Scalable AI: Bridging Theory, Understanding, and Practice

EE 290 / 194 · Spring 2026

Instructors	Prof. Anant Sahai; Prof. Jiantao Jiao)
Teaching staff	Haocheng Xi (Teaching); Paul Zhou (Teaching); Venkat Srinivasan (NVIDIA infrastructure and technology)
Lecture	Tuesdays & Thursdays, 9:30am–11:00am (521 Cory Hall)
Office hours	Tuesdays, 11:00am–12:00pm (immediately after lecture; location announced on Ed)
Assignment checkoffs	Checkoff slots by sign-up
Resources	Website: `https://scalable-ai.eecs.berkeley.edu/` Forum (Ed): `https://edstem.org/us/courses/93630` Gradescope: `4DDRWY`
Questions	**Post on Ed.** We do *not* use Slack or other channels for course Q&A. Please only use Ed for all discussions.
Compute	Groups of 4–6. 1 H100 node (8×H100) per group (8 total nodes on GCP). Each group is assigned a single static external IP address for SSH'ing in and working.

Course Description

The central inquiry of this course is: *How do we build, train, and deploy large-scale AI systems by treating them as full-stack engineered artifacts, where hardware constraints, software stacks, and optimization dynamics jointly determine model behavior and performance?*

This course examines the principles required to build, train, and deploy large-scale AI models. We treat large-scale AI as an **end-to-end engineering discipline**, where a model is a computational graph that must be trained, specialized, evaluated, deployed, monitored, and iterated on—all under hard constraints from hardware, data, and serving economics.

At modern scales, large language models are constrained as much by hardware and systems realities as by algorithms. Matrix multiplication, memory bandwidth, interconnect topology, numerical precision, and optimizer stability define the feasible design space and shape both training and inference behavior. Architectural choices (dense vs. sparse, MoE, long-context mechanisms, parallelism strategies) and optimizer choices (including emerging optimizers like Muon and SOAP) directly determine convergence, efficiency, and deployability.

This course follows the lifecycle end to end:

Architecture → Pre-training → Post-training → Efficient inference → Applications → Research greenfields.

Throughout, we will work directly with the NVIDIA ecosystem and the software stack that underpins modern AI infrastructure. We will learn how the world built its AI infrastructure on NVIDIA—and what we can do to make it better.

Course questionnaire (Week 1–2). We will release a short onboarding questionnaire in the **first lecture (Jan 20)**. We will finalize enrollment/compute allocations and related decisions by the **end of the second week (Jan 29)**.

Communication policy (Ed only). All course questions **must** be posted on **Ed** (https://edstem.org/us/courses/93630). We answer, announce, and archive on Ed only.

Commitment and grading basis. This course requires sustained weekly engagement (group work, compute usage, and in-person checkoffs). **Letter grade only:** you may not take this course Satisfactory/Non-Satisfactory. Additionally, because the work is continuous and group-dependent, you should not enroll if you anticipate major recurring time commitments that would interfere with course participation (e.g., intensive interviewing or other major obligations).

Grading and Policies

This course has **no exams**. Evaluation is based on assignments, a semester-long research project, scribing, and participation.

Component	Weight
Research project	50%
Assignments	35%
Scribing (2 lectures per student)	5%
Attendance & participation	10%

Failing any one of them fails the class. You must do every component satisfactorily or else you will fail the class. This includes: the research project, all assignments, scribing, and participation.

Submission platform (Gradescope). All written submissions and code submissions will be submitted through **Gradescope** (4DDRWY), unless explicitly stated otherwise. Oral presentations require sign-up and in-person attendance to deliver the presentation.

Assignments (35% total: 25% core + 10% enhancements)

There are five **group-based assignments**, each spanning roughly two to three weeks.

Assignments include both:

- **Conceptual questions:** short written questions to test understanding.
- **Hands-on experiments:** implement, profile, scale, and analyze real workloads; report results; and reason about performance bottlenecks and tradeoffs.

How assignments are graded. Assignments are graded using a combination of:

- **Code grading:** Correctness, completeness, and (when applicable) performance profiling/measurement quality.
- **Written component:** Answers for conceptual questions, clarity of explanations, plots/tables, and reasoning about tradeoffs.
- **Oral checkoff/presentation (in-person):** each group will give a short oral presentation/checkoff for each assignment to demonstrate understanding and defend results.

Two grading components (within the 35%):

- **Core / working (25%):** The rubric above.
- **Enhancements (10%):** Contribute meaningfully by adding questions, or reframing the programmatic part to be more challenging.

Late policy (Assignments). You have **3 total slack days** across the semester for assignments. At most one slack day may be applied to any single assignment. Slack days extend the deadline by 24 hours. No other late work is accepted without an approved exception.

Assign.	Topic	Release	Due
A1	Parallelism Assignment	Jan 27	Feb 10
A2	Pre-Training Assignment	Feb 10	Feb 24
A3	Post-Training Assignment	Feb 24	Mar 10
A4	Inference and Serving Assignment	Mar 17	Apr 7
A5	Applications Assignment	Apr 7	Apr 21

Research Project (50% total: 40% technical + 10% impact)

The research project is **group-based** and open-ended, with an emphasis on producing something useful to the community.

Technical quality (40%). Assessed on clarity of hypothesis, technical depth, correctness, experimental rigor, and quality of the final artifact (code + report + other tangible artifacts).

Impact (10%). The goal is to have impact on the broader community. Examples include a PR merged into a public open-source repository, releasing a reproducible benchmark, a public technical report, or an arXiv preprint. Impact is scored on *evidence* of external usefulness.

Peer review (required). We will require **peer review** as part of the project process: each group will provide a structured, constructive review of another group's report. Peer reviews will be submitted via **Gradescope**. The **Technical Quality** aspect of the grading will incorporate peer feedback as well as the teaching staff feedback.

Milestone	Due	Notes
Research directions released	Jan 22	Posted on course site; brainstorming begins.
Group formation	Jan 29	Groups of 4–6; node assigned.
Hypothesis statement v1	Feb 12	One paragraph hypothesis + success metric.
Project proposal v2	Feb 26	2–3 pages: method, evaluation plan, risks.
Midterm check-in	Mar 19	Milestone report; check-in meetings scheduled around this date.
Draft report	Apr 26	Draft report submitted for peer review (Gradescope).
Final presentations	Apr 28/30	Short talks in the week **prior** to RRR week.
Peer review due	May 1	Peer reviews submitted (Gradescope).
Poster session	May 5	RRR week poster session.
Final deliverables	May 10	Code + final report.
Impact update close	May 15	Proof of merge/acceptance or submission to ArXiv, etc.

Scribing (5%)

Each student is expected to **scribe 2 lectures**. Scribe assignments will be posted on Ed. Scribing may use **LLM assistance** (e.g., for drafting), but the final scribe must be accurate, edited, and clearly written. If you use external sources or AI tools, you must cite them clearly.

Attendance & participation (10%)

Attendance is required. Participation includes:

- Contributing to in-class discussions and project check-ins.
- Posting and answering questions on Ed.
- **Providing constructive comments on lecture notes and slides** (we will solicit feedback threads on Ed; improving course notes/slides is part of the participation grade).

Academic integrity

Collaboration is encouraged *within your group*. You must not copy code or reports from other groups. If you use external code or AI assistants, you must cite the source clearly and ensure you understand the work you submit.

Infrastructure

This course uses shared GPU infrastructure. If you have questions about access, quotas, networking, or failures, **post on Ed** (include timestamps, job IDs, logs, and a short repro description).

- **Cluster and allocation:** We will use **8 total nodes on GCP**. Each group receives **one** H100 node (8×H100) for the semester. You will be given a static IP address that you can use to log into the node.
- **NVIDIA compute support:** Please route requests through **Ed** so issues are tracked and shared.
- **Multi-node experiments:** If your project requires more than one node, you must coordinate with another team to **share nodes** for a bounded window of time. Since each node has its own external IP, multi-node work may require coordinating host files and access rules across *multiple* team IPs.
  - Plan early and post on Ed to request staff help if infrastructure help is needed.
  - Agree on a schedule with the other team(s) to avoid interference.
  - Keep runs reproducible (versioned code, pinned configs, logged seeds, and saved artifacts) so you can hop around if the need arises.

Lecture and Deadline Schedule

Schedule assumes Tuesday/Thursday meetings. Guest speakers and exact ordering are subject to change. Deadlines (assignments and project milestones) are included directly in the schedule below. We may use late-semester buffer time as needed.

Date	Lecture / Event	Notes
Part 1: Architecture		
Jan 20	**L1.** Course Overview and the Modern AI Stack	
Jan 20	***Milestones/Releases:*** *Questionnaire released*	
Jan 22	**L2.** All About Performance	
Jan 22	***Project release:*** *Research directions released*	
Jan 27	**L3.** Architectures To Break Bottlenecks: MoE, Sparse & Long-Context Architectures	
Jan 27	***Release:*** *A1 released (Parallelism Assignment)*	
Jan 29	**L4.** Parallelism Strategies	
Jan 29	***Deadlines/Milestones:*** *Group formation due; questionnaire decisions finalized*	
Feb 3	**L5.** NeMo AutoModel Guest Lecture	Guest (AutoModel)
Part 2: Pre-Training of Language Models		
Feb 5	**L6.** Introduction to Pre-Training	
Feb 10	**L7.** Powering Pre-Training: NeMo Curator	Guest (Curator)
Feb 10	***Deadline/Release:*** *A1 due; A2 released (Pre-Training Assignment)*	
Feb 12	**L8.** Case Study: The Pre-Training of Nano-V3	
Feb 12	***Deadline:*** *Hypothesis statement v1 due*	
Feb 17	**L9.** Optimizer Fundamentals	
Feb 19	**L10.** Looking To The Future: Emerging Optimizers	Guest

Date	Lecture / Event	Notes
	Part 3: Post-Training of Language Models	
Feb 24	**L11.** Intro To the LLM Post-Training Lifecycle and Evaluation	
Feb 24	***Deadline/Release:*** *A2 due; A3 released (Post-Training Assignment)*	
Feb 26	**L12.** The Data Powering Post-Training: SFT Data Engineering and RL Environments	
Feb 26	***Deadline:*** *Project proposal v2 due*	
Mar 3	**L13.** Building SFT Datasets: Foundations of SFT Data Generation and Tooling	Guest
Mar 5	**L14.** NeMo Data Designer Deep Dive	
Mar 10	**L15.** Using The NeMo RL Stack For RL Post-Training	Guest
Mar 10	***Deadline:*** *A3 due (Post-Training Assignment)*	
Mar 12	**L16.** Case Study: Post-Training of Nemotron-NanoV3	Guest
	Part 4: Efficient Inference	
Mar 17	**L17.** Deployment Preparation: Speculative Decoding, Quantization, Pruning, and NAS	Guest
Mar 17	***Release:*** *A4 released (Inference and Serving Assignment)*	
Mar 19	**L18.** Fundamentals and Overview of High-Performance Inference Frameworks	
Mar 19	***Deadline:*** *Midterm check-in report due*	
Mar 24	*Spring Recess*	*No class*
Mar 26	*Spring Recess*	*No class*
Mar 31	**L19.** High-Performance Inference using Dynamo and TRT-LLM	Guest
Apr 2	**L20.** High-Performance Inference using vLLM and SGLang	
	Part 5: LLM Applications and Use Cases	
Apr 7	**L21.** Fundamentals of Context Engineering	
Apr 7	***Deadline/Release:*** *A4 due; A5 released (Applications Assignment)*	
Apr 9	**L22.** Agentic Applications	Guest
Apr 14	**L23.** Safety Guardrails	Guest
	Part 6: Research Greenfields	
Apr 16	**L24.** Diffusion Language Models	
Apr 21	**L25.** Advanced RL Algorithms	
Apr 21	***Deadline:*** *A5 due (Applications Assignment)*	
Apr 23	**L26.** Multi-Agent Systems and Architecture	
Apr 26	*Draft project report due*	*Submit on Gradescope*
Apr 28	**Project Presentations (Session A)**	Week prior to RRR week

Date	Lecture / Event	Notes
Apr 30	**Project Presentations (Session B)**	Week prior to RRR week
May 1	*Peer review due*	*Submit on Gradescope*
May 5	*Poster session (RRR Week)*	*No formal lecture*
May 10	*Final deliverables due*	*Code + final report*
May 15	*Impact update*	*Proof of Impact Document*