The University of Texas at Austin
Chandra Department of Electrical and Computer Engineering
Cockrell School of Engineering

Spring 2023

ECE 361E: Machine Learning and Data Analytics for Edge AI
HW4 - In depth explanations

All plots in this homework need to have: a title, labels on both axes, a grid and a legend with corresponding labels for each line in the plot. Use ***matplotlib.pyplot*** to draw figures and save your figures as PNG files. Save all data for the plots in a CSV file so there is no need to re-train the models in case of an error. The training code for this homework should run on GPUs from Maverick2; the inference should run on the edge devices. You **do *not* need** to change the loss, the optimizer, the batch size or the number of training epochs. You **need** to change the learning rate as instructed in the question.

For better readability, for all numbers that are over 999 in value, put comma signs, e.g., 1,800,903 instead of 1800903. For time measurements, use the **time.time()** function.

Environment Setup

For TensorFlow and TFLite on TACC

1. Load necessary modules:

   module load python3/3.7.0 cuda/11.0 cudnn/8.0.5

   Of note, these modules are different from the ones required for running PyTorch on Maverick2.

2. Create and activate a new Python virtual environment.
3. Install the necessary Python packages:

   pip install -r requirements.txt

   You can find the *requirements.txt* under the folder *HW4_files*

For TFLite on your own machine:

1. Install TFLite runtime with

   pip install tflite-runtime

Problem 1: TensorFlow Evaluation

General remark: In the first part of this assignment, you need to experiment with TensorFlow implementations of the VGG11 and VGG16 networks using the CIFAR10 dataset. **The models you create in this homework will actually be Keras models**, but since Keras is an integrated library in TensorFlow, we can also refer to these models as being TensorFlow models.

Question 2

Use the **Appendix A1.1** to set up the Maverick2 environment, and **Appendix A3.1** to train both models in parallel, using the same Maverick2 instance. Save the trained models. Use the ADAM optimizer with a **learning rate = 0.0001** (different from HW3!) to train these models.

Question 3

The ***fit*** method will return its output into a variable called ***history***. To find the loss and accuracy, use ***history.history***, which is a dictionary containing 4 important keys: *loss*, *accuracy, val_loss* and *val_accuracy*. Each key contains a list for loss and accuracy values for each training epoch.

Use ***model.evaluate()*** to evaluate the model on the testing dataset and obtain the loss and the accuracy of the model (if you mentioned as metric for the ***compile()*** method only ***'accuracy'***). Use

model.save_weights() to save the model weights in the ***.ckpt*** format. As a parameter, give a string: the name of the model with the corresponding ***_tf*** suffix followed by the ***.ckpt*** extension. To load the model later on, define the model instance and then use ***model.load_weights()*** with the same parameter as before.

Problem 2: Deployment on the Edge using TensorFlow Lite

In this part of the homework, you need to deploy the VGG11 and VGG16 models on Odroid MC1 and Raspberry Pi 3B+ using the TFLite framework. Use info in **appendices A2.1 and A3.3** to connect to each device.

Question 1

The first step is to instantialize a converter ***tf.lite.TFLiteConverter*** with the ***from_keras_model()*** method which takes the Keras model (with its corresponding weights already loaded) as input. The next step is to obtain the TFLite model using the ***convert()*** method from the previously obtained converter.

Question 2

The memory for RaspberryPi 3B+ and Odroid MC1 processors is the amount of RAM memory consumed when doing inference. Use ***Appendix A3.5*** to check the amount of RAM memory needed before running inference, and write it down. After you start doing inference, check that RAM memory again and write the difference between the current value and the previous one in ***Table 2***. The actual inference happens when ***interpreter.invoke()*** is being executed.

Question 3

The power measurements for both RaspberryPi 3B+ and Odroid MC1 devices need to be done as in **HW2**. For temperature, check **Appendix A3.4** for how to collect temperature measurements from RaspberryPi. Because you have multiple temperature sensors for the Odroid MC1 processor, the temperature you need to use throughout this homework is the average value of the 4 big core temperatures.

Problem 3: ONNX vs TensorFlow Lite

In this problem you need to use MobileNet-v1 (a model specifically designed and optimized for edge devices) to deploy it on both Raspberry Pi 3B+ and Odroid MC1 processors. You need to explore what it means in terms of latency, energy consumption and accuracy performance to use a model specifically designed for edge devices using ***TFLite***. Finally, you need to compare the entire workflow (i.e., from training until the deployment phase) using ***ONNX*** and ***TFLite*** and consider your favorite one.