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
Appendix A3

A3.1. Running Tasks in Parallel on TACC

Running a single Python program on a computing node with 4 GPUs is a waste of resources. As this class focuses on efficiency, we show you how to run commands in parallel, each of them on a different GPU, for better resource utilization.

1) To run in parallel, in the ***config.slurm*** file provided in ***HW3_files*** place the two commands you want to run in parallel one after the other.
2) Add **"&"** at the end of each command; this will make the processes run in the background.
3) After inserting all commands, write **"wait"**. If the processes are running in the background, you want to wait until all of them finish running, and not kill the entire script (with all its running processes) when the first command from the background finishes running.
4) Example:

```
python main.py --option=1 &
python main.py --option=2 &
wait
```

A3.2. Installing ONNX for Linux on your personal computer

1) The following steps are for using Conda on your computer. Python 3.9 is highly recommended since this is the version used by the edge devices.

   -
     ```
     $ sudo apt-get update
     ```

   -
     ```
     $ sudo apt-get install -y python3 python3-dev python3-setuptools gcc libtinfo-dev zlib1g-dev build-essential cmake libedit-dev libxml2-dev
     ```

   -
     ```
     $ conda create --name ml python=3.9
     ```

   -
     ```
     $ conda activate ml
     ```

2) Make sure you have vim installed or run `$ sudo apt install vim`
3) While inside the virtual environment created in step 1), execute the following.

   -
     ```
     $ pip install onnx onnxruntime==1.16.3
     ```

NOTE: This is the version of `onnxruntime` already installed on the edge devices. You must have the same version on your laptop otherwise you will encounter errors during deployment.

A3.3. Connecting to the Devices

1) For RaspberryPi : **`ssh student@sld-rpi-<your RPi number>.ece.utexas.edu`**
2) For Odroid MC1: **`ssh student@sld-mc1-<your MC1 number>.ece.utexas.edu`**

A3.4. Monitoring Power and Temperature on RaspberryPi 3B+ and MC1

1) For Odroid MC1, you already have measured temperatures in **HW2**. For Raspberry Pi 3B+ you need to measure temperature as follows:

```
import gpiozero
cpu_temp = gpiozero.CPUTemperature().temperature
```

2) To measure the power for Raspberry Pi 3B+ use the same method as for Odroid MC1 in **HW2**.

- Keep in mind that the **`nmcli`** command is only for the Odroid and you need to verify the telnet connection on the Raspberry Pi by running **`telnet 192.168.4.1`**
- In the rare case the connection is not OK for Raspberry Pi, please let us know as soon as possible.

A3.5. Monitoring Memory Consumption on Edge Devices

For both RaspberryPi 3B+ and Odroid MC1, to get the memory consumption you need to run the following command:

```
free -m
```

which gives you the total, used and free RAM memory in [MB]. The memory consumption stabilizes during the inference process so you can see exactly how much memory the inference consumes compared to the idle state of the device. To see the memory consumption refreshing every 0.1 seconds use:

```
watch -n0.1 free -m
```

Another alternative is to use `htop` command to monitor both the cores on which the model is running inference and also the memory.