Topics for AI/ML on HPC systems

Grigory Shamov,
May 22, 2026

University of Manitoba

Why AI/ML on HPC? Next to HPC? etc.

- **HPC has high performance hardware and software**
  - NVIDIA (and AMD) GPUs
  - High bandwidth, Low-latency Interconnect (NVidia Infiniband)
  - Scalable, parallel File Systems (VAST, CEPH, DDN Infinia) or local SSD NVMe
- **HPC community has a data centre infrastructure**
  - Power; high efficiency cooling
- **Many AI shops do use same HPC technology**
  - **SLURM, Linux, etc**.
  - Some do use Kubernetes and containerized workflows
- **So HPC should be a natural fit for AI. … Right?**

This talk is about using our HPC infra (2026!)

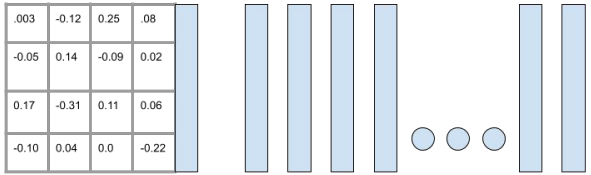
- HPC presents a few issues for running AI workflows:
  - How to maintain the (largely Python-based) software on HPC?
  - Good to have some ability to interactive workflows for debugging
  - How to adapt it to the SLURM based, batch workflow
    - Need to match HPC resources with AI workflow efficiently

- **2026**: Agentic AI adds a new set of issues
  - Local models vs best models? Policy issues.
  - Running microservices on HPC is hard.


Artificial Intelligence and Machine Learning revolution
https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/35179.pdf "The Unreasonable Efficiency of Data"
http://www.incompleteideas.net/IncIdeas/BitterLesson.html "The Bitter Lesson"
https://dl.acm.org/doi/10.5555/3295222.3295349 "Attention is all you need"

What are those AI and “models”

- Layers of numerical weights, and metadata about them
- Specific to the model’s software and hardware (precision)
  - (“transformers” would have attention layer, etc.)



- **Training/tuning** optimizes the weights of all or some layers
- **Inference** converts a prompt into a numerical representation , passes through model layers and generates outputs

https://poloclub.github.io/transformer-explainer/

What are those AI and “models”

- **Training/tuning** optimizes the weights of all or some layers
- **Inference** converts a prompt into a numerical representation , passes through model layers and generates outputs

Training on HPC is a natural fit

- Needs fast GPUs, needs fast storage and interconnect.
- Can be done in our lovely Batch Mode, as a job, using many GPUs

- If the training is really what you need….

DRI (Digital Research Infrastructure) for AI

IN MANY WAYS, AI VINDICATES THE "HPC WAY"

- ***AI needs fast interconnects***. We had them, the cloud and the enterprise did not.
  - Microsoft deployed 40,000 KM of *Infiniband*, in 2023, built for the HPC market ~1999,
- ***AI needs message passing***. MPI, the message passing interface, was built Open Source in the HPC community, ~1993
  - Now the standard library for transformer-based generative AI (e.g. ChatGPT, DeepSpeed, OpenAI etc.).
- ***AI needs heterogeneity*** – GPUs for general purpose computing – the hardware building block for AI - came out of the HPC world ("GPGPU" ~2004).
- ***AI needs fast, large scale filesystems –*** not object stores
- ***AI needs liquid cooling*** – even 5 years ago, many datacenter providers were convinced they could just use air, now none are. HPC systems switched to liquid cooling a long time ago.
- This means AI needs HPC hardware (probably good) and HPC programmers (good if you are one, bad if you need to hire one).

A slide by Dan Stazione, Director of TACC

***tamIA**, a real AI supercomputer of LavalU*

What are those AI and “models”

- **Training/tuning** optimizes the weights of all or some layers
- **Inference** converts a prompt into a numerical representation , passes through model layers and generates outputs

- **LoRA, QLoRA** (Low Rank Adaptation of Large Language Models)
  - optimizes weights of a few layers to improve an existing model

- **RAG** (Retrieval Augmented Generation) augments the prompt context with an external knowledge (numerically encoded into a “vector database” by another model)
  - Provides more context. Success depends on the “context window” of a given model.

Agentic AI

- Agentic more or less “builds” on RAG, creating an **agentic loop**
  - Reasons, calls tools , updates the context (memory) iteratively
  - Can even self-evolve or self-optimize the model
  - Breaking news: A. Karpathy’s LLM Wiki , “automated research”

- Can execute tools: if you can do a CLI work, agent can do it too.
  - Or can use MCP protocol with tools that support it

- Can be multi-agentic with “teams” of agents (PM managing QA and coders)

HPC use cases?

- Internal use: analysis of user's usage;
  - A predictive, dynamic job Scheduling
  - RAGs for HPC documentation.
- Training large models; LoRAs
- Developing parallel codes for HPC.
  - Stating intent rather than low-level programming
  - From prompt/copilot to automatic agentic generation and testing
    - An example : https://github.com/Katagiri-Hoshino-Lab/VibeCodeHPC
- Automating usage of HPC systems
  - (https://clusterpilot.sh/)
  - Tools like Globus-MCP for building agents https://github.com/globus/globus-mcp
- AI co-Scientists on HPC
  - Automating running computational experiments on HPC (i.e. LAMMPS)
    - https://sc25.supercomputing.org/proceedings/posters/poster_files/post284s2-file3.pdf

.

Issues with AI / ML agentic uses on HPC

- Policy and security issues
  - Using commercial models leaks data (esp for agentic use)
  - Responsibility wrt ToS ? Agents runs as a user. Safety / sandboxing
  - Network tunnels, (lack of) endpoint control, again DLP
- Ethical and organzational rules (check UManitoba AI guidelines!)
- Technical issues
  - Running microservices is harder in batch jobs
  - Local models require a lot of resources to be useful
- Positives.
  - There are free GPUs, storage and networks in HPC.
  - Agents consume CPU and memory but LLMs run elsewhere, ok.
  - With AI SCIP, much more hardware and perhaps a LLMaaS would materialize

Hands-on: a local LLM with llama.cpp and OpenWebUI

- Download the models in a proper GGUF format from HuggingFace

- Run a SLURM job that starts **llama.cpp** service
- Connect to the llama.cpp via a browser and chat with it.
    - Works? Is the small model any good? Ask a question about Grex.

- Run a Virtual Desktop job.
- Start an OpenWebUI container and connect to it via a browser
    - Connect the llama service to OpenWebUI
    - Ingest Grex documentation (all markdown!) from https://github.com/um-grex/docs as a RAG KB
    - Ask the model specific questions about Grex (partitions etc)


University
of Manitoba