Literature Review on the Variance Reduction Technique in Optimization and Beyond

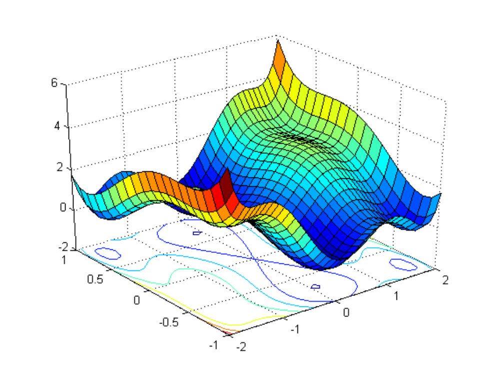
Wenjie Li

May 2020

Introduction

Introduction i

Gradient Descent (GD) is stable but slow, Stochastic Gradient Descent (SGD) is fast but unstable. Is it possible to combine the advantages of both of them?



Nonconvex Smooth Optimization Problem

$$min_{x \in \mathbb{R}^d} F(x) \tag{1}$$

where $F(x) = \frac{1}{n} \sum_{i=1}^{n} f_i(x)$ and each $f_i(x)$ is a non-convex, L-smooth function, with average bounded variance σ^2.

$$\|\nabla f_i(x) - \nabla f_i(y)\| \leq L\|x - y\|, \forall i \tag{2}$$

Sample $j \in \{1, 2, \cdots, n\}$ uniformly

$$\mathbb{E}[\nabla f_j(x)] = \nabla F(x), \mathbb{E}\|\nabla f_j(x) - \nabla F(x)\|_2^2 \leq \sigma^2 \tag{3}$$

We will talk about non-smoothness later.

Introduction iii

Convergence Criterion: ϵ-first order stationary point

$$\mathbb{E}[\|\nabla F(x_T)\|_2] \leq \epsilon \tag{4}$$

We use the number of stochastic gradient computations (Stochastic first-order oracle, $\nabla f_i(x)$). to measure the algorithm performance

SVRG Algorithm

SVRG Algorithm i

Algorithm 1 Stochastic Variance Reduction Gradient Algorithm (Johnson & Zhang, 2013)

Input: Number of stages T, initial x_1, step sizes $\{\alpha_t\}_{t=1}^{T}$
for $t = 1$ **to** T **do**
    $g_t = \nabla F(x_t)$
    $y_1^t = x_t$
    for $k = 1$ **to** K **do**
        Randomly pick $i \in [n]$
        $v_k^t = \nabla f_i(y_k^t) - \nabla f_i(y_1^t) + g_t$
        $y_{k+1}^t = y_k^t - \alpha_t v_k^t$
    end for
    $x_{t+1} = y_j^t$ with j uniformly sampled from $\{1, 2, \cdots, K\}$
end for

SVRG Algorithm ii

Key Idea: Use the full-batch gradient g_t and the snapshot x_t to reduce the variance of stochastic gradients. The convergence rate is proved by Reddi et al. (2016a).

SVRG Algorithm iii

Results by Reddi et al. (2016a)

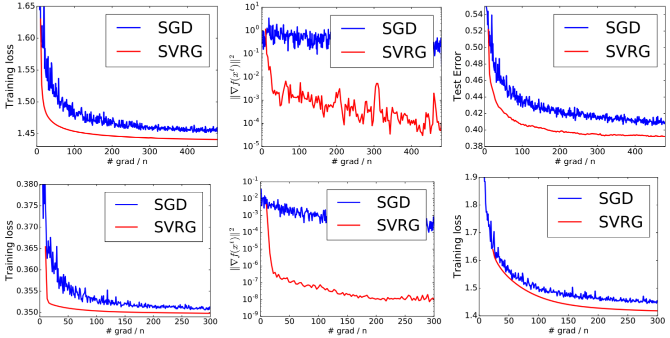


Figure 1: Neural network results for CIFAR-10, MNIST and STL-10 datasets. The top row represents the results for CIFAR-10 dataset. The bottom left and middle figures represent the results for MNIST dataset. The bottom right figure represents the result for STL-10.

SVRG Algorithm iv

Other Variants: SCSG (Lei et al., 2017), SNVRG (Zhou et al., 2018),and a lot more.

Tabelle 1: Comparison Between Different Algorithms

Algorithms	SFO computations
GD	$O(n/\epsilon^2)$
SGD	$O(1/\epsilon^4)$
SVRG (Reddi et al., 2016a)	$O(n^{2/3}/\epsilon^2)$
SCSG (Lei et al., 2017)	$O((n/\epsilon^2 \wedge 1/\epsilon^{10/3}))$
SNVRG (Zhou et al., 2018)	$\tilde{O}(n^{1/2}/\epsilon^2 \wedge 1/\epsilon^3)$

n is the total number of samples

SVRG Algorithm v

Algorithm 2 SCSG Algorithm

Input: Number of stages T, initial x_1, step sizes $\{\alpha_t\}_{t=1}^T$, batch sizes $\{B_t\}_{t=1}^T$, mini-batch sizes $\{b_t\}_{t=1}^T$

for $t = 1$ **to** T **do**

  Randomly sample a batch $\mathcal{I}_t$ with size B_t

  $g_t = \nabla f_{\mathcal{I}_t}(x_t)$

  $y_1^t = x_t$, Generate $K_j \sim \mathrm{Geom}(B_j/(B_j + b_j))$

  for $k = 1$ **to** K_j **do**

    Randomly pick sample $\tilde{\mathcal{I}}_t$ of size b_t

    $v_k^t = \nabla f_{\tilde{\mathcal{I}}_t}(y_k^t) - \nabla f_{\tilde{\mathcal{I}}_t}(y_1^t) + g_t$

    $y_{k+1}^t = y_k^t - \alpha_t v_k^t$

  end for

  $x_{t+1} = y_{K+1}^t$

end for

SVRG Algorithm vi

Key Idea: Use the geometric random variable K_j to control the number of inner loop iterations.

SNVRG (Zhou et al., 2018): A little more complicated, use more than one reference points and reference gradients to make the algorithm even faster. But with more assumptions on each batch size for each reference point.

SVRG Algorithm vii

Algorithm 1 One-epoch-SNVRG($\mathbf{x}_0$, $F, K, M, \{T_l\}, \{B_l\}, B$)

Input: initial point $\mathbf{x}_0$, function F, loop number K, step size parameter M, loop parameters $T_l, l \in [K]$, batch parameters $B_l, l \in [K]$, base batch size $B > 0$.
$\mathbf{x}_0^{(l)} \leftarrow \mathbf{x}_0$, $\mathbf{g}_0^{(l)} \leftarrow 0, 0 \le l \le K$
Uniformly generate index set $I \subset [n]$ without replacement, $|I| = B$
$\mathbf{g}_0^{(0)} \leftarrow 1/B \sum_{i \in I} \nabla f_i(\mathbf{x}_0)$
$\mathbf{v}_0 \leftarrow \sum_{l=0}^{K} \mathbf{g}_0^{(l)}$
$\mathbf{x}_1 = \mathbf{x}_0 - 1/(10M) \cdot \mathbf{v}_0$
for $t = 1, ..., \prod_{l=1}^{K} T_l - 1$ **do**
  $r = \min\{j : 0 = (t \mod \prod_{l=j+1}^{K} T_l),\ 0 \le j \le K\}$
  $\{\mathbf{x}_t^{(l)}\} \leftarrow$ Update_reference_points($\{\mathbf{x}_{t-1}^{(l)}\}, \mathbf{x}_t, r), 0 \le l \le K$.
  $\{\mathbf{g}_t^{(l)}\} \leftarrow$ Update_reference_gradients($\{\mathbf{g}_{t-1}^{(l)}\}, \{\mathbf{x}_t^{(l)}\}, r), 0 \le l \le K$.
  $\mathbf{v}_t \leftarrow \sum_{l=0}^{K} \mathbf{g}_t^{(l)}$
  $\mathbf{x}_{t+1} \leftarrow \mathbf{x}_t - 1/(10M) \cdot \mathbf{v}_t$
end for
$\mathbf{x}_{\text{out}} \leftarrow$ uniformly random choice from $\{\mathbf{x}_t\}$, where $0 \le t < \prod_{l=1}^{K} T_l$
$T = \prod_{l=1}^{K} T_l$
Output: $[\mathbf{x}_{\text{out}}, \mathbf{x}_T]$

Nonconvex Nonsmooth Optimization

Nonconvex Nonsmooth Optimization i

Similar to the setting of nonconvex smooth optimization, now we add some non-smoothness into the discussion.

$$min_{x\in\mathbb{R}^d} F(x) = f(x) + h(x) \tag{5}$$

where $f(x) = \frac{1}{n}\sum_{i=1}^{n} f_i(x)$ and each $f_i(x)$ is a non-convex, L-smooth function, with average bounded variance σ^2. $h(x)$ is a convex function that can be non-smooth.

With the same bounded variance and smoothness of f_i assumptions

Nonconvex Nonsmooth Optimization ii

Algorithm 3 ~~SVRG~~ ProxSVRG Algorithm (Reddi et al., 2016b)

Input: Number of stages T, initial x_1, step sizes $\{\alpha_t\}_{t=1}^{T}$,
for $t = 1$ **to** T **do**
  $g_t = \nabla f(x_t)$
  $y_1^t = x_t$
  for $k = 1$ **to** K **do**
    Randomly pick $i \in [n]$
    $v_k^t = \nabla f_i(y_k^t) - \nabla f_i(y_1^t) + g_t$
    $\cancel{y_{k+1}^t = y_k^t - \alpha_t v_k^t}.$
    $y_{k+1}^t = \mathrm{argmin}_y \{\alpha_t \langle v_k^t, y \rangle + \alpha_t h(x) + \frac{1}{2}\|y - y_k^t\|^2\}$
  end for
  $x_{t+1} = y_j^t$ with j uniformly sampled from $\{1, 2, \cdots, K\}$
end for

Nonconvex Nonsmooth Optimization iii

Algorithm 4 ~~ProxSVRG~~ ProxSVRG+ Algorithm (Li & Li, 2018)

Input: Number of stages T, initial x_1, step sizes $\{\alpha_t\}_{t=1}^T$, batch sizes $\{B_t\}_{t=1}^T$, mini-batch sizes $\{b_t\}_{t=1}^T$
for $t = 1$ **to** T **do**
    $\cancel{g_t = \nabla f(x_t)}$
    Randomly sample a batch $\mathcal{I}_t$ with size B_t
    $g_t = \nabla f_{\mathcal{I}_t}(x_t)$
    $y_1^t = x_t$
    for $k = 1$ **to** K **do**
        ~~Randomly pick i~~ Randomly sample a mini-batch $\tilde{I}_j$ of size b
        $v_k^t = \nabla f_{\tilde{I}_j}(y_k^t) - \nabla f_{\tilde{I}_j}(y_1^t) + g_t$
        $y_{k+1}^t = \mathrm{argmin}_y\{\alpha_t \langle v_k^t, y\rangle + \alpha_t h(x) + \frac{1}{2}\|y - y_k^t\|^2\}$
    end for
end for

Nonconvex Nonsmooth Optimization iv

Define the generalized gradient

$$\tilde{g}_{X,t} = \frac{1}{\alpha_t}(x_t - x_{t+1}) \tag{6}$$

and its corresponding term when the algorithm uses non-stochastic gradients, i.e. when it uses $\nabla f(x_t)$ instead of $\nabla f_{\mathcal{I}_j}(x_t)$ in the mirror descent update rule.

$$g_{X,t} = \frac{1}{\alpha_t}(x_t - x_{t+1}^{+}),\ \text{when } x_{t+1}^{+} = \text{argmin}_x\{\langle \nabla f(x_t), x\rangle + \alpha_t h(x) + B_{\psi_t}(x, x_t)\} \tag{7}$$

Convergence criterion

$$\mathbb{E}[\|g_{X,t^*}\|] \leq \epsilon \tag{8}$$

Nonconvex Nonsmooth Optimization v

Tabelle 2: Comparison Between Different Algorithms

ALGORITHMS	SFO COMPUTATIONS
ProxGD (Ghadimi et al., 2016)	$O(n/\epsilon^2)$
ProxSVRG/SAGA (Reddi et al., 2016b)	$O(n/(\epsilon^2\sqrt{b}) + n)$
ProxSVRG+ (Li & Li, 2018)	$O(n/(\epsilon^2\sqrt{b}) \wedge (1/(\epsilon^2\sqrt{b})) + \frac{b}{\epsilon^2})$

b is the mini-batch size

Can we generalize to adaptive mirror descent algorithms, i.e. replace $\frac{1}{2}\|y - y_k^t\|^2$ with general Bregman Divergences $B_{\psi_{tk}}(x, x_t)$?

$$B_{\psi_{tk}}(x, y) = \psi_{tk}(x) - \psi_{tk}(y) + \langle \nabla \psi_{tk}(x), y - x \rangle \tag{9}$$

The answer is yes.

We have already proved the possibility of adding variance reduction to general mirror descent algorithms, as long as the proximal functions $\psi_{tk}(x)$ have a lower bound for its strong convexity. i.e.

$$\exists m > 0, s.t. \nabla^2 \psi_{tk}(x) \succeq mI \tag{10}$$

Nonconvex Nonsmooth Optimization vii

Examples for the strong convesity assumption :

- $\psi_{tk}(x) = \frac{1}{2}\|x\|_2^2$, the algorithm reduces to ProxSVRG+ (Li & Li, 2018) and $m = 1$.
- $\psi_{tk}(x) = f_{tk}(x) + \frac{c}{2}\|x\|_2^2$, where each $f_{tk}(x)$ is a convex function and $m = c$.
- $\psi_{tk}(x) = \frac{1}{2}\langle x, H_{tk}x\rangle$, $H_{tk} \succeq mI$, the algorithm covers all the adaptive optimizers (AdaGrad, RMSProp) with constant m added to the denominator to avoid division by zero.

We have proved that the convergence rate of adaptive mirror descent with variance reduction is the same as ProxSVRG+. So we have extended the results by Li & Li (2018) to a more general setting.

Specifically, adaptive algorithms (AdaGrad, RMSProp etc) works with variance reduction.

Future work:

Can we prove better convergence results for adaptive algorithms specifically?

Run more experiments with the other adaptive algorithms

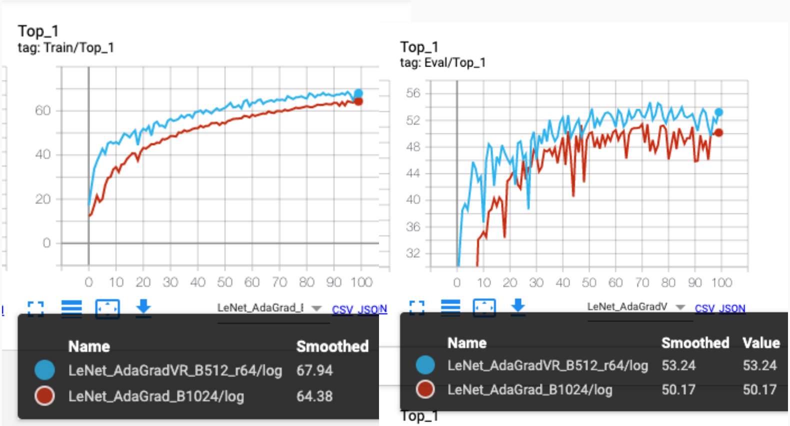
Experiments

Experiments i

We have evaluted the performance of AdaGrad and AdaGrad + VR with LeNet on CIFAR-10 (And also with fully connected networks on MNIST)

Tabelle 3: Parameters Setting

Algorithms	AdaGrad	AdaGrad + VR
LR	0.001	0.001
Batch Size	1024	512 * 64, 512



(a) CIFAR-10 Training Acc.

(b) CIFAR-10 Testing Acc.

Abbildung 1: Training and Testing Top-1 accuracy on CIFAR-10

This is it ☺

Literatur

Saeed Ghadimi, Guanghui Lan, and Hongchao Zhang. Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. *arXiv preprint arXiv:1308.6594*, 2016.

Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. *Advances in Neural Information Processing Systems 27*, 2013.

Lihua Lei, Cheng Ju, Jianbo Chen, and Michael I Jordan. Non-convex finite-sum optimization via scsg methods. *Advances in Neural Information Processing Systems 30*, pp. 2348–2358, 2017.

Zhize Li and Jian Li. A simple proximal stochastic gradient method for nonsmooth nonconvex optimization. *Advances in Neural Information Processing Systems 31*, pp. 5564–5574, 2018.

Sashank Reddi, Suvrit Sra, Barnabas Poczos, and Alexander J Smola. Proximal stochastic methods for nonsmooth nonconvex finite-sum optimization. *Advances in Neural Information Processing Systems 29*, 2016b.

Sashank J. Reddi, Ahmed Hefny, Suvrit Sra, Barnabas Poczos, and Alex Smola. Stochastic variance reduction for nonconvex optimization, 2016a.

Dongruo Zhou, Pan Xu, and Quanquan Gu. Stochastic nested variance reduction for nonconvex optimization. *Advances in Neural Information Processing Systems 32*, 2018.