Lecture 20 - scribbles - SAGA

Wednesday, March 29, 2017
10:27

today: • variance reduced SGD
• prox method, catalyst, etc..

Variance reduction idea:

here $f(x) = \frac{1}{n}\sum_i f_i(x)$ $\qquad \nabla f = \frac{1}{n}\sum_i \nabla f_i$ (R.V.)

$= \mathbb{E}[\nabla f_I] \approx \mathbb{E}[X]$ $\quad I \sim \mathrm{Unif}(\{1,\dots,n\})$

SGD approximate $\mathbb{E}[X]$ with just X (i.e. ∇f_I)

general idea:

goal: estimate $\mathbb{E}X$ using M.C. samples

suppose: $\mathbb{E}Y$ is cheap to compute and Y is correlated with X

consider estimator $\Theta_\alpha \triangleq \alpha(X-Y) + \mathbb{E}Y$ ($\alpha \in [0,1]$) to approximate $\mathbb{E}X$
(here X & Y are r.v.)

$\mathbb{E}\Theta_\alpha = \alpha\,\mathbb{E}X + (1-\alpha)\mathbb{E}Y$ $\rightsquigarrow$ unbiased if $\mathbb{E}Y = \mathbb{E}X$ [never happens; otherwise use $\mathbb{E}Y$] $\alpha = 1$

Variance $\mathrm{Var}(\Theta_\alpha) = \alpha^2[\mathrm{Var}(X) + \mathrm{Var}(Y) - 2\mathrm{Cov}(X,Y)]$

α^2: variance reduction aspect

for $\alpha = 1$ (unbiased): $\Theta_\alpha = X + [\mathbb{E}Y - Y]$ (correction)

for SGD setting:

~~[illegible]~~.

$\mathbb{E}X$ is batch gradient ; X is ∇f_i

SAG/SAGA algorithm Y is g_i [past stored gradient]

$\mathbb{E}Y = \frac{1}{n}\sum_i g_i$

SAG algorithm : $\alpha = \frac{1}{n}$

~~standard~~ SAG $w_{t+1} = w_t - \gamma \frac{1}{n} \sum_j g_j^{t+1}$

$\sum [g_i^t + \nabla f_i(w_t) - g_i^t]$

SAG: $w_{t+1} = w_t - \gamma \left[\nabla f_i(w_t) - g_i^t + \frac{1}{n} \sum_j g_j^t \right]$ (biased)

SAGA: $w_{t+1} = w_t - \gamma \left[\nabla f_i(w_t) - g_i^t + \frac{1}{n} \sum_j g_j^t \right]$ (unbiased i.e. $\mathbb{E}[\,] | w_t = \nabla f(w_t)$)

SVRG (stochastic variance reduced gradient): $w_{t+1} = w_t - \gamma \left[\nabla f_i(w_t) - \nabla f_i(w_{old}) + \frac{1}{n} \sum_j \nabla f_j(w_{old}) \right]$ (for fixed w_{old} from outer loop)

SVRG for $k = 0, \ldots$ (outer loop)

compute $g_{ref} = \frac{1}{n} \sum_i \nabla f_i(w^{(k)})$

$w_0^{(k)} = w^{(k)}$

for $t = 0, \ldots, t_{max}$

sample i_t

$w_{t+1}^{(k)} = w_t^{(k)} - \gamma \left[\nabla f_{i_t}(w_t^{(k)}) - \nabla f_{i_t}(w_0^{(k)}) + g_{ref} \right]$

end

$w^{(k+1)} = w_{t_{max}}^{(k)}$

Questions:
- what is t_{max}?
- what is γ?

SAG: (−) need to store g_i^t $O(nd)$ (+) no t_{max} to tune, no weird 2 loops

SVRG: (−) tweak t_{max}, weird 2 loops, 2 gradients per iteration (+) only need to store $w_0^{(k)}$

SVRG convergence result: need $\gamma \leq \approx \frac{1}{L}$

$t_{max} \geq \approx \frac{L}{\mu} = K$ $\Rightarrow$ in practice, $t_{max} = \max\{n, K\}$

innerbank account: "adaptivity to local vs strong convexity"

important concept: "adaptivity to (local & strong) convexity"

SAG result $\gamma \simeq \frac{1}{L}$ ⇒ algorithmic parameters ~~do not depend on μ~~ (strong convexity)

SVRG $\gamma \simeq \frac{1}{L}$ but $t_{max} = \max\{n, \frac{L}{\mu}\}$ ← depends on μ

for just convex fct. ($\mu = 0$), get $\min_t [\mathbb{E} f(x_t) - f^*] = O(\frac{1}{t})$

[contrast with $\frac{L}{\sqrt{t}}$ for SGD]

SAGA "simple" convergence result:

if you use $\gamma = \frac{a}{L}$ for SAGA for $a \leq 1$ i.e. $\mathbb{E} f(x_t) - f^* \leq (1-\rho)^t C_0$

then rate $\rho \geq \frac{1}{5} \min\{\frac{1}{n}, \frac{a}{K}\}$

two effects:
- bigger step-size ⇒ bigger variance
- smaller $\|\cdot\|$ ⇒ slower gradient descent rate (determined by $\frac{1}{K}$)

if $K < n$; then as long as $\frac{a}{K} \geq \frac{1}{n}$ i.e. $\boxed{1 \geq a \geq \frac{K}{n}}$, we get same rate (roughly)
→ "robustness" to step size when $K \ll n$

SAGA complexity to reach ε error $O((n+K)\log\frac{1}{\varepsilon})$ "SGD steps"

Practical aspects of SAG/SAGA:

a) storage: if $f_i(w) = h(x_i^T w)$ $\nabla f_i(w) = \underbrace{h'(x_i^T w)}_{\text{scalar}} x_i$

i.e. instead of $O(nd)$ storage, only need $O(n)$

b) initialization? hack which is to use $\frac{1}{|S_t|}\sum_{i\in S_t} g_i^t$ where $S_t = \{i : i \text{ has been visited before } t\}$

c) step-size?
- $\frac{1}{L}$
- cheap line search heuristic { while $f_i\left(w_t - \frac{1}{\hat{L}}\nabla f_i(w_t)\right) \geq f_i(w_t) - \frac{1}{2\hat{L}}\|\nabla f_i(w_t)\|^2$ set $\hat{L}^{new} = 2\hat{L}^{old}$

(comes from FISTA)

$\hat{L}$ is surrogate for L

use step-size $\frac{1}{\hat{L}}$

a) non-uniform sampling? → sample $i \sim \frac{L_i}{\sum_j L_j}$ then get rate with $\frac{\frac{1}{n}\sum_i L_i}{\mu}$ instead of $\frac{\max_i L_i}{\mu}$

e) stopping criterion? you can use $\frac{1}{n}\sum_j g_j^t$ as approximate $\nabla f(w_t)$

f) sparse features? two tricks 1) "lagged update" → $w_t^d = w_{t-\Delta}^d - (\Delta\alpha)\, g_{old}^d$ [Schmidt et al.]

sparse modification $w_{t+1} = w_t - \alpha\left(\nabla f_i(w_t) - g_i^t + P_{S_i}\left(\frac{1}{n}\sum_j g_j^t\right)\right)$

[Lekeland et al. 2017] weighted projection on support of x_i

2) $f_i(w) = \frac{\lambda\|w\|^2}{2} + h(x_i^T w)$

$w_{t+1} = (1-\lambda\alpha)w_t - \gamma\left[\text{sparse}\right]$ $\quad S_i = \{d : (x_i)_d \neq 0\}$

state w as {direction, scale}

Pointers

- variance reduction perspective on SAG / SAGA / SVRG -- in SAGA paper:
  - A. Defazio, F. Bach and S. Lacoste-Julien
    SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives
    NIPS 2014
  - other pointers:
    - SVRG paper:
      Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. NIPS 2013
    - note that a variant of SVRG that is *adaptive to local strong convexity* is given in the following paper (where the end of the inner loop is decided randomly: at every inner loop iteration, with probability 1/n, you end the inner loop):
      T. Hofmann, A. Lucchi, S. Lacoste-Julien, and Brian McWilliams
      Variance Reduced Stochastic Gradient Descent with Neighbors,
      NIPS 2015
- the practical aspects of SAG are described in the massive journal version:
  - M. Schmidt, N. Le Roux, F. Bach.
    Minimizing Finite Sums with the Stochastic Average Gradient
    Mathematical Programming, 162:83-162, 2017 | arxiv
  - an alternative to the complicated "lagged updates" when you have sparse features is the Sparse SAGA algorithm; see Section 2 of:
    ASAGA: Asynchronous Parallel SAGA,
    R. Leblond, F. Pedregosa and S. Lacoste-Julien
    AISTATS 2017