Hogwild!

Neil Giridharan

March 20, 2019

Outline

- Sparse Separable Cost Functions
- Motivating Examples
- Previous Work
- Hogwild!
- Convergence
- Experiments and Results
- Discussion

Sparse Separable Cost Functions

Let $f : X \subseteq \mathbb{R}^n \rightarrow \mathbb{R}$
Define $f(x) = \sum_{e \in E} f_e(x_e)$
$e \subseteq 1, ..., n$
x_e: The components of x that are indexed by e
When $|E|$ and n are large, but length of x_e is small then f is sparse

Example

$x = (2, 3, 5, 7, 11)$
$e = (3, 5)$
$x_e = (5, 11)$
$f_e(x_e) = |x_e|$
$f_e(x_e) \approx 27.31$

Motivating Examples

- Netflix Prize Problem
- Neutrinos and Muons
- Image Segmentation

Netflix Prize

Figure 1: Netflix Prize Winner

Sparse Matrix Completion Visualization

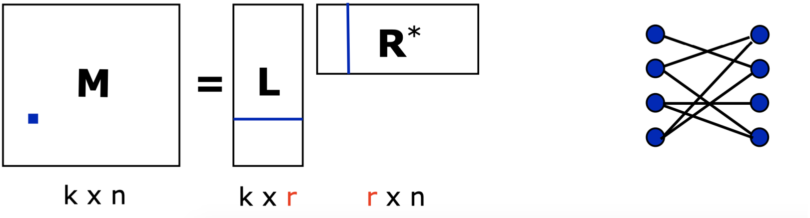


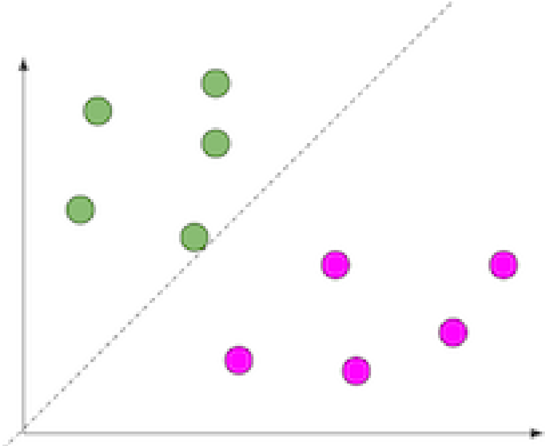
Figure 2: Matrix Completion

Sparse Matrix Completion

M is a $n_r x n_c$ low rank matrix with some entries filled
E contains set of (u, v) which is uth row of L and vth column of R
Idea is to estimate M from the product of LR^* matrices
$minimize_{(L,R)} \sum_{(u,v)\in E}(L_u R_v^* - M_{uv})^2 + \frac{\mu}{2}||L||_F^2 + \frac{\mu}{2}||R||_F^2$
The above regularization term depends on L and R
$minimize_{(L,R)} \sum_{(u,v)\in E}(L_u R_v^* - M_{uv})^2 + \frac{\mu}{2|E_{u-}|}||L_u||_F^2 + \frac{\mu}{2|E_{-v}|}||R_v||_F^2$
$E_{u-} = \{v : (u, v) \in E\}$ and $E_{-v} = \{u : (u, v) \in E\}$

Neutrinos and Muons

- If a neutrino hits a water molecule then it could potentially emit a muon
- Need to distinguish between muons coming from neutrinos and other muons
- Going upward vs Going downward muons



Sparse SVM

Let $E = \{(z_1, y_1), ..., (z_{|E|}, y_{|E|})\}$
$z \in \mathbb{R}^n$, y are labels
x is the hyperplane in the previous figure
Formulate as $minimize_x \sum_{\alpha \in E} max(1 - y_\alpha x^T z_\alpha, 0) + \lambda ||x||_2^2$
Regularization term depends on all of x
Let e_α be non-zero components of z_α
Let d_u be the number of training examples that are non-zero in $u (u = 1, 2, ..., n)$
$minimize_x \sum_{\alpha \in E} max(1 - y_\alpha x^T z_\alpha, 0) + \lambda \sum_{u \in e_\alpha} \frac{x_u^2}{d_u}$

Image Segmentation

- Can reduce image segmentation to a minimum cut problem
- Min cuts have had success even compared to normalized cuts

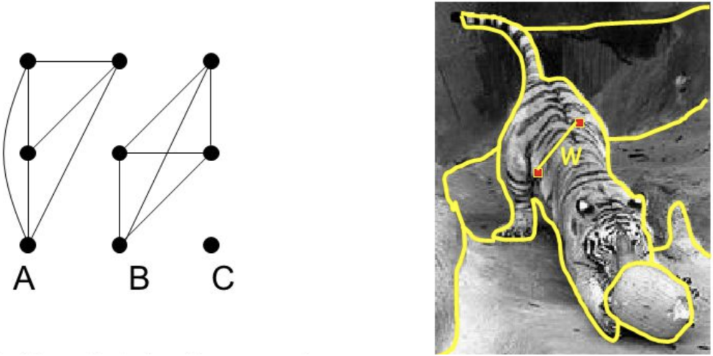


Figure 4: Image Segmentation

Sparse Graph Cuts

Let W be a sparse, non-negative similarity matrix, with edges corresponding to nonzero entries
Each node is associated with a D dimensional simplex in the set S_D
$S_D = \{\zeta \in \mathbb{R}^D : \zeta_v \geq 0 \sum_{v=1}^{D} \zeta_v = 1\}$
$minimize_x \sum_{(u,v)\in E} w_{uv} ||x_u - x_v||_1$, $x_v \in S_D$ for $v = 1, ..., n$

Previous Work

- Approaches are inspired from numerical methods books
- Master/Worker: One processor writes to memory, other processors computes gradients
- Round Robin: One processor updates gradient, tells other processors it's done
- Massive overhead due to lock contention and communication
- What happens with no locking and no communication?

Hogwild!

Assume component wise addition is atomic
$b_v = 1$ on vth component, 0 otherwise
$G_e(x) \in \mathbb{R}^n$, gradient of f_e multiplied by $|E|$
$G_e(x) = 0$ on the components $\neg e$
γ - step size

Algorithm 1: Hogwild!

Sample e uniformly at random from E;
Read current state x_e, evaluate $G_e(x)$;
for $v \in e$ **do**
  $x_v \leftarrow x_v - \gamma b_v^T G_e(x)$
end for

Graph Statistics

$\Omega := max_{e \in E}|e|$: Max cardinality of a hyperedge
$\Delta := \frac{max_{1 \leq v \leq n}|\{e \in E : v \in e\}|}{|E|}$: Normalized Max degree
$\rho := \frac{max_{e \in E}|\{e' \in E : e' \cap e \neq \emptyset\}|}{|E|}$: Normalized Max edge degree

Statistic	Approximation
Ω	$2r$
Δ	$O(\log n/n)$
ρ	$O(\log n/n)$

Table 1: Matrix Completion statistics.

Convergence

Continuous differentiability: $||\nabla f(x') - \nabla f(x)|| \leq L||x' - x||$,
$\forall x', x \in X$
Strongly convex: $f(x') \geq f(x) + (x' - x)^T \nabla f(x) + \frac{c}{2}||x' - x||^2$,
$\forall x', x \in X$
Bounded Gradients: $||G_e(x_e)||_2 \leq M$
Define $D_0 := ||x_0 - x_*||^2$
τ bounds the lag between when gradient is computed and when it's used at a particular step
$\epsilon > 0, v \in (0, 1)$
$k \geq \frac{2LM^2(1+6\tau\rho+6\tau^2\Omega\Delta^{\frac{1}{2}})\log(LD_0/\epsilon)}{c^2 v \epsilon}$
When graph is disconnected ($\Delta = 0, \rho = 0$), rate equals serial convergence rate
$E[f(x_k) - f(x_*)] \leq \epsilon$, x_* unique minimizer

Results

Data set	size (GB)	ρ	Δ	time (s)	speedup
RCV1	0.9	0.44	1.0	9.5	4.5
Netflix	1.5	2.5e-3	2.3e-3	301.0	5.3
KDD	3.9	3.0e-3	1.8e-3	877.5	5.2
Jumbo	30	2.6e-7	1.4e-7	9453.5	6.8
DBLife	3e-3	8.6e-3	4.3e-3	230.0	8.8
Abdomen	18	9.2e-4	9.2e-4	1181.4	4.1

Table 2: Hogwild! statistics

12 core machine: 10 cores for gradients, 2 cores for data shuffling

Experiments

- RR is being destroyed by communication delay
- RR does get a nearly linear speedup when gradient computation time is slow
- Atomic Incremental Gradient (AIG): Locks memory associated with one edge, performs update, unlocks
- Graph Cut experiences a plateau after about 5 cores
- Believe it is an issue with data movement and poor spatial locality

Experiments

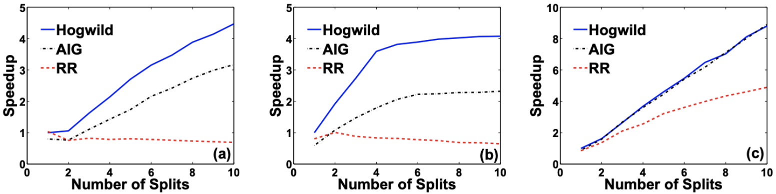


Figure 5: a) RCV1 b) Abdomen c) DBLife

Matrix Completion Experiments

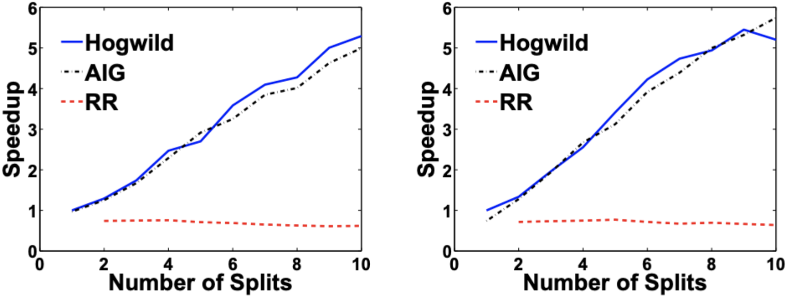


Figure 6: a) Netflix b) KDD

Conclusion

Is Hogwild a good approach to parallelizing machine learning algorithms?

Answer Follow · 27 Request

1 Answer

Kenneth Tran, ML Scientist @ MSR
Answered Apr 3, 2016 · Upvoted by Alberto Bietti, PhD student in machine learning. Former ML engineer

It may be good or bad, depending on your problems.

Good

It is effective, i.e. little to no loss of convergence, and scales well if the features are sparse (i.e. number of non-zeros feature values are relatively small).

Bad

1. Hurts convergence rate if the features are not sparse. As a consequence, doesn't work well on (deep) neural nets because even if the data is sparse, the hidden layers are typically not.
2. Results are not reproducible --> nightmare for testing and debugging.

TL;DR: generally, I'm not a fan of Hogwild-style algorithms although I published one (Scaling Up Stochastic Dual Coordinate Ascent).

Figure 7: One Perspective

Discussion

- Could hybrid locking schemes outperform Hogwild!? Especially when certain terms are accessed more frequently
- What about hybrid algorithms that combine SGD with L-BFGS?
- What would be the impact of better spatial locality? For example could having a biased sampling (rather than uniformly with replacement) improve matrix completion?
- How much do you agree with the person above?