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

Spring 2026

ECE 361E: Machine Learning and Data Analytics for Edge AI
HW3 - In depth explanations

IMPORTANT! Do not use VSCode/PyCharm or any other IDE to connect to the edge devices in this class. They consume a lot of memory and impact everything that runs on the devices. Only use a plain terminal to connect to the devices.

In this homework you should first *train your models exclusively* using the GPUs of Lonestar6, and then perform inference on the edge devices. **Use only the A100-Small nodes on Lonestar6.** Of note, you do ***not*** need to change the loss, the optimizer, the batch size, the number of training epochs or the learning rate when training. For time measurements, use the **time.time()** function.

IMPORTANT! All plots (in all homeworks) must 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 in case of an error or system malfunctioning. For the entire HW3 use a sampling period of 200 milliseconds [ms] for power and temperature readings.

IMPORTANT! As a general rule for measuring power and temperature on edge devices, leave at least 2 minutes between running the benchmarks to allow the RaspberryPi 3B+ and Odroid MC1 thermals and loads to reach the idle steady-state. Of note, avoid using *print* statements in the measurement file since they may cause a small delay in the measurements/inference code that compounds over time.

For better readability, for all numbers that are over 999 in value, put comma signs, e.g.,1,800,903 instead of 1800903. Similarly, use 0.XXXX for decimal values (i.e., do *not* use more than 4 decimal digits) for any results written in tables, e.g., use 0.9384 instead of 0.938411291111.

IMPORTANT! Everything you need is already provided to you on the devices. You are *not* allowed to install anything on the edge devices. By extension, you are not allowed to make any modification through software to the edge devices unless we mention it explicitly in the homework. When solving the homework, create your own directory and do not modify the *HW2_files*, *HW3_files* or *HW4_files* folders and their contents.

Problem 1: PyTorch Evaluation

In this problem, you will compare the training accuracy, memory consumption and training time of a medium size model (VGG11) and a large size model (VGG16) using the CIFAR10 dataset.

Question 1

Compared to the VGG models (VGG11 and VGG16) for the ImageNet dataset presented in ***Lecture 9***, there are two differences in its implementation for the CIFAR10 dataset: instead of 4096 neurons per fully connected layer, you now have 512, and instead of 1,000 classes you have now only 10 classes.

Question 2

Use the **Appendix A1.1** to set up the Lonestar6 environment and the **Appendix A3.1** to train both models in parallel, on a single Lonestar6 computation node.

For reporting FLOPs, make sure to use **M for millions**. For example, if you have 123,405,890 FLOPs write in the table **123.4M**.

Problem 2: Deployment on Edge Devices Using ONNX

In this part of the homework, you will deploy the VGG11 and VGG16 models on real edge devices (i.e., Odroid MC1 and Raspberry Pi 3B+) using the ONNX framework. You need to compare how these two models perform when doing inference on edge devices. Use **Appendices A2.1** and **A3.3** to connect to each device properly.

Question 1

You can either convert the PyTorch models to ONNX on TACC or on your local machine, but not on the edge devices. If you do it on your local machine, make sure you follow the instructions in **Appendix A3.2** to install ONNX locally on your machine before you start working on this problem.

The first three parameters for the convert function are: the model itself, the random input tensor with the same input size as the model, and the file name to export to.

For Odroid MC1, use:

```
torch.onnx.export(your_model, random_input_tensor,
                  model_name.onnx, export_params=True, opset_version=13)
```

For RaspberryPi 3B+, use:

```
torch.onnx.export(your_model, random_input_tensor,
                  model_name.onnx, export_params=True, opset_version=16)
```

Question 2

You need to send all ONNX models (obtained in **Problem 2, Question 1**) and the ***deploy_onnx.py*** file to each device over SSH using the ***scp*** command (similar to **HW2**). **The testing accuracy you obtain when deploying the models** on the edge devices must be the *same* testing accuracy you obtained in **Problem 1, Question 2** for the VGG11 and VGG16 models, respectively.

The RAM memory consumption for RaspberryPi 3B+ and Odroid MC1 is the amount of RAM memory consumed when doing inference. Using the info in the **Appendix A3.5,** check the amount of RAM memory ***before running inference***, and write it down (this is the idle memory consumption). ***After you start doing inference***, check the memory usage value again (this is the inference memory consumption + idle memory consumption) and write the difference between the current value and the previous one in ***Table 2*** (i.e., how much memory the inference process consumes compared to the idle state).

NOTE: The inference is when ***sess.run()*** is being executed, *not* when considering all the image loading and preprocessing times.

For the Odroid only, use **`taskset`** to do inference only on the 4 *big cores*:

```
taskset -all-tasks 0xF0 python [script]
```

IMPORTANT! For both devices, use `htop` to make sure that all 4 cores running inference are near 100% utilization.

Question 3

For both RaspberryPi 3B+ and Odroid MC1 devices, the power measurement process is identical (since both RaspberryPi and Odroid are connected to the Smart Power 2 device) and needs to be done as in **HW2**. Check **Appendix A3.4** to see how to collect temperature measurements for the RaspberryPi device.

Because you have multiple temperature sensors for the Odroid MC1, the temperature you need to use throughout this homework will have to be the *average* of all four big core temperatures.

NOTE: Besides the above note related to inference time, when computing the energy consumption, we consider the time it takes for ***the entire script*** to execute. This is like when you measured the energy consumption in HW2, i.e., you considered the energy consumed for a complete run of a benchmark. The benchmark in this case is the entire inference over 10,000 images.

Problem 3: MobileNet-v1 on Edge Devices

In this problem you will deploy the MobileNet-v1 model on Raspberry Pi 3B+ and Odroid MC1 devices. As discussed in **Lecture 9**, the MobileNet-v1 model is specifically designed and optimized for edge devices, so this architecture is very relevant to edge applications.

You need to explore (in terms of training and inference latency, energy consumption and accuracy performance) what it means to have a model that is optimized for edge devices, as opposed to more general-purpose models such as VGG11 and VGG16.