LLM Infrastructure on HPC: Workflows, Constraints, and Solutions (Text Lab)

Ahmad Alhineidi
Data Science Lab (DSL)
29.06.2026
PASC 26
MS2C – Serving Inference: Leveraging HPCs in the Age of Generative AI

DSL

UNIVERSITÄT BERN

Content

- University of Bern HPC setup
- Text Lab overview & usage (Live demo)
- GPUs and LLMs choice

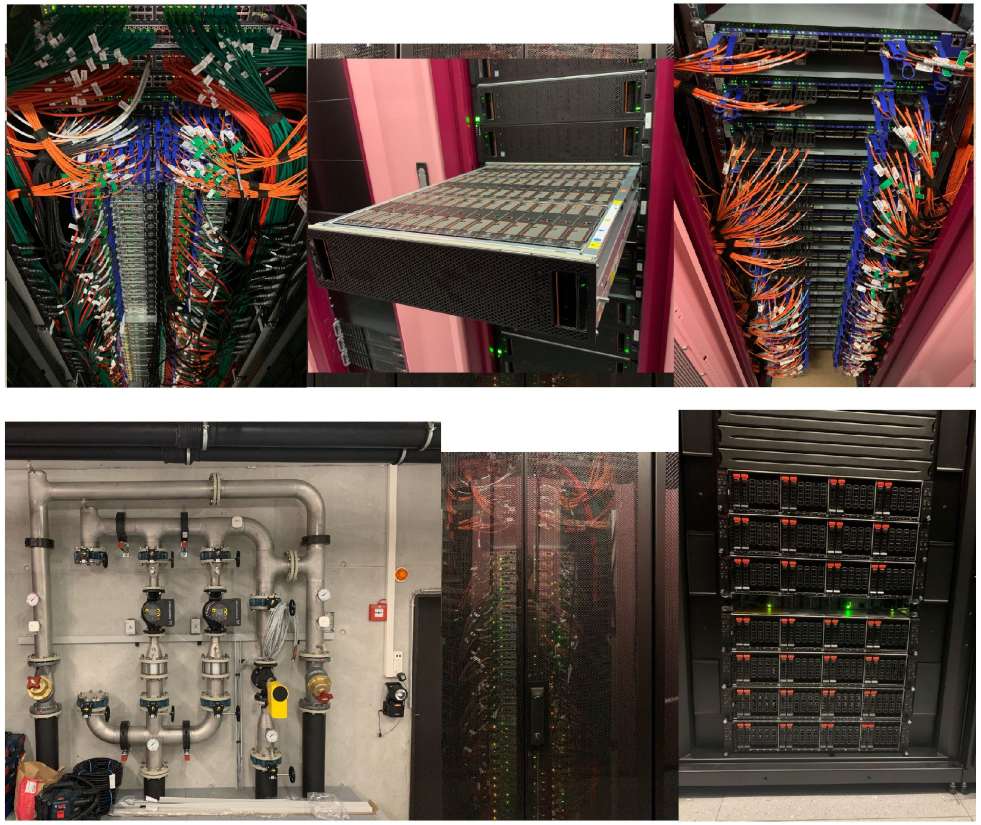
Image source: link

DSL

UNIVERSITÄT BERN

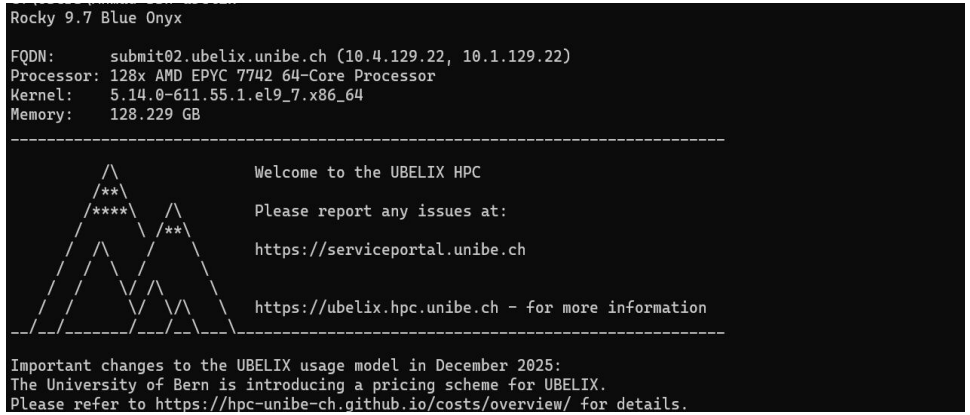
UniBe HPC - UBELIX

- GPU partition
- 21 GPU nodes · 174 GPUs total
- RTX3090, RTX 4090, A100, H100, H200, RTX PRO 6000 Blackwell
- Most GPU nodes: 128 CPU cores and ~740 GB RAM (rtx4090/h100/h200)



UniBe HPC - UBELIX

- Open OnDemand deployed since 2025
- Improves HPC accessibility, useability
- Allows to easily build custom interactive apps



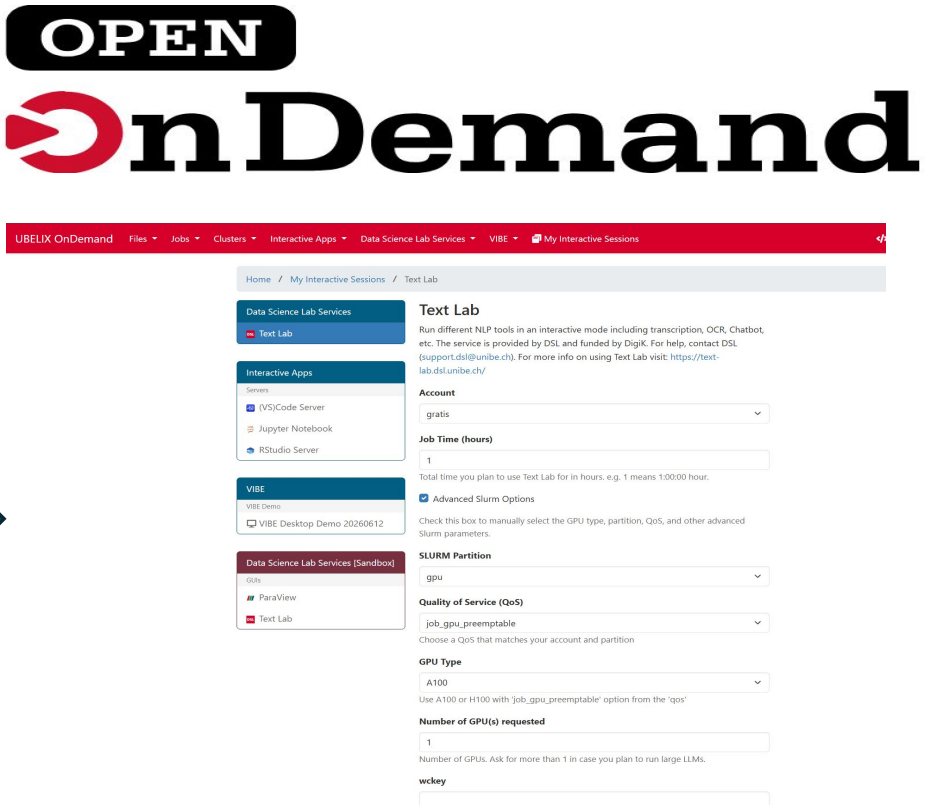
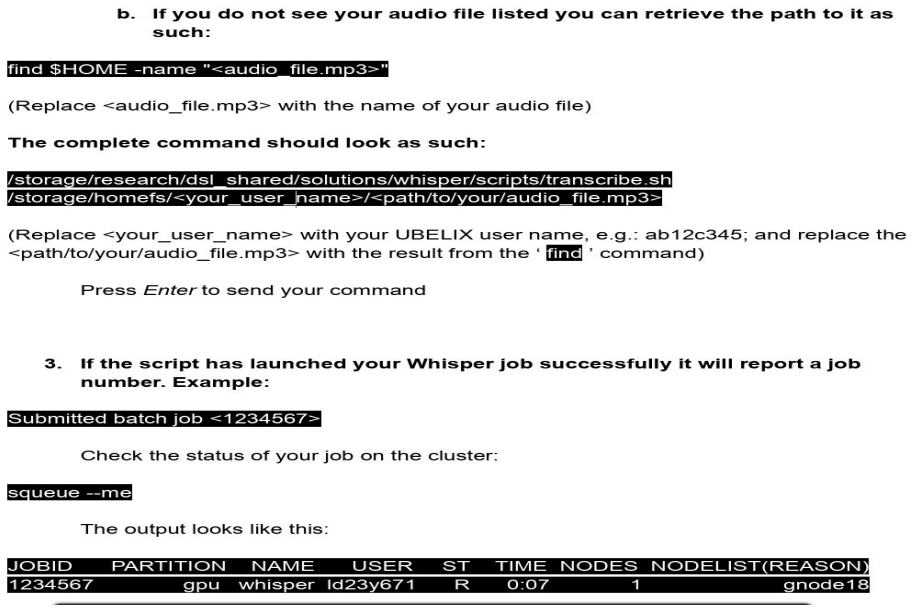


Image source: link

Text Lab - a bit of History

- Whisper container on UBELIX for audio transcription
- Very long Manuel
- Includes command-line interface
- ☹ confusion, more support, where is my data? How do I transfer my data to the cluster? Where is the output?
- Why not just make a User interface? Text Lab



b. If you do not see your audio file listed you can retrieve the path to it as such:

```
find $HOME -name "<audio_file.mp3>"
```

(Replace <audio_file.mp3> with the name of your audio file)

The complete command should look as such:

```
/storage/research/dsl_shared/solutions/whisper/scripts/transcribe.sh
/storage/homefs/<your_user_name>/<path/to/your/audio_file.mp3>
```

(Replace <your_user_name> with your UBELIX user name, e.g.: ab12c345; and replace the <path/to/your/audio_file.mp3> with the result from the ' find ' command)

Press *Enter* to send your command

3. If the script has launched your Whisper job successfully it will report a job number. Example:

```
Submitted batch job <1234567>
```

Check the status of your job on the cluster:

```
squeue --me
```

The output looks like this:

```
JOBID    PARTITION   NAME    USER    ST   TIME  NODES  NODELIST(REASON)
1234567      gpu   whisper  ld23y671   R   0:07      1        gnode18
```

Open-source LLMs

- Managed to match/come close to performance of closed models
- Can download and run locally
- Still requires expensive hardware to serve efficiently
- Quantization and smaller models are cheaper and can perform well in certain tasks

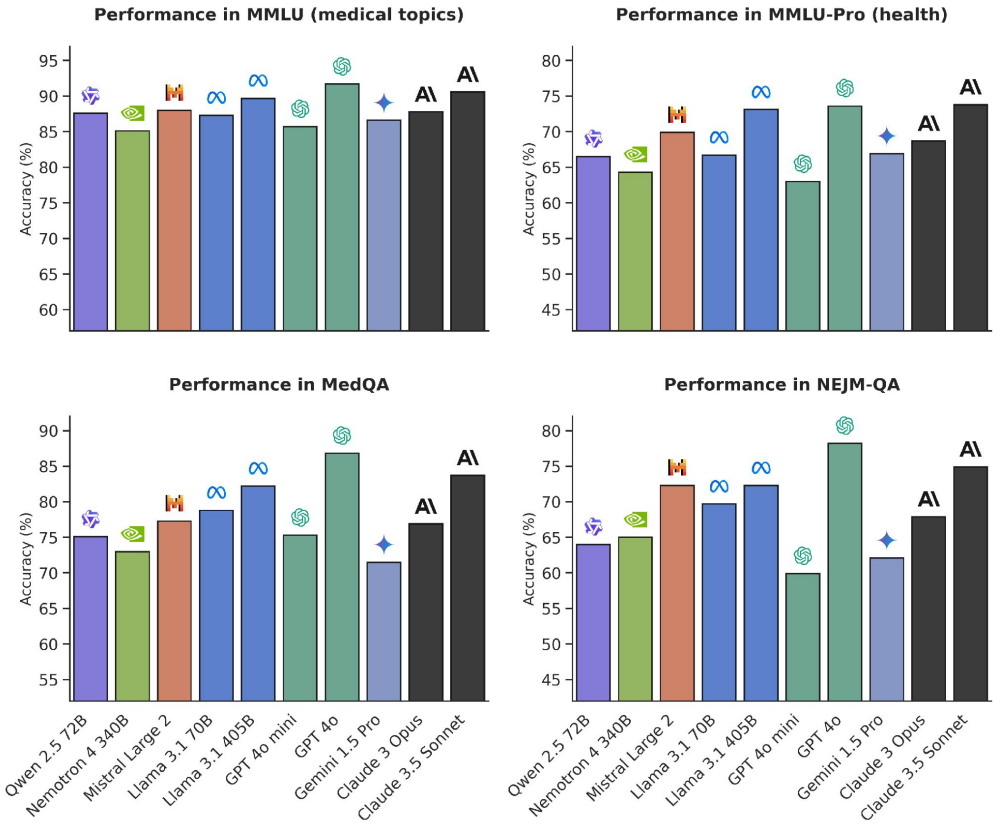


Image source: link

DSL

UNIVERSITÄT BERN

Text Lab - motivation

- Make different “SOTA” models available & easily accessible == zero code
- Data privacy (Runs locally, University infrastructure)
- Experiment with different models / HPC deployment

DSL

UNIVERSITÄT BERN

What is Text Lab?

- Framework with different NLP/Text Processing features
- Runs within the University internal network and infrastructure

What can you do with Text Lab?

- Chat, transcription, OCR, Data visualization, knowledge graphs
- Live Demo
- User's guide: https://text-lab.dsl.unibe.ch/

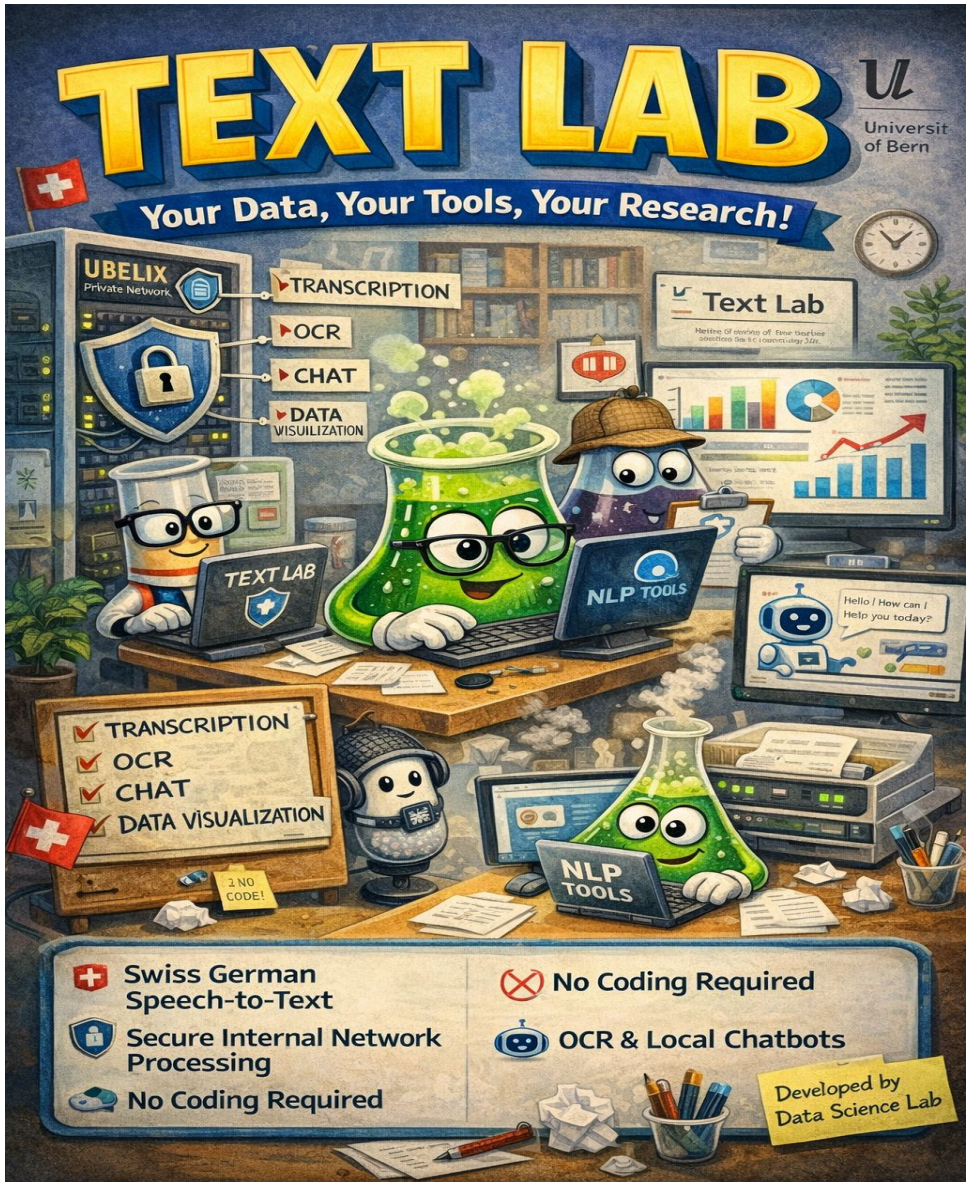


Image source: AI generated, openai

DSL

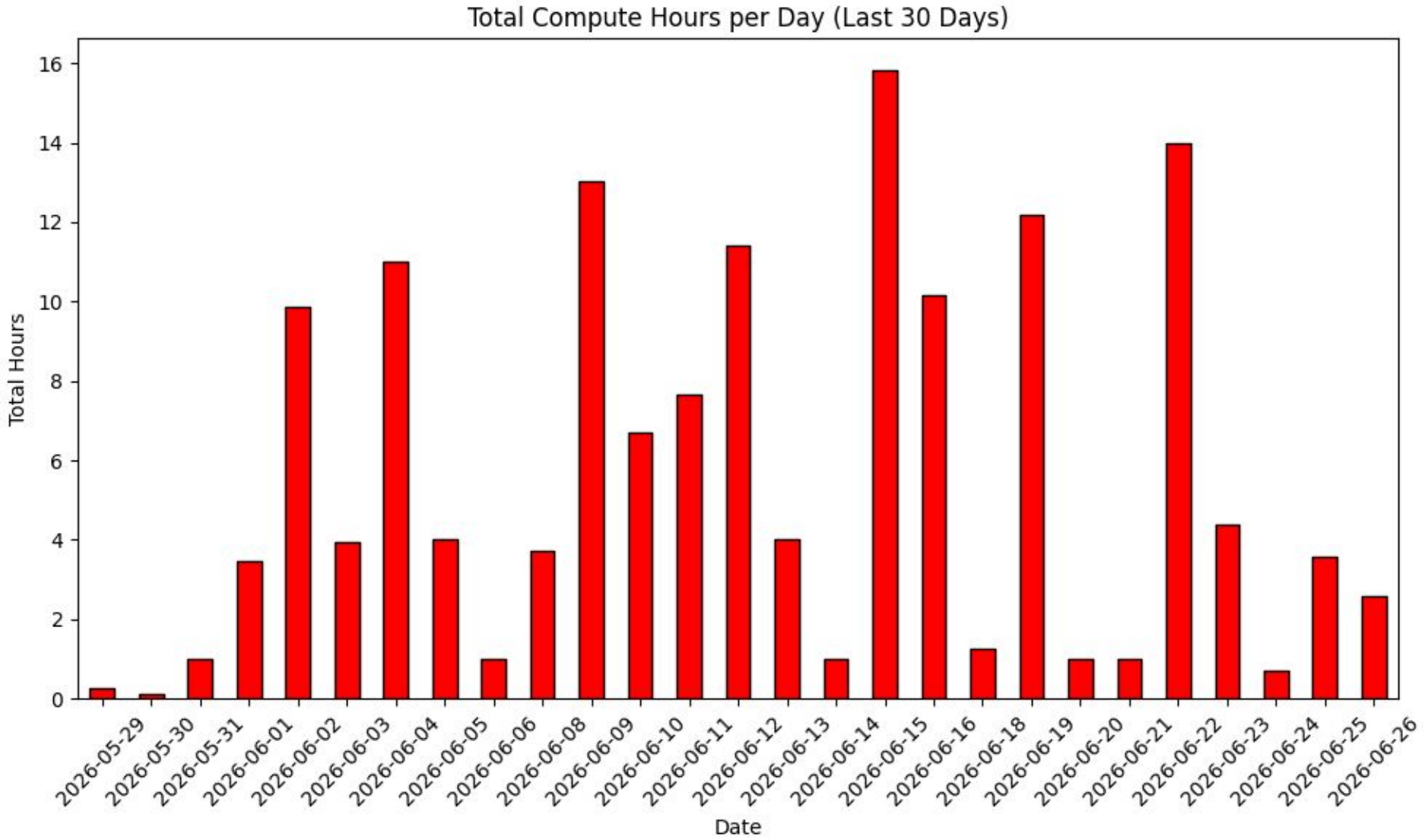
UNIVERSITÄT BERN

Usage

- Anyone really using Text Lab?
- We’re moving from prototype to useful, functional tools
- Next step to know suitable users communicate the tool

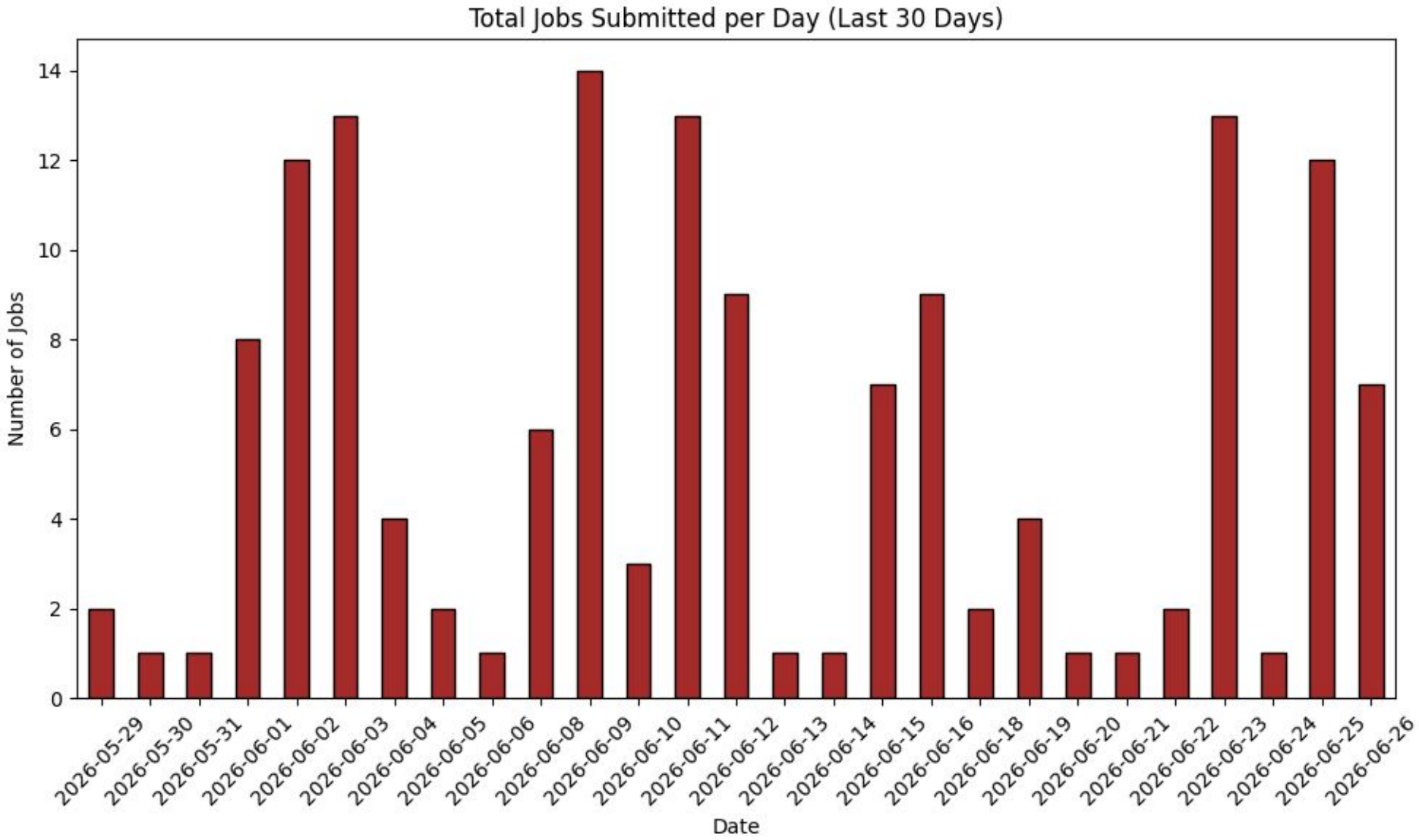
DSL

UNIVERSITÄT BERN



DSL

UNIVERSITÄT BERN



DSL

UNIVERSITÄT BERN

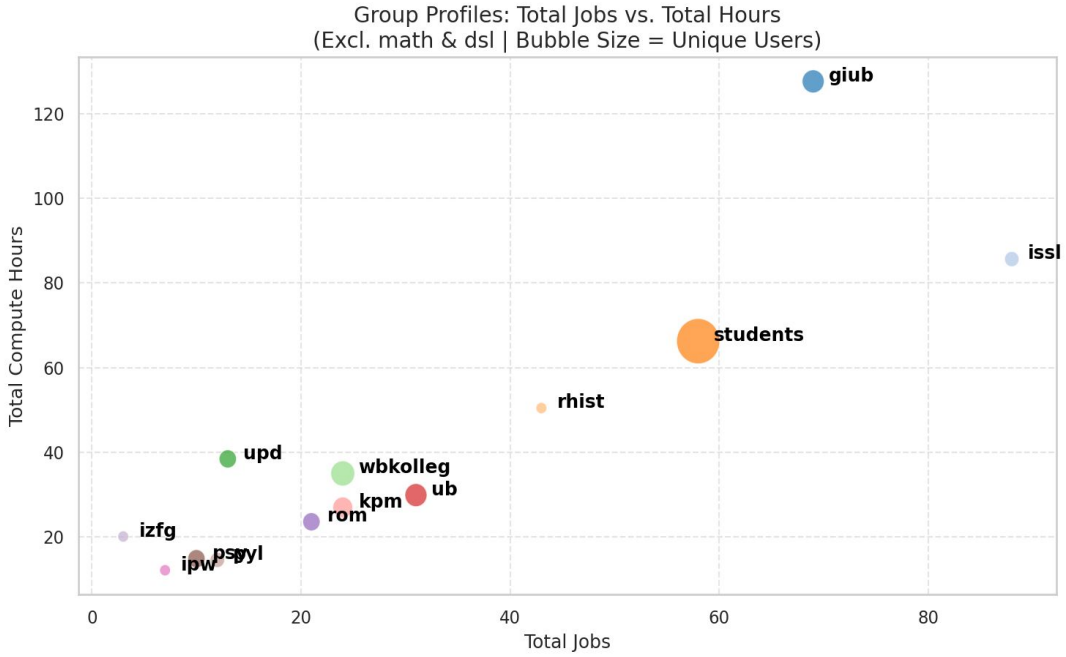
Usage - last 30 days

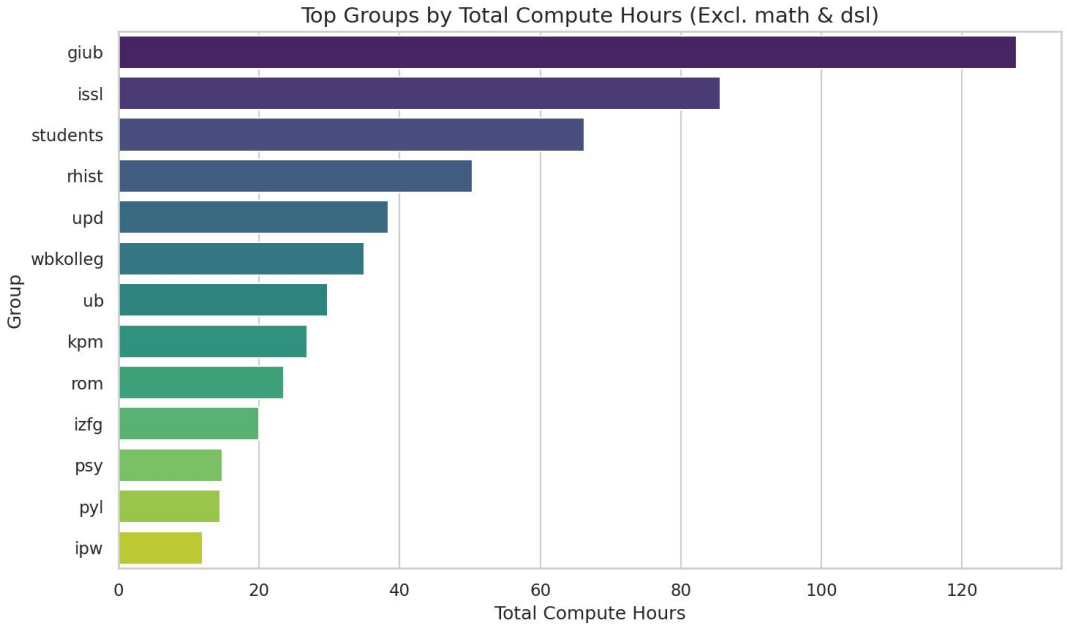
- Average Compute Hours per Day: 4.96
- Average Jobs per Day: 5.63
- Total Jobs Run: 169 (2026-05-29 to 2026-06-26)
- Total Unique Users: 47
- Total Unique Groups: 24

DSL

UNIVERSITÄT BERN

Usage

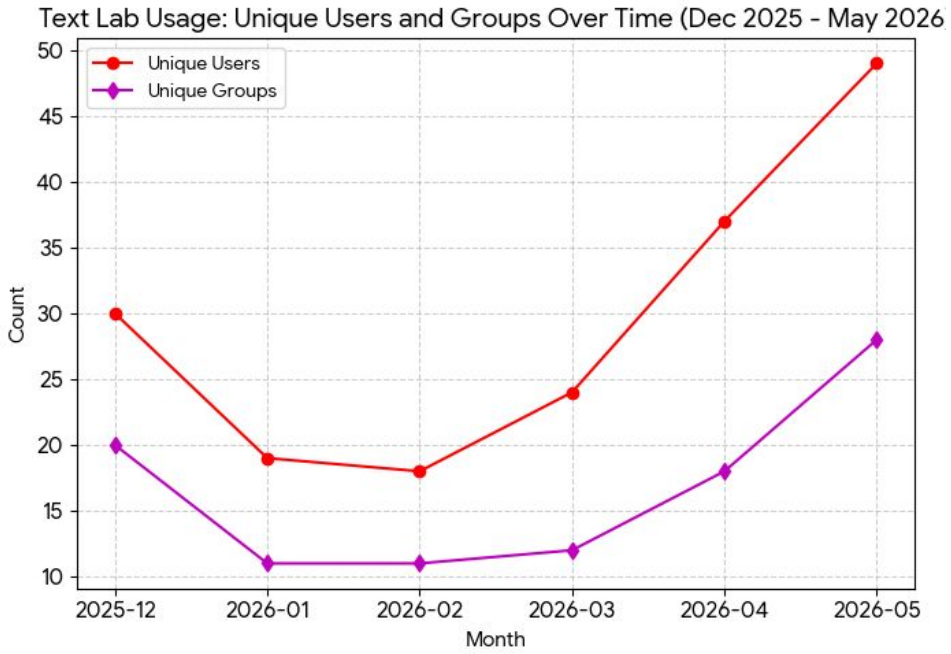


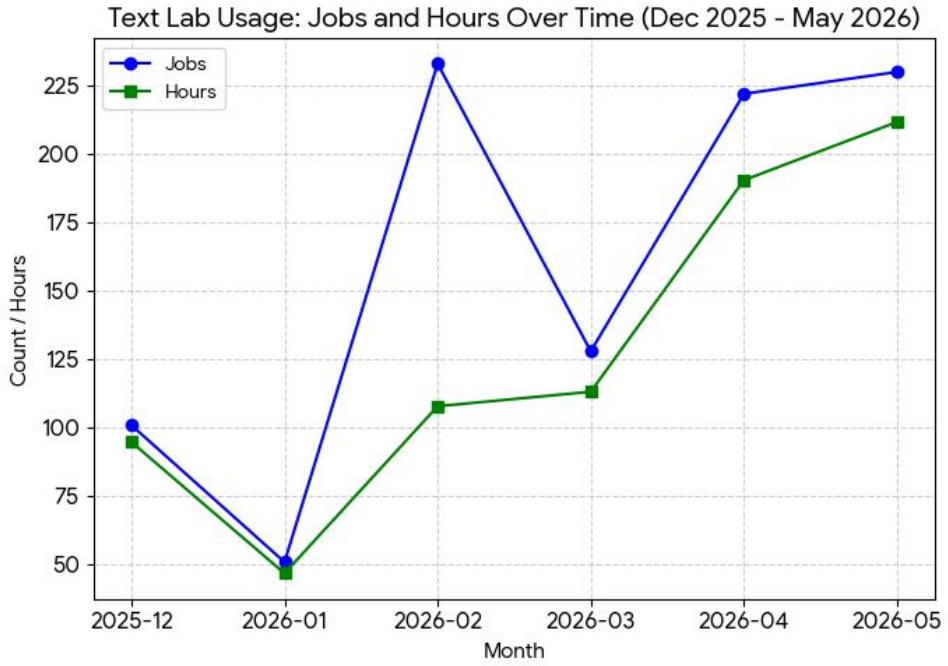


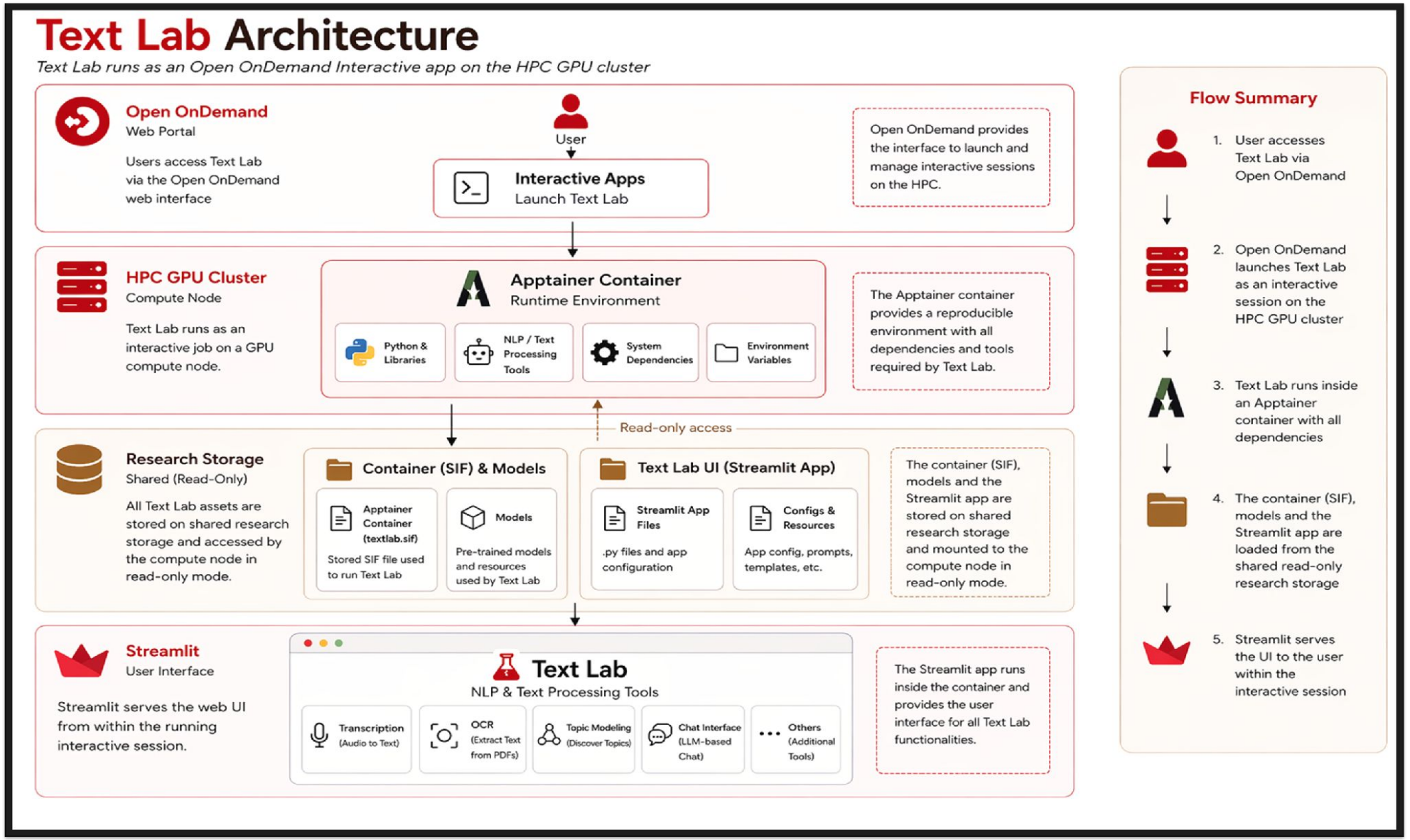
DSL

UNIVERSITÄT BERN

Usage






Text Lab Architecture
Text Lab runs as an Open OnDemand Interactive app on the HPC GPU cluster
Open OnDemand
Web Portal
Users access Text Lab via the Open OnDemand web interface
User
Interactive Apps
Launch Text Lab
Open OnDemand provides the interface to launch and manage interactive sessions on the HPC.
HPC GPU Cluster
Compute Node
Text Lab runs as an interactive job on a GPU compute node.
Apptainer Container
Runtime Environment
Python & Libraries
NLP / Text Processing Tools
System Dependencies
Environment Variables
The Apptainer container provides a reproducible environment with all dependencies and tools required by Text Lab.
Read-only access
Research Storage
Shared (Read-Only)
All Text Lab assets are stored on shared research storage and accessed by the compute node in read-only mode.
Container (SIF) & Models
Apptainer Container (textlab.sif)
Stored SIF file used to run Text Lab
Models
Pre-trained models and resources used by Text Lab
Text Lab UI (Streamlit App)
Streamlit App Files
.py files and app configuration
Configs & Resources
App config, prompts, templates, etc.
The container (SIF), models and the Streamlit app are stored on shared research storage and mounted to the compute node in read-only mode.
Streamlit
User Interface
Streamlit serves the web UI from within the running interactive session.
Text Lab
NLP & Text Processing Tools
Transcription (Audio to Text)
OCR (Extract Text from PDFs)
Topic Modeling (Discover Topics)
Chat Interface (LLM-based Chat)
Others (Additional Tools)
The Streamlit app runs inside the container and provides the user interface for all Text Lab functionalities.
Flow Summary
1. User accesses Text Lab via Open OnDemand
2. Open OnDemand launches Text Lab as an interactive session on the HPC GPU cluster
3. Text Lab runs inside an Apptainer container with all dependencies
4. The container (SIF), models and the Streamlit app are loaded from the shared read-only research storage
5. Streamlit serves the UI to the user within the interactive session

DSL

UNIVERSITÄT BERN

LLMs Choice

- Open-source models range from less than 1 B □ more than 600B (~404GB model weights)
- **Hardware requirement example 1**: Qwen 3.5 122B (81GB), **4** RTX 4090, OR **2** A100, OR **2** H100
- **Hardware requirement example 2**: Qwen-coder 480B (290 GB) **13** RTX4090 !?, OR **4** A100, OR **4** H100
- Choice between acceptable performance and Hardware restrains
- Model used between ~9 – 30B parameters. Can run on single GPU
- Perform reasonable with MCP, given good environment
- Less problematic with other transformer models for transcription (whisper large ~ 3GB), OCR, etc

DSL

UNIVERSITÄT BERN

Local LLMs (The Storage Problem)

- Initial approach: each user download model on ~HOME in first run
- **Models are huge.** A single modern open-weight LLM is 5–40 GB
- Many users, one cluster. If every researcher downloads their own copy into their home directory, storage cost multiplies by the number of users — N users × M models = N×M copies of the same bytes
- **Models update frequently** (Gemma1, 2, 3, 4, etc). Regular updates on Text Lab
- If Text Lab becomes popular storage problem

DSL

UNIVERSITÄT BERN

Local LLMs (The Storage Problem)

- The intuitive first approach was to embed the models inside the Apptainer SIF image itself. One file to rule them all!
- First problem, build time is very long for every update.
- Second problem, introduce latency loading the model

DSL

UNIVERSITÄT BERN

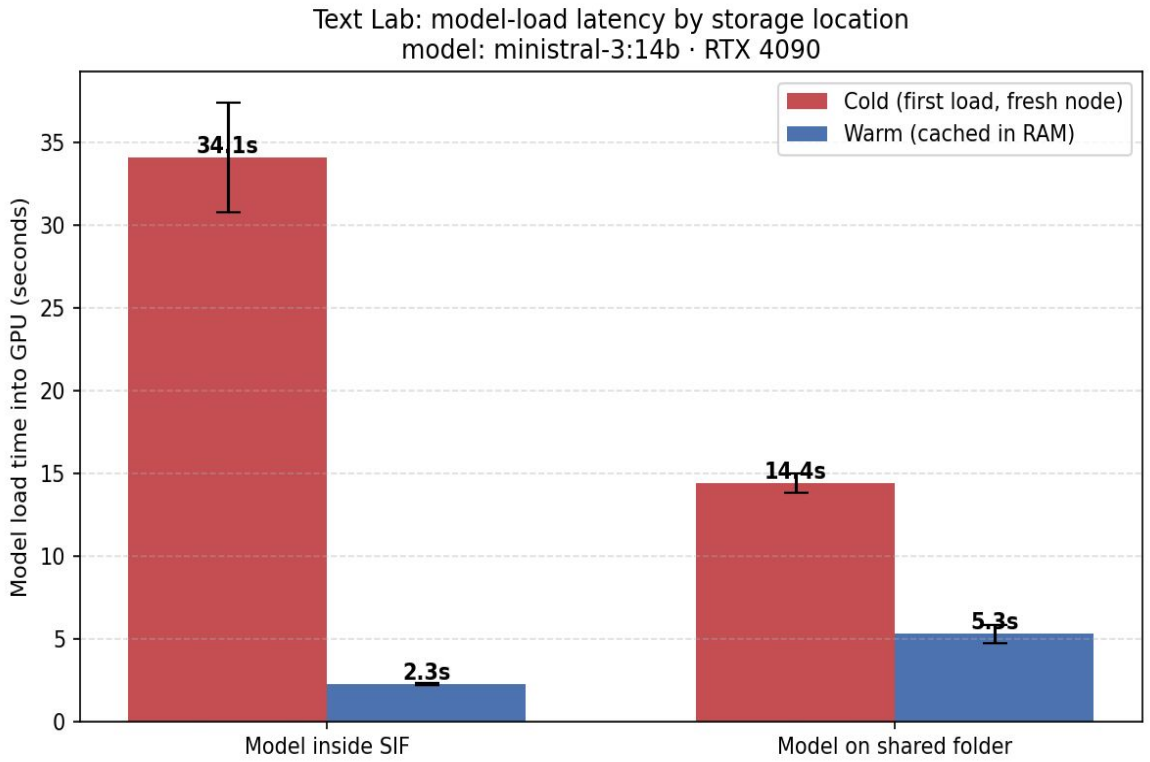
Local LLMs (The Storage Problem)

- Current solution: One Shared, Read-Only Cache
- Store each model once, in a central read-only folder on cluster storage, shared by all Text Lab users.
- **Read-only = safe & consistent.** No user can corrupt or accidentally delete the weights; everyone runs the exact same, validated model.
- **Zero footprint in user space.** Researchers consume *no* home-directory quota for model weights. Their quota stays for their own data.
- **De-duplication at scale.** Storage cost is M models (one copy each), not N×M. As user count grows, the savings grow with it.

LLM inside SIF vs. Shared Folder Benchmark

- Cold start from inside the SIF is ~2.4× slower than from the shared folder.
- The larger the model, the longer the wait for the first load
- Ollama reports load_duration (nanoseconds) in every /api/generate response
- Model: ministral-3:14b, ollama version 0.24.0, GPU: RTX 4090

Storage location	Cold load (mean±std)	Warm load (mean±std)	Cold runs
Model inside SIF	34.1 ± 3.301 s	2.268 ± 0.084 s	3
Model on shared folder	14.392 ± 0.587 s	5.289 ± 0.567 s	3



DSL

UNIVERSITÄT BERN

Local LLMs (The Storage Problem) – Storage saving

- Shared read-only cache: ~393 GB on disk, 14 models, stored exactly once. (Some older models clean-up)

Text Lab users	Storage saved
10	~ 3.8 TB
25	~9.6 TB
50	~19.2 TB
100	~38.4 TB
200	~76.8 TB

DSL

UNIVERSITÄT BERN

Which GPU should I use?

- RTX4090, A100, H100, H200??
- The larger VRAM, the better?
- Certain GPUs are less available than others

DSL

UNIVERSITÄT BERN

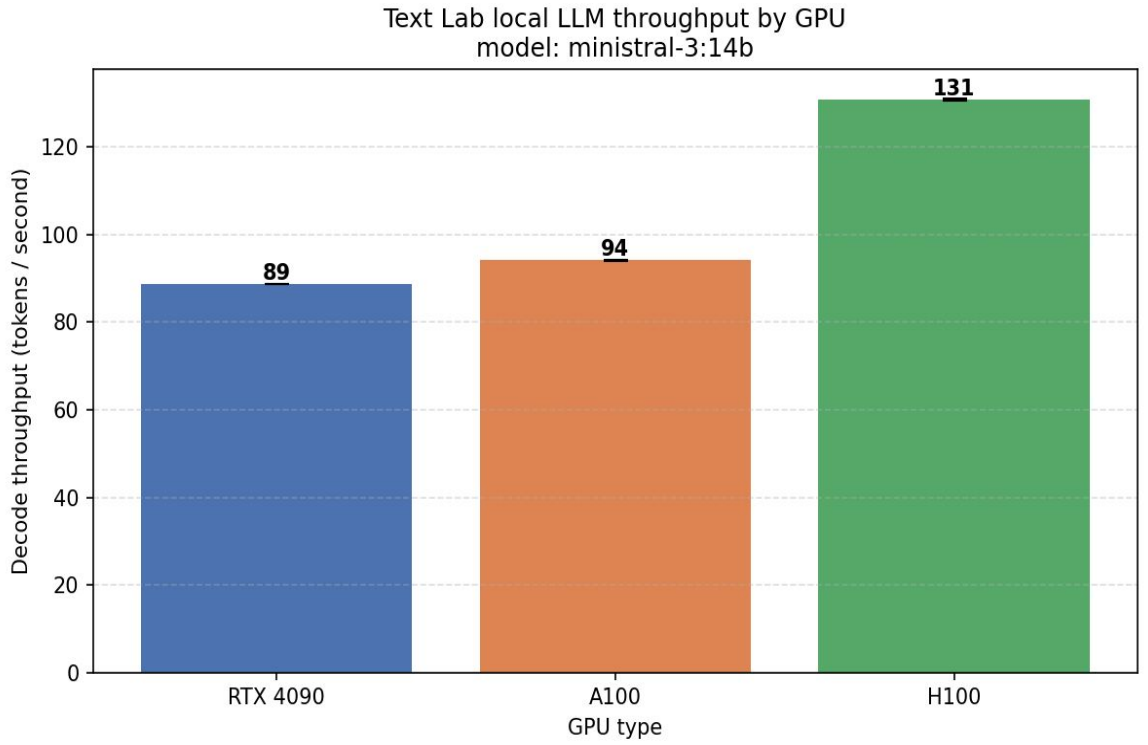
Text Lab - tokens per second GPU benchmark

- Measures how fast a local LLM generates text (**tokens per second**) on different UBELIX GPU type

Text Lab - tokens per second GPU benchmark

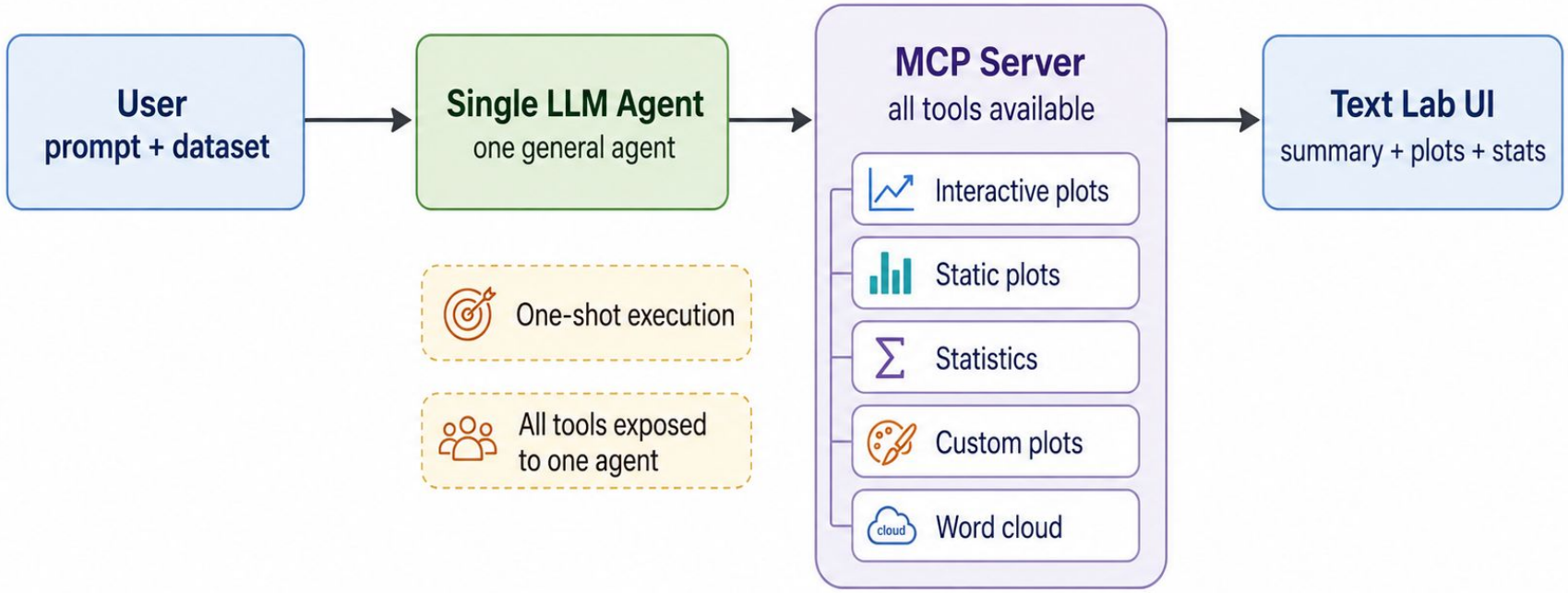
- Experiment set up:
Model: ministral-3:14b
Ollama version: 0.24.0
Flash attention: true
Sampling options: {"num_predict": 256, "temperature": 0, "seed": 42, "top_p": 1.0, "top_k": 0, "num_ctx": 4096}

tokens_per_sec = eval_count / eval_duration * 1e9
eval_count = tokens generated, **eval_duration** = nanoseconds spent generating them,
and **1e9** converts nanoseconds to seconds.



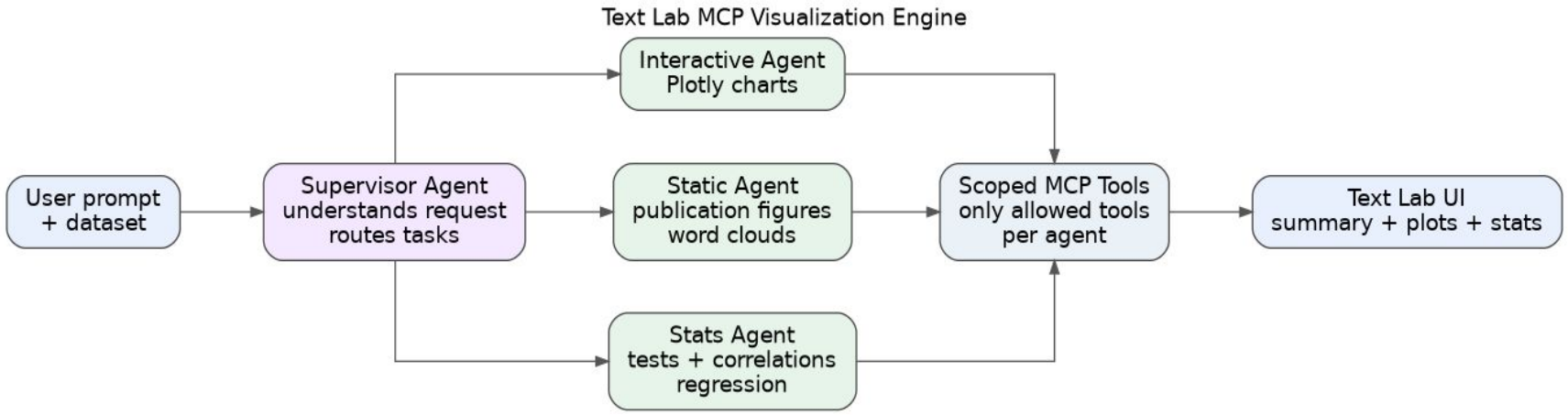
MCP AI visualizer feature

- Initial approach
- 1 agent (LLM)
- Access to all tools
- Failure when scaling

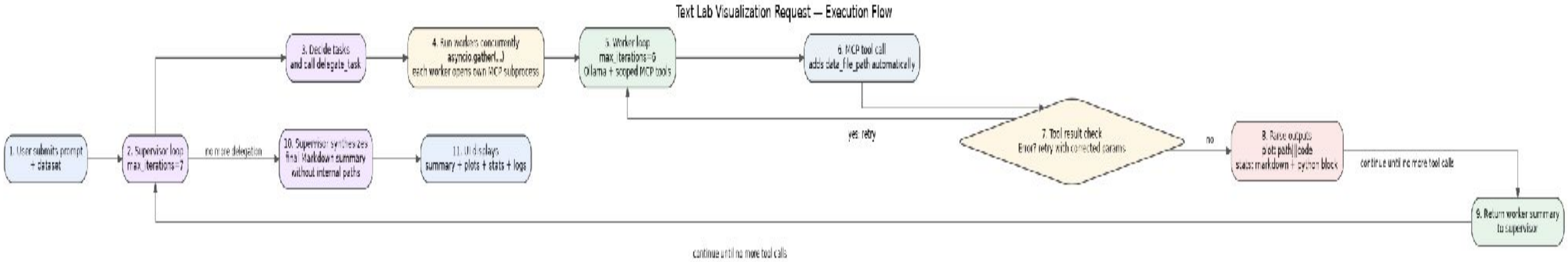


MCP AI visualizer feature

- Multi-agent architecture
- 1 supervisor agent, 3 specialized agents
- Each LLM has access to own tools only
- Integration when failed



MCP AI visualizer feature



DSL

UNIVERSITÄT BERN

What's Next?

- Improve reach out
- More features (Machine Translation, Metadata extraction)
- Maybe communicate with other university with similar setup (HPC + Open OnDemand)

DSL

UNIVERSITÄT BERN

Thank you!

- Questions
- Try Text Lab on: https://ondemand.hpc.unibe.ch/