THE UNIVERSITY OF AUCKLAND
Te Whare Wānanga o Tāmaki Makaurau
NEW ZEALAND

ENGINEERING
DEPARTMENT OF ELECTRICAL,
COMPUTER, AND SOFTWARE ENGI

The HASEL Deep Learning Workstations

User Access Manual

Dr. Reza Shahamiri

Department of Electrical, Computer, and Software Engineering

Faculty of Engineering

The University of Auckland

Version 1.1

Contents

Introduction

This document provides a user guide on how to access Human Aspects of Software Engineering Lab's Deep Learning (DL) Workstations remotely, how to run Jupyter Notebook, install conda, and copy and access your data.

Although researchers can have physical access to these machines via the lab (providing you already have HASEL access), the preference is via remote access and `ssh` terminals in which users connected to the UoA network (or VPN) can simply log in to the machines via their own devices and run Jupyter Notebook to train their DL models.

These DL machines are not for code and architecture design. In other words, users need to code their solutions and conduct the initial trials to (almost) finalize their DL architectures on their own machines or online platforms such as Google Collab, and then train and finetune their models on these workstations to ensure maximum availability of the workstation for other researchers since we have a limited number of Graphical Processing Units (GPUs) available.

The Workstations

Currently the HASEL lab is equipped with three DL workstations as follow:

1. **Zeus** is a quad RTX6000 GPU setup linked with NVLink, AMD Threadripper 3990X 64 cores CPU, and 256GB of RAM.
2. **Apollo** is a dual RTX 3090 GPU setup. It has 256GB of RAM on board, and its CPU is an AMD Threadripper 3970X with 32 cores and 128 MB cache.
3. **Artemis** is identical to Apollo.

User Signup and Login Credentials

The workstations are not connected to the UoA's user identification system hence we need to manually create your user account on the workstation you need access to. To do so please complete the HASEL Deep Learning Workstation Signup form. The academic staff will be given access to Zeus and students to Apollo and Artemis by default. If you need access to other machines for yourselves or your students, please mention that in your request. However, we may not be able to accommodate your request if the requested machine is already fully loaded.

Please be advised that your user account will be created on the workstation you request access to, and the same account does not work on the other workstations.

Once a request is submitted, we will reply with your username, a default password (that you need to change), which workstation you can access, and the validity duration of your account (see below).

Student Access

Student access should be authorized by their academic supervisor. If you are a student, please complete the form, and then ask your supervisor to authorize your request by emailing us at reza.shahamiri@auckland.ac.nz, also supplying a timeframe for which the student's account to remain valid (i.e. active).

User Account Validity Duration

To promote faire and coordinated access to the workstations, all user accounts will be periodical with a limited validity date. Once a user account is expired, the user will not have access to the machines. If you still require accessing the workstations, you can submit a new request, and your request will be processed based on the number of active users of the time. In case of no available user slot available, your request will be added to a waiting list; this may also apply to new users.

PhD students will have an active account for one year and other students for six months by default. We consider shorter or longer time frames based on the demand on the workstations, the waiting list status, and priorities.

Once a user account is expired, we may delete the user profile and files if data storage is needed – hence, **please ensure to copy your data back to your local machine before your account is expired**. The HASEL Lab takes no responsibility for any data loss; hence having a backup of your code and trained model is highly recommended.

IP Addresses

Connecting to the workstations is via their IP addresses allocated by the UoA. The IP address for each workstation is given below:

- Zeus: **10.104.144.223**
- Apollo: **10.104.144.2**
- Artemis: **10.104.144.222**

Connect to the Workstation Remotely

Once your account is created, you need to lunch Terminal (in macOS) or Command Prompt (in Windows), or other SSH terminals of your choice, **on your local machine**. Your local machine must be connected to the UoA network ((either via Wi-Fi or Ethernet cable); if you are off-campus, you need to connect to the UoA's VPN. Please see The UoA VPN instructions here.

If Open SSH is not installed in your Windows OS, you need to add it as explained here. Alternatively, you can use other ssh clients, such as PuttySSH.

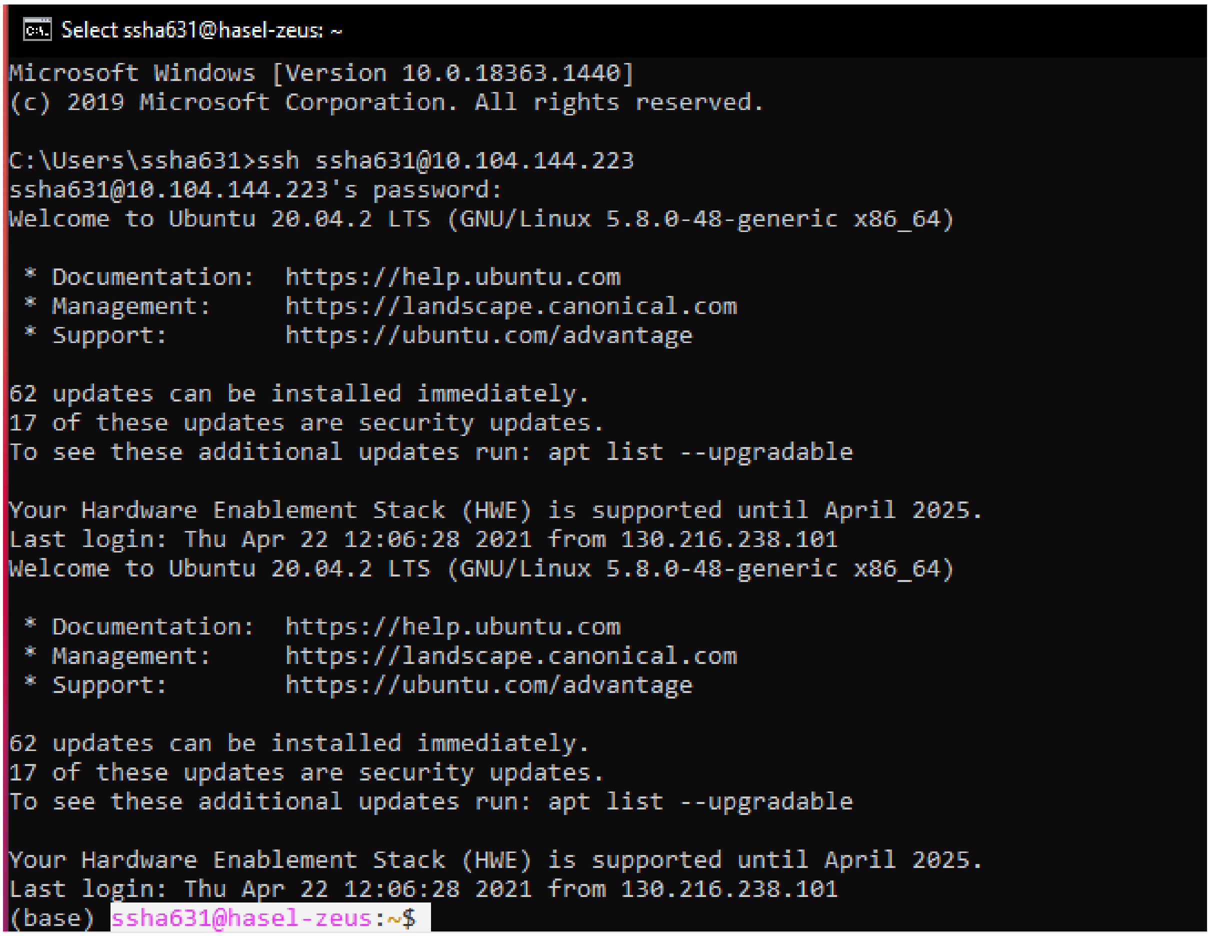
After launching your ssh terminal, you can connect to the workstation using your upi and the workstation IP address via:

```
ssh upi@IP
```

For example, assuming your upi is *ssha631* and your workstation is *Zeus* **:**

```
ssh ssha631@10.104.144.223
```

You will be asked to type your password, and then you should connect to the workstation. You should see the following in your terminal. From now on, all your commands will be executed on the remote workstation:



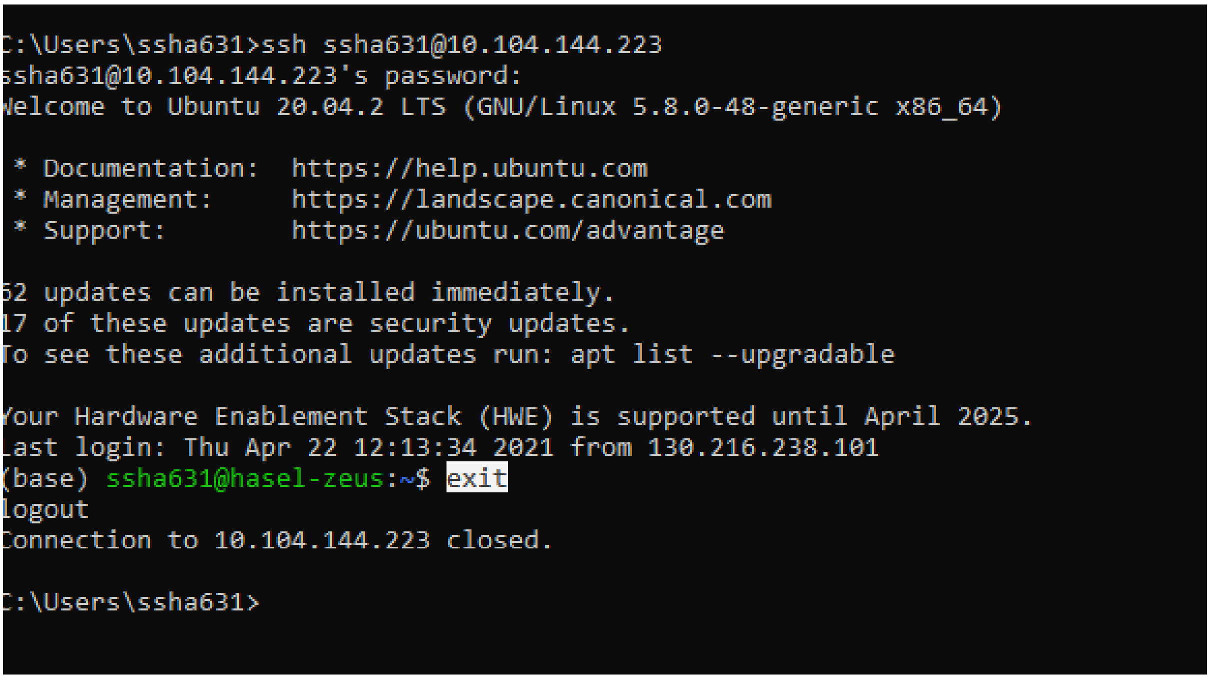
```
Microsoft Windows [Version 10.0.18363.1440]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\ssha631>ssh ssha631@10.104.144.223
ssha631@10.104.144.223's password:
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.8.0-48-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

62 updates can be installed immediately.
17 of these updates are security updates.
To see these additional updates run: apt list --upgradable

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Thu Apr 22 12:06:28 2021 from 130.216.238.101
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.8.0-48-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

62 updates can be installed immediately.
17 of these updates are security updates.
To see these additional updates run: apt list --upgradable

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Thu Apr 22 12:06:28 2021 from 130.216.238.101
(base) ssha631@hasel-zeus:~$
```

Changing Password

To change your password after connecting to the workstation, simply type `passwd` and follow the on-screen instructions.

Closing the Connection

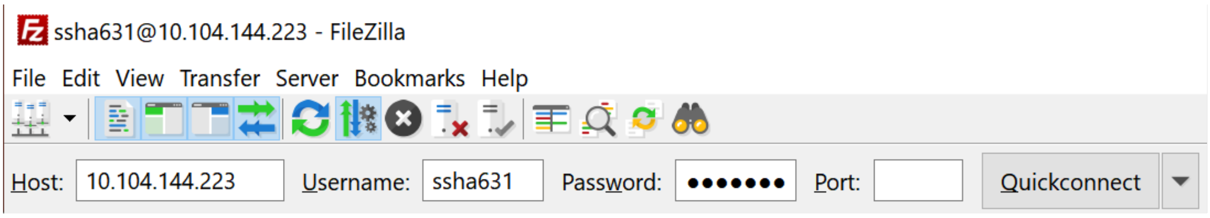
It is important that you close the SSH connection after you are done. To do so, type `exit` in the SSH terminal:



```
C:\Users\ssha631>ssh ssha631@10.104.144.223
ssha631@10.104.144.223's password:
Welcome to Ubuntu 20.04.2 LTS (GNU/Linux 5.8.0-48-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

52 updates can be installed immediately.
17 of these updates are security updates.
To see these additional updates run: apt list --upgradable

Your Hardware Enablement Stack (HWE) is supported until April 2025.
Last login: Thu Apr 22 12:13:34 2021 from 130.216.238.101
(base) ssha631@hasel-zeus:~$ exit
logout
Connection to 10.104.144.223 closed.

C:\Users\ssha631>
```

Transferring Data

There are multiple ways to transfer your data between your local machine and the workstations. A simple way is via an FTP Client such as FileZila Client:



ssha631@10.104.144.223 - FileZilla

Host: 10.104.144.223 Username: ssha631 Password: ●●●●●●● Port:

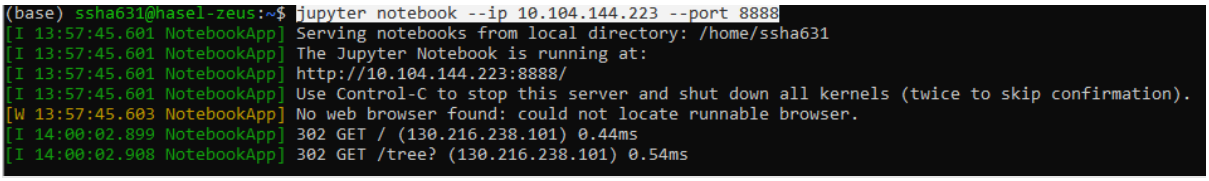
Running Jupyter Notebook Remotely

Preparation:

1. Ensure a ssh connection is already established to the remote workstation, and
2. Ensure that password for Jupyter Notebook is set via `jupyter notebook password`. Setting the password is only needed for the first time that you run Jupyter Notebook.

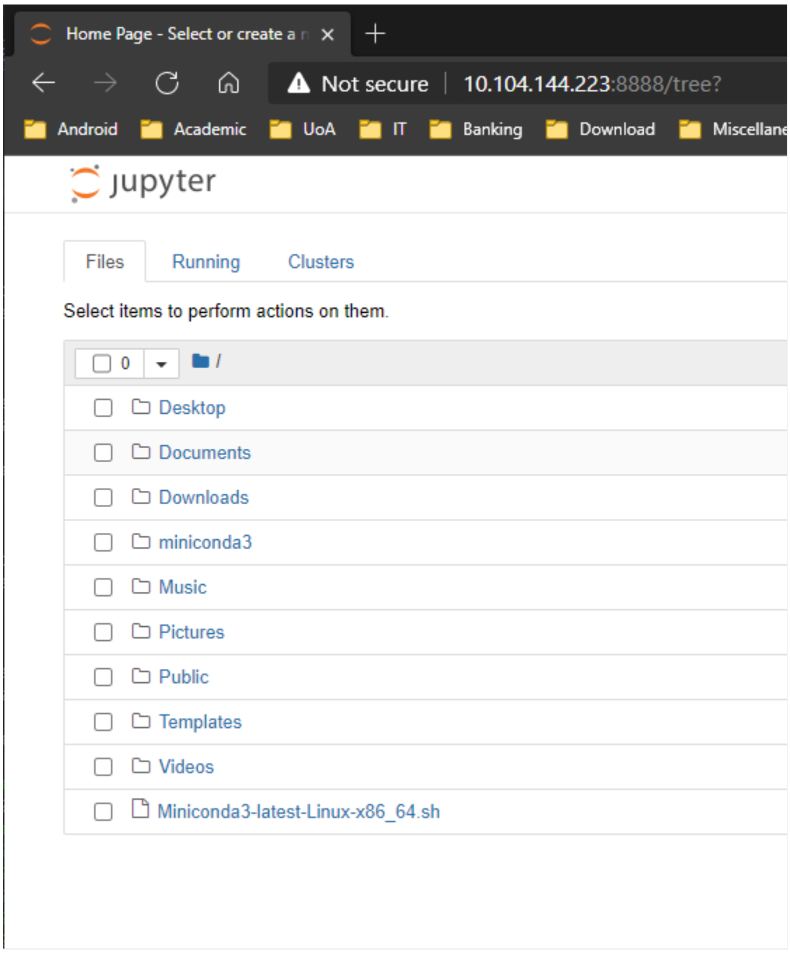
Once the password is set, you can run Jyputer Notebook by specifying the IP of the remote workstation and a free port: `jupyter notebook --ip remote_ip --port port_no`

For example, `jupyter notebook --ip 10.104.144.223 --port 8888`:



```
(base) ssha631@hasel-zeus:~$ jupyter notebook --ip 10.104.144.223 --port 8888
[I 13:57:45.601 NotebookApp] Serving notebooks from local directory: /home/ssha631
[I 13:57:45.601 NotebookApp] The Jupyter Notebook is running at:
[I 13:57:45.601 NotebookApp] http://10.104.144.223:8888/
[I 13:57:45.601 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[W 13:57:45.603 NotebookApp] No web browser found: could not locate runnable browser.
[I 14:00:02.899 NotebookApp] 302 GET / (130.216.238.101) 0.44ms
[I 14:00:02.908 NotebookApp] 302 GET /tree? (130.216.238.101) 0.54ms
```

Jupyter notebook is now ready, and you can access it via your local browser. To do so, run the browser on your local machine and navigate to http://IP:PORT, http://10.104.144.223:8888/ in the above example, and enter the password you selected for Jupyter Notebook. You are now ready to open your notebook and run them on the workstation:



10.104.144.223:8888/tree?

Select items to perform actions on them.

- Desktop
- Documents
- Downloads
- miniconda3
- Music
- Pictures
- Public
- Templates
- Videos
- Miniconda3-latest-Linux-x86_64.sh

Please be advised that connections will be refused if the port is already occupied. Try another port number of your choice.

Important: Please terminate the notebook and close Jupyter Notebook connection after you are done. Otherwise, the allocated port may not be released.

Lambda Deep Learning Stack

The workstations come pre-installed with Lambda Deep Learning stack. For more information, please see here.

If you need other packages or different versions of the installed packages, you need to:

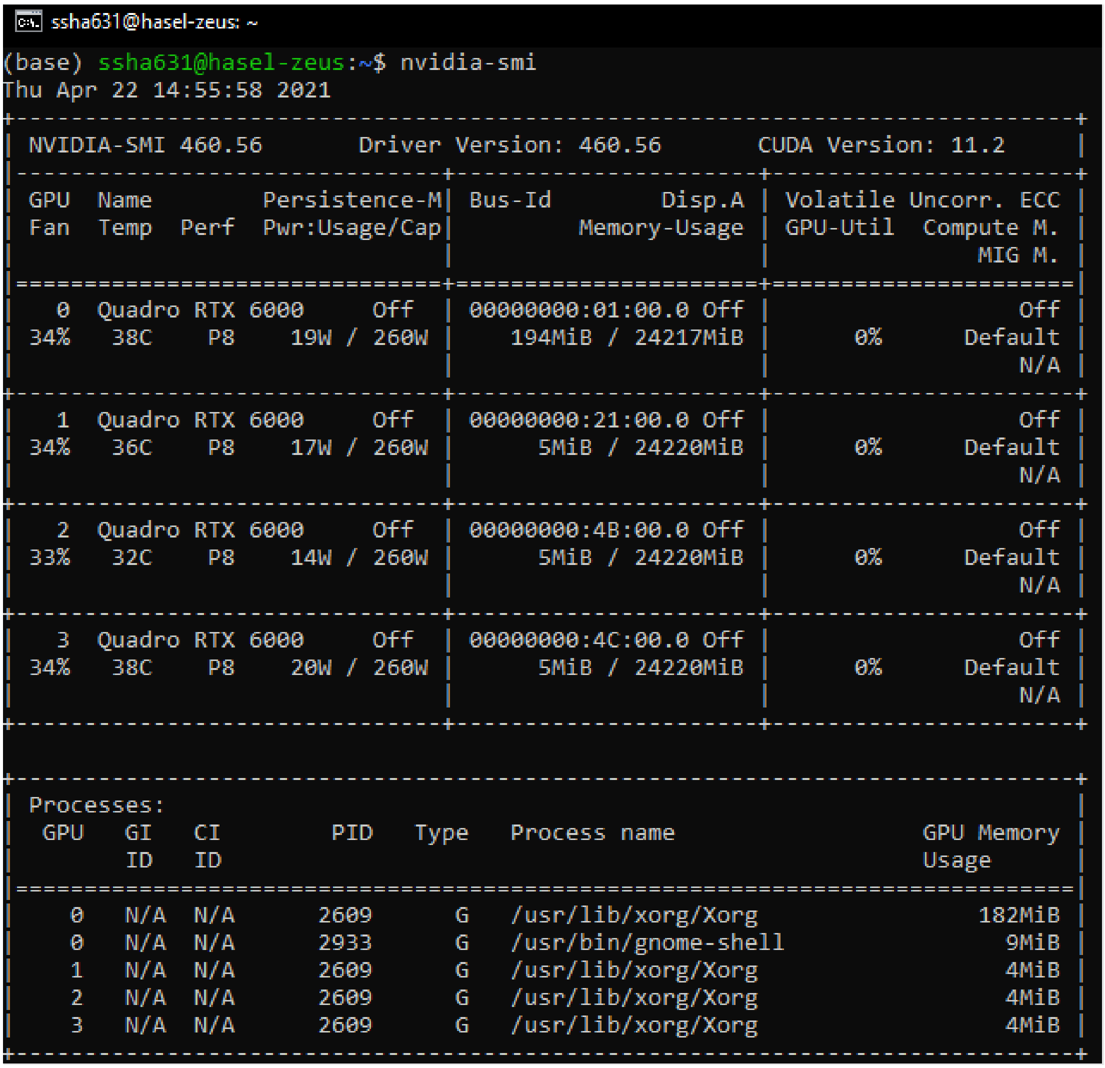
1. Install Miniconda as explained below,
2. Create a conda environment,
3. Activate the environment `conda activate myenv`
4. Use conda to install the packages you need.

Install Miniconda Remotely

1. Download Miniconda for Linux on your local machine from here
2. From your local machine, transfer the downloaded file as explained before.
3. Login to the remote server via `ssh upi@host`
4. On the remote workstation, navigate (via ssh) to the location you copied the miniconda downloaded file.
5. Install miniconda via `bash the_downloaded_miniconda_installation.sh`

Check GPU Availability

While GPUs are being used, they operate in locked mode, which means they are not available for other DL processes. Any attempt to use them will result in an error. You can check which GPUs are available (aka free) using `nvidia-smi` command and looking at “Volatile GPU-Util” column. If 0%, then the GPU is not being used:



```
(base) ssha631@hasel-zeus:~$ nvidia-smi
Thu Apr 22 14:55:58 2021
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 460.56       Driver Version: 460.56       CUDA Version: 11.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Quadro RTX 6000     Off  | 00000000:01:00.0 Off |                  Off |
| 34%   38C    P8    19W / 260W |    194MiB / 24217MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   1  Quadro RTX 6000     Off  | 00000000:21:00.0 Off |                  Off |
| 34%   36C    P8    17W / 260W |      5MiB / 24220MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   2  Quadro RTX 6000     Off  | 00000000:4B:00.0 Off |                  Off |
| 33%   32C    P8    14W / 260W |      5MiB / 24220MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   3  Quadro RTX 6000     Off  | 00000000:4C:00.0 Off |                  Off |
| 34%   38C    P8    20W / 260W |      5MiB / 24220MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|    0   N/A  N/A      2609      G   /usr/lib/xorg/Xorg                182MiB |
|    0   N/A  N/A      2933      G   /usr/bin/gnome-shell                9MiB |
|    1   N/A  N/A      2609      G   /usr/lib/xorg/Xorg                  4MiB |
|    2   N/A  N/A      2609      G   /usr/lib/xorg/Xorg                  4MiB |
|    3   N/A  N/A      2609      G   /usr/lib/xorg/Xorg                  4MiB |
+-----------------------------------------------------------------------------+
```

If a GPU is not free, you can configure your notebook to use another GPU, as explained below.

Select GPU(s) in TensorFlow/Keras

In order to set which GPU(s), and how many GPUs you intend to use in Tensorflow 2.x, you need to use Strategy:

1. Create an strategy instance: `strategy = tf.distribute.MirroredStrategy()`
   - Here, you can supply which GPU(s) you intend to use as well. For example:

```
strategy = tf.distribute.MirroredStrategy(["GPU:0", "GPU:1"])
```

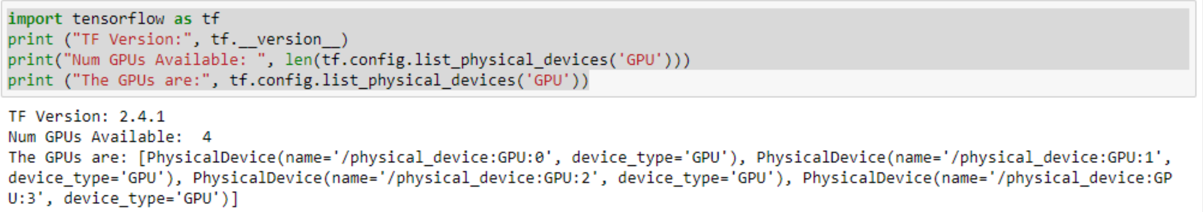
   - Otherwise, all GPUs will be used.
   - Please check for GPU availability first (via `nvidia-smi`)
2. Define and compile the Keras model inside the strategy instance:

```
with strategy.scope():
        //Build the keras model
        //Compile the model
```

View Installed GPUs

You can use `tf.config.list_physical_devices('GPU')` to check if TensorFlow is using GPUs, and how many GPUs are visible to TensorFlow. Please be noted that this command does not check if GPUs are free:

```
import tensorflow as tf
print ("TF Version:", tf.__version__)
print("Num GPUs Available: ",
len(tf.config.list_physical_devices('GPU')))
print ("The GPUs are:", tf.config.list_physical_devices('GPU'))
```



```
import tensorflow as tf
print ("TF Version:", tf.__version__)
print("Num GPUs Available: ", len(tf.config.list_physical_devices('GPU')))
print ("The GPUs are:", tf.config.list_physical_devices('GPU'))
```

```
TF Version: 2.4.1
Num GPUs Available:  4
The GPUs are: [PhysicalDevice(name='/physical_device:GPU:0', device_type='GPU'), PhysicalDevice(name='/physical_device:GPU:1',
device_type='GPU'), PhysicalDevice(name='/physical_device:GPU:2', device_type='GPU'), PhysicalDevice(name='/physical_device:GP
U:3', device_type='GPU')]
```

Data Server

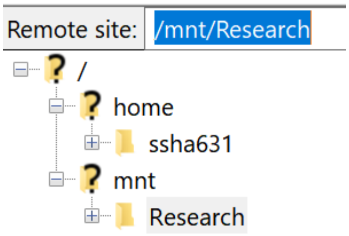
Each workstation has a 2TB HDD. Additionally, the workstations are connected to a private data server to enable the users to store their data. This data server has a 20TB SSD setup with RAID 5 configuration, which means 10TB storage is available for users and another 10TB to backup the data. The workstations are connected to the data server over 10GB Ethernet connection, ensuring ultra-fast speed.

All users can select to store their data on the data server, especially if you work with multiple workstations. You can navigate to the data server folder (explained below) and run your notebook from that location.

There are two folders on the data server in which you can use to store your data:

1. **Research**
    - Location: `/mnt/Research`
    - Use this to store anything related to your research.
2. **DataSets**
    - Location: `/mnt/DataSets`
    - Use this space for your data sets.

You can use the same FTP approach to transfer your data to the data server. However, please ensure to change the working remote directory to `/mnt/Research` or `/mnt/DataSets` in your FTP client:



Remote site: /mnt/Research

```
/
  home
    ssha631
  mnt
    Research
```

Please create a folder with your upi as the folder name. This will help us identify the corresponding owner of the stored data. Any data outside upi folders will be deleted.

To run Jyputer Notebook from the data server location, change the working directory to one of the data server directories (for example, `cd /mnt/Research`) after you ssh into your workstation, then run Jupyter Notebook as explained before.

Data Privacy and Protection

All data stored on the workstations will be in your private profile, where you and administrators can access them. By default, they are not accessible by other users. On the other hand, any data on the data server is shared with all users.

Please be advised that we do not have any specific data protection strategy instead of the default user protection mechanism supplied by the operating system. Furthermore, although your data on the data server are mirrored, there is no backup mechanism on the workstations, and any HDD failure may result in data loss. Thus, we recommend that you backup your data and code frequently over your personal computers.

Contact us

You can contact Reza Shahamiri at reza.shahamiri@auckland.ac.nz or Emmanuel Baguia at e.baguia@auckland.ac.nz if further information is needed.