THE UNIVERSITY *of* EDINBURGH
informatics

# Introduction to the Informatics GPU Cluster

Researching Responsible and Trustworthy Natural Language Processing

9 October 2024

Frank Keller
Slide credit: Tom Sherborne

## Overview

- GPUs in NLP
- Working with GPUs
- What is a cluster and Slurm
- When to use a cluster
- How to access and use it
- Walkthrough running experiments
- Getting help
- Sort demo

## Reading the room

- ✅ I’m comfortable with shells/bash, ssh and remote access
- ✅ I am comfortable writing code for my experiments
- ✅ I know how to use CUDA and run GPU experiments
- ✅ I have used a cluster (any cluster) before
- ✅ I have used a Slurm managed cluster before

## GPUs in NLP

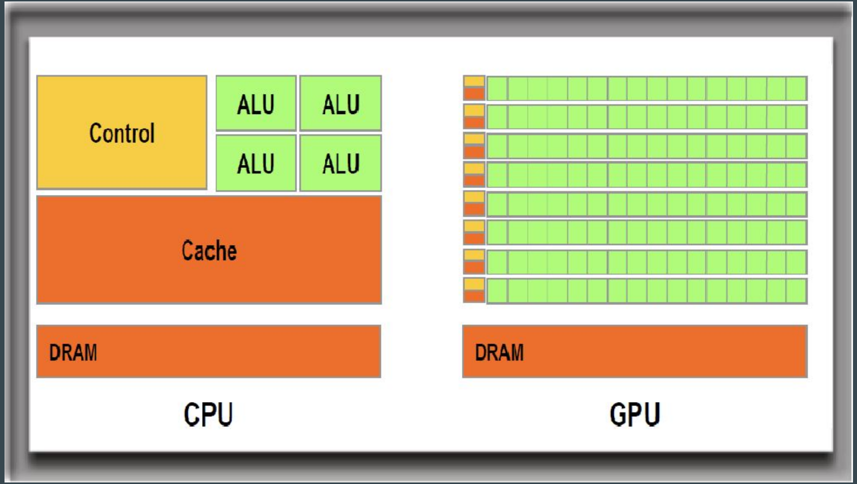
## Machine Learning demands many calculations

```
>>> import torch

>>> a = torch.randn((1024,512))

>>> b = torch.randn((2048,1024))

>>> torch.matmul(b,a)

# Approximately 1B operations!
```

- CPUs have few, high power processing cores
- On a CPU, each product must be calculated sequentially leading to slow processing.
- But each operation is a simple instruction, so can this be sped up?
- Can we delegate processing to many smaller processing cores?

## What is a GPU?

- GPUs enable rapid parallel processing of operations.
- Many small cores working in parallel rather than a few large CPU cores.
- Useful for graphical tasks and gaming but now a must-have tool for NLP and AI



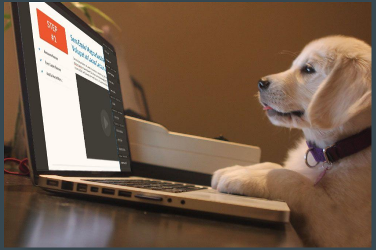
## Working with GPUs

```
>>> import torch
>>> a = torch.randn((1024,512))
>>> b = torch.randn((2048,1024))
>>> torch.matmul(b,a)
7.39s to compute 1000x
>>> a = a.cuda()
>>> b = b.cuda()
>>> torch.matmul(b,a)
2.36s to compute 1000x
```

- We use the NVIDIA CUDA interface to integrate GPUs into our code.
- All our code today is in PyTorch which plugs into CUDA without us writing GPU instructions. Other options exist if desired.
- Write code as normal then move matrices to the GPU for speed up.
- GPUs have their own memory.
  - Small models fit entirely on a GPU
  - Or process data through a GPU model in batches.
  - Or distribute your model across multiple GPUs

## Workflow of using a GPU in NLP

CPU runs main process for model



you write code

input

your data

.cuda()

GPU does heavy lifting

output

model + prediction

## What is a cluster?

## What is a cluster?

- An arrangement of servers to execute computationally intensive work on dedicated high-performance machines in the background.

- You log into the head node, format your experiments and then submit scripts as “jobs”.

- Your jobs are assigned a compute node (with a GPU) which runs your script and accesses a shared or local file system for data.

- Jobs are assigned, managed and controlled using a scheduler program.
  Informatics uses the Slurm scheduler.

## Why use a cluster?

### Single GPU experiments

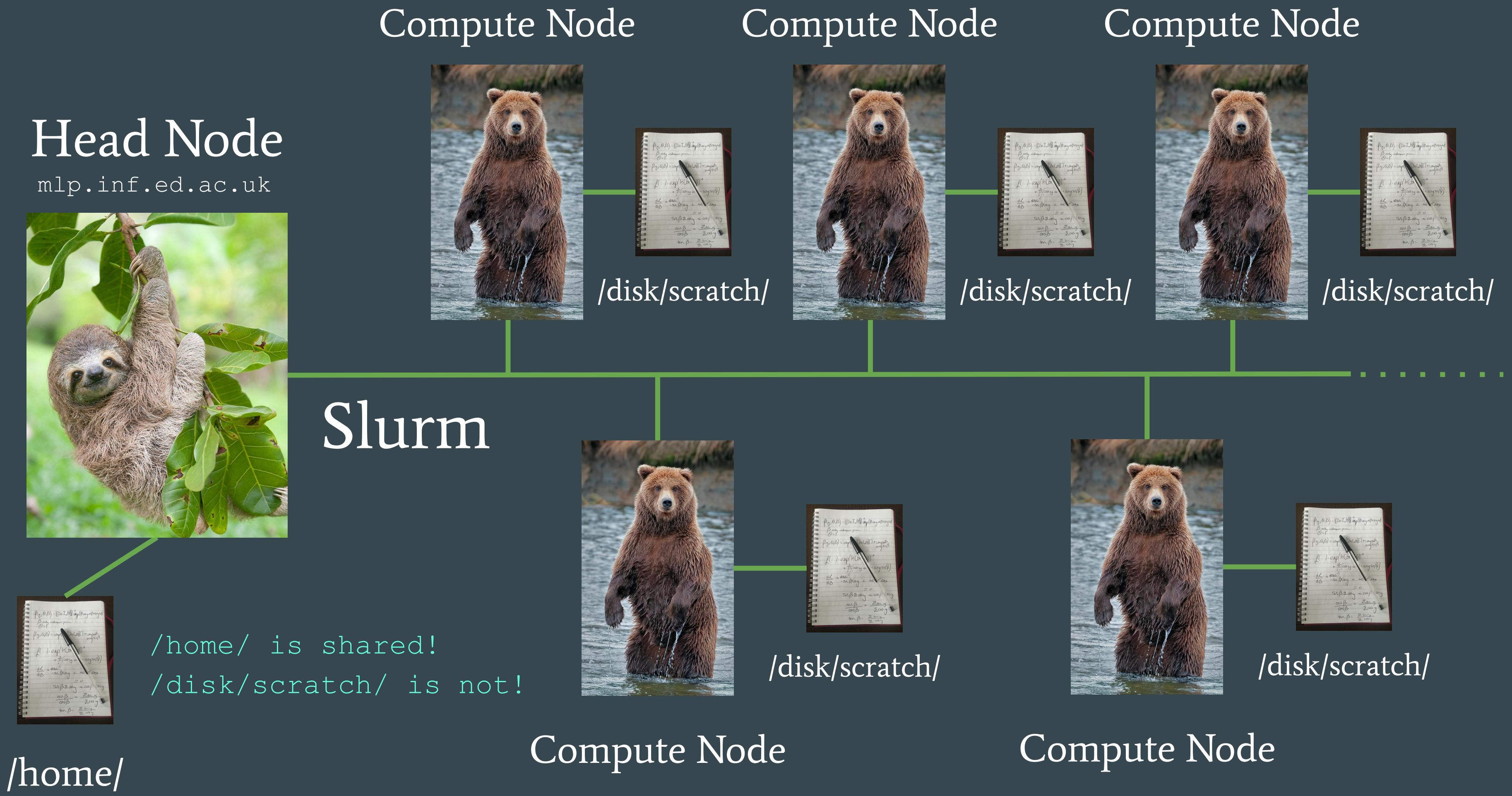
- ✅ Debug models during development with direct shell access to model e.g. using PDB
- ❌ GPU also required to run monitor and other processes.
- ❌ One experiment at a time.
- ❌ Computer possibly not usable during experiments.

### Cluster experiments

- ❌ No direct access to shell. Hard to debug errors .
- ✅ GPU dedicated to your experiment.
- ✅ Run many parallel experiments.
- ✅ Sharing GPUs maximises usage without grinding your own PC to a halt.

# What do we have in Informatics?

- Head node: `mlp.inf.ed.ac.uk`; Lustre file system with lots of space
- ILCC partitions:
  - 48 GPUs in in the `ILCC-Standard` partition
  - 16 GPUs in the `ILCC-CDT` partition
  - A combination of NVIDIA RTX2080 Ti / NVIDIA RTX1080 Ti cards with 11GB memory.
- PGR partitions:
  - `damnii[01-12]` nodes in the `PGR-Standard` partition.
  - Each node has 8 NVIDIA RTX 2080 Ti GPUs with 11GB memory.
  - `crannog[01-07]` nodes in the `PGR-Standard` partition.
  - Each node has 4 NVIDIA A40 GPUs with 48GB memory.
- Alternative: EIDF GPU cluster; more GPUs, bigger and faster GPUs. But: uses Kubernetes as control infrastructure; Slurm is a lot easier to use.



/home/ is shared!
/disk/scratch/ is not!

## Disk space on the cluster

- Like on DICE, you will have a home folder as `/home/${USER}/`
- Move data between machines using `rsync` or `scp`.
- Your user space is on a Lustre file system that all nodes can access.
  - The file system is large but comparatively slow.
  - There are no backups! Got important work? Copy it out of the cluster.

- Each compute node has a local disk drive at `/disk/scratch/`
  - This is fast to read and write to during an experiment.
  - Save weights and large files here during training.
    - Copy what you need back to your user space at the end.
    - Delete everything else you haven't stored from here at the end.

## What is Slurm?

- Slurm is an open source scheduler that controls the allocation and execution of jobs on our cluster.

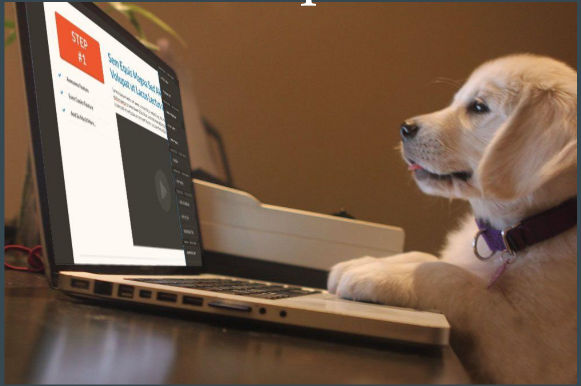
- You write your experiment script then...
  - You submit your script to the Slurm controller while logged into the head node.
  - Slurm finds an available compute node and assigns resources to execute your script.
  - You can monitor your job output and status using Slurm monitoring commands.
  - No free compute node? Slurm places your jobs in a queue to execute when a GPU is free.

## Slurm commands

- `sbatch` - submit a job for hands-off execution on the cluster.
- `srun` - request an interactive shell session on a compute node (for debugging)
- `squeue` - check the execution of your jobs and the queue of waiting jobs
- `sinfo` - check cluster information
- `scontrol` - update job configuration (won't be covering today)

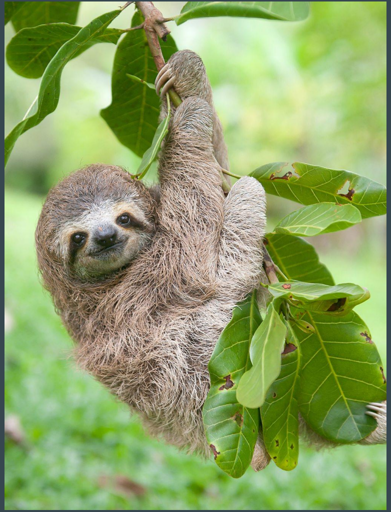
# Use ssh to access the head node

Local computer



ssh

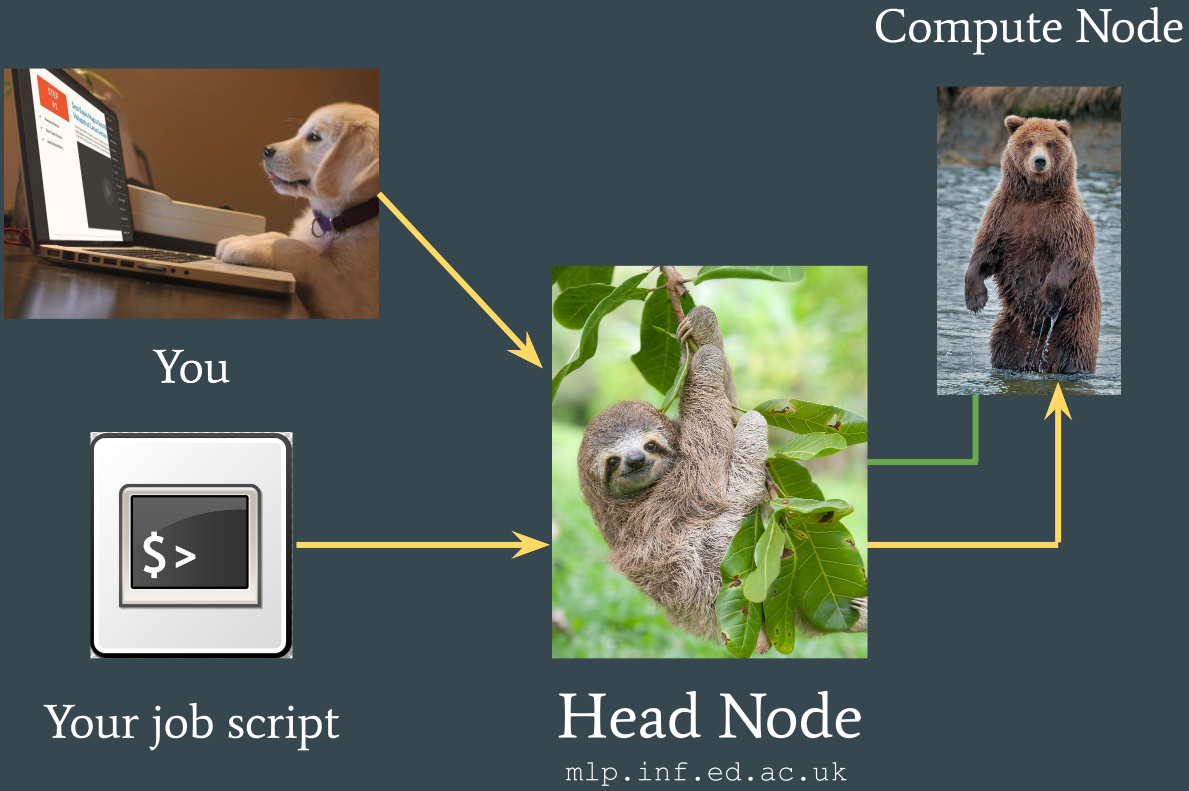
Head Node



For example:

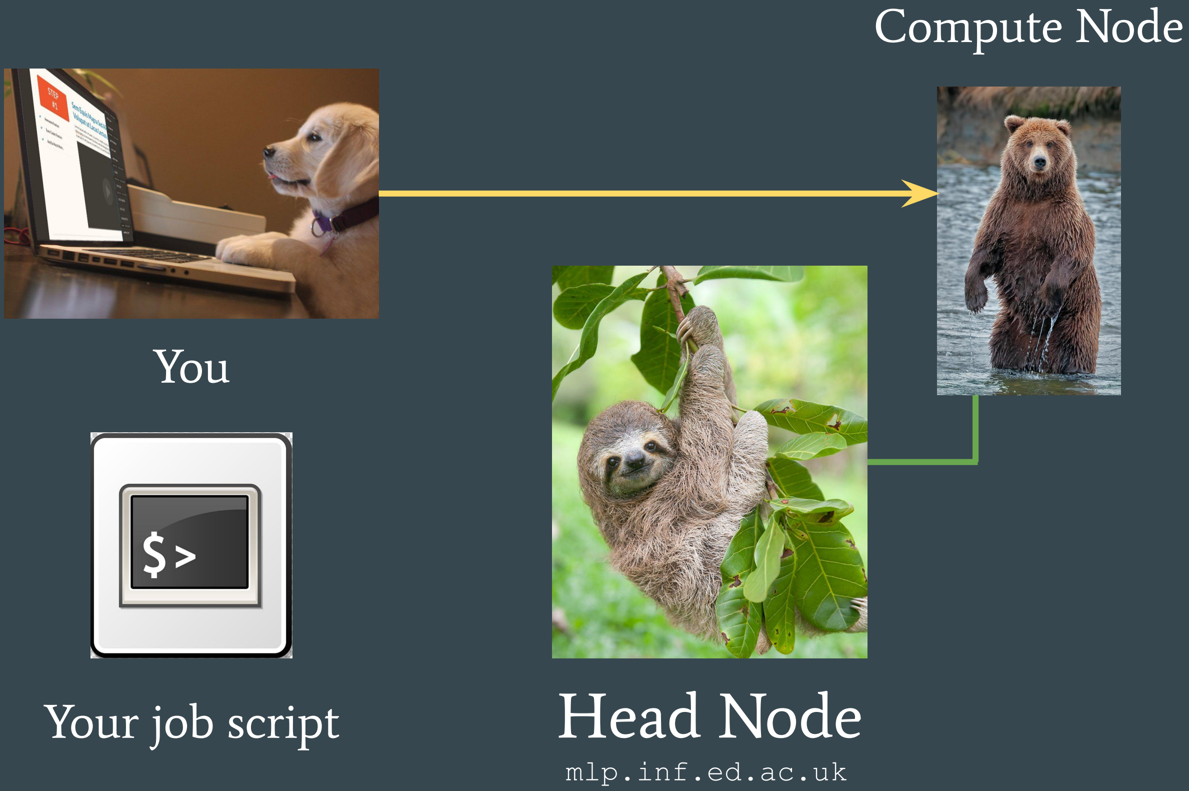
```
    ssh ${USER}@mlp.inf.ed.ac.uk
    ssh ${USER}@${cluster_name}.inf.ed.ac.uk
```

# sbatch



- You ssh onto the head node.
- Submit your job using `sbatch`.
- Slurm assigns the job to a compute node and then executes the job in the background.

## `srun`



- Slurm assigns you an interactive session on the compute node (like ssh)
- Useful if your job is going wrong somewhere/debugging.
- No automatic processing and job is not a background process.

# Comparing sbatch and srun

## `sbatch`

- ✅ Your experiment runs as a background process without direct supervision.
- ✅ Run all your experiments in parallel on compute nodes.
- ✅ The intended use case for cluster computing.
- ✅ Go home and rest. Your work is happening while you sleep!

## `srun`

- ✅ Gives you an ssh-like session on a compute node. Useful if something has gone wrong and you need to check your model on the cluster.
- ❌ Hoards GPU resources if used excessively.
- ❌ The cluster becomes less useful and effective.
- ❌ Encourages poor experiment design and babysitting your jobs.

## Everything all together...

- Assume that experiments are **bash scripts** that specify all steps of computation.
- We access a cluster by sshing on to the **head node**.
- Submit an experiment job using `sbatch` to request a **compute node** to run the job.
- **Slurm** manages the allocation, execution and running of jobs.

## Cluster Workflow

## Anatomy of an `sbatch` script

```
#SBATCH Args here....
```

```
conda activate pt
```

```
rsync data /home/ to /disk/scratch/
```

```
python train.py
```

```
python predict.py
```

```
rsync results /disk/scratch/ to /home/
```

```
rm -rf /disk/scratch/${USER}/exp
```

You will need...

- Slurm configuration
- A Python environment
- Training and test data
- The model to train (`model.py`)
- Training command (`train.py`)
- Prediction command (`predict.py`)

## Conda Environments

- Miniconda provides isolated runtime environments for your Python code. This manages your packages so you can be sure what dependencies you are using in your programme.

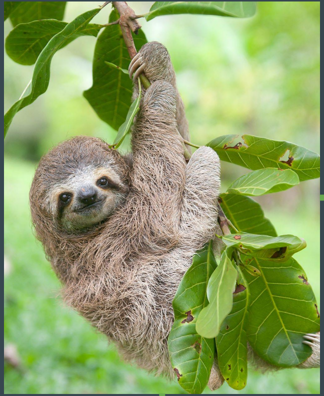
- Install a specification of packages to an environment and use it for all your experiments!

- Different experiments have different specifications? Use a new environment!

- We will install tools such as PyTorch in an environment.

# Workflow: data

Head Node



1

input data —> compute node scratch

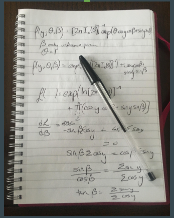
/home/ <— results from compute node scratch

3

Compute Node

training and testing

2



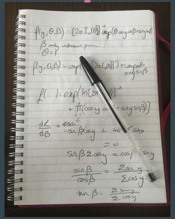
git + results

miniconda

input data

logs

/home/



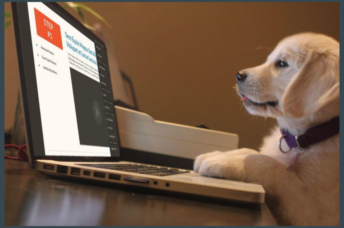
results

input data

scratch

# Workflow: code

## Local Computer

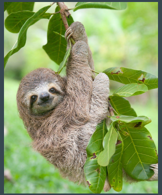


1. Write your code and get it working with a conda virtual environment

2. Version control your code with git and put it in a repository online with GitHub

ssh

## Head Node



3. Download your code by cloning repo from GitHub

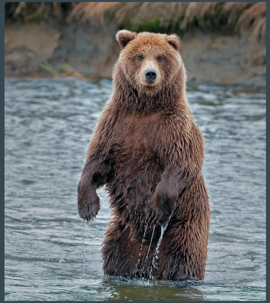
4. Create/activate conda environment

5. Get your input data onto the file system e.g. scp, rsync

8. Run your jobs with sbatch

Slurm

## Compute node



6. Test your code on an srun interactive session

7. Last minute code edits on command line editors like vim or emacs

## Need more power?

- You can request more than 1 GPU using the `gres` argument to `sbatch`
  - `--gres=gpu:1` requests 1 GPU
  - `--gres=gpu:2` requests 2 GPU on 1 compute node...
  - Make sure your configuration can fit into the cluster
  - You will probably need to adjust the TCP port for your job

- Need even more power?
  - It is possible to request multiple nodes.
  - This will make your experimental setup more complicated
  - Bear in mind the Informatics cluster is not the best place to retrain Llama...

## Now what?

## Getting help

- `#computing` channel in the CDT in NLP slack
  - Peer support from other cluster users
  - Also useful if you want to help other people out!
- Ask your research group
  - Likely any senior-ish PhD students have got the hang of the cluster.
  - Most students are happy to help share their knowledge.
- Submit Tickets to Computing Support 🎫
  - Try and be as specific as possible.
    What do you think is the error?
    Is it reproducible?
    What Slurm job # caused this?

## Cluster etiquette

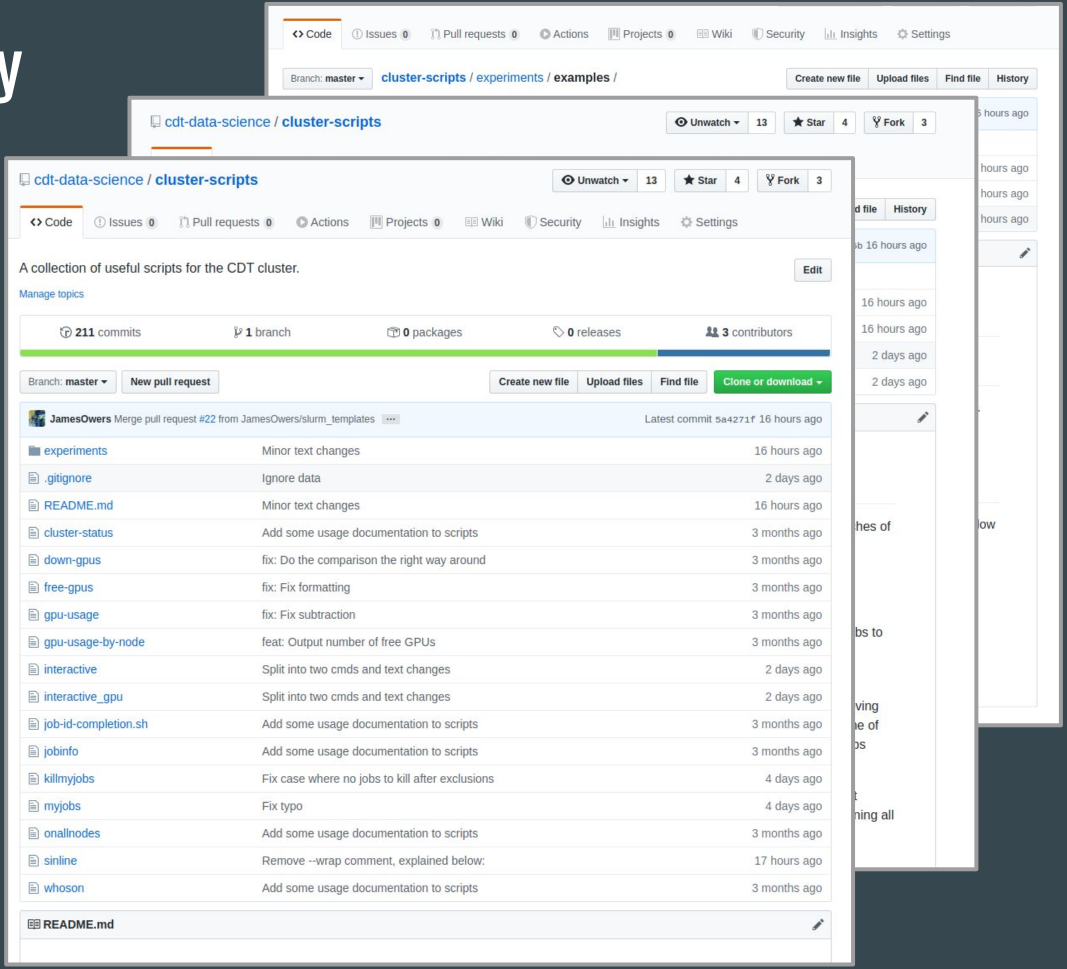
- Be nice!
- Running a lot of jobs? Consider staggering so many users can use the queue
- Or use Array jobs (not covered today but included in the demo)
- If you see someone misbehaving then consider emailing them (they may be unaware)
- Similarly, another user may notify you if they see a process of yours acting improperly (e.g. running Python on the head node)

# The cluster-scripts repository

Repo here:
https://github.com/cdt-data-science/cluster-scripts

1. scripts to make your life easier
2. examples for quick learns
3. templates for running experiments fast

We will use this in today's demonstration!



cdt-data-science / cluster-scripts

A collection of useful scripts for the CDT cluster.

211 commits · 1 branch · 0 packages · 0 releases · 3 contributors

JamesOwers Merge pull request #22 from JamesOwers/slurm_templates … Latest commit 5a4271f 16 hours ago

| | | |
|---|---|---|
| experiments | Minor text changes | 16 hours ago |
| .gitignore | Ignore data | 2 days ago |
| README.md | Minor text changes | 16 hours ago |
| cluster-status | Add some usage documentation to scripts | 3 months ago |
| down-gpus | fix: Do the comparison the right way around | 3 months ago |
| free-gpus | fix: Fix formatting | 3 months ago |
| gpu-usage | fix: Fix subtraction | 3 months ago |
| gpu-usage-by-node | feat: Output number of free GPUs | 3 months ago |
| interactive | Split into two cmds and text changes | 2 days ago |
| interactive_gpu | Split into two cmds and text changes | 2 days ago |
| job-id-completion.sh | Add some usage documentation to scripts | 3 months ago |
| jobinfo | Add some usage documentation to scripts | 3 months ago |
| killmyjobs | Fix case where no jobs to kill after exclusions | 4 days ago |
| myjobs | Fix typo | 4 days ago |
| onallnodes | Add some usage documentation to scripts | 3 months ago |
| sinline | Remove --wrap comment, explained below: | 17 hours ago |
| whoson | Add some usage documentation to scripts | 3 months ago |

README.md

## Common mistakes

- Conda environment not set up properly to use a GPU.
  - Check `torch.cuda.is_available()==True` in an interactive session.
- Training fails due to Out Of Memory errors.
  - Consider adjusting batch sizes to reduce peak GPU memory.
  - Or reformat your model to use multiple GPUs.
- Nothing happens when I submit using `sbatch`?
  - Check your SBATCH arguments. SBATCH will fail **silently** if the arguments contain an error.
- My job stops after a few seconds
  - The `/disk/scratch/` of a compute node might be full. Identify the node and submit a ticket!

## Demonstration

...