Case Study 3: fMRI Prediction

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

Machine Learning for Big Data
CSE547/STAT548, University of Washington
Emily Fox
February 6th, 2014

©Emily Fox 2014

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

©Emily Fox 2014
Stochastic Coordinate Descent for LASSO (aka Shooting Algorithm)

Repeat until convergence

- Pick a coordinate $j$ at random
  - Set:
    \[ \beta_j = \begin{cases} 
    (c_j + \lambda)/a_j & c_j < -\lambda \\
    0 & c_j \in [-\lambda, \lambda] \\
    (c_j - \lambda)/a_j & c_j > \lambda 
    \end{cases} \]
  - Where:
    \[ a_j = 2 \sum_{i=1}^{N} (x_i^j)^2 \]
    \[ c_j = 2 \sum_{i=1}^{N} x_i^j (y_i - \beta^T x_i^T) \]

Can be done more efficiently. Proof: HW!

Shotgun: Parallel SCD [Bradley et al '11]

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

**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:**
\[
\Delta \beta = \begin{pmatrix}
\delta \beta_j \\
0 \\
0 \\
\delta \beta_j \\
0
\end{pmatrix}
\]

Convergence Analysis

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

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

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

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

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

There are interferences in these updates if features are corr. Can we quantify this?
Stepping Back…

- Stochastic coordinate ascent
  - Optimization: pick a coord. $j$, find min $\beta_j$
  - Parallel SCD:
    - Pick $P$ coord.
    - Issue: coordinates may interfere on $P$ coord.
    - Solution: spectral bound possible interference based $P$

- Natural counterpart: SGD
  - Optimization: pick a datapoint $i$, $\beta \leftarrow \beta - \eta \nabla F(x^i; \beta)$
  - Parallel: pick $P$ datapoints + ind. update $\beta$
  - Issue: can interfere on all coord.
  - Solution: bound interference by exploiting sparsity in $X$

---

Parallel SGD with No Locks

- Each processor in parallel:
  - Pick data point $i$ at random
  - For $j = 1 \ldots p$:
    $$\beta_j \leftarrow \beta_j - \eta \nabla F(x^i; \beta)_j$$

- Assume atomicity of: $\beta_j \leftarrow \beta_j + \alpha$
  - other interferences
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:
  $\min_{\beta} \frac{1}{N} \sum_{i=1}^{N} F(x_i; \beta)$
- Distribute data: $P$ machines solves a problem $D_k$
  $|D_k| = \frac{N}{P} = n$
  Randomly assign data $P_1$, ..., $P_p$
  Solve problems independently
  $\beta^{(k)} = \min_{\beta} \frac{1}{n} \sum_{x \in D_k} F(x; \beta)$
- Merge solutions
  $\bar{\beta} = \frac{1}{P} \sum_k \beta^{(k)}$
- Why should this work at all????

For Convex Functions…

- Convexity:
  $F(\beta_1) + F(\beta_2) \geq F(\bar{\beta})$
- Thus:
  $\max (F(\beta_1), F(\beta_2)) \geq F(\bar{\beta})$
Hopefully…

- Convexity only guarantees:
  \[ F(\hat{\beta}) \leq \max_k F(\beta^k) \]
- But, estimates from independent data!

\[ \hat{\theta}_1, \hat{\theta}_2, \hat{\theta}_3, \hat{\theta}_4 \]

Can we leverage this to improve this bound?

**Analysis of Distribute-then-Average**

[Zhang et al. '12]

- Under some conditions, including strong convexity, lots of smoothness, and more…
  \[ \hat{\beta}_n = \arg\min_{\beta} \frac{1}{N} \sum_{i=1}^{N} F(x_i; \beta) \]
- If all data were in one machine, converge at rate:
  \[ E[\|\hat{\beta}_n - \beta^*\|^2] = O\left(\frac{1}{N}\right) \]
- With \( P \) machines, converge at a rate:
  \[ E[\|\hat{\beta} - \beta^*\|^2] = O\left(\frac{1}{N} + \frac{1}{n^2}\right) \]

\( n \) obs. per machine
\( N \) obs.
\( P \) proc.

\( \frac{n}{P} \) unavoidable "bias" from parallelism

\( \frac{1}{n^2} \) negligible compared to \( \frac{1}{N} \). great parallelism

\( \frac{1}{N} \) hard observation.
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:
    - sparsity patterns in machine $i$ can be very different from those in machine $j$ => average $\beta$ can lose sparsity

- 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:
  \[
  \min_x \sum f(x) + g(x)
  \]

- LASSO example:
  \[
  \min \|y - X\beta\|_2^2 + \lambda \|\beta\| \]

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

- Try this instead:
  \[
  \min_{x,z} \left\{ f(x) + g(z) \right\} \quad \text{s.t.} \quad x = z
  \]
  Still convex!

- Solve using method of multipliers
- Define the augmented Lagrangian:
  \[
  \mathcal{L}_\rho(x,z,y) = f(x) + g(z) + y^T(x-z) + \frac{\rho}{2} \|x-z\|^2
  \]

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

ADMM Algorithm

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

- Alternate between:
  1. \(x \leftarrow \arg\min_x L_\rho(x,z,y)\)
  2. \(z \leftarrow \arg\min_z L_\rho(x,z,y)\)
  3. \(y \leftarrow y + \rho(x-z)\)
ADMM for LASSO

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

- **Objective:**
  \[ \min_{\beta, z} \left\{ \frac{1}{2} ||x - X\beta||^2 + \lambda \|z\|_1 \right\} \text{ s.t. } \beta = z \]

- **Augmented Lagrangian:**
  \[ L_\rho(\beta, z, a) = \frac{1}{2} ||y - X\beta||^2 + \lambda \|z\|_1 + a^T (\beta - z) + \frac{\rho}{2} \|\beta - z\|_2^2 \]

- **Alternate between:**
  1. \[ \beta \leftarrow \arg\min_\beta L_\rho(\beta, z, a) = (X^T X + \rho I)^{-1} (X^T y + \rho z - a) \]
  2. \[ z \leftarrow \arg\min_z L_\rho(\beta, z, a) = S(\beta + \frac{a}{\rho}, \frac{1}{\rho}) \]
  3. \[ a \leftarrow a + \rho (\beta - z) \]

**Reference**


---

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