Introduction of using HPC for Deep Learning on Windows

10-August-2020

Prepared by: ZHANG JINHUI

National University of Singapore

1. Prerequisite

HPC Access

Register via https://nusit.nus.edu.sg/services/hpc/getting-started-hpc/register-for-hpc/

Software

Download MobaXterm via https://mobaxterm.mobatek.net/download.html

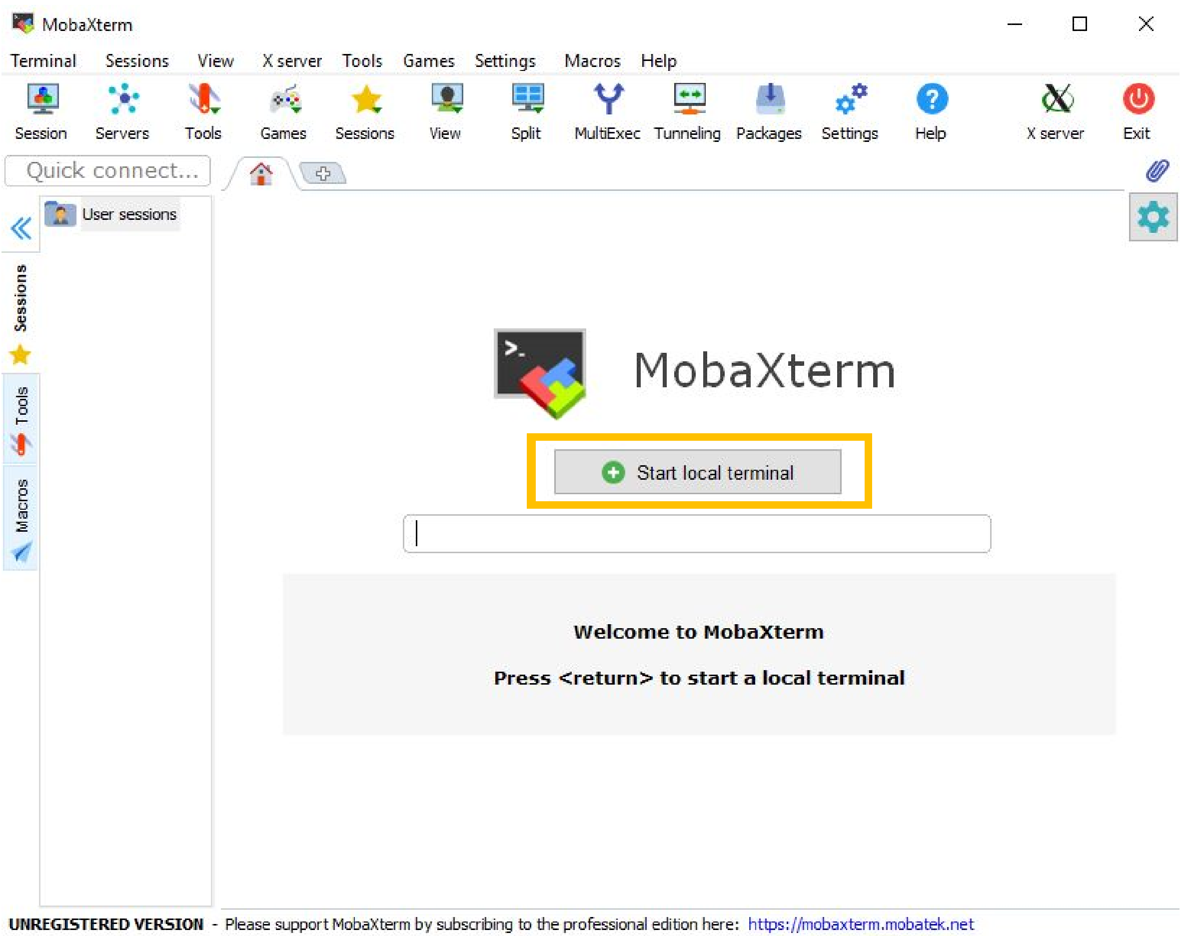
Free version works well. Portable or installer version has no functionality difference. Feel free to choose either.

NUS VPN

If you are not inside NUS campus, connect to nVPN before deployment
https://nusit.nus.edu.sg/services/wifi_internet/nvpn/

2. Access and File Transfer

2.1 Run MobaXterm to access the login node



This is the landing page that you will see when you first open MobaXterm

To start, open a local terminal

When first starting up MobaXterm, the system will ask for firewall permission.

Allow MobaXterm to be exempted form the firewall to allow it to function

2. Access and File Transfer

2.2 Accessing the login node

```
• MobaXterm Personal Edition v10.9 •
(X server, SSH client and network tool

➤ Your computer drives are accessible through the /
➤ Your DISPLAY is set to 172.23.190.69:0.0
➤ When using SSH, your remote DISPLAY is automatica
➤ Each command status is specified by a special sym

• Important:
This is MobaXterm Personal Edition. The Professiona
allows you to customize MobaXterm for your company:
your own logo, your parameters, your welcome messag
either an MSI installation package or a portable ex
We can also modify MobaXterm or develop the plugins
For more information: https://mobaxterm.mobatek.net

[2018-11-07 11:52.08]  ~
[Michael.MichaelLaptop] ➤ ssh atlas7 -l mpemic
```

To access a login node, type in:

`ssh <login_node> -l <username>`

at the line.

Replace `<login_node>` with any of the 5 login nodes and `<username>` with your NUSNET username

In this example, I used `atlas7` as the login node and `mpemic` as the username

`ssh atlas7 -l mpemic`

After confirming that the command is correct, press the "enter" key on the keyboard

Note: For deep learning, simply login to atlas9 instead to access *volta_gpu*

2. Access and File Transfer

2.2 Accessing the login node

```
[2018-11-07 12:05.22]  ~
[Michael.MichaelLaptop] ➤ ssh atlas7 -l mpemic
mpemic@atlas7's password: 
```

You will then be prompted to enter your password

This password is your NUSNET password.

Do not be alarmed if there is no character that appears as you type in your password. *This is normal behavior.* Your keyboard is working fine and the system registers your input.

Just type your password normally, being extra careful, and press enter afterwards.

Press "enter" after you are done.

It is recommended that you save your password so future logins will be easier

2. Access and File Transfer

2.2 Accessing the login node

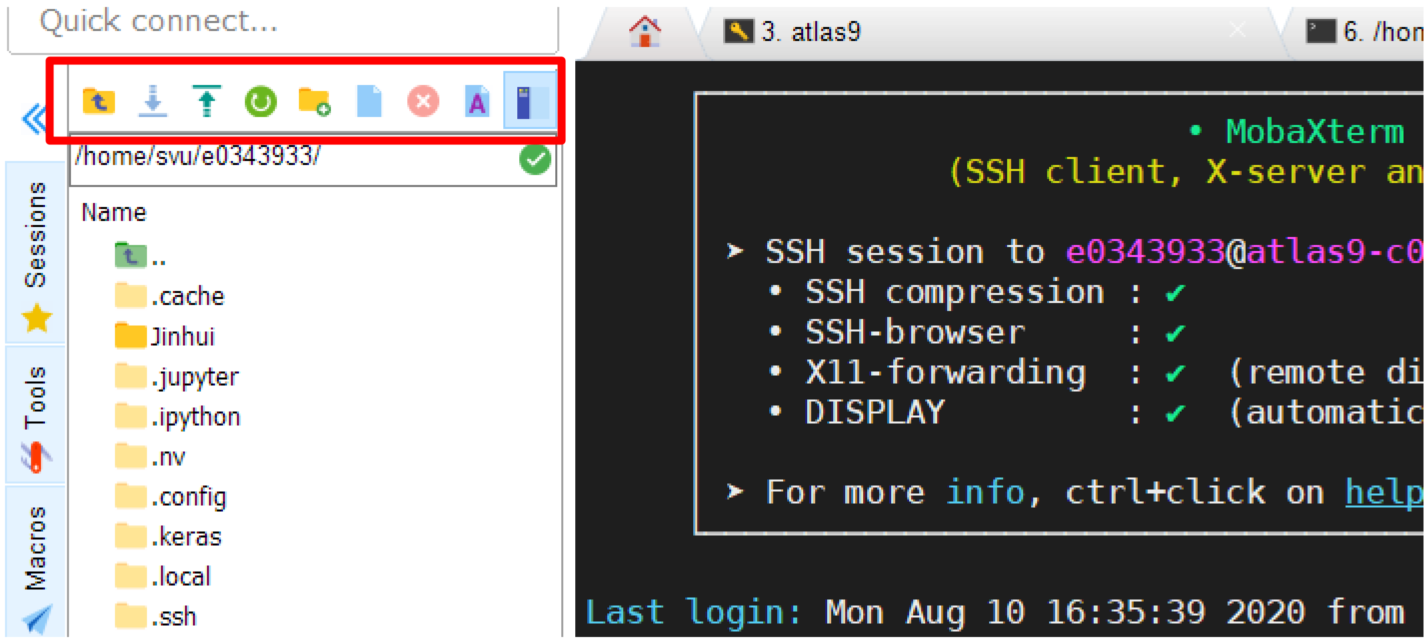
```
[2018-11-07 12:18.21]  ~
[Michael.MichaelLaptop] ➤ ssh atlas7 -l mpemic
Last login: Wed Nov  7 12:10:02 2018 from 172.23.190.69
*****************************************************
# Use PBS Job Scheduler to Submit and Manage Jobs    #
#                                                    #
# Help info available via command: hpc pbs -help     #
*****************************************************
[mpemic@atlas7-c10 ~]$ 
```

If you have arrived at this page, you are logged in.

2. Access and File Transfer

File transfer

After login to atlas 9, you still need to upload your data, codes before deployment. Given the error of downloading FileZilla, simply upload/download files via MobaXterm directly.



3. Deployment

3.1 Short, Interactive Jobs

For debugging

For checking if your code works

3.2 Long, Batch Jobs

For running working code for long periods of time.

3.1 Interactive Jobs

Run short interactive jobs meant for **debugging** your code on GPU.

qsub -I -l select=1:mem=50GB:ncpus=10:ngpus=1 -l walltime=02:00:00 -q volta_login

- -I : Interactive
- Mem: Max 96GB, Min 50GB
- ncpus: 6 to 20
- ngpus: 1 to 2 (Choose less, wait less)
- -q : volta_login queue

Default Walltime: 1 hour

Max Walltime: 4 hours

3.1 Interactive Jobs

● Once the interactive job is launched, you will be brought into **volta01**.

● In Volta01, you can launch a container of your choice to debug your deep learning python script.

qsub -I -l select=1:mem=50GB:ncpus=10:ngpus=1 -l walltime=02:00:00 -q volta_login
singularity exec $image bash

● You can replace $image with the container image of your choice, e.g.:

/app1/common/singularity-img/3.0.0/tensorflow_1.12_nvcr_19.01-py3.simg

● List all available containers

ls /app1/common/singularity-img/3.0.0/

3.1 Interactive Jobs

Example of using Jupyter Notebook

Red: must change
Green: configurable
Black: fixed

1. Submit Request

#request

qsub -I -l select=1:mem=50GB:ncpus=10:ngpus=1 -l walltime=02:00:00 -q volta_login

#select container (tensorflow)

singularity exec /app1/common/singularity-img/3.0.0/tensorflow_2.0.0_cuda10.0-cudnn7-devel-ubuntu18.04-py36.simg jupyter notebook --no-browser --port=8889 --ip=0.0.0.0

2. Open a new terminal, and run

ssh -L 8888:volta01:8889 nusnet_id@atlas9

If you receive an error that says port 8889 is already in use, choose another port

3. Open a browser session, and use URL

http://localhost:8888

3.2 Batch Jobs

- Up to 2 GPUs per job
  - Please request only what you need.
  - ENSURE THAT YOUR CODE IS MODEL OR DATA PARALLEL
  - Using >1 GPU has to be **explicitly programmed** in your code.
  - Choose LESS, wait LESS

- Most applications use only 1 GPU by default
  - Requesting >1 GPUs does not make it faster. The other GPUs will not be utilised at all.

- Most applications will not scale well with > 1 GPUs

- 1 GPU is most likely more than enough as the V100-32GB has large memory capacity

3.2 Batch Jobs

Configuration

Max RAM = 300gb

Max No. of CPU cores = 20

Max No. of GPUs = 2

Max Walltime = 72:00:00

Minimum No. of CPU cores = 5

Minimum No. of GPU = 1

Default Walltime = 04:00:00

Request CPU Core in increments of 1

3.2 Batch Jobs

Create a job script

Go back to the MobaXterm program.

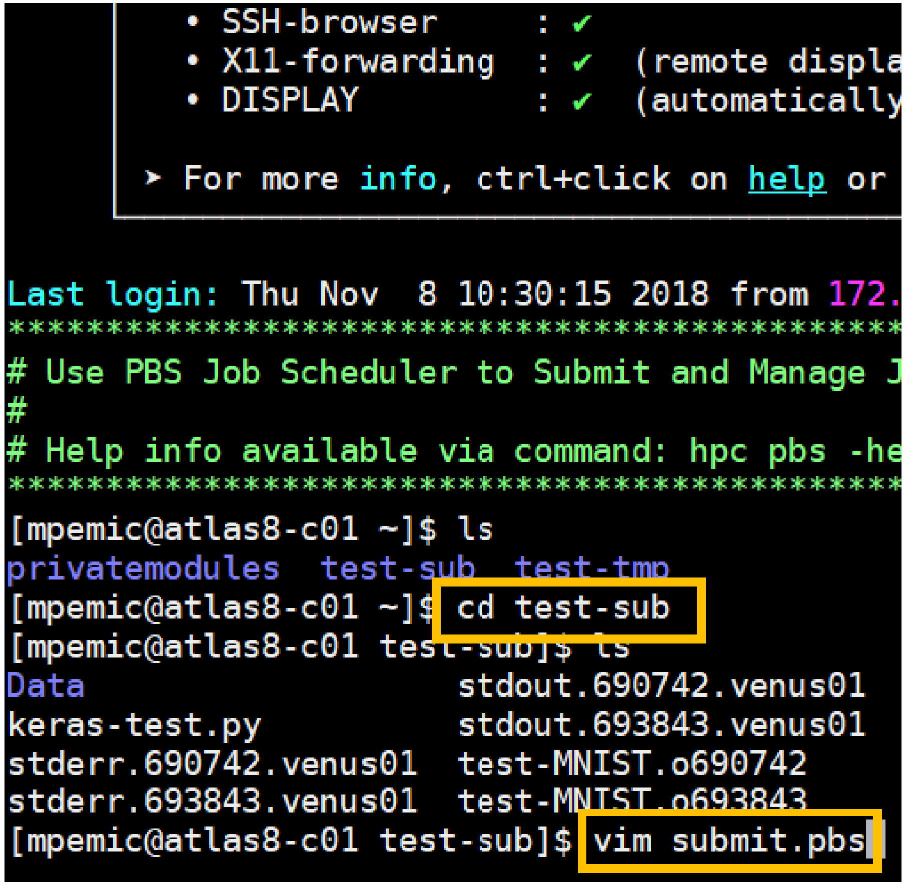
Go into the directory where the python file and Dataset is located.

cd <folder_name>

Create a new file with:

vim <script_file_name>.pbs

In this example, I named the folder `test-sub` and the file name `submit.pbs`



3.2 Batch Jobs

Example of “submission.pbs”

Green is user configurable
Black is fixed

```
#!/bin/bash
#PBS -P volta_pilot
#PBS -j oe
#PBS -N tensorflow
#PBS -q volta_gpu
#PBS -l select=1:ncpus=20:mem=100gb:ngpus=2
#PBS -l walltime=24:00:00

cd $PBS_O_WORKDIR;
np=$(cat ${PBS_NODEFILE} | wc -l);

image="/app1/common/singularity-img/3.0.0/tensorflow_1.12_nvcr_19.01-py3.simg"

singularity exec $image bash << EOF > stdout.$PBS_JOBID 2> stderr.$PBS_JOBID

NCCL_DEBUG=INFO ; export NCCL_DEBUG
mpirun -np 2 -x NCCL_DEBUG python keras_mnist_advanced.py

EOF
```

3.2 Batch Jobs

Submission to HPC

- Press `esc` key, then type `:wq`, and enter (this will save the job script and go back to main command).

```
[mpemic@atlas8-c01 test-sub]$ vim submit.pbs
[mpemic@atlas8-c01 test-sub]$ ls
Data                    stdout.690742.venus01
keras-test.py           stdout.693843.venus01
stderr.690742.venus01   submit.pbs
stderr.693843.venus01   test-MNIST.o690742
[mpemic@atlas8-c01 test-sub]$ qsub submit.pbs
```

- Type `qsub <script_file_name>.pbs` and enter to submit the job.

```
[mpemic@atlas8-c01 test-sub]$ qsub train-model.pbs
696172.venus01
[mpemic@atlas8-c01 test-sub]$ ls
Data                    stderr.693843.venus01   submit.pbs           test
keras-test.py           stdout.690742.venus01   test-MNIST.o690742   trai
stderr.690742.venus01   stdout.693843.venus01   test-MNIST.o693843
[mpemic@atlas8-c01 test-sub]$ qstat
Job id            Name              User              Time Use S Queue
----------------  ----------------  ----------------  -------- - -----
696172.venus01    test-MNIST        mpemic                   0 Q azgpu
[mpemic@atlas8-c01 test-sub]$
```

- You can check the status by typing `qstat`.

Reference

[1] Michael Surjawidjaja and Nicholas Ho: “HPC and NSCC DGX for dummies”
[2] Ku Wee Kiat: “Volta Cluster User Guide"
http://bobcat.nus.edu.sg:3080/dl_ml/DL_on_NUS_HPC.pdf