# Clarification

Our servers are not for CPU computing. CPU intensive programs may jam other users' GPU processes and may shut down the server due to the CPU RAM limit.

If you want to access high performance computing for CPU, you should check the Duke Computing Cluster which provides abundant CPU computing resources.

<https://dcc.duke.edu/>

# Hostname

[netid@research-tarokhlab-01.oit.duke.edu](mailto:netid@research-tarokhlab-01.oit.duke.edu)

to

[netid@research-tarokhlab-14.oit.duke.edu](mailto:netid@research-tarokhlab-14.oit.duke.edu)

Titan Xp 1

RTX 2080 2-3

RTX 3090 4-14

**14 server nodes and each has 4 GPUs**

# **On campus network or using VPN**

**# To avoid typing password and Duo verification during ssh or scp**

ssh-keygen -t rsa

Upload your ssh public key (copy id\_rsa.pub content) at https://oit.duke.edu/selfservice

under "advanced user options", "update your ssh public keys"

# To upload files

scp -i [id\_rsa.pub path] [your file] netid@research-tarokhlab-03.oit.duke.edu:

# To download files

scp netid@research-tarokhlab-03.oit.duke.edu:./output.zip .

# To Log in

ssh -i [id\_rsa.pub path] netid@research-tarokhlab-03.oit.duke.edu

# To check GPU status

NVIDIA-SMI

# To run python code with GPU

CUDA\_VISIBLE\_DEVICES="0" python train\_model.py

**0 here means the id of GPU device. There are 4 GPUs on each server and id is 0-3**

# To use data and save output

create symbolic link using

ln -s /hpc/group/tarokhlab/netid/data/ ./src/data

ln -s /scratch/netid/output ./src/output

**You should not save your model in your home directory /hpc/home**

**The SSD device is mounted at /scratch and /hpc/group/tarokhlab**

**See # File Storage for details**

# To prevent program halt due to lost connection

**use screen on the remote server**

<https://linuxize.com/post/how-to-use-linux-screen/>

* *create a screen*

screen -S netid

* *detach a screen*

screen -D netid (not attaching)

ctrl+a d (attaching)

* *reattach a screen*

screen -r netid

* *list screens*

screen -ls

* *kill a screen*

screen -X -S [session # you want to kill] quit

# To kill a process list by NVIDIA-SMI

kill -9 [process d]

# To kill all your processes on all GPUs

kill -9 $(nvidia-smi | sed -n 's/|\s\*[0-9]\*\s\*\([0-9]\*\)\s\*.\*/\1/p' | sort | uniq | sed '/^$/d')

# To (un)zip files

zip netid.zip -r [zip path]

unzip src.zip -d [folder name]

**# Use Jupyter notebook**

1. ssh netid@research-tarokhlab-xx.oit.duke.edu

2. Start Jupyter (you can first enter a screen and Jupiter in the screen so that you can still have a working shell)

3. ssh -NL localhost:1234:localhost:8889 netid@research-tarokhlab-xx.oit.duke.edu

4. Here 8889 is the host on GPU server, 1234 is the local host. You do not need to initiate a jupyter notebook on your local machine

5. Be sure to restart kernel once you are not using GPU, o.w. the program will occupy memory of GPU

**# File Storage**

**/scratch** – very fast, local NVME drives on each server, ~7TB each.  No redundancy and no backups, data loss can occur due to failures, system activities, and manual errors.  Use: storage needed during computation and model development.

**/hpc/home** – standard speed network attached storage, a single volume that will be mounted across all servers.  Robust enterprise storage with internal redundancy and full backups have been enabled.  Initial size 1TB. Use: home directories for individual lab members.

**/hpc/group/tarokhlab**- standard speed network attached storage, a single volume that will be mounted across all servers.  Robust enterprise storage with internal redundancy, (no backups but there is a self-service 7-day snapshot). This is the 1TB existing volume that we mentioned that Research Computing provides at no cost as a resource as a general-purpose storage volume across some RC services.  It is also mounted to the Duke compute cluster and available via Globus for transferring files in and out of RC storage. Use: shared data sets for the lab, or other project-based resources that will be used across the lab.

Anaconda is installed in **/hpc/group/tarokhlab/anaconda3/** and **/scratch/anaconda3**

Matlab is installed in **/hpc/group/tarokhlab/matlab2020b/**

**Attention:** Installing your own anaconda3 in /scratch/{netid}/anaconda3 is recommended.

We have two generally available GPU partitions for use in the DCC:

**gpu-common** for jobs that will run on DCC GPU nodes

**scavenger-gpu** for GPU jobs that will run on lab-owned nodes in “low priority” (kill and requeue preemption).

Researchers that have purchased GPU nodes have a high priority partition for their GPUs that will preempt any jobs submitted to scavenger-gpu and are running on their nodes.

Examples of batch submission are here: <https://dcc.duke.edu/dcc/slurm/?h=#gpu-jobs> and there’s also OnDemand for interactive Jupyterlab and RStudio jobs: <https://dcc.duke.edu/OpenOnDemand/> via the OnDemand server [https://dcc-ondemand-01.oit.duke.edu](https://dcc-ondemand-01.oit.duke.edu/).