

A white wolf is the central focus, walking towards the viewer in a snowy, futuristic cityscape. The scene is bathed in a cool blue light, with vertical beams of light and digital artifacts scattered throughout. The wolf's fur is detailed and appears to be blowing in a light breeze. The background features a grid of light and dark rectangular shapes, suggesting a digital or architectural environment.

LUMI

Using the LUMI
web-interface

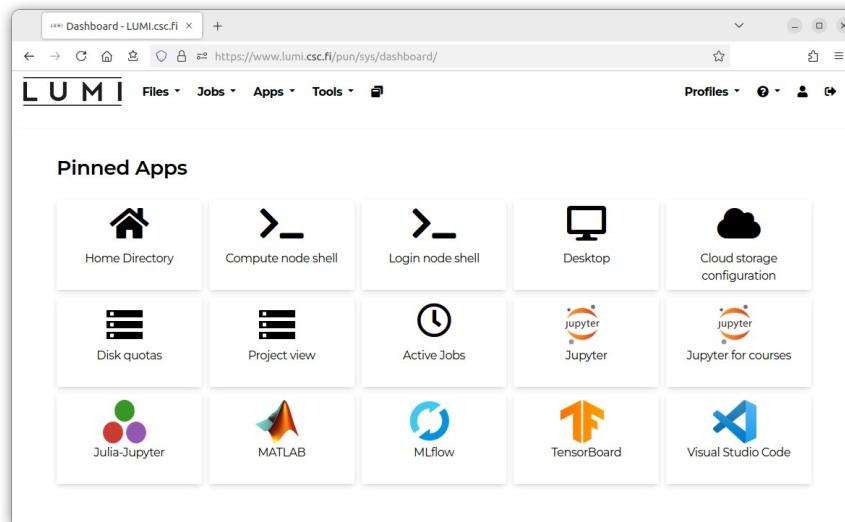
Mats Sjöberg, Lukas Prediger – CSC – IT Center for Science, Finland

Two ways of accessing LUMI



- via SSH connection
 - terminal access only

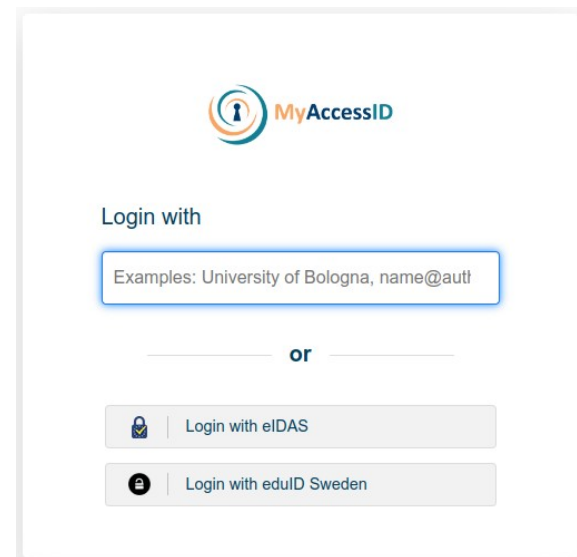
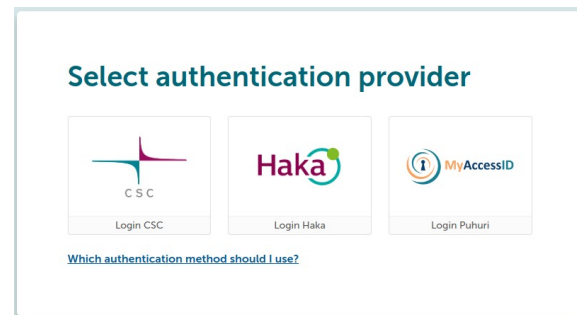
- via the LUMI web-interface
 - browser based
 - terminal + various apps



Accessing the LUMI web-interface

- Go to <https://www.lumi.csc.fi>
- Click “Go to login”
- Select login method
 - In most cases you should select MyAccessID / Puhuri
- MyAccessID: type the name of your institution / university
- You should be directed to your institution's login page

LUMI



Pinned Apps



Home Directory



Compute node shell



Login node shell



Desktop

Cloud storage
configuration

Disk quotas



Project view



Active Jobs



Jupyter



Jupyter for courses



Julia-Jupyter



MATLAB



MLflow



TensorBoard



Visual Studio Code

Files browser and shell (login node)

LUMI



Home Directory



Login node shell

The screenshot shows the LUMI Files browser interface in a web browser. The address bar shows the URL: `https://www.lumi.csc.fi/pun/sys/dashboard/files/fs/users/mvsjober`. The interface includes a navigation bar with "Files", "Jobs", "Apps", and "Tools". A message states: "The web interface has been updated to release 3. MATLAB and VisIt are now available in the Desktop app. Additionally, the web version of MATLAB is also available as an interactive app." Below this is a toolbar with buttons for "Open in Terminal", "Refresh", "New File", "New Directory", "Upload", "Download", "Copy/Move", and "Delete". The main content area shows a file tree on the left with "Home Directory" expanded, listing various project directories under "/projappl/" and "/scratch/". The main pane shows the current directory: `/ users / mvsjober /`. A table lists files and directories with columns for Type, Name, Size, and Modified at.

Type	Name	Size	Modified at
Folder	appl_sync_logs	-	29/04/2024 10:26:10
Folder	bin	-	01/03/2024 09:32:01
Folder	code	-	19/03/2024 09:25:23
Folder	Desktop	-	07/09/2023 12:18:16
Folder	Documents	-	07/09/2023 12:18:21
Folder	Downloads	-	07/09/2023 12:18:21

The screenshot shows the LUMI Login node shell terminal. The browser address bar shows the URL: `https://www.lumi.csc.fi/pun/sys/shell/ssh/default/users/mvsjober`. The terminal output shows the user `uan09` at the host `uan09.can` in the initial directory `users/mvsjober`. The user has executed the `ls` command, listing the contents of the current directory:

```
uan09:/scratch/project_465801063/mvsjober/Getting_Started_with_AI_workshop $ ls
01_Introduction_to_LUMI/          05_Running_containers_on_LUMI/      09_Hyper-parameter_tuning_us
02_Using_the_LUMI_web_interface/  06_Bulding_containers_from_conda_pip_environments/  10_Extreme_scale_AI/
03_Your_first_AI_training_job_on_LUMI/  07_Virtual_environments_to_iterate_and_test/      11_Coupling_AI_and_HPC/
04_Workarounds_and_checking_jobs/      08_Scaling_to_multiple_GPUs/         bonus_material/
uan09:/scratch/project_465801063/mvsjober/Getting_Started_with_AI_workshop $
```

Jupyter notebook



- Launching Jupyter actually launches a job in the cluster (more on that in the next lecture)
- You need to fill in some things:
 - Project
 - Partition
 - Resources
 - What Python installation to use
- Max resources for a single GPU job
 - 1/8 of a node:
 - Number of CPU cores: 7
 - Memory: 60 GB

Jupyter

Interactive Jupyter session

[Documentation](#)

Project

project_465001063

Partition

dev-g

The selected partition will reserve 1 GCD (MI250).

Resources

Number of CPU cores

7

SMT is enabled for the selected partition. 2 threads per core will be allocated.

Memory (GB)

60

Time

3:00:00

d-hh:mm:ss, or hh:mm:ss

Jupyter notebook



LUMI

- The job is submitted to the normal Slurm queue
- Once it has started “Connect to Jupyter” button will appear

Jupyter (7102485) 1 node | 14 cores | Running

Host: [>_ nid005025](#) Cancel

Created at: 2024-05-13 07:40:28 UTC

Time Remaining: 2 hours and 54 minutes

Session ID: 34afe7b3-65b1-4c8c-8c17-64840910f8ba

If you run into issues, please include the following log file in the support ticket: [output.log](#)

Project: project_465001063
Partition: dev-g
Cores: 14
Memory: 61440M
GPUs (MI250): 1
Module: pytorch

[Connect to Jupyter](#)

Note: device called "cuda" even though it's AMD GPU – for "historical reasons"

GPT-neo-IMDB... - Jupyter x +

https://www.lumi.csc.fi/node/nid005002/3053/lab/tree/mvsjober/Getting_Started_with_AI_workshop/02_Using_the_LUMI_web

File Edit View Run Kernel Git Tabs Settings Help

Launcher x GPT-neo-IMDB-introduction x +

Python 3 (ipykernel)

Introduction to the Ongoing Example

In this notebook you will get to know the example machine learning task we will consider for most of the exercises throughout the course: We will finetune the GPT-neo language model by EleutherAI on the Stanford IMDB movie review data set to obtain a model specialised in generating movie reviews.

Since both the model and the data set are available from huggingface.co, we will use the libraries provided by HuggingFace, which present a slightly higher level abstraction of training with PyTorch.

This notebook does not yet perform any training but demonstrates loading the model and allows you to perform inference, i.e., generating some text with it. It also loads the training data set for you to explore.

We begin by loading the required Python modules...

```
[1]: import torch
import os
from datasets import load_dataset
from transformers import AutoModelForCausalLM, AutoTokenizer
```

...and determining the device on which to run the model. Even though LUMI uses AMD MI250x GPUs, PyTorch still use `cuda` when we mean "GPU". The following should print: "Using device: cuda". If this is not the case, then we have made a mistake in allocating resources for the job or loading the proper software environment.

```
[2]: device = torch.device('cuda') if torch.cuda.is_available() else torch.device('cpu')
print(f"Using device: {device}")
```

Using device: cuda

```
[3]: print(torch.cuda.get_device_name(0))
```

AMD Instinct MI250X

Meet the Pre-Trained Base Model

Now we load the model using the `transformers` module. First we set an environment variable to point to a shared cache directory which `transformers` uses when loading the model, so it does not have to download the same model repeatedly:

Simple 0 8. 1 main Python 3 (ipykernel) | Idle Mode: Command Ln 1, Col 1 GPT-neo-IMDB-introduction.ipynb 0

Limitations of interactive Jupyter runs

- Using GPUs interactively is *inefficient usage of resources*
 - Most of the time, when you are editing code or grabbing a coffee, the GPU is idle (but nobody else can use it!)
 - Because of this interactive use is limited to a single GPU
- Running multiple copies of your job (e.g., hyperparameter search) is not possible
- Solution:
 - Use Jupyter for development and experimentation
 - Use terminal interface for real runs

→ more on this in the next session

Our running example for this course

- Finetuning GPT-Neo LLM for generating movie reviews on the IMDb data set
- Using Hugging Face's datasets and transformers on top of PyTorch as training library

GPT-Neo 1.3B

- 1370 M parameters
- BF16
- ~2.67 GB

Stanford IMDb data set

- 100 000 movie reviews
- Varying lengths (low hundreds of words)
- 25 000 reserved for testing

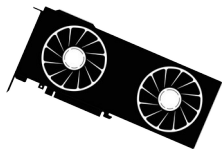


Our running example for this course

LUMI

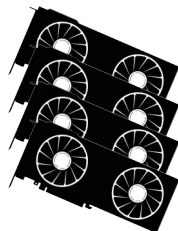


02: Get familiar with LUMI web interface and the example



03: Using Slurm scheduler to train on a single GPU

04: Checking on training jobs and some common problems



08: Training on multiple GPUs (on a single GPU node)

09: Training across multiple nodes

Course practicalities: Reservations

- A portion of the cluster is reserved for the course
- When starting jobs you need to give the reservation names, otherwise you apply for resources in the general queues
 - in the web-interface, select from the drop down menu
- Day 1: AI_workshop
- Day 2: AI_workshop_2
- However, if you use a reservation that is not active, your job will not run until the reservation becomes active.