Guide to
High-performance Computing

Nicklas Hansen

What is the HPC?

The High-performance Computing (HPC) cluster is a computing cluster available to all students enrolled at DTU. It offers remote execution of scripts via a queue system and can handle very resource-intensive tasks such as training deep neural networks due to its powerful CPUs and GPUs in particular. For deep learning, we are mostly interested in the GPU nodes.

Connecting to the HPC

You can connect to DTU's HPC cluster either by terminal or by a graphical user interface. Terminal connection is faster but graphical interface is easier, so use whatever you are more comfortable with. To connect by SSH through terminal, use the following command:

```
ssh <student ID>@login2.gbar.dtu.dk
```

Note that a SSH connection requires a locally stored SSH key. If you do not have an SSH key already, refer to this link for a guide on how to generate SSH key pairs. If you prefer the graphical user interface, refer to this link for a guide on how to connect to the cluster through the ThinLinc client (requires a stable internet connection).

Installing dependencies

Now that you're connected to the HPC, you have access to UNIX commands and a remote user directory linked to your account (i.e. it is persistent). To run Python and GPU-accelerated scripts, you can load pre-installed *modules* with the command below. You will need to make sure that your versions of Python, CUDA and CUDNN are compatible. I recommend the following versions:

```
module load python3/3.6.2
module load cuda/8.0
module load cudnn/v7.0-prod-cuda8
```

You can call

```
module avail
```

to get an overview of other available modules and/or versions.

If you want to avoid loading modules every time you connect to the HPC, you can add the commands above to your **.bashrc** file located in the root directory using your favorite editor, e.g.:

```
nano .bashrc
```

To install PyTorch and any other dependencies that you might need for your project, use the following command:

```
pip3 install --user torch torchvision matplotlib seaborn
```

Pip installs are persistent and you will only need to install packages once. Don't forget the 3 in pip as you will otherwise install packages for Python 2.x. *Note that you should avoid conda on the HPC due to compatibility issues*.

Transferring files to the HPC

You can use Rsync (UNIX only), git or cloud services such as Google Drive or OneDrive to synchronize files between your local machine and the HPC (with Rsync obviously being the faster and more convenient option). A transfer of directory *~/mydir* from your local machine to the remote user directory using Rsync can be executed as follows:

```
rsync -av ~/mydir <student ID>@transfer.gbar.dtu.dk:
```

You will have to use your imagination (or the link above) to transfer in the reverse direction.

Using GPUs on the HPC

You can use GPUs in two different modes of operation: interactively or by submitting jobs to a queue. You should primarily use the interactive nodes for development and debugging purposes as the number of interactive nodes is limited and threads running on the interactive nodes are typically killed after a few hours. See this page for a guide on how to connect to an interactive node.
To submit a job for the GPU queue, you may want to create a bash script similar to what I have prepared for you here:

```
#!/bin/sh
#BSUB -q gpuk80
#BSUB -gpu "num=1"
#BSUB -J myJob
#BSUB -n 1
#BSUB -W 10:00
#BSUB -R "rusage[mem=32GB]"
#BSUB -o logs/%J.out
#BSUB -e logs/%J.err
<loading of modules, dependencies etc.>
```

```
echo "Running script..."
python3 main.py
```

which you can submit using the command:

```
bsub < jobscript.sh
```

For an overview of BSUB and available options, check out this link. When submitting jobs to the queue you need to load modules etc. as part of your bash script. When running on an interactive node, you can put it in your .bashrc.

Let me know on Slack if there's any issues.