Christof Teuscher

ECE 410/510: Hardware for AI and ML

Week 5: Recap, outlook, and reminders

Portland State University
Department of Electrical and Computer Engineering (ECE)
www.teuscher-lab.com
teuscher@pdx.edu

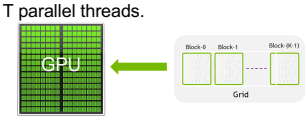
teuscher Lab teuscher-lab.com

Portland State UNIVERSITY

What did you learn last week?

Recap

- GPU execution model: SIMT (Single Instruction, Multiple Threads)
- GPU overhead:
  - Malloc
  - Copy data to GPU
  - Free
- A CUDA kernel is a function that runs on the GPU.
- Warp (32 threads, SIMD), thread block, kernel grid
- Parallelism is not always easy to find
- kernel<<M,T>>: The kernel launches with a grid of M thread blocks. Each thread block has T parallel threads.



SETH'S BLOG

View Online

The steps vs. the concept

If you memorize the steps, you have a direct, simple and fast path to obtain the result.

Until the world changes.

Even the tiniest shift in the system will render your memorization useless.

On the other hand, if you understand the concept, you'll be able to produce the steps whenever you need them.

Trends in AI Supercomputers

Konstantin F. Pilz [*1] James Sanders [2] Robi Rahman [2] Lennart Heim [2 3]

Abstract

Frontier AI development relies on powerful AI supercomputers, yet analysis of these systems is limited. We create a dataset of 500 AI supercomputers from 2019 to 2025 and analyze key trends in performance, power needs, hardware cost, ownership, and global distribution. We find that the computational performance of AI supercomputers has doubled every nine months, while hardware acquisition cost and power needs both doubled every year. The leading system in March 2025, xAI's Colossus, used 200,000 AI chips, had a hardware cost of $7B, and required 300 MW of power—as much as 250,000 households. As AI supercomputers evolved from tools for science to industrial machines, companies rapidly expanded their share of total AI supercomputer performance, while the share of governments and academia diminished. Globally, the United States accounts for about 75% of total performance in our dataset, with China in second place at 15%. If the observed trends continue, the leading AI supercomputer in 2030 will achieve 2×10^{22} 16-bit FLOP/s, use two million AI chips, have a hardware cost of $200 billion, and require 9 GW of power. Our analysis provides visibility into the AI supercomputer landscape, allowing policymakers to assess key AI trends like resource needs, ownership, and national competitiveness.

https://arxiv.org/pdf/2504.16026

Computational performance of AI supercomputers

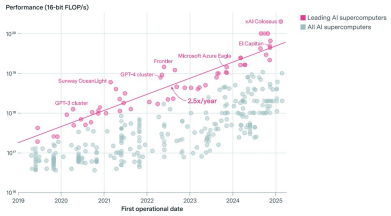


Figure 1: The performance of leading AI supercomputers (in FLOP/s, for 16-bit precision) has doubled every 9 months (a rate of 2.5× per year).

https://arxiv.org/pdf/2504.16026

Cost of AI supercomputers

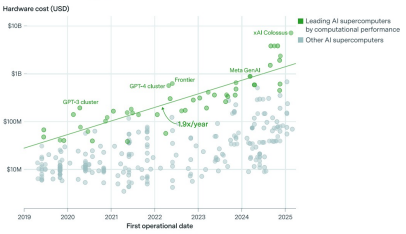


Figure 2: The cost of leading AI supercomputers (in 2025 USD) has doubled roughly every year.

https://arxiv.org/pdf/2504.16026

AI chip quantity in AI supercomputers

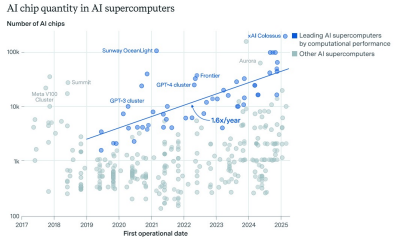


Figure 6: The number of AI chips in the leading AI supercomputers grew by 1.6× per year (90% CI: 1.5–1.8×). We start the regression in 2019, but gathered data further back to determine which 2019 AI supercomputers were in the top-10. Our pre-2019 data is too limited to include in the regression. See Section 2 for full methods.[11]

https://arxiv.org/pdf/2504.16026

Breakdown of factors contributing to training compute growth

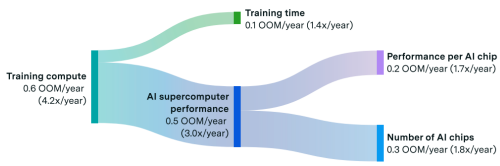


Figure 9: Overview of drivers of increasing training compute. OOM stands for orders of magnitude. AI supercomputer metrics are based on private sector systems and the highest computational performance across precisions.

https://arxiv.org/pdf/2504.16026

Power requirement of AI supercomputers

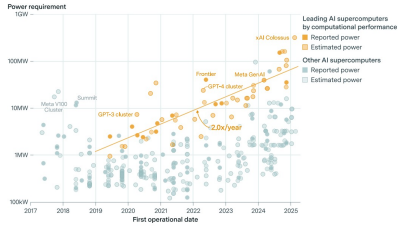


Figure 10: Peak data center power requirements of leading 10 AI supercomputers doubled every year (90% CI: 1.6–2.2× per year). We display reported power requirements whenever available. If not, we estimate capacity based on the number and type of chips used.

https://arxiv.org/pdf/2504.16026

Energy efficiency of AI supercomputers

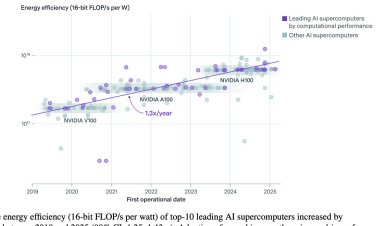


Figure 11: The energy efficiency (16-bit FLOP/s per watt) of top-10 leading AI supercomputers increased by 1.34× per year between 2019 and 2025 (90% CI: 1.25–1.43×). Adoption of new chips was the primary driver of energy efficiency improvements, with data center infrastructure efficiency only playing a minor role. We use reported power requirement whenever available and estimated power otherwise. For detailed methods and limitations, see Appendix B.4.

https://arxiv.org/pdf/2504.16026

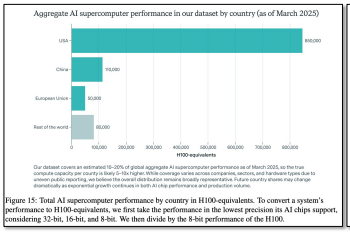


Figure 15: Total AI supercomputer performance by country in H100-equivalents. To convert a system's performance to H100-equivalents, we first take the performance in the lowest precision its AI chips support, considering 32-bit, 16-bit, and 8-bit. We then divide by the 8-bit performance of the H100.

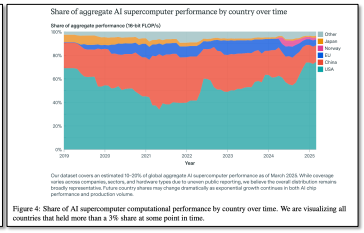


Figure 4: Share of AI supercomputer computational performance by country over time. We are visualizing all countries that held more than a 3% share at some point in time.

https://arxiv.org/pdf/2504.16026

Naser AI in Civil Engineering (2025) 4:10
https://doi.org/10.1007/s43503-025-00053-x

AI in Civil Engineering

CASE STUDY Open Access

A step-by-step tutorial on machine learning for engineers unfamiliar with programming

M. Z. Naser[1,2*]

Abstract

Machine learning (ML) has garnered significant attention within the engineering domain. However, engineers without formal ML education or programming expertise may encounter difficulties when attempting to integrate ML into their work processes. This study aims to address this challenge by offering a tutorial that guides readers through the construction of ML models using Python. We introduce three simple datasets and illustrate how to preprocess the data for regression, classification, and clustering tasks. Subsequently, we navigate readers through the model development process utilizing well-established libraries such as NumPy, pandas, scikit-learn, and matplotlib. Each step, including data preparation, model training, validation, and result visualization, is covered with detailed explanations. Furthermore, we explore explainability techniques to help engineers understand the underlying behavior of their models. By the end of this tutorial, readers will have hands-on experience with three fundamental ML tasks and understand how to evaluate and explain the developed models to make engineering projects efficient and transparent.

Keywords Machine learning, Engineering, Tutorial, Python

https://doi.org/10.1007/s43503-025-00053-x

Week	Monday	Wednesday (Codefest)
2	HW/AI/ML overview + codesign overview	Start main project: pick workload, start analysis, benchmark, …
3	GPU architecture and programming for AI	Drafting a HW architecture, creating a model
4	Deep neural networks on GPUs	Coding HW description
5	Transformers on GPUs	First simulation + refinement
6	In-memory computation	Improving initial design
7	Neuromorphic chips: TrueNorth, Loihi, Akida	Simulation + refinement
8	Neuromorphic computing with mem-devices	Synthesizing design + benchmarking
9	Hardware accelerators for embedded systems	Final improvements
10	Emerging technologies and future directions	Final tests, validation, verification, benchmarking

Memorial Day

This week

Monday
- CUDA
- Mapping deep neural nets (DNN/CNN) on GPUs
- TPUs
- Transformers on GPUs

Wednesday
- Codefest #5
  - Going to the gate level
  - Getting first performance numbers