Case Study 3: fMRI Prediction

“Scalable” LASSO Solvers:
Parallel SCD (Shotgun)
Parallel SGD
Averaging Solutions
ADMM

Scaling Up LASSO Solvers

• A simple SCD for LASSO (Shooting)
  – Your HW, a more efficient implementation! 😊
  – Analysis of SCD
• Parallel SCD (Shotgun)
• Other parallel learning approaches for linear models
  – Parallel stochastic gradient descent (SGD)
  – Parallel independent solutions then averaging
• ADMM
Stochastic Coordinate Descent for LASSO (aka Shooting Algorithm)

• Repeat until convergence
  – Pick a coordinate \( j \) at random
    • Set:
      \[
      \hat{\beta}_j = \begin{cases} 
      (c_j + \lambda)/a_j & c_j < -\lambda \\
      0 & c_j \in [-\lambda, \lambda] \\
      (c_j - \lambda)/a_j & c_j > \lambda 
      \end{cases}
      \]
    • Where:
      \[
      a_j = 2 \sum_{i=1}^{N} (x_i^j)^2 \\
      c_j = 2 \sum_{i=1}^{N} x_i^j (y_i - \beta_{-j}' x_i)
      \]

Shotgun: Parallel SCD [Bradley et al ’11]

Lasso:
\[
\min_{\beta} F(\beta) \quad \text{where} \quad F(\beta) = \|X\beta - y\|_2^2 + \lambda \|\beta\|_1
\]

Shotgun (Parallel SCD)
While not converged,
  • On each of \( P \) processors,
    • Choose random coordinate \( j \),
    • Update \( \beta_j \) (same as for Shooting)
Is SCD inherently sequential?

Lasso: \( \min_{\beta} F(\beta) \) where \( F(\beta) = ||X\beta - y||_2^2 + \lambda \| \beta \|_1 \)

Coordinate update:
\[ \beta_j \leftarrow \beta_j + \delta \beta_j \]
(closed-form minimization)

Collective update:
\[ \Delta \beta = \begin{pmatrix} \delta \beta_1 \\ 0 \\ 0 \\ \delta \beta_j \\ 0 \end{pmatrix} \]

there are interferences in these update if features are correlated.
Can we quantify this?

Convergence Analysis

Lasso: \( \min_{\beta} F(\beta) \) where \( F(\beta) = ||X\beta - y||_2^2 + \lambda \| \beta \|_1 \)

Theorem: Shotgun Convergence
Assume \( P < p/\rho + 1 \)
where \( \rho = \) spectral radius of \( X^T X \)

\[
E\left[F(\beta^{(T)})\right] - F(\beta^*) 
\leq \frac{p\left(\frac{1}{2} \| \beta^* \|_2^2 + F(\beta^{(0)})\right)}{TP}
\]

Nice case:
Uncorrelated features
\( \rho = \frac{1}{\mu} \Rightarrow P_{\text{max}} = \frac{1}{\mu} \)

Bad case:
Correlated features
\( \rho = \frac{1}{\mu} \Rightarrow P_{\text{max}} = \frac{1}{\mu} \) (at worst)

\( \uparrow \) linear speed up, up to \( P \) processors
Stepping Back…

- Stochastic coordinate ascent
  - Optimization:
  - Parallel SCD:
  - Issue:
  - Solution:

- Natural counterpart:
  - Optimization:
  - Parallel
  - Issue:
  - Solution:

Parallel SGD with No Locks
[e.g., Hogwild!, Niu et al. ‘11]

- Each processor in parallel:
  - Pick data point i at random
  - For j = 1...p:

- Assume atomicity of:
Addressing Interference in Parallel SGD

- Key issues:
  - Old gradients
  - Processors overwrite each other’s work

- Nonetheless:
  - Can achieve convergence and some parallel speedups
  - Proof uses weak interactions, but through sparsity of data points

Problem with Parallel SCD and SGD

- Both Parallel SCD & SGD assume access to current estimate of weight vector

- Works well on shared memory machines

- Very difficult to implement efficiently in distributed memory

- Open problem: Good parallel SGD and SCD for distributed setting...
  - Let’s look at a trivial approach
Simplest Distributed Optimization Algorithm Ever Made

• Given $N$ data points & $P$ machines
• Stochastic optimization problem:
• Distribute data:

• Solve problems independently

• Merge solutions

• Why should this work at all????

For Convex Functions...

• Convexity:

• Thus:
Hopefully...

- Convexity only guarantees:

- But, estimates from independent data!

Analysis of Distribute-then-Average

[Zhang et al. ‘12]

- Under some conditions, including strong convexity, lots of smoothness, etc.
- If all data were in one machine, converge at rate:

- With $P$ machines, converge at a rate:
Tradeoffs, tradeoffs, tradeoffs,...

- Distribute-then-Average:
  - “Minimum possible” communication
  - Bias term can be a killer with finite data
    - Issue definitely observed in practice
  - Significant issues for L1 problems:

- Parallel SCD or SGD
  - Can have much better convergence in practice for multicore setting
  - Preserves sparsity (especially SCD)
  - But, hard to implement in distributed setting

Alternating Directions Method of Multipliers

- A tool for solving convex problems with separable objectives:

- LASSO example:

- Know how to minimize $f(\beta)$ or $g(\beta)$ separately
ADMM Insight

• Try this instead:

• Solve using method of multipliers
• Define the augmented Lagrangian:

  – Issue: L2 penalty destroys separability of Lagrangian
  – Solution: Replace minimization over \((x, z)\) by alternating minimization

ADMM Algorithm

• Augmented Lagrangian:

  \[ L_\rho(x, z, y) = f(x) + g(z) + y^T(x - z) + \frac{\rho}{2}||x - z||_2^2 \]

• Alternate between:

  1. \(x \leftarrow\)
  2. \(z \leftarrow\)
  3. \(y \leftarrow\)
ADMM for LASSO

\[ L_\rho(x, z, y) = f(x) + g(z) + y^T(x - z) + \frac{\rho}{2}||x - z||^2_2 \]

• Objective:

• Augmented Lagrangian:

\[ L_\rho(\beta, z, \alpha) = \]

• Alternate between:

1. \( \beta \leftarrow \)

2. \( z \leftarrow \)

3. \( \alpha \leftarrow \)

ADMM Wrap-Up

• When does ADMM converge?
  - Under very mild conditions
  - Basically, \( f \) and \( g \) must be convex

• ADMM is useful in cases where
  - \( f(x) + g(x) \) is challenging to solve due to coupling
  - We can minimize
    - \( f(x) + (x-a)^2 \)
    - \( g(x) + (x-a)^2 \)

• Reference
What you need to know

- A simple SCD for LASSO (Shooting)
  - Your HW, a more efficient implementation! 😊
  - Analysis of SCD
- Parallel SCD (Shotgun)
- Other parallel learning approaches for linear models
  - Parallel stochastic gradient descent (SGD)
  - Parallel independent solutions then averaging
- ADMM
  - General idea
  - Application to LASSO