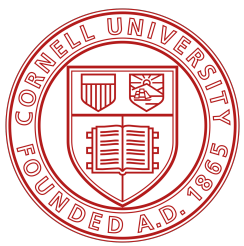
Distributed Optimization for Machine Learning

- co-listed in ECE 5290/7290 and ORIE 5290

Tianyi Chen

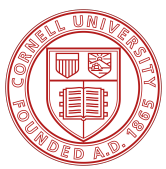
School of Electrical and Computer Engineering
Cornell Tech, Cornell University

August 25, 2025



Who I am, where to find me

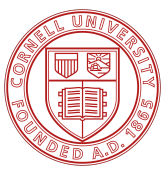
- Tianyi Chen (tianyi.chen@cornell.edu)
- Associate Professor, Electrical and Computer Engineering
- Where? We meet here at Bloomberg Center 161
- When? Mondays and Wednesdays 1:25PM - 2:40PM
- My office hours, Wednesdays at 5pm - 6pm
  - Also by appointment, as long as you have something interesting to chat



Teaching assistant

- A great TA to help you with your homework and projects

- Yuheng Wang
- Ph.D. candidate in Operations Research at Cornell University
- Location: TBD
- Email: yw634@cornell.edu
- Office hour: TBD



Large-scale distributed model training

Why taking this course at this time?

- Large-scale model training: Parallel and distributed training
- Private distributed learning: Federated learning

Course Content and Importance in ECE

Course workload and grading

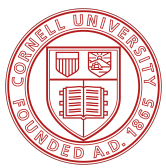
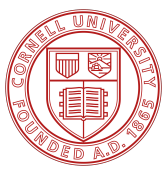


Table of Contents

Why taking this course at this time?
- Large-scale model training: Parallel and distributed training
- Private distributed learning: Federated learning

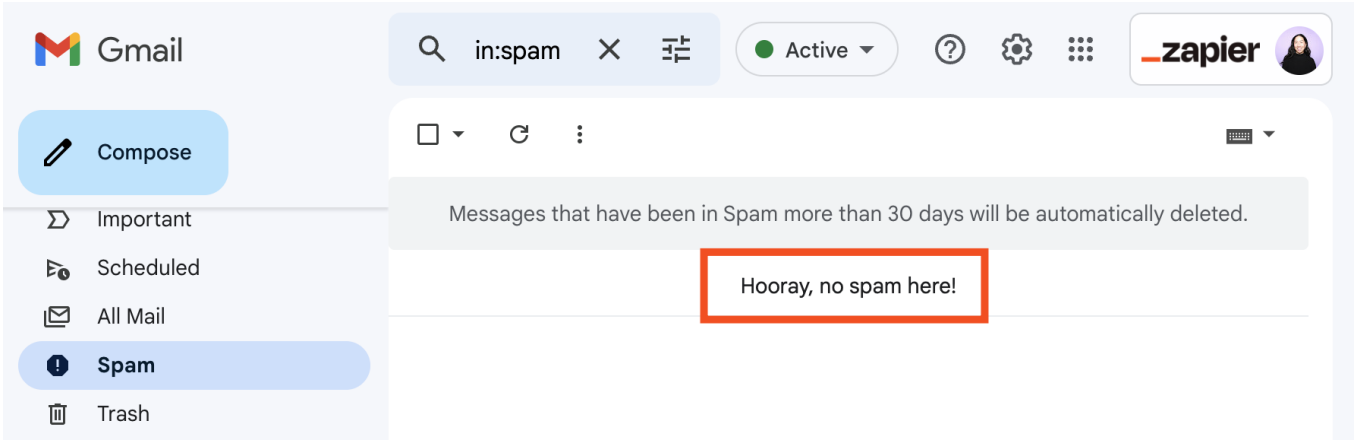
Course Content and Importance in ECE

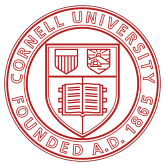
Course workload and grading



ML and AI in everyday life - Pre-Generative AI

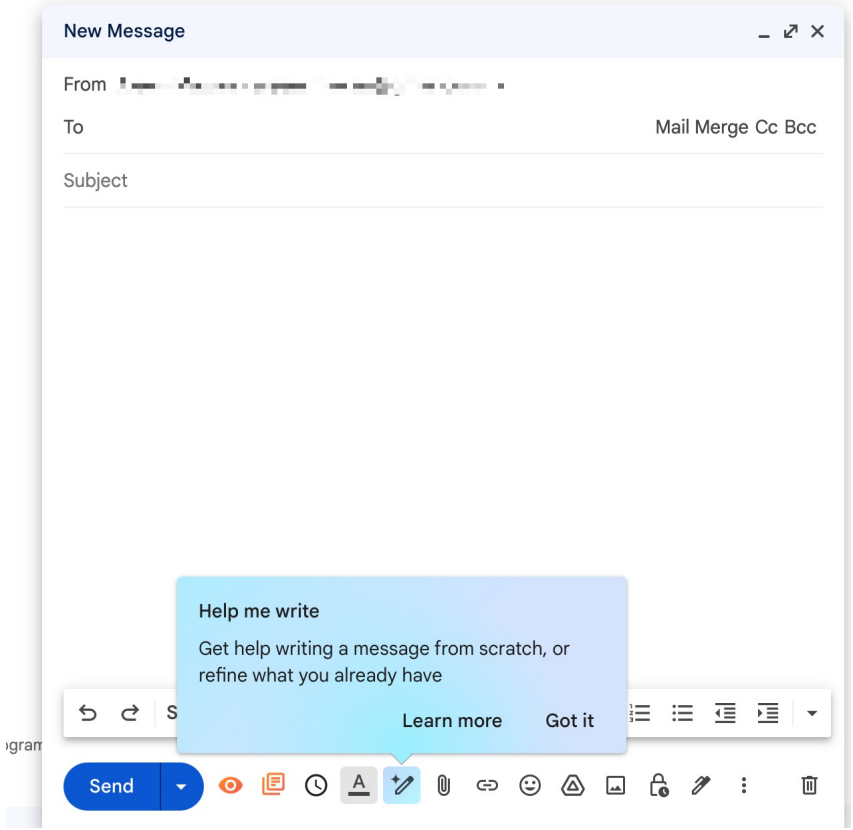
Machine learning is used in classifying spam emails, such as in Gmail.

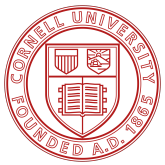




Generative AI in everyday life

Machine learning is nowadays used in drafting and replying emails.



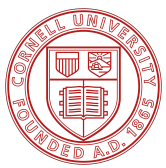


Why optimization is crucial for ML and AI?

Machine learning (ML) learns pattern from historical data. Assume a old-school ML task - model **House Price** based on **Square Footage**.

Square Footage	**Price ($1000s)**
800	150
1200	250
1500	280
2000	350
2400	450
3000	500

- **Feature (x):** Input variable for predictions (e.g., *square footage*).
- **Label (y):** The "answer" or output we want to predict (e.g., *price*).
- **Training Example:** A single row of data, like (1200 sq. ft., $250k).



Why optimization is crucial for ML and AI?

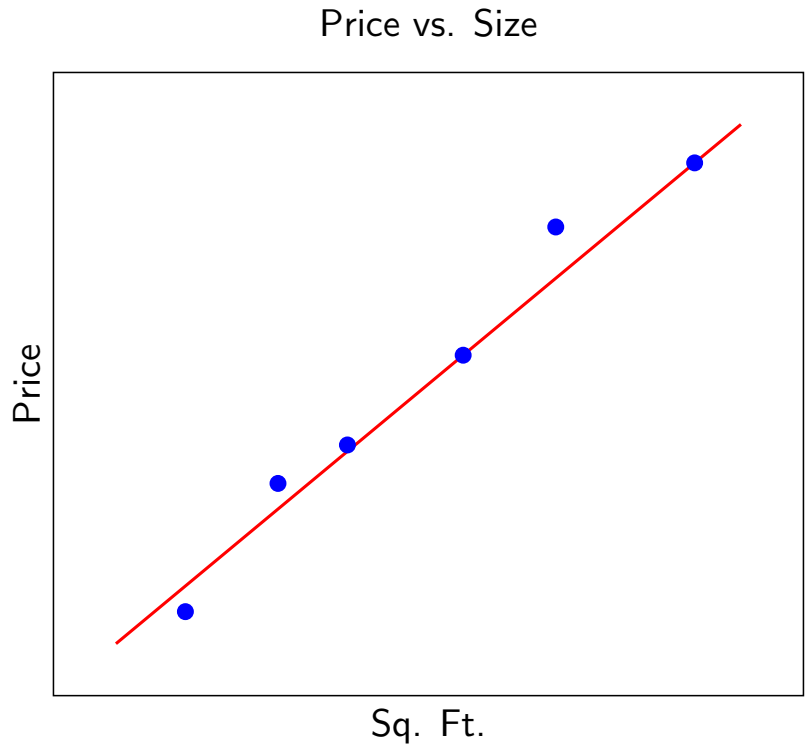
Our ML task is to model **House Price** based on **Square Footage**.

Square Footage $\rightarrow x$

Price $\rightarrow y$

Base Price $\rightarrow \theta_0$

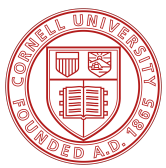
Price per SqFt $\rightarrow \theta_1$



Our Model:

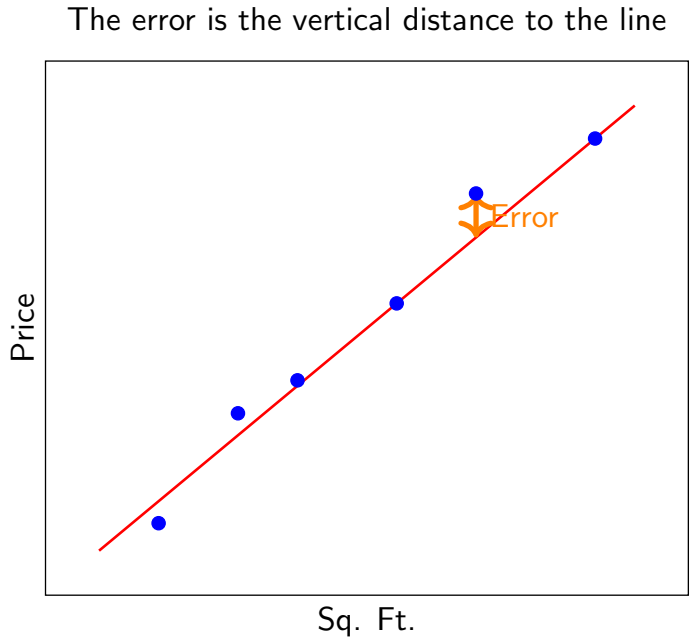
$$h_\theta(x) = \theta_0 + \theta_1 x$$

But what are the good parameters θ_0, θ_1 of our ML model?



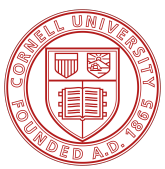
Why optimization is crucial for ML and AI?

We need to measure the total **error** of our model. The error is the distance between the actual price and the predicted price.



The **Mean Squared Error (MSE)** loss averages the square of all errors:

$$L(\theta) = \frac{1}{m}\sum_{i=1}^{m}(\text{Predicted}_i - \text{Actual}_i)^2.$$



Why optimization is crucial for ML and AI?

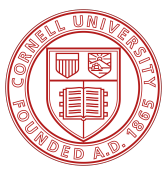
The goal of "training" is to find the parameters (θ_0, θ_1) that **minimize the loss function**.

$$\min_{\theta_0,\theta_1} \quad L(\theta_0, \theta_1)$$

Specifically, we want to solve:

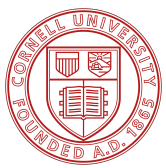
$$\min_{\theta_0,\theta_1} \frac{1}{2m} \sum_{i=1}^{m} (\underbrace{(\theta_0 + \theta_1 x_i)}_{\text{Predicted}_i} - \underbrace{y_i}_{\text{Actual}_i})^2$$

Based the **training set**, an optimization algorithm uses to "learn" the best parameters θ_0, θ_1 to minimize the loss functions.



What is distributed optimization?

- Distributed optimization is an approach where multiple entities, or "devices," connected through a distributed system aim to solve an optimization problem by sharing data and/or compute
- Distributed system: A system whose components are located on different networked devices, which communicate and coordinate their actions by passing messages to one another



They may have a common goal or individual interests

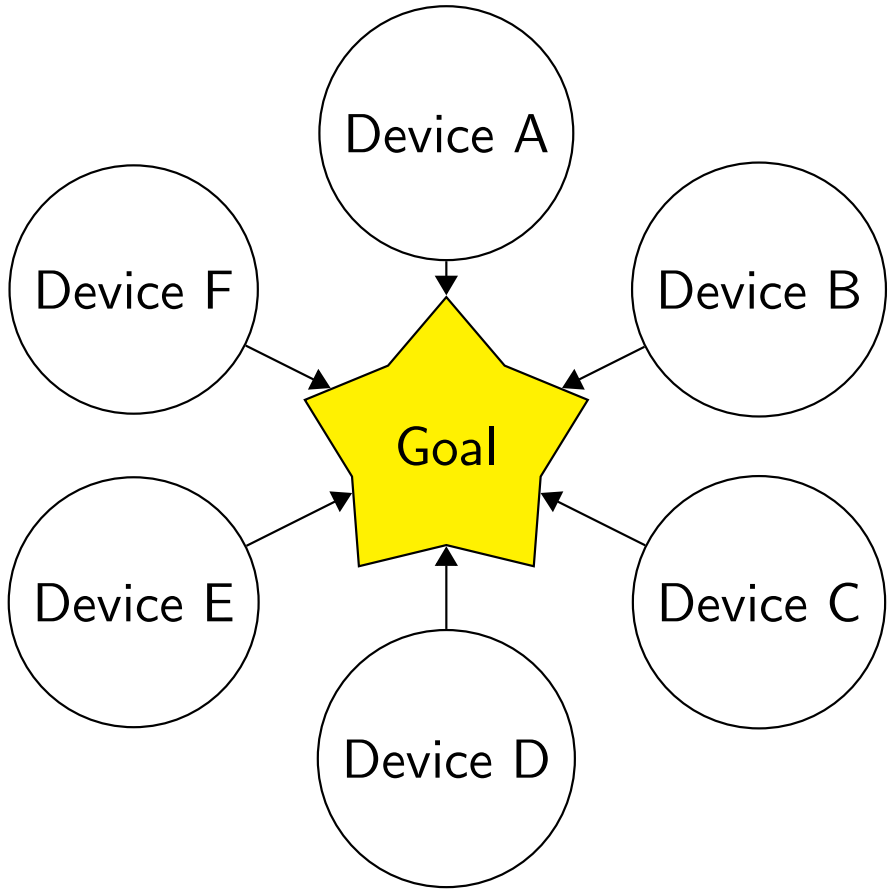
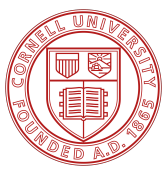


Figure: Distributed devices with a common goal.



They may have a common goal or individual interests

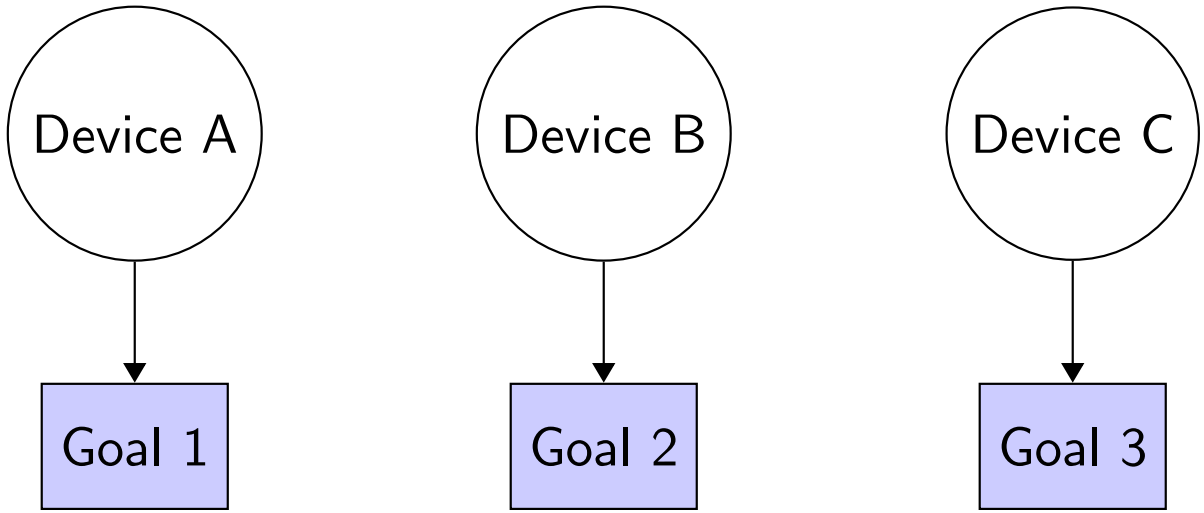
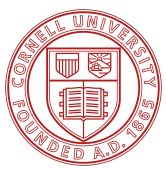
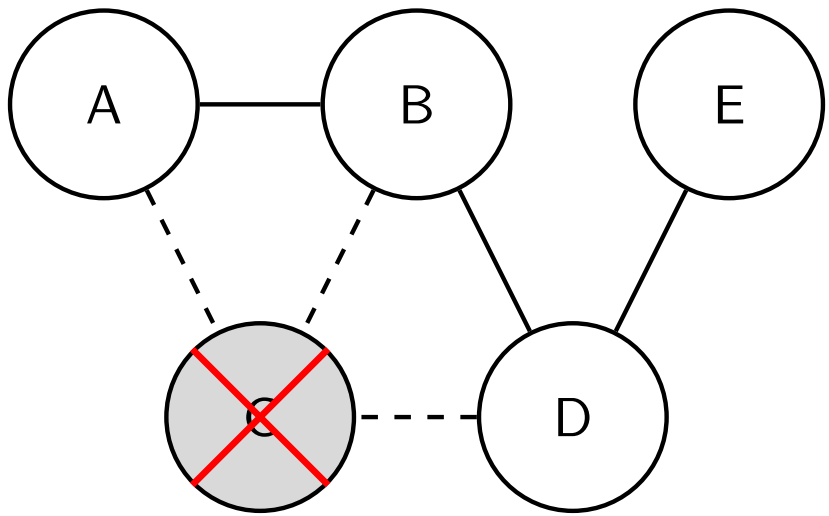


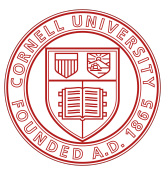
Figure: Distributed devices with individual interests.



What is unique about distributed systems?

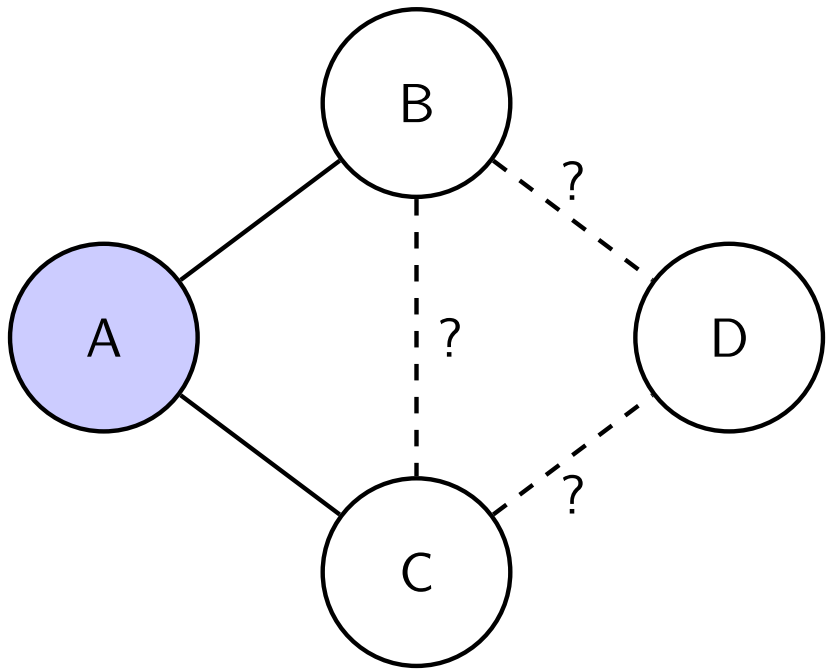
- Typical properties of distributed systems
  - $\Rightarrow$ Tolerate failures in individual devices

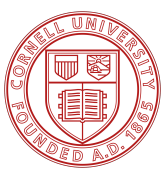




What is unique about distributed systems?

- Typical properties of distributed systems
  - $\Rightarrow$ System structure (topology, latency) not known in advance

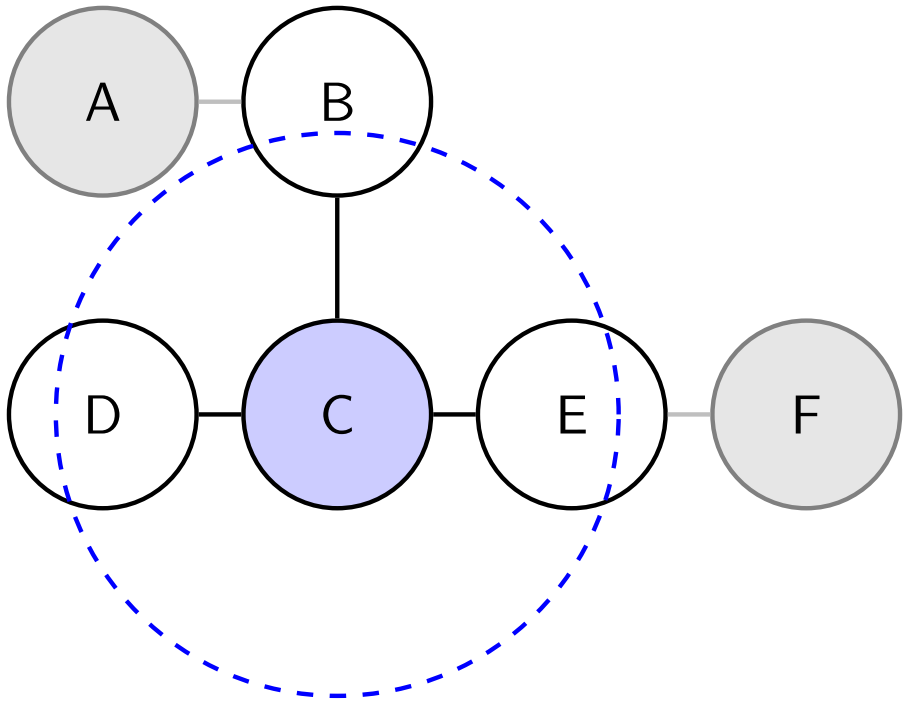


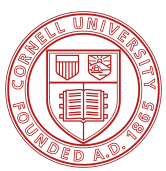


What is unique about distributed systems?

- Typical properties of distributed systems

  $\Rightarrow$ Each device has only a limited, incomplete view of the system

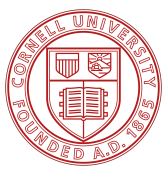




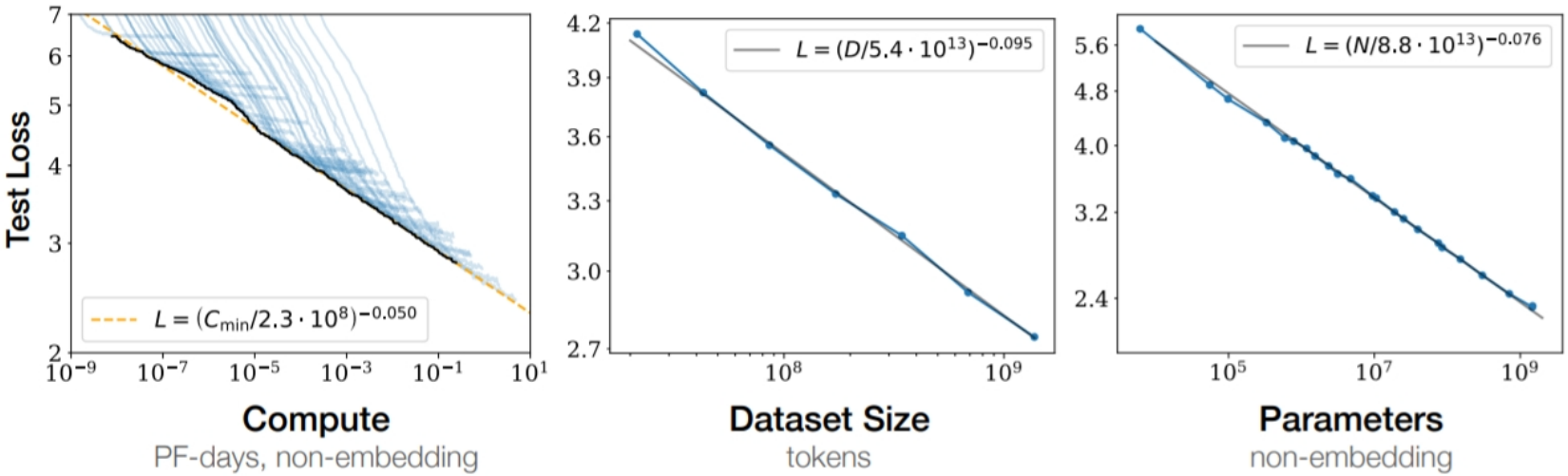
Why distributed optimization?

Why we need to perform AI and optimization in distributed systems?

For **scalability** and **trustworthiness**!

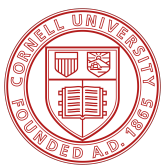


Recent success of generative AI and large models



- Better LLM = **Larger dataset**, **Bigger model**, **Longer training**

Scaling Laws from "Training Compute-Optimal Large Language Models, 2022"



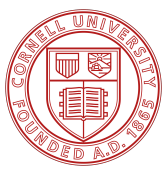
Large-scale distributed model training

Why taking this course at this time?

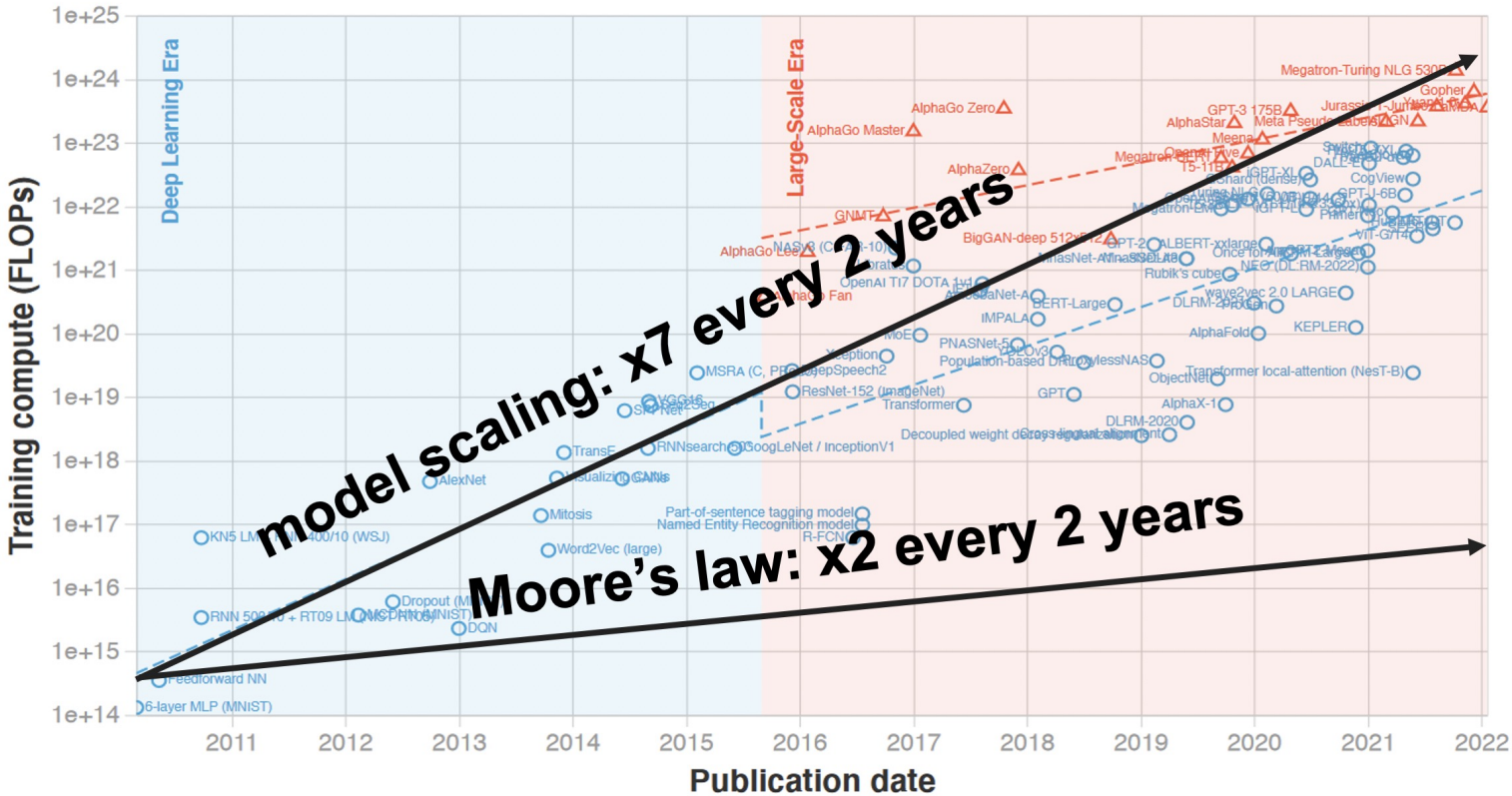
- Large-scale model training: Parallel and distributed training
- Private distributed learning: Federated learning

Course Content and Importance in ECE

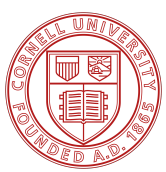
Course workload and grading



Training large models requires large compute



Sevilla et al., "Compute trends across three eras of machine learning," **IJCNN 2022.**



Thousands of GPUs are needed to train LLMs

2 example models

GPT-3 (2020)

50,257 vocabulary size
2048 context length
175B parameters
Trained on 300B tokens

Model Name	n_{params}	n_{layers}	d_{model}	n_{heads}	d_{head}	Batch Size	Learning Rate
GPT-3 Small	125M	12	768	12	64	0.5M	6.0×10^{-4}
GPT-3 Medium	350M	24	1024	16	64	0.5M	3.0×10^{-4}
GPT-3 Large	760M	24	1536	16	96	0.5M	2.5×10^{-4}
GPT-3 XL	1.3B	24	2048	24	128	1M	2.0×10^{-4}
GPT-3 2.7B	2.7B	32	2560	32	80	1M	1.6×10^{-4}
GPT-3 6.7B	6.7B	32	4096	32	128	2M	1.2×10^{-4}
GPT-3 13B	13.0B	40	5140	40	128	2M	1.0×10^{-4}
GPT-3 175B or "GPT-3"	175.0B	96	12288	96	128	3.2M	0.6×10^{-4}

Table 2.1: Sizes, architectures, and learning hyper-parameters (batch size in tokens and learning rate) of the models which we trained. All models were trained for a total of 300 billion tokens.

Training: (rough order of magnitude to have in mind)

- O(1,000 - 10,000) V100 GPUs
- O(1) month of training
- O(1-10) $M

LLaMA (2023)

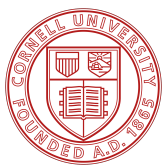
32,000 vocabulary size
2048 context length
65B parameters
Trained on 1-1.4T tokens

params	dimension	n heads	n layers	learning rate	batch size	n tokens
6.7B	4096	32	32	$3.0e^{-4}$	4M	1.0T
13.0B	5120	40	40	$3.0e^{-4}$	4M	1.0T
32.5B	6656	52	60	$1.5e^{-4}$	4M	1.4T
65.2B	8192	64	80	$1.5e^{-4}$	4M	1.4T

Table 2: **Model sizes, architectures, and optimization hyper-parameters.**

Training for 65B model:

- 2,048 A100 GPUs
- 21 days of training
- $5M



Recent GPU clusters scale with more than 10000 GPUs

MegaScale: Scaling Large Language Model Training to More Than 10,000 GPUs

Ziheng Jiang[1,*] Haibin Lin[1,*] Yinmin Zhong[2,*] Qi Huang[1] Yangrui Chen[1] Zhi Zhang[1]
Yanghua Peng[1] Xiang Li[1] Cong Xie[1] Shibiao Nong[1] Yulu Jia[1] Sun He[1] Hongmin Chen[1]
Zhihao Bai[1] Qi Hou[1] Shipeng Yan[1] Ding Zhou[1] Yiyao Sheng[1] Zhuo Jiang[1]
Haohan Xu[1] Haoran Wei[1] Zhang Zhang[1] Pengfei Nie[1] Leqi Zou[1] Sida Zhao[1]
Liang Xiang[1] Zherui Liu[1] Zhe Li[1] Xiaoying Jia[1] Jianxi Ye[1] Xin Jin[2,†] Xin Liu[1,†]

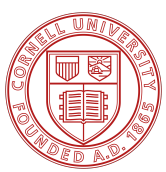
[1]*ByteDance* [2]*Peking University*

Feb. 23, 2024

Abstract

We present the design, implementation and engineering experience in building and deploying MegaScale, a production system for training large language models (LLMs) at the scale of more than 10,000 GPUs. Training LLMs at this scale brings unprecedented challenges to training efficiency and stability. We take a full-stack approach that co-designs the algorithmic and system components across model block and optimizer design, computation and communication overlapping, operserving billions of users, we have been aggressively integrating AI into our products, and we are putting LLMs as a high priority to shape the future of our products.

Training LLMs is a daunting task that requires enormous computation resources. The scaling law [3] dictates that the model size and the training data size are critical factors that determine the model capability. To achieve state-of-the-art model capability, many efforts have been devoted to train large models with hundreds of billions or even trillions of parameters on hundreds of billions or even trillions of tokens. For example, GPT-3 [4] has 175 billion parameters and



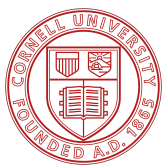
Llama 3 uses 24000 GPUs in LLM training

Apr. 18, 2024

Build the future of AI with Meta Llama 3

To train our largest Llama 3 models, we combined three types of parallelization: data parallelization, model parallelization, and pipeline parallelization. Our most efficient implementation achieves a compute utilization of over 400 TFLOPS per GPU when trained on 16K GPUs simultaneously. We performed training runs on two custom-built 24K GPU clusters. To maximize GPU uptime, we developed an advanced new training stack that automates error detection, handling, and maintenance. We also greatly improved our

[Introducing Meta Llama 3: The most capable openly available LLM to date]



Distributed training is extremely challenging

Stability

Definition: maintain consistent performance under a steady load and to recover quickly from faults.

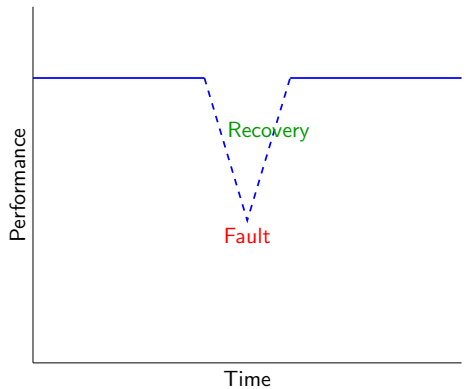


Figure: The system hits a snag but quickly returns to normal.

Scalability

Definition: A system's ability to handle a growing amount of load by adding more GPUs.

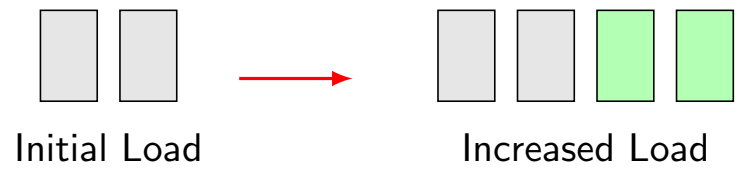
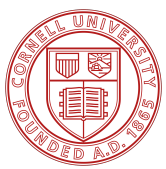
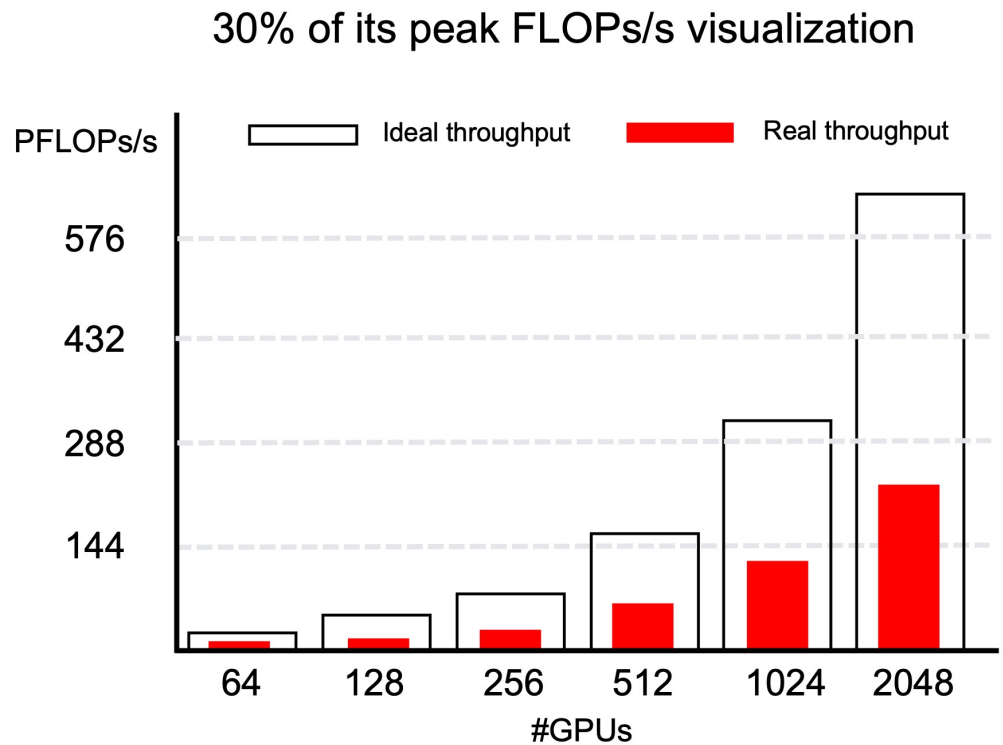


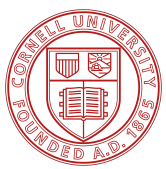
Figure: As load increases, we add more GPUs (green) to maintain performance.



Distributed training is extremely challenging

- The communication overhead and GPU idle time hamper scalability
- Each GPU can only achieve 30%-55% of its peak compute power
- The system achieves 30% scalability - inefficient





Privacy-preserving distributed learning

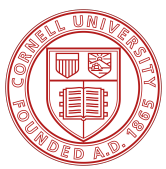
Why taking this course at this time?

Large-scale model training: Parallel and distributed training

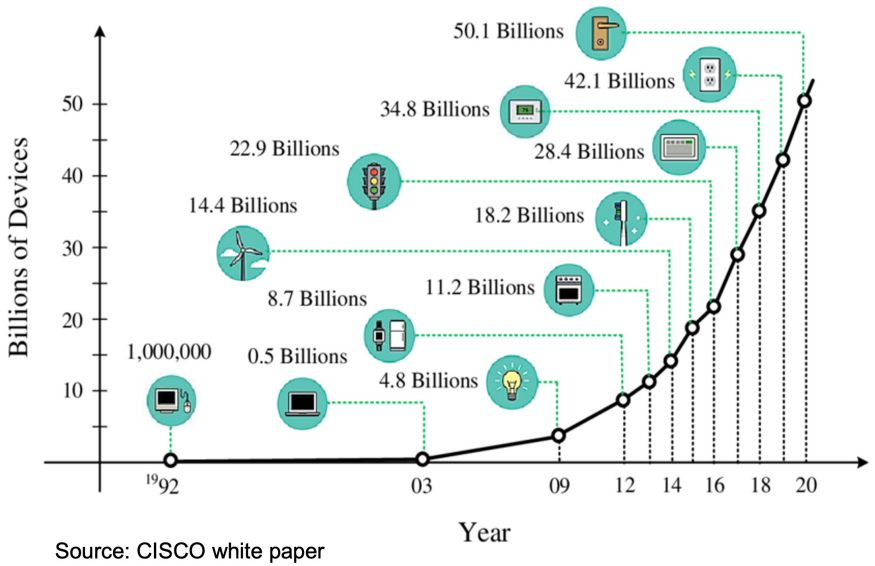
Private distributed learning: Federated learning

Course Content and Importance in ECE

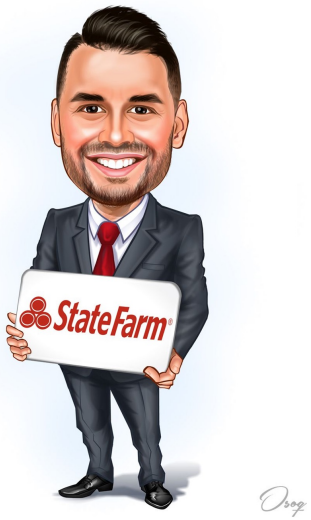
Course workload and grading



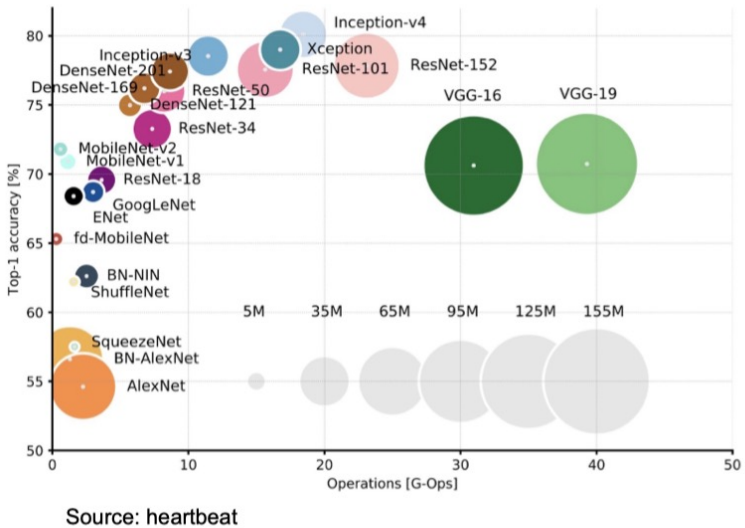
Where are the high-quality data today?



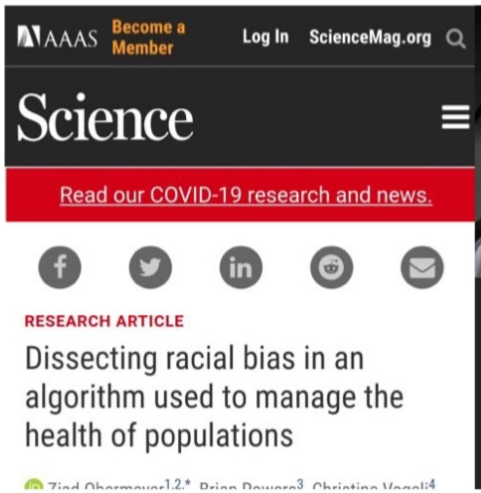
Source: CISCO white paper

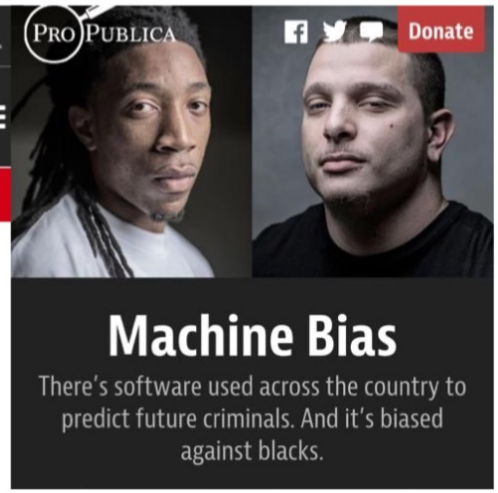


The importance of data sharing



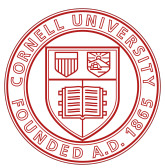
Source: heartbeat



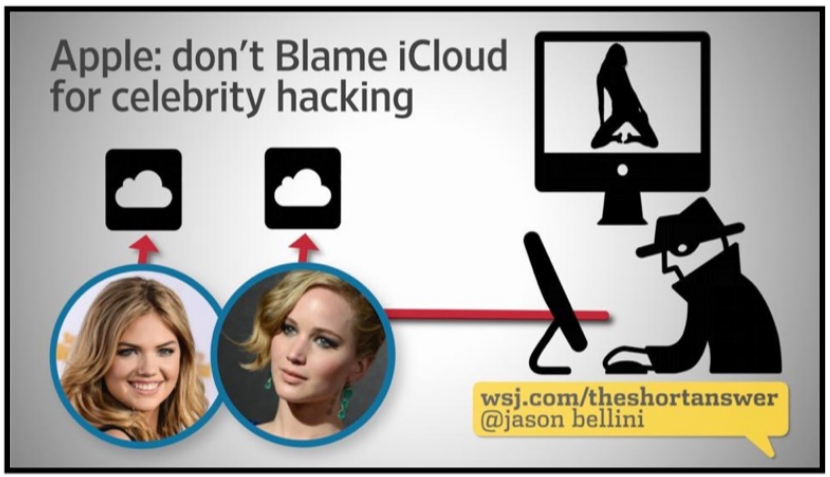


Train large AI/ML models

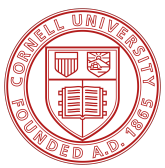
Reduce data bias



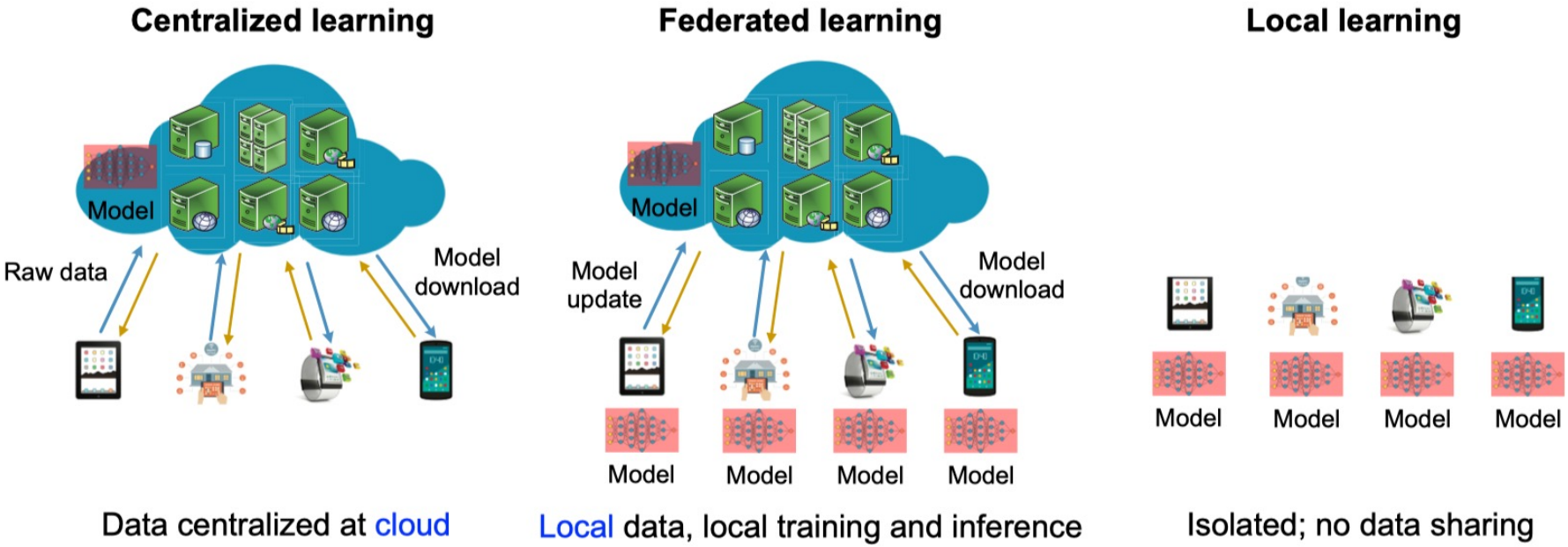
Stringent data privacy regulation



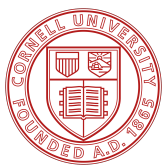
Picture credit: WSJ, The Economist



A representative paradigm - Federated learning



B. McMahan and D. Ramage, "Federated learning: Collaborative machine learning without centralized training data," *Google Research Blog*, April 2017.

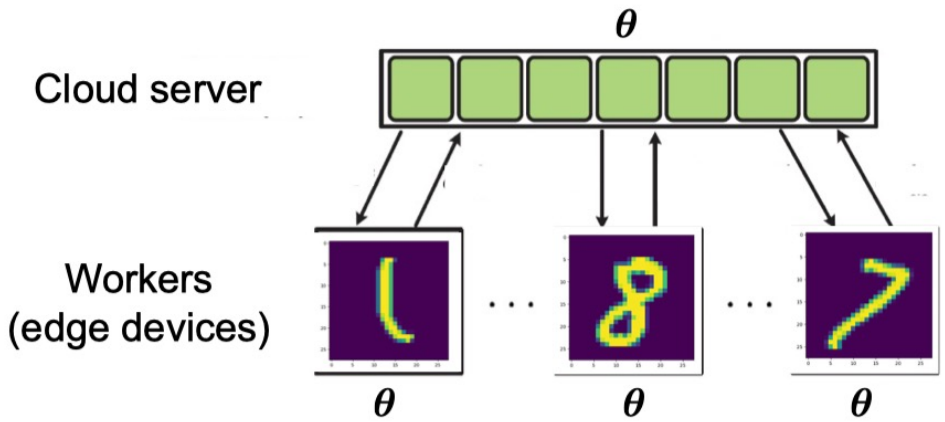


Distributed optimization for federated learning

Unifying model

$$\underset{\boldsymbol{\theta}\in\mathbb{R}^d}{\text{minimize}}\ \mathcal{L}(\boldsymbol{\theta}) \quad \text{with} \quad \mathcal{L}(\boldsymbol{\theta}) := \sum_{m\in\mathcal{M}} \mathcal{L}_m(\boldsymbol{\theta})$$

e.g., loss of classification; (nonlinear) regression



- Worker $m \in \mathcal{M} := \{1, \ldots, M\}$ keeps local data $\{\mathbf{x}_n, y_n,\ n \in \mathcal{N}_m\}$

- Iterative solvers: gradient descent (GD), stochastic GD, Adam, momentum methods …

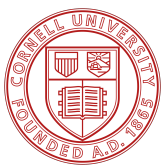
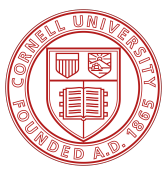


Table of Contents

Why taking this course at this time?
- Large-scale model training: Parallel and distributed training
- Private distributed learning: Federated learning

Course Content and Importance in ECE

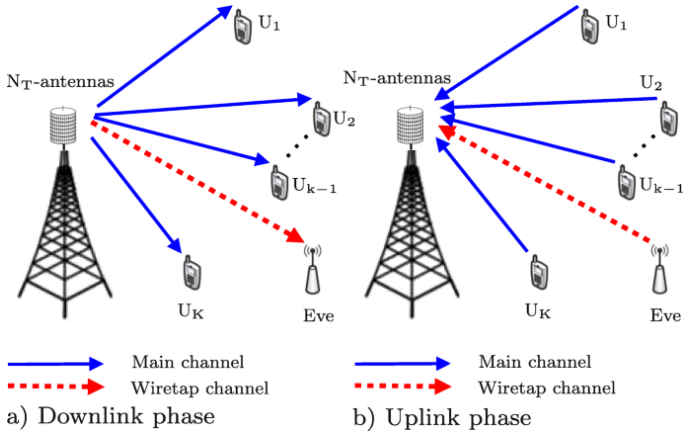
Course workload and grading



Distributed optimization appears in many ECE areas

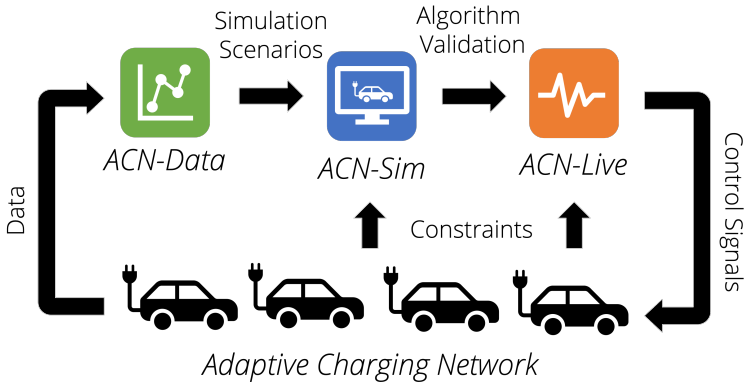
Parameter estimation

Wireless physical-layer designs



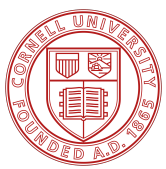
Linear quadratic control

EV scheduling and load balancing



But different from optimization in other ECE areas

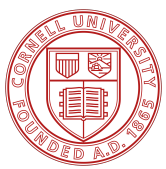
Aspect	Traditional ECE Optimization	Machine Learning Optimization
Objective basis	Function derived from system dynamics, physics, signal models, or cost structures. Often explicit.	Primarily empirical risk (average loss on training data) plus regularization terms to aid generalization.
Scale & complexity	Varies, but often lower/moderate dimension. Can be convex, linear, or non-linear with known structure.	Typically high-dimensional, non-convex and often stochastic (due to data batches).
Key challenge	Ensuring model fidelity, handling physical/operational constraints, computational tractability.	Ensuring high efficiency in terms of model and data size; ensuring the learned model generalizes rather than memorizes.
Dominant methods	Diverse toolbox: Convex Optimization, Gradient Descent, DP, Kalman Filters, Simplex, etc.	Heavily relies on Stochastic Gradient Descent and its adaptive variants due to scale & data structure.



What will be covered in this class?

Learning and Optimization in Distributed Settings

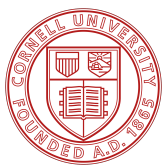
- Abstract yet fundamental models under distributed systems
- (Mostly) generic yet simple optimization algorithms
  - $\Rightarrow$ Use several examples to illustrate their utility
- Rigorous analysis of distributed optimization algorithms
- Numerical simulations using toolboxes (homework + project)



Alert!!!

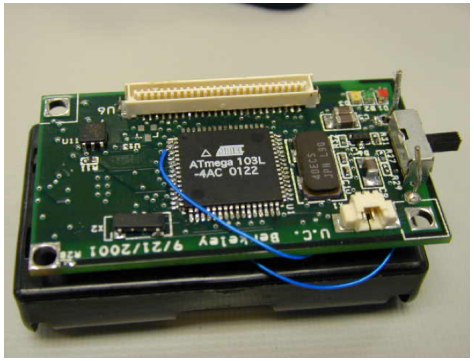
Neither about **hardware implementation** nor **system designs**.

A course about **algorithms and applications** of distributed AI and ML.

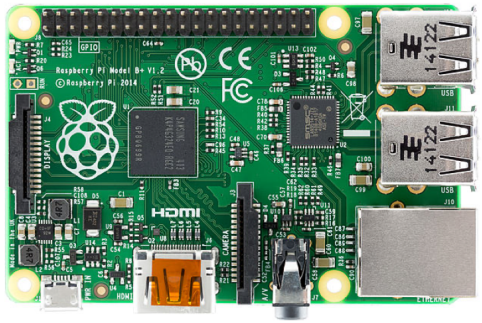


What will not be covered in this class?

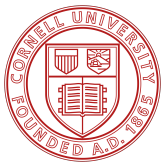
- Hardware wireless sensor
- Cost $100-400



- Raspberry Pi



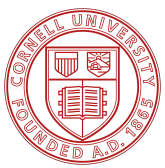
Bottom line: no physical systems; no hand-on experiences using hardware



Recommended textbooks and monographs

- Leon Bottou, Frank E. Curtis, and Jorge Nocedal, "*Optimization Methods for Large-Scale Machine Learning*," SIAM Review

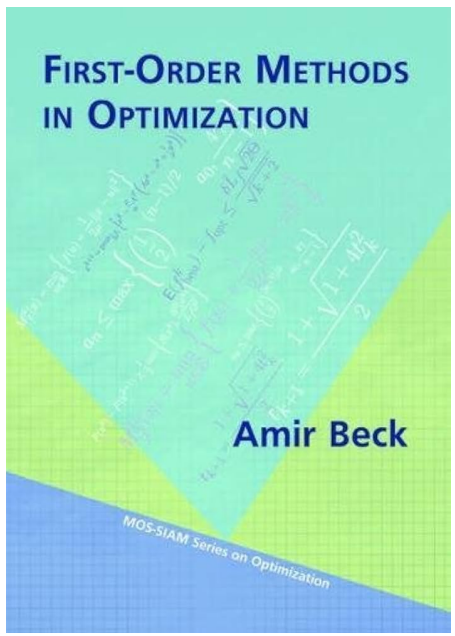
  This monograph provides a focused overview and synthesis of stochastic gradient methods (SGD) specifically tailored to the challenges and structure of large-scale machine learning problems. It also excels at discussing the interplay between optimization algorithm design and ML-specific goals like generalization.

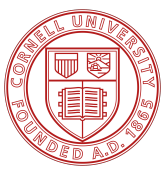


Recommended textbooks and monographs

- Amir Beck, "*First-Order Methods in Optimization*," SIAM Series Optm. This book heavily emphasizes the gradient-based algorithms

  (Gradient Descent, Momentum, Nesterov Acceleration, Proximal Gradient methods) used in large-scale ML training. It provides the foundational theory for why these algorithms work.

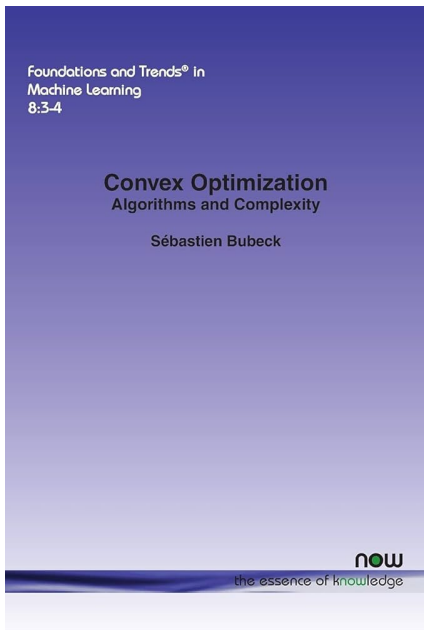


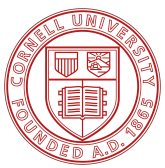


Recommended textbooks and monographs

- Sebastien Bubeck, "*Convex Optimization: Algorithms and Complexity*," Foundations and Trends in Optimization

  This book has comprehensive coverage of convex optimization theory with a strong emphasis on algorithmic complexity. It delves into understanding the efficiency limits (lower bounds) and performance guarantees (upper bounds) of algorithms.

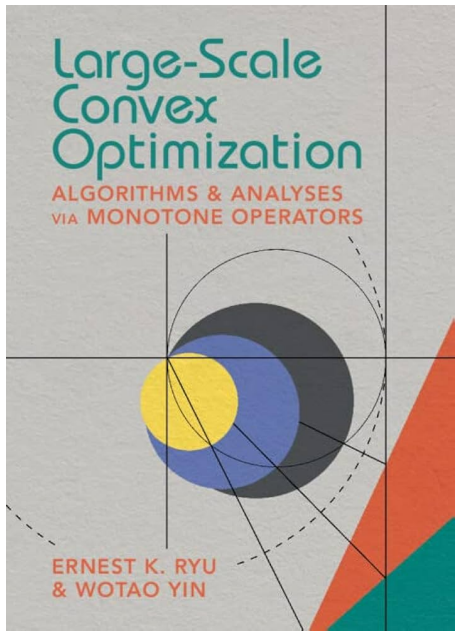


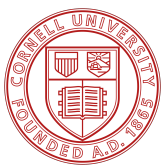


Recommended textbooks and monographs

- Ernest Ryu, Wotao Yin, "*Large-Scale Convex Optimization: Algorithms & Analyses via Monotone Operators*," Cambridge University Press.

  This books is unique in its use of monotone operator theory as a unifying framework to analyze a vast array of optimization algorithms, particularly those relevant to large-scale problems. This includes many splitting methods (like ADMM, Douglas-Rachford) often used for structured optimization problems common in signal processing, statistics, and certain areas of ML.





Four thematic blocks (*represents optional content)

Basics of probability, machine learning, and empirical risk minimization

(I) First-order optimization (4 lectures)

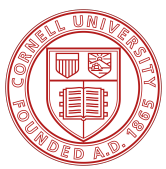
- Gradient methods
- Subgradient methods
- Accelerated gradient methods

Reading: [Bubeck, Ch. 3.1-3.5, 3.7, 4.1-4.3; Beck Ch. 3, 5, 9 ,10]

(II) Stochastic and nonconvex optimization (4 lectures)

- Stochastic approximation and stochastic gradients
- Finite-sum minimization and variance reduction
- Escaping saddle point methods

Reading: [Bubeck, Ch. 6; Beck Ch. 8; and reference papers]



Four thematic blocks (cont'd)

(III) Advanced topics on distributed optimization (8 lectures)

- Consensus averages and gossips
- Distributed optimization: Local averages and decentralized gradients
- Alternating direction methods of multiplier (ADMM)
- Multi-level optimization: implicit gradient and penalty methods*

Reading: [Ryu and Yin, Ch. 6, 8, 11; Beck Ch. 15; and reference papers]

(IV) Applications to distributed learning and computing (8 lectures)

- Federated learning over wireless networks
- Large language model (LLM) pre-training and post-training
- Analog in-memory training and inference
- Hardware-aware low-precision training*

Reading: [Suggested reference papers]

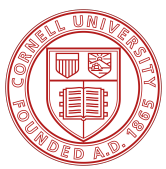
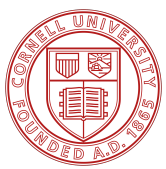


Table of Contents

Why taking this course at this time?
- Large-scale model training: Parallel and distributed training
- Private distributed learning: Federated learning

Course Content and Importance in ECE

Course workload and grading



Prerequisites

(I) **Calculus, linear algebra and probability**

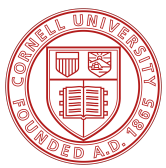
- Integrals, derivatives, limits, infinite series
- Vector/matrix notation, systems of linear equations, eigenvalues
- Expectations, moments (mean, variance), and conditional probability

(II) **Machine learning concepts**

- Familiarity with supervised learning (e.g., regression, classification, empirical risk minimization)
- No advanced background required - core ideas will be reviewed

(III) **Programming in Python or Matlab**

- Needed for homework and projects
- If you know programming you can learn Matlab in one afternoon



Course grading

(I) Homework sets worth 30 points (6 points per homework)

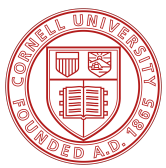
- A hands-on "guided exploration" where students implement large-scale optimization algorithms based on course materials and other resources in real-world applications using Python/MATLAB.

(II) One in-class exam worths 40 points

- **October 27**, closed book, 2 pages hand-written notes allowed

(III) Comprehensive assignments worth 30 points

- Literature review: culminate in a well-structured written report, demonstrating the ability to contextualize and assess existing work
- Or, Course project: apply concepts from the course to a practical or theoretical problem in large-scale distributed optimization



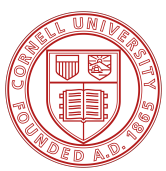
About homework - Analytical

(A) Analytical: Proximal operator of a convex function g is defined as:

1. $$\text{prox}_g(v) = \text{argmin}_x \; g(x) + \frac{1}{2}||x - v||_2^2. \quad \text{(Prox)}$$
2. Explain concisely why (Prox) has a unique solution.
3. What optimality condition must be satisfied for x^* to be the minimizer of (Prox)?
4. Let $\alpha \geq 0$. Predict the behavior of

$$\text{prox}_{\alpha g}(v) = \text{argmin}_x \; \alpha g(x) + \frac{1}{2}||x - v||_2^2$$

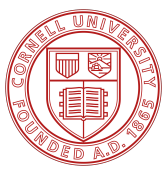
as $\alpha \to 0$ and as $\alpha \to \infty$.



About homework - Programming

(B) Programming: Consider the LASSO problem:

1. $$x^* = \text{argmin}_{x \in \mathbf{R}^d} \frac{1}{2}||Ax - b||_2^2 + \lambda \|x\|_1 \quad \text{(LASSO)}$$
2. Write the expression for x_{t+1} in the proximal gradient descent method (also known as ISTA - Iterative Shrinkage-Thresholding Algorithm) for solving (LASSO), using the soft-shrinkage operator $S_\alpha(\cdot)$. Assume stepsizes α_t are given.
3. Write a Python function '[xT, objhist] = istaLasso(A, b, lambda-reg, x0, alpha, T)' that returns x_T, the T-th iterate of ISTA for (LASSO) with initial iterate 'x0' and constant stepsize 'alpha'. It should also return a vector 'objhist' containing the objective values at iterates $x_0, \ldots, x_T$. Use your 'softShrink' function.



About course project

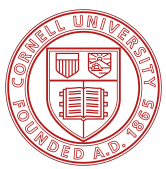
This project is designed to complement your understanding of the material through in-depth reading and/or hands-on research.

(A) Educational projects (5290 students)

- Develop detailed lecture notes and a 20-minute corresponding presentation on a topic related to but not covered in the course. This must include an empirical evaluation of the methods discussed. Educational projects should include a problem set, complete with solutions, designed to test understanding.

(B) Research projects (7290 students)

- Conduct original research related to the course content. This may involve theoretical analysis, algorithm development, or novel applications. Applying existing methods to a new dataset without a significant technical contribution is not acceptable. Research results will be presented in a 20-minute presentation and in a report.



About course project (cont'd)

This project is designed to complement your understanding of the material through in-depth reading and/or hands-on research.

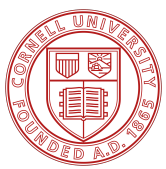
The project consists of several key milestones:

(A) Progress Report

- Submit a draft of your semi-final report. Research projects should include a complete introduction, related work, and preliminary results. Educational projects should provide detailed lecture notes, including mathematical details and problem set drafts.

(B) Final Deliverables

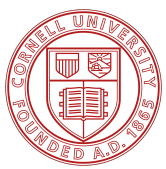
- A GitHub repository with reproducible code and data scripts.
- A slide deck for your 20-minute presentation.
- A final workshop-style report.



Final grading

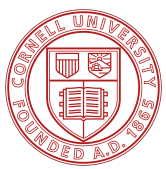
- Overall grading
- At least 60 points are required for passing (C grade)
- At least 75 points to get B grade
- At least 90 points to get A grade

  $\Rightarrow$ Goal is for everyone to get an A



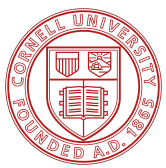
Online learning platforms

- Canvas - Major hub (https://canvas.cornell.edu/)
  - $\Rightarrow$ Post notes, lecture slides, and recommended readings
  - $\Rightarrow$ Post major announcements
- Ed Discussion
  - $\Rightarrow$ Interactive discussion for students and instructors
  - $\Rightarrow$ Encourage peer-to-peer support while instructors can endorse
- Gradescope
  - $\Rightarrow$ Support for paper-based scanning and online submissions
  - $\Rightarrow$ Grade problem sets, quizzes, and exams with rubrics



Recap and preview

- What we have talked about today?
  - $\Rightarrow$ Why we want to learn this course?
  - $\Rightarrow$ What will be covered in this course?
  - $\Rightarrow$ What will not be covered in this course?
  - $\Rightarrow$ How to get an A in this course?



Fill out course survey

To help us tailor this course to your background, please take a few minutes to complete our brief, anonymous welcome survey.