFLite model loaded successfully!")

* **Output**:

text

Copy

TFLite model loaded successfully!

**5. Challenges and Resolutions**

* **Challenge**: Encountered an error installing TensorFlow locally (ERROR: Could not find a version that satisfies the requirement tensorflow).
  + **Resolution**: Created a fresh virtual environment, ensured Python 3.10 was used, and installed TensorFlow 2.10.0. Alternatively, used Google Colab for training to bypass local installation issues.
* **Challenge**: Slow training on local machine.
  + **Resolution**: Switched to Google Colab with GPU runtime for faster training.
* **Challenge**: Model loading error in VS Code.
  + **Resolution**: Ensured my\_model.h5 was in the correct project folder and TensorFlow was installed in the virtual environment.

**Log Example** (TensorFlow installation):

text

Copy

(.venv) PS C:\Users\qpsmith\Desktop\ITAI377\_Lab03> python -m pip install tensorflow

Collecting tensorflow

Downloading tensorflow-2.10.0-cp310-cp310-win\_amd64.whl (455.9 MB)

Successfully installed tensorflow-2.10.0

**6. Conclusion**

Lab 03 successfully demonstrated the process of training a CNN on the MNIST dataset, achieving a test accuracy of approximately 98.60%. The model was converted to TFLite format and saved as model.tflite, ready for future edge deployment. The use of Google Colab (if applicable) and VS Code streamlined the workflow, despite initial setup challenges. This lab reinforced key concepts in deep learning and edge computing.