Distributed Machine Learning

COS 518: Advanced Computer Systems
Lecture 13

Michael Freedman
(Slides heavily based on Daniel Suo)

Outline

- What is machine learning?
- Why is machine learning hard in parallel / distributed systems?
- A brief history of what people have done

Some definitions

- Give computers the ability to act without being explicitly programmed
- Program that learns from experience to perform some task better
- More practical: predictive models that have some parameters that are informed by data

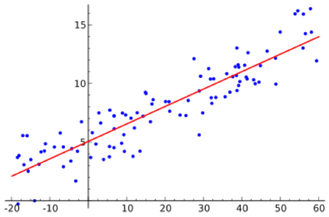
Three broad classes

Primarily classified by the 'feedback'

- **Supervised:** use example inputs with corresponding 'answers' (labels) -> learn mapping
- **Unsupervised**: find structure in data without labels (but is anything ever *really* unsupervised?)
- **Reinforcement**: learn policy of behavior in a dynamic environment from rewards / punishments

Example: Linear regressions

- Could be thought of as a machine learning algorithm
- Want to make predictions
- Determine parameters from data



Example: Linear regressions

- **Model**: $y = \boldsymbol{w} \cdot \boldsymbol{x} + \boldsymbol{b}$
- **Training**:
  - Objective / cost / loss: squared error

$$\mathcal{L}(\beta) = \sum_{i=1}^{N} (y_i - f(x_i))^2$$

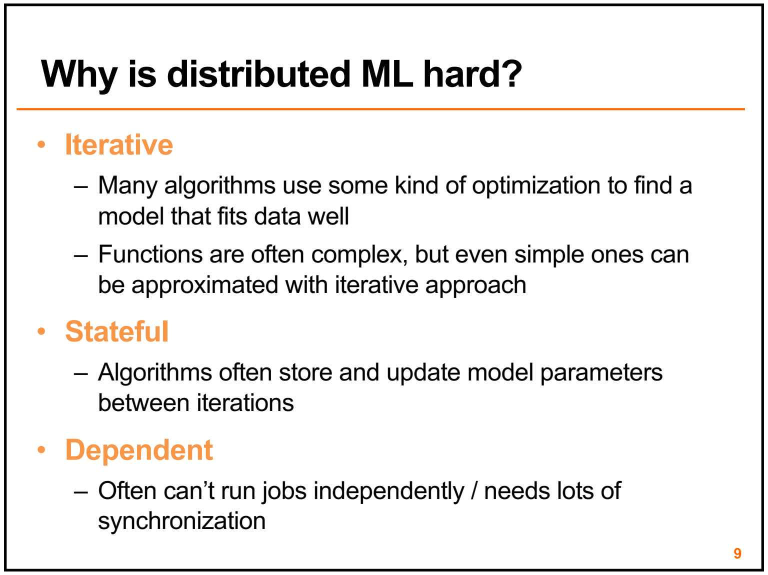
  - Training: minimize the sum squared error
- **Inference**: just plug inputs into our model with parameters from training

But isn’t this statistics?

- Lots of overlap, but some notable differences
- **Partly interest**
  - Statistics: survey design, sampling, industrial statistics
  - ML: what is learning, what can be learned
- **Partly cultural**
  - Statistics: complicated models we can explain but don’t work
  - ML: whatever improves prediction performance goes

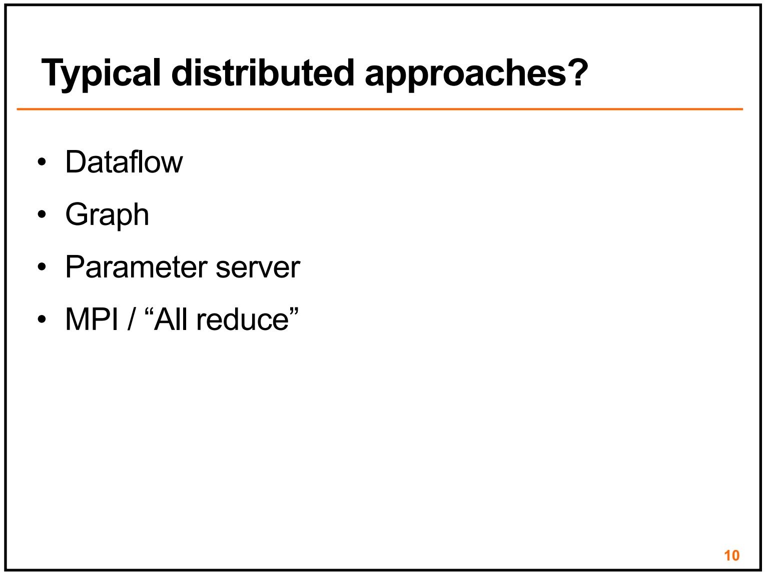
Representation learning

- What if we don’t have data collected and organized into a bunch of features?
- What if features are hard / impossible to define?
- Quick example: k-means clustering
  - Initialize centroids
  - Cluster
  - Re-compute centroid
  - Repeat
- We can create hierarchies of representations
- Deep learning!



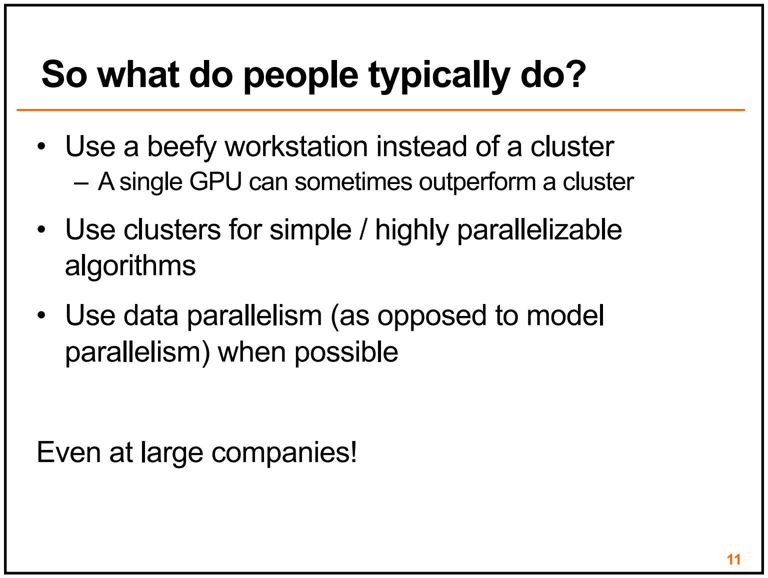
Why is distributed ML hard?

- **Iterative**
  - Many algorithms use some kind of optimization to find a model that fits data well
  - Functions are often complex, but even simple ones can be approximated with iterative approach
- **Stateful**
  - Algorithms often store and update model parameters between iterations
- **Dependent**
  - Often can't run jobs independently / needs lots of synchronization



Typical distributed approaches?

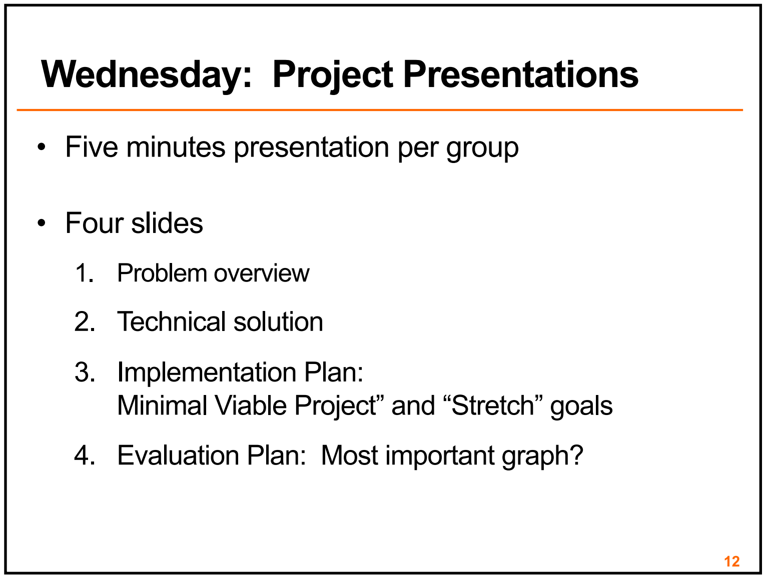
- Dataflow
- Graph
- Parameter server
- MPI / "All reduce"



So what do people typically do?

- Use a beefy workstation instead of a cluster
  - A single GPU can sometimes outperform a cluster
- Use clusters for simple / highly parallelizable algorithms
- Use data parallelism (as opposed to model parallelism) when possible

Even at large companies!



Wednesday: Project Presentations

- Five minutes presentation per group
- Four slides
  1. Problem overview
  2. Technical solution
  3. Implementation Plan:
     Minimal Viable Project" and "Stretch" goals
  4. Evaluation Plan: Most important graph?