Document for Conceptual Design

Eduardo Cabrera-Lopez

ITAI 3377

Lab: L02 - TensorFlow Lite Deployment

Method Selected: Conceptual

First part: conceptual arrangement of the development environment

Python Set-up

To set Python:

Get the newest edition by visiting https://www.python.org.

Run the installation then check "Add Python to PATH".

Click ‘Install Now' to complete setup.

4. Check installation to ensure Python --version TensorFlow and TensorFlow Lite Installations

Install TensorFlow using pip.

TensorFlow Lite finds place within TensorFlow. Regarding stationary:

pip install tflite- runtime

Jupyter Notes book Installement

Install using pip to a notebook.

To start: Jupyter notebook

Part 2: Training of an AI Model Architectural Conceptual Development

Keras allows a basic neural network:

Input form: (28, 28).

Layers: Dense (10, Softmax), Dense (128, ReLU), Flattening

Data loading and preprocessing

Load using Keras: mnist.load_data() returns x_train, y_train and x_test, y_test.

x_train, x_test = x_train / 255.0, x_test / 255.0

Model Building and Instruction

Adam was the optimizer.

The loss is sparse categorical crossentropy.

Measurements: accuracy

Epochs: five

Get and train using: model.compile(...) model. fit ()

Third part: conceptual conversion and model saving

Conversion's Goals

Change the model to run effectively on edge hardware.

Actions to Convert and Save

Load model: tf.keras.models.load_model('mnist_model.h5').

From a keras model, converter = tf.lite.TFLiteConverter; tflite_model = converter.convert().

3. Save open ("mnist_model.tflite," "wb") as f: write (model tflite)

Part 4: TensorFlow Lite Conceptual Application of the Model Deployment Methodology Interpretor: tf.lite. Interpreter with model path "mnist_model.tflite"; allocate tensors here

Get specific information:

outputs = interpreter.get_output_details; inputs = interpreter.get_input_details

Valuating the Model

One should get ready for input: enlarge and translate test images

2. Define tensor input.

3. Call for interpreter

4. Get and understand tensor prediction from output data: np.argmax