**Homework 1 Student Name:** Tanmay Samak

AuE 8930: Computing and Simulation for Autonomy

Instructor: Prof. Bing Li, Clemson University, Department of Automotive Engineering

\* Refer to [Syllabus](https://tinyurl.com/computing-autonomy) for homework (late) submission, grading and plagiarism policies;

\* Submission due Mon. 9/11/2023 11:59 pm via Canvas, include:

* This document (with answers), and with your program results/visualization;
* A .zip file of (modified) source code and data if any, which the TA might run.

**Question 1**

Training a Pytorch deep learning model on Palmetto cluster (60 points)

(Recommended to use Jupyter Notebook in Palmetto [OpenOnDemand](https://openod.palmetto.clemson.edu/) for edit/debug/run)

**Palmetto Cluster and Setup**

* Login into your Palmetto account & request a node with required specifications by specifying a hardware resource configuration, making sure to include GPU. (For below all questions, make sure to use same configuration).
* Transfer the sample code into your account using Globus (if using Terminal) or JupterHub.

**Create a Conda virtual environment in the terminal**

*module add anaconda3/2022.05-gcc/9.5.0*

A conda virtual environment allows you to run/install a version of Python and package as needed within it.

This environment, once created/modified is saved and can be accessed later through the code:

*conda create -n NAME\_OF\_ENV python=3.6 # (Create Environment)*

*source activate NAME\_OF\_ENV # (Activate Environment)*

*source deactivate NAME\_OF\_ENV # (Deactivate Environment)*

**Install necessary packages in the terminal**

Add cuda and cudnn module:

*module add cuda/11.1.1-gcc/9.5.0*

*module add cudnn/8.0.5.39-11.1-gcc/9.5.0-cu11\_1*

Install Pytorch and Torchvision libraries using conda ([reference](https://pytorch.org/get-started/locally/))

*conda install pytorch torchvision torchaudio cudatoolkit=11.1 -c pytorch-lts -c nvidia*

**Generate Kernel for JupyterHub**

(Attention: if you install those modules under a certain conda environment)

You may encounter this error when running the base.ipynb in Jupyter Hub:

"no module named torch"

It means your Jupyter notebook is running in the default python environment, but your torch module is installed in your Conda virtual environment. You will need to run Jupyter notebook in your virtual env.

Here is a tutorial: <https://janakiev.com/blog/jupyter-virtual-envs/>

**Training deep learning model for Image Classification**

Sample code is in Canvas/Files can be downloaded from: [Homework\_1\_sample\_code.zip](https://clemson-my.sharepoint.com/:u:/g/personal/bli4_clemson_edu/EYxlIdnUZU5KrcKe5qJSvYoB8LWCaijiVT1dV4pc2Y3j9g?e=EulwYv)

which includes: base.ipynb, common.py and models.py. The base.ipynb allows you to use your web browser as the GUI to run/edit/debug.

You also need to make ‘data’ and ‘models’ folder before running the ‘base.ipynb’. The directory structure should look like:

Table

Description automatically generated with low confidence

There are multiple steps in the sample code files:

* Load the training and test datasets from torchvision ([reference](https://pytorch.org/vision/stable/datasets.html))  
  Training Data can be obtained from various online sources, self-procured or can even be imported from a library like Pytorch.
* Define a Convolutional Neural Network ([reference](https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148))
* Define a loss function ([reference](https://blog.algorithmia.com/introduction-to-loss-functions/))
* Train the network on the training data with different number of Epochs ([reference](https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9)).

**(1) Show screenshots of successful installation and procedure of the setup. (15 points)**

A screenshot of a computer

Description automatically generated

* 1. Open MobaXterm application (Windows) and SSH into Palmetto Cluster.

A screenshot of a computer

Description automatically generated

* 1. When prompted, enter the mode of authentication (Duo Push/SMS).

A screenshot of a computer program

Description automatically generated

* 1. Login with your credentials (e.g., verify the Duo Push).

A screenshot of a computer program

Description automatically generated

* 1. Clone the GitHub Repository of course to the ~HOME directory on Palmetto Cluster.

A screenshot of a computer

Description automatically generated

* 1. Create a new directory for HW1 and change directory to that location.

A screenshot of a computer

Description automatically generated

* 1. Move (drag-and-drop) new files for HW1 to `HW1/Tanmay` directory on Palmetto Cluster.

A screenshot of a computer

Description automatically generated

* 1. Confirm uploaded files and folders via terminal.

A screenshot of a computer screen

Description automatically generated

* 1. Refer `/etc/hardware-table` for hardware specification of the available compute resources.

A screenshot of a computer

Description automatically generated

* 1. Submit a new interactive job called `aue8930\_hw1` with X11 port forwarding, which has the given hardware specifications.

A screenshot of a computer

Description automatically generated

1.10 Add Anaconda module.

A screenshot of a computer screen

Description automatically generated

1.11 Create Conda virtual environment called `aue8930`.

A screenshot of a computer

Description automatically generated

1.12 Add CUDA and CUDNN modules.

A screen shot of a computer

Description automatically generated

1.13 Activate Conda virtual environment and install `pytorch`.

A screen shot of a computer screen

Description automatically generated

1.14 Install `ipykernel` (required to add the virtual environment to Jupyter).

A screen shot of a computer screen

Description automatically generated

1.15 Add Conda virtual environment to Jupyter.

A screenshot of a computer

Description automatically generated

1.16 Verify Jupyter kernels.

A screenshot of a computer program

Description automatically generated

1.17 Change directory to `HW1/Tanmay` and launch JupyterLab.

A screenshot of a computer

Description automatically generated

1.18 An interactive browser (e.g., Firefox) window should pop-up (from the Palmetto Node).

A screenshot of a computer

Description automatically generated

1.19 Open the IPYNB and change the kernel.

**(2) Run the existing sample code “base.ipynb” (5 points)**

**During the training, what’s your GPU usage percentage? (You can open another terminal and use “nvidia-smi –l” to monitor the usage info of GPU and GPU memory.)**

A screenshot of a computer

Description automatically generated

2.1 A100 GPU usage (~23%) and GPU RAM usage (~2.4%) for baseline implementation.

A screenshot of a computer

Description automatically generated

2.2 Accuracy (train=76.43%; val=77.69%; test=78.62%) for baseline implementation.

**(3) Modify the code for better performance (change the batch size) (10 points)**

**During the training, what’s your GPU usage percentage?**

A screenshot of a computer

Description automatically generated

3.1 Duplicate the baseline IPYNB for batch-size experiment.

A screenshot of a computer

Description automatically generated

3.2 Change batch size and model name for saving.

A screenshot of a computer

Description automatically generated

3.3 A100 GPU usage (~25%) and GPU RAM usage (~4.1%) for implementation with increased batch size.

A screenshot of a computer

Description automatically generated

3.4 Accuracy (train=78.51%; val=77.99%; test=78.05%) with increased batch size.

**(4) Modify the code for better performance (use two GPUs) (10 points)**

**During the training, what’s your GPU info percentage? (TIPS:** [**reference API**](https://pytorch.org/tutorials/beginner/former_torchies/parallelism_tutorial.html)**)**

A screenshot of a computer

Description automatically generated

4.1 Submit a dual-GPU interactive job with X11 port forwarding.

A screenshot of a computer

Description automatically generated

4.2 Add Anaconda, CUDA and CUDNN modules.

A screenshot of a computer

Description automatically generated

4.3 Activate Conda virtual environment and install `matplotlib`

A screenshot of a computer program

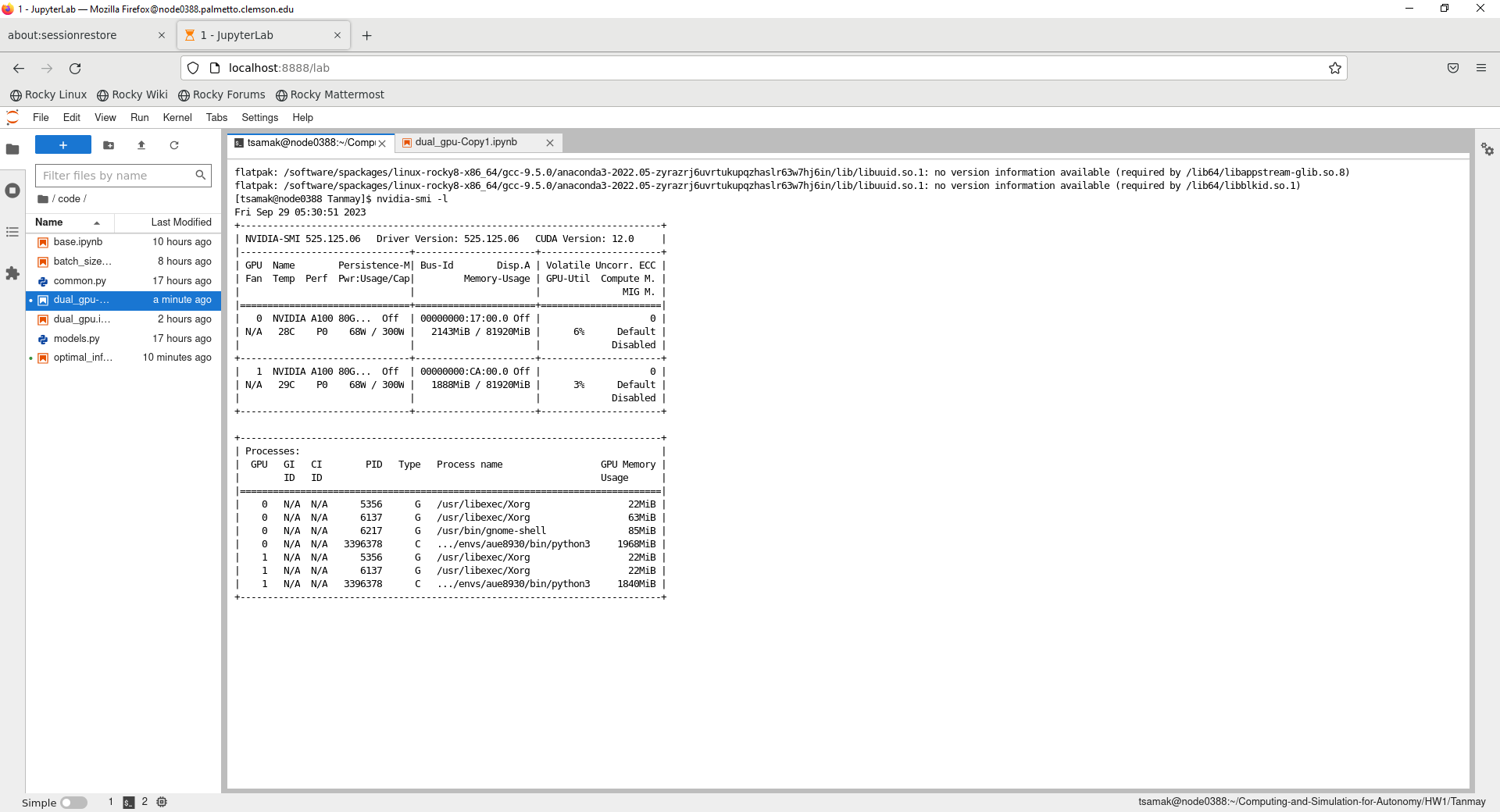
Description automatically generated

4.4 Change directory to `HW1/Tanmay` and launch JupyterLab.

A screenshot of a computer program

Description automatically generated

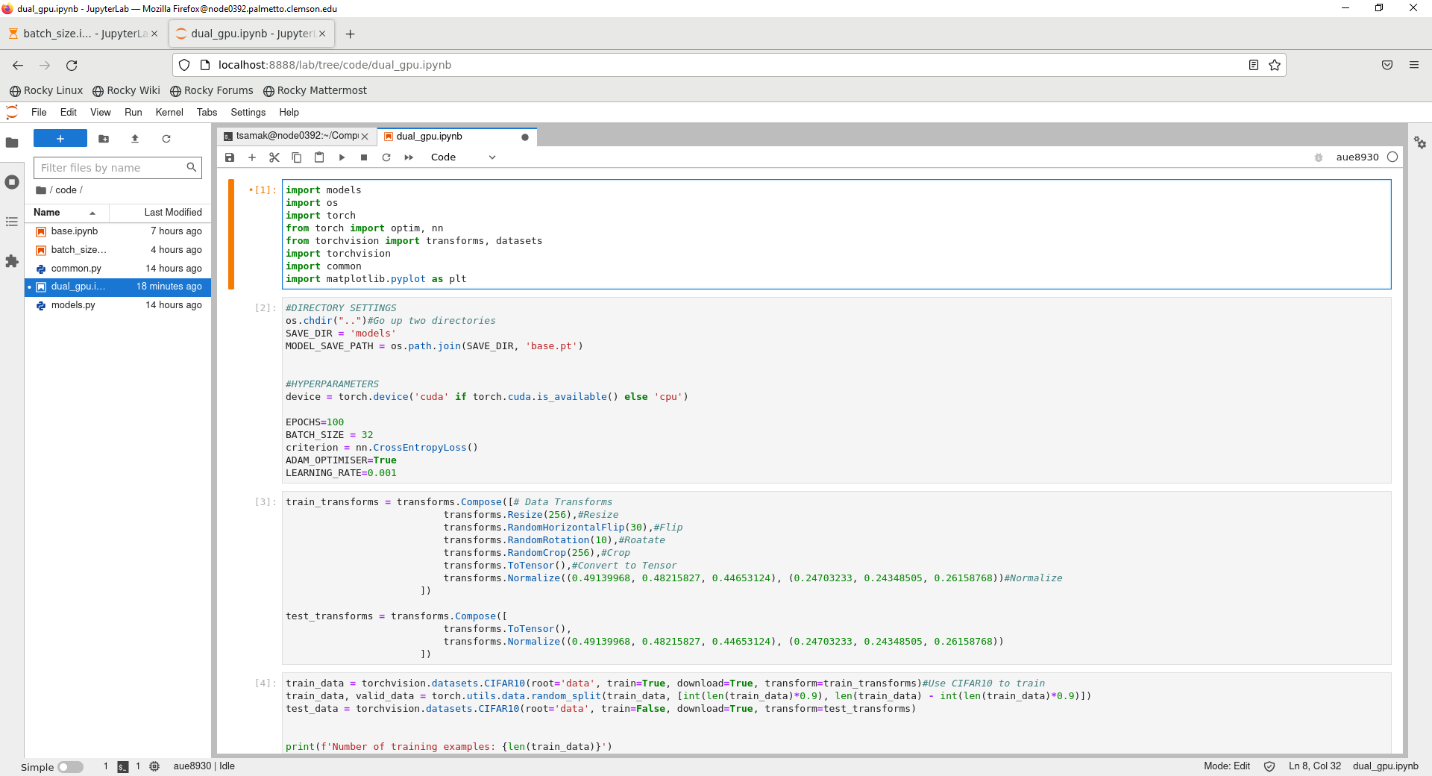
4.5 Modify the model section to allow data parallelism.



4.6 2xA100 GPU usage (GPU0=~6%; GPU1=~3%) and GPU RAM usage (GPU0=~2.6%; GPU1=~2.3%) for implementation with dual-GPU configuration.

**(5) Plot the accuracy against the number of training Epochs on a Graph. (10 points)**

**(TIPS: you need to import matplotlib, modify the code of “for epoch in range (EPOCHS):” by saving the “epoch” and “train\_acc”, and plot its relationship in the end)**

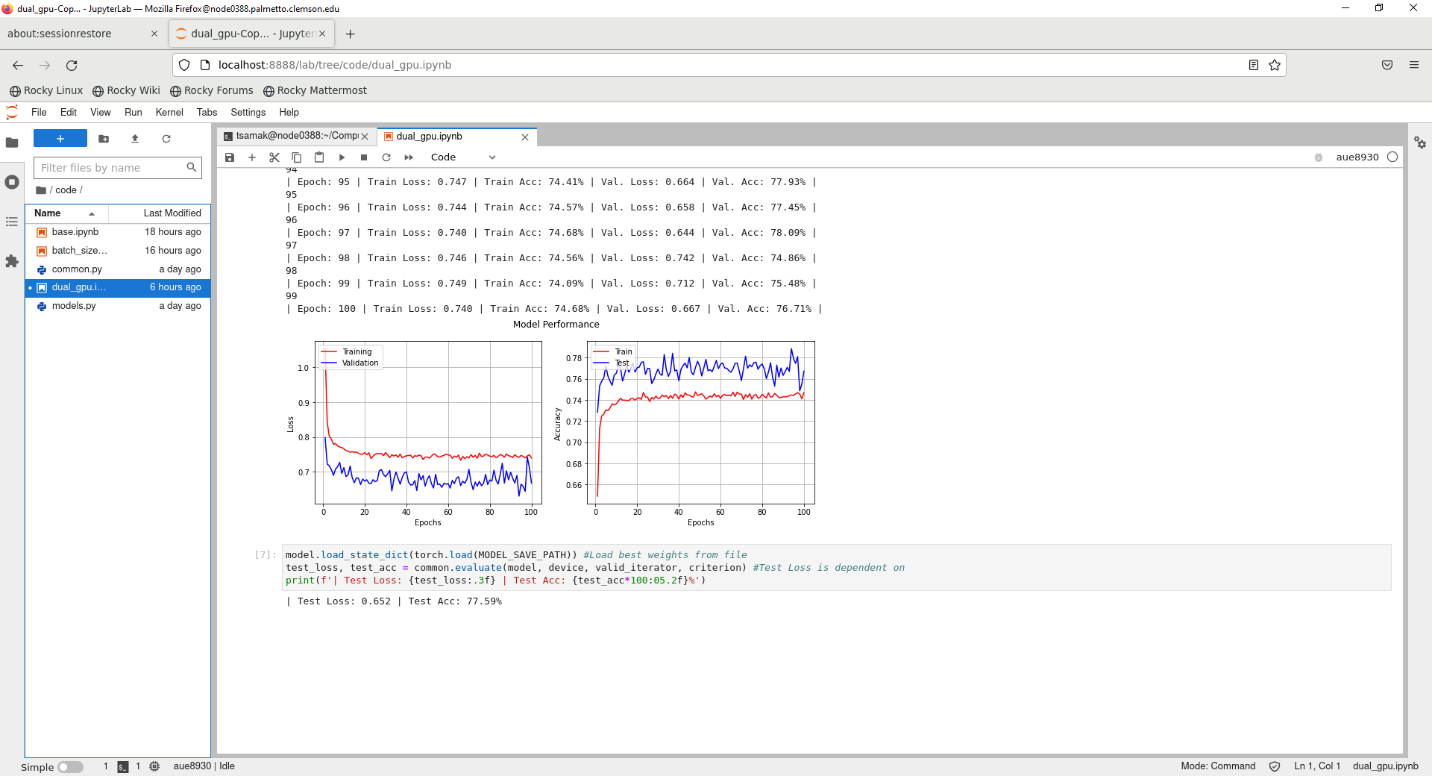


5.1 Import `matplotlib`

A screenshot of a computer

Description automatically generated

5.2 Modify the training section to keep track of loss and accuracy metrics and plot them.



5.3 Training and validation loss and accuracy plots for 100 epochs; testing loss and accuracy.

**(6) Could you improve on the network model, train it for better accuracy? (optional, 5 points)**

**(This question is optional. Extra 5 points until reach the cap of 100)**

A screenshot of a computer

Description automatically generated

6.1 Change batch size (512), and implement weight decay and robust augmentations.

A screenshot of a computer

Description automatically generated

6.2 Implement exponential learning rate scheduling with gamma = 0.9.

|  |  |
| --- | --- |
| Hyperparameters:  *BATCH\_SZIE = 512*  *WEIGHT\_DECAY = 1e-6*  *LR\_SCHEDULER.EXPONENTIAL\_LR(gamma=0.9)* | Augmentations:  *RANDOM\_HORIZONTAL\_FLIP(50%)*  *RANDOM\_CROP(256, padding=4, padding\_model=‘reflect’)* |

A screenshot of a computer

Description automatically generated

6.3 2xA100 GPU usage (GPU0=~10%; GPU1=~0%) and GPU RAM usage (GPU0=~6.4%; GPU1=~4.8%) for implementation with dual-GPU configuration.

A screenshot of a computer

Description automatically generated

6.4 Training, validation and testing loss and accuracy metrics for 100 epochs.

***Note:*** The training became significantly stable with the modification made to the pipeline. As a result, the training, validation and testing loss and accuracy metrics fall really close, which ultimately points towards superior performance of this model (as compared to the baseline implementation.

**(7) Perform a model inference for a certain image, which you can choose from anywhere. The image shall include the object which belongs to the category of the training dataset. (10 points)**

**(TIPS: if you are using CIFAR10 datasets, its categories are shown in this** [**reference**](https://www.cs.toronto.edu/~kriz/cifar.html)**)**

A screenshot of a computer

Description automatically generated

7.1 Mode inference on a test image with ground truth and predicted labels.

**Question 2**

Write a 2~3 pages survey report on a particular High-Performance-Computing application related to engineering/vehicles (40 points). The grading of this question will be based on the contents which the survey covers:

- What is the problem to be solved (5 points);

- The importance of the problem to be solved (5 points);

- The challenges of solving this problem (10 points);

- Existing solutions of solving this problem (15 points);

- Other grading factors (such as novelty, organization, etc.) (5);

\* You are encouraged to include any drawing/table in the report;

\* Attention: use like [1] to cite a content you referred to, with reference list in the end. You should never literally copy contents from other places;

TIPS: you should survey and read multiple academic papers, academic papers. Then, summarize for the above.