CMSC828G: Special Topics in Systems for Machine Learning

Course Overview

Instructor: Yafan Huang

DEPARTMENT OF
COMPUTER SCIENCE

Instructor: Dr. Yafan Huang

- Ph.D. from Computer Science, University of Iowa
- 5-year visiting Ph.D. student, Argonne National Laboratory
- 1st year Assistant Professor, UMD CS and UMIACS
- Research Interests: High-performance Scientific and AI Systems

UMIACS
University of Maryland Institute for Advanced Computer Studies

Argonne NATIONAL LABORATORY

Class Information

- Course Website: (*or just google Yafan Huang and check the personal website*)
  https://hyfshishen.github.io/teaching/teaching-26fall-cmsc828g/

- ELMS-Canvas: Course discussions, announcements, submissions.
- Instructor Email: yafan@umd.edu
- Instructor Office Hour: By appointments at IRB 5146 or after class.
- TA: Shweta Bhardwaj (Email: shweta12@umd.edu)
- TA Office Hour: IRB 3119, Friday, 12:30 pm to 2:30 pm.

What will you learn?

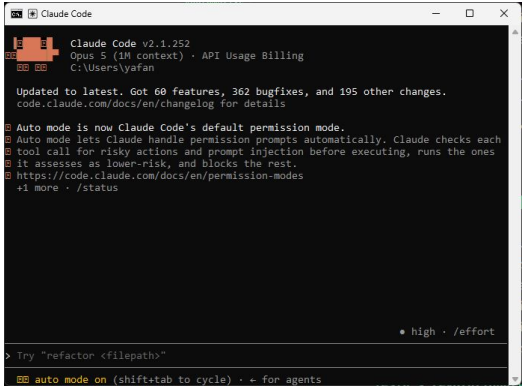
- Understand system innovations for modern machine learning (ML) workloads
- Explore cutting-edge ML systems research through top conference papers
- Learn to review, present, and develop ideas from research papers
- Gain hands-on experience through ML systems projects

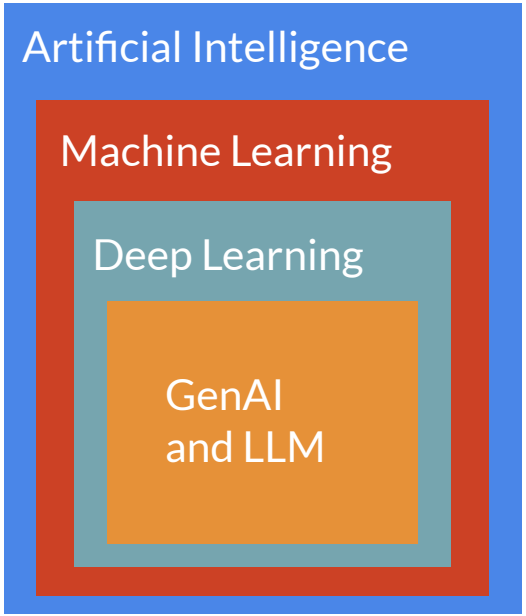
- **Note**: In this course, our discussion of ML primarily focuses on large language models (LLMs). Building agents and other ML models will not be primary focus.

Overview

The Success of AI Today

ChatGPT





- Even for our class, we mention AI in the Course Policy! 😂

Evoluation of Large Language Models (2017-2026+)

$$\text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

Transformer
Foundation
Jun. 2017

OpenAI

ChatGPT
Mainstream
Nov. 2022

vLLM

PagedAttention
Serving
Jun. 2023

Kimi
OpenAI
Claude

AI Agents
Harness, Loop, ...
2025-2026+

GPT-3
Model Scaling
May. 2020

OpenAI

Llama
Open Weight
Feb. 2023

Meta

Reasoning Models
Test-time Scaling
2024-2026+

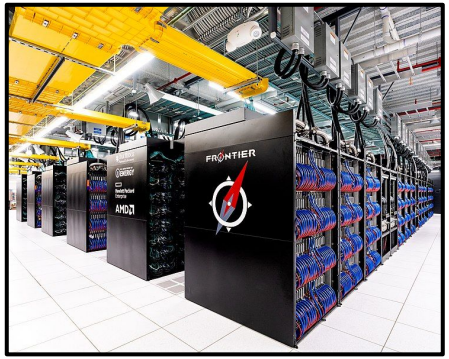
OpenAI
Claude
deepseek

What's behind? High-Performance Computing Systems

- Three key drivers of modern AI: algorithms, data, and computation.
- All three increasingly rely on high-performance computing (HPC) systems.

Summit (2018)

4,608 Nodes
9,216 IBM Power9 CPUs
27,648 NVIDIA V100 GPUs



Frontier (2021)

9,472 Nodes
9,472 AMD EPYC CPUs
37,888 AMD Instinct GPUs

Aurora (2025)

10,624 Nodes
21,248 Intel Xeon CPUs
63,744 Intel GPUs

Solstice (2026-28)

~20,000 Nodes
~40,000 High-end CPUs
~100,000 Blackwell GPUs[1]

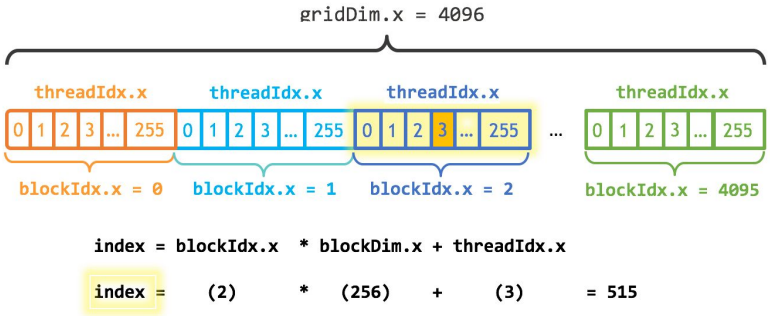
[1] [Argonne Press Release] https://www.anl.gov/article/argonne-expands-nations-ai-infrastructure-with-powerful-new-supercomputers

System for Machine Learning

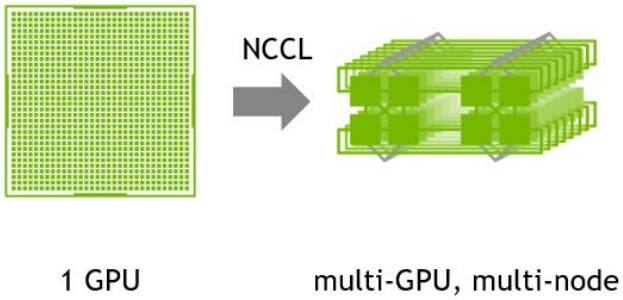
- Parallel Computing
  - GPU Programming, Multi-GPU / Multi-Node Communication
- Performance Optimization
  - Kernels, Memory, Communication, Compression, Quantization
- ML System Software
  - Compilers, Serving, Scheduling, Reliability
- Hardware and Architecture
  - GPUs, Memory, Interconnects, AI Chips (e.g., Cerebras…)

Parallel Computing

- Modern ML exploits parallelism at multiple levels, from GPU threads to distributed multi-GPU systems.



Intra-GPU Parallelism: CUDA[1]



Inter-GPU/Node Parallelism: NCCL[2]

- Make large-scale ML possible!

[1] Figure from https://developer.nvidia.com/blog/even-easier-introduction-cuda/
[2] Figure from https://developer.nvidia.com/nccl

Performance Optimization

- Efficient ML requires optimizing computation and data movement across the system stack (from intra-GPU to inter-GPU/node).
    - Kernels: Kernel fusion, Tiling, Latency Control
    - Memory: Locality, Data Movement, KV Cache
    - Communication: Data Awareness, Topology Awareness
    - Compression: Data Movement / Memory Footprint Reduction
    - Quantization: Lower Precision

- Make ML faster and more resource efficient!

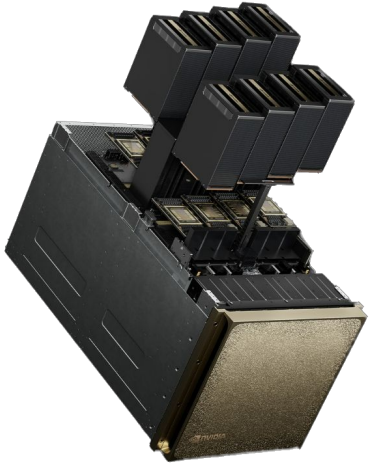
ML System Software

- Modern ML relies on system software to efficiently compile, serve, schedule, and reliably execute workloads.
  - Compilers: TVM, MLIR
  - Serving: vLLM, SGLang
  - Scheduling: Batching, Request Scheduling
  - (Reliability: Algorithm-based Fault Tolerance, Deterministic Replay)

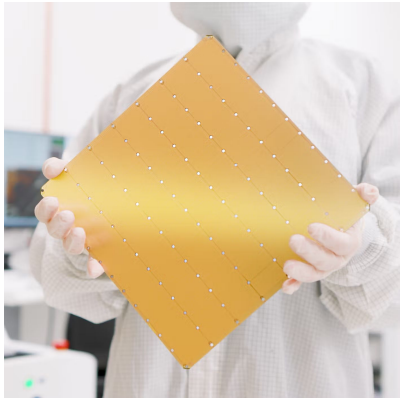
Hardware and Architecture

- Modern ML relies on hardware innovations at multiple levels, from GPU microarchitecture to large-scale AI systems.

Illustrating 1 SM in NVIDIA Ampere GPU[1]



NVIDIA DGX B200 Server Node (8x NVIDIA B200 GPUs)



AI Chip: One Cerebras WSE (40+ GB On-chip Memory)

[1] Figure from https://developer.nvidia.com/blog/nvidia-ampere-architecture-in-depth/

Logistics

Course Topics

- Introduction to HPC (1 week)
- GPU Programming Basics (1 week)
- Transformer and LLM Training (2 weeks)
- Optimizing GPU Kernels (0.5 weeks)
- Compilers and MLIR (0.5 weeks)
- Optimizing LLM Inference (4 weeks)
- Fault Tolerance for LLM (1 week)
- Emerging AI Chips (0.5 weeks)

Course Structure

- Topic Overviews: Instructor-led introduction to concepts or research directions.

- Paper Discussions: Student-led presentation and discussion of selected papers.
  - Check course schedule when lecture is marked with **[Reading].**

- Guest Lectures (if any): Invited talks on selected systems topics.
  - Dates and times may vary and will be announced in advance.

Textbook

- No required textbook.
- Primary readings: research papers from top-tier conferences.
- Optional REFERENCES:
  - HPC: *The Art of HPC* book by Victor Eijkhout.
  - Deep Learning (DL): *Deep Learning* book by Ian Goodfellow et al.
  - MLSys: *Introduction to Machine Learning System* books by Vijay Janapa Reddi.
  - GPU Programming: *Modern GPU Programming for MLSys* by MLC Community.

Grading Criteria

- Paper Readings: 5%
- Paper Presentation: 15%
- Midterm Exam 1: 20%
- Midterm Exam 2: 20%
- Final Project: 40%

Paper Readings

- Choose one of the two assigned papers for each paper discussion.
- Submit a one-page reading report:
    - Motivation, key ideas, main results, and your thoughts and questions.
- Due at 10:00 PM the day before class.
- Up to three reports may be skipped without penalty. No make-up reports for skipped readings.
- Paper reading list see Presentation Assignments.

Three Questions While Reading a System Paper

- Why does it work this way?
- Does it have to work this way?
- Can we do better?

Paper Presentation

- One individual paper presentation per student through the semester.
- 30 minutes in all: ~20 minutes presentation + 5-7 minutes QA + 3 minutes setup.
- Evaluation: Technical understanding, clarity and organization, presentation quality, and QA performance.
- What makes a good presentation?
    - Make it understandable.
    - Identify the key insights.
    - Help your audience learn.

Presentation Assignment

- Email Yafan for the Google Spreadsheet link.

Comment in the cell for the paper you'd like to present

	A	B	C	D
1	**Topics**	**Conferences**	**Papers**	**Presenter**
2	**Pipeline and Hybrid Parallel Training**	VLDB 2023	PyTorch FSDP: Experiences on Scaling Fully Sharded Data Parallel	
3		OSDI 2025	WLB-LLM: Workload-Balanced 4D Parallelism for Large Language Model Training	
4	**Sparsity in Training**	MLSys 2023	MegaBlocks: Efficient Sparse Training with Mixture-of-Experts	
5		OSDI 2025	ZEN: Empowering Distributed Training with Sparsity-driven Data Synchronization	
6	**Optimizing GPU Kernels**	ICS 2024	Accelerated Auto-Tuning of GPU Kernels for Tensor Computations	
7		SC 2024	cuSZp2: A GPU Lossy Compressor with Extreme Throughput and Optimized Compression Ratio	
8	**Compilers and MLIR**	OSDI 2025	KPerfIR: Towards an Open and Compiler-centric Ecosystem for GPU Kernel Performance Tooling on Modern AI Workloads	
9		MLSys 2026	ApproxMLIR: Accuracy-Aware Compiler for Compound ML System	
10	**Speculative Decoding**	ICML 2024	Medusa: Simple LLM Inference Acceleration Framework with Multiple Decoding Heads	
11		ASPLOS 2024	SpecInfer: Accelerating Large Language Model Serving with Tree-based Speculative Inference and Verification	
12	**Attention Approximation**	NeurIPS 2023	H2O: Heavy-Hitter Oracle for Efficient Generative Inference of Large Language Models	
13		COLM 2024	Mamba: Linear-Time Sequence Modeling with Selective State Spaces	
14	**Long Context Optimizations**	ICLR 2024	Efficient Streaming Language Models with Attention Sinks	
15		ICLR 2024	Ring Attention with Blockwise Transformers for Near-Infinite Context	
16	**Quantization**	NeurIPS 2023	QLORA: Efficient Finetuning of Quantized LLMs	
17		ICLR 2023	GPTQ: Accurate Post-Training Quantization for Generative Pre-trained Transformers	
18	**KV Cache Compression I**	ICML 2024	KIVI: A Tuning-Free Asymmetric 2bit Quantization for KV Cache	
19		NeurIPS 2024	KVQuant: Towards 10 Million Context Length LLM Inference with KV Cache Quantization	
20	**KV Cache Compression II**	SIGCOMM 2024	CacheGen: KV Cache Compression and Streaming for Fast Large Language Model Serving	
21		EuroSys 2025	CacheBlend: Fast Large Language Model Serving for RAG with Cached Knowledge Fusion	
22	**Mixture-of-Experts Inference**	SC 2022	DeepSpeed Inference: Enabling Efficient Inference of Transformer Models at Unprecedented Scale	
23		SIGCOMM 2025	MegaScale-Infer: Efficient Mixture-of-Experts Model Serving with Disaggregated Expert Parallelism	
24	**Reliable LLM Training**	ACL 2025	Understanding Silent Data Corruption in LLM Training	
25		OSDI 2026	Safeguarding LLM Training at Scale: Online SDC Detection and Insights from 35 Million GPU Hours	
26	**Reliable LLM Inference**	SC 2025	FT-Transformer: Resilient and Reliable Transformer with End-to-End Fault Tolerant Attention	
27		ICS 2026	Not All Errors Are Equal: A Systematic Study of Error Propagation in Large Language Model Inference	
28	**Emerging AI Chips**	OSDI 2025	WaferLLM: Large Language Model Inference at Wafer Scale	
29		MLSys 2026	SHIP: SRAM-Based Huge Inference Pipelines for Fast LLM Serving	

Midterm Exams

- Two in-class written midterm exams (20% each).
- Midterm Exam 1: Primarily covers topics related to LLM training.
- Midterm Exam 2: Primarily covers topics related to LLM inference.
- Both: Relevant HPC, GPU, and ML concepts.

Final Project

See the course website for more details and important dates.

- Work in teams of 2-3. Individual projects require instructor approval.
- Choose a project related to systems for machine learning.
  - Explore a new idea, extend prior work, or analyze an existing system.
  - You are always welcome to discuss with instructor.
- Final Project Pitch Presentation is at the middle of the semester.
- Final Presentation is at the end of the semester.
- Final Deliverables
  - Presentation (slides), Implementation, Final Report in MLSys paper format.

Policy - Use of Generative AI

- Generative AI tools are allowed as learning and research aids.
- Submitted work must reflect your own understanding and intellectual contribution.
    - Disclose substantive use of generative AI in submitted coursework. For example, write a AI-use disclosure section in the final project report.
- Generative AI tools are not permitted during midterm exams.

Questions?

- Always feel free to reach out and discuss with Yafan.

- Always see more details from course website:
  https://hyfshishen.github.io/teaching/teaching-26fall-cmsc828g/