Oren Moreno

ITAI 3377

February 3, 2025

Professor Patricia McManus

L02 Deploying an AI Model on a Simulated Edge Device

I have elected to complete the Practical Option (Option B) for this lab.

**Conceptual Report:**

A screenshot of a computer

Description automatically generated

This step verifies that the correct version of Python and TensorFlow are installed and imports the libraries necessary for the lab. We can see that we are running Python 3.11.11 and TensorFlow 2.18.0.

A screenshot of a computer

Description automatically generated

Here we define the train and test split variables and load the dataset. Then we make sure that the images are all the same size by normalizing them. After normalization we output 9 sample images from the dataset.

A white rectangular object with colorful text

Description automatically generated

In this step we define the model’s architecture. In this case, we use a model with a Flatten layer that converts 28x28 images into a 1D array, a Hidden layer with 128 ReLU activated neurons, and a 10 softmax neuron output layer.

A screenshot of a computer

Description automatically generated

Next, we compile the model using the “adam” training optimizer, a sparse categorical crossentropy loss function, and track accuracy during training. We then define our training run to consist of 5 epochs and specify the training and validation data.

A screenshot of a computer

Description automatically generated

Once training is complete, we save the model as “mnist\_model.h5.”

A screenshot of a computer

Description automatically generated

In this step, we load the model to the “model” variable and create the object that will convert the model into a TFLite model. Once the converter object is defined, the conversion is carried out to convert the model into a lightweight TFLite model format. After conversion, the model is saved as “mnist\_model.tflite.”

A screenshot of a computer

Description automatically generated

We then load the TFLite model and use the interpreter to display the details of the model’s expected inputs and outputs. From this code’s output, we can see that the model expects 1 28x28 image as input and outputs a [1,10] array which will be the probabilities for digits 0-9. After this, we run a test image through the model.

A screenshot of a computer

Description automatically generated

Finally, we get the model’s prediction probabilities and output its top prediction along with an image and the correct label of the image. We can see that the model correctly identified the digit.

**Reflective Journal:**

Introduction:

The implementation of neural networks and deployment of machine learning models have evolved significantly, driven by the increasing demands of mobile applications and edge computing devices. Traditional machine learning workflows often present challenges in transitioning from development to deployment, while the complexity of model optimization poses difficulties for efficient implementation. The objective of this reflection is to examine the challenges encountered during the TensorFlow implementation assignment, the knowledge gained through the process, and the practical applications of TensorFlow Lite in modern computing scenarios.

Discussion:

The primary challenges encountered during this implementation centered on code formatting and conceptual understanding of neural network architecture. The provided code required careful attention to Python syntax requirements, particularly in areas of proper indentation and sequential layer definition. The process of addressing these challenges revealed the importance of precise code structure. The assignment's technical components presented several learning opportunities through their practical application. The normalization of input data through pixel value scaling (x\_train / 255.0) demonstrated the critical nature of data preprocessing in neural network training. Similarly, the implementation of the model architecture, comprising Flatten, Dense, and output layers, provided practical insight into neural network design principles. The model's structure efficiently transformed 2D image data into processable formats while maintaining the capability to identify complex patterns necessary for digit recognition.

In the context of real-world AI deployment, TensorFlow Lite demonstrates particular significance through its optimization capabilities. The conversion process from the original TensorFlow model to TensorFlow Lite format illustrates the practical considerations necessary for deploying machine learning models in resource-constrained environments. This optimization proves especially valuable in mobile computing scenarios, where processing power and memory availability present consistent constraints. The implementation of TensorFlow Lite enables several key applications in modern computing environments. In mobile devices, the optimized model format facilitates real-time processing capabilities for applications such as text recognition and image classification. IoT deployments benefit from TensorFlow Lite's efficient resource utilization, enabling local processing of sensor data and real-time decision making without constant cloud connectivity. Additionally, edge computing implementations leverage TensorFlow Lite to reduce latency and enhance privacy through local data processing, particularly valuable in scenarios requiring immediate response times or handling sensitive information.

Conclusion:

This implementation assignment provided valuable insights into both the technical challenges and practical applications of machine learning model deployment. The experience highlighted the importance of proper code structure and understanding of neural network architectures while demonstrating the practical significance of model optimization for real-world applications. TensorFlow Lite's role in enabling efficient model deployment across various computing scenarios underscores its importance in modern AI implementations. These findings emphasize the critical nature of understanding both development and deployment considerations in machine learning applications.