Optimization for Deep Learning

Lecture 9: Stochastic Variance-Reduced Gradient Method

Kun Yuan

Peking University

Main contents in this lecture

- Finite-sum minimization
- SAGA
- SVRG

Stochastic optimization

- Consider the stochastic optimization problem:

$$\min_{x \in \mathbb{R}^d} \quad f(x) = \mathbb{E}_{\xi \sim \mathcal{D}}[F(x; \xi)] \tag{1}$$

  - ξ is a random variable indicating data samples
  - $\mathcal{D}$ is the data distribution; unknown in advance
  - $F(x; \xi)$ is differentiable in terms of x

- Many applications in signal processing and machine learning

Finite-sum minimization

- In real practice, we typically have finite data samples

$$\mathcal{M} = \{\xi_1, \xi_2, \cdots, \xi_m\}$$

  where m is the sample size

- Suppose in distribution $\mathcal{D}$, each data will be sampled uniformly randomly, i.e.,

$$\mathbb{P}(\xi = \xi_i) = \frac{1}{m}, \quad \forall\ i,$$

  Problem (1) becomes finite-sum minimization

$$\min_{x \in \mathbb{R}^d} \quad f(x) = \mathbb{E}_{\xi \sim \mathcal{D}}[F(x; \xi)] = \frac{1}{m} \sum_{i=1}^{n} F(x; \xi_i)$$

- Finite-sum minimization is a special example of stochastic optimization

Stochastic gradient descent with finite samples

- Applying SGD to finite-sum minimization, we achieve

  Sample $i_k \sim [m]$ **uniformly** and **randomly**

  $$x_{k+1} = x_k - \gamma F_{i_k}(x_k), \quad \text{where} \quad F_{i_k}(x_k) = F(x_k; \xi_{i_k})$$

  which is referred to as SGD with finite samples.

- If we assume the stochastic gradient is unbiased and has bounded variance, all convergence theories in Lecture 6 apply to SGD with finite samples.

Can we further improve the convergence rate?

- Recall the convergence performance of SGD:

$$\frac{1}{K}\sum_{k=1}^{T}\mathbb{E}\|\nabla f(x_k)\|^2 \le \frac{2\Delta_0}{\gamma K} + \gamma L\sigma^2$$

- If we use constant γ, the first term decays quickly, but meanwhile, SGD cannot converge exactly due to bias $\gamma L\sigma^2$

- If we use decaying $\gamma = 1/\sqrt{K}$, SGD can converge exactly, but meanwhile, the first term decays at a slower rate $O(1/\sqrt{K})$

- There is a **trade-off** between convergence rate and precision

Can we further improve the convergence rate?

- The root reason to cause trade-off is the existence of σ^2
- Can we remove it from the convergence of SGD?
- Not possible in general stochastic optimization; we do not have the closed-form of $f(x) = \mathbb{E}[F(x;\xi)]$
- But it is possible in finite-sum minimization due to its special structure

Stochastic variance-reduced gradient

- In SGD, stochastic gradient $\nabla F_{i_k}(x)$ has constant variance:

$$\mathbb{E}[\nabla F_{i_k}(x)] = \nabla f(x), \qquad \mathbb{E}\|\nabla F_{i_k}(x) - \nabla f(x)\|^2 \leq \sigma.$$

- Now we construct a new stochastic gradient (Defazio et al., 2014)

$$g_k(x) = \nabla F_{i_k}(x) - \nabla F_{i_k}(\alpha_{i_k}) + u_k, \text{ where } u_k = \frac{1}{m}\sum_{j=1}^{m} \nabla F_j(\alpha_j)$$

  - quantity $\{\alpha_j\}_{j=1}^m$ are m auxiliary variables with each $\alpha_j \in \mathbb{R}^d$
  - index $i_k \in [m]$ is sampled uniformly randomly.

Stochastic variance-reduced gradient

- Stochastic gradient g_k is unbiased:

$$\begin{aligned}&\mathbb{E}[\nabla F_{i_k}(x) - \nabla F_{i_k}(\alpha_{i_k}) + u_k] \\ =&\frac{1}{m}\sum_{j=1}^{m}[\nabla F_j(x) - \nabla F_j(\alpha_j)] + \frac{1}{m}\sum_{j=1}^{m}\nabla F_j(\alpha_j) = \nabla f(x)\end{aligned}$$

- The variance is examined as

$$\begin{aligned}&\mathbb{E}\|\nabla F_{i_k}(x) - \nabla F_{i_k}(\alpha_{i_k}) + u_k - \nabla f(x)\|^2 \\ =&\ \frac{1}{m}\sum_{j=1}^{m}\|\nabla F_j(x) - \nabla F_j(\alpha_j) + \frac{1}{m}\sum_{j=1}^{m}\nabla F_j(\alpha_j) - \frac{1}{m}\sum_{j=1}^{m}\nabla F_j(x)\|^2 \\ \le&\ \frac{1}{m}\sum_{j=1}^{m}\|\nabla F_j(x) - \nabla F_j(\alpha_j)\|^2 \le \frac{L^2}{m}\sum_{j=1}^{m}\|x - \alpha_j\|^2\end{aligned}$$

Stochastic variance-reduced gradient

- The variance of g_k

$$\mathbb{E}\|g_k - \nabla f(x)\|^2 \le \frac{L}{m}\sum_{j=1}^{m}\|x-\alpha_j\|^2$$

will vanish to 0 if $\alpha_j \to x$. We name g_k as **variance-reduced gradient**

- To make $\alpha_j \to x$, we construct $\{\alpha_j\}_{j=1}^m$ as follows

$$\alpha_j^{(k+1)} = \begin{cases} x_k & \text{if index } j \text{ is sampled, i.e., } i_k = j; \\ \alpha_j^{(k)} & \text{otherwise.} \end{cases}$$

Only one α_i is updated; the others remain unchanged.

Illustration of the α update

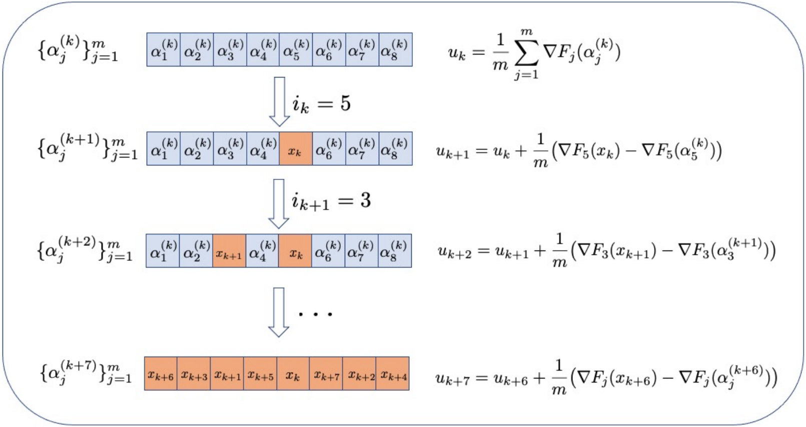

Figure: Illustration of the α and u update.

Stochastic variance-reduced gradient

- With the construction of $\{\alpha_j\}_{j=1}^m$, we can update u as

$$
\begin{aligned}
u_{k+1} &= \frac{1}{m}\sum_{j=1}^m \nabla F_j(\alpha_j^{(k+1)}) \\
&= \frac{1}{m}\sum_{j\neq i_k} \nabla F_j(\alpha_j^{(k)}) + \frac{1}{m}\nabla F_{i_k}(x_k) \\
&= \frac{1}{m}\sum_{j=1}^m \nabla F_j(\alpha_j^{(k)}) + \frac{1}{m}[\nabla F_{i_k}(x_k) - \nabla F_{i_k}(\alpha_{i_k}^{(k)})] \\
&= u_k + \frac{1}{m}[\nabla F_{i_k}(x_k) - \nabla F_{i_k}(\alpha_{i_k}^{(k)})]
\end{aligned}
$$

which can be updated very efficiently

SAGA algorithm

Initialize x_0 arbitrarily; let $\alpha_j^{(0)} = x_0$ and $u_0 = \frac{1}{m}\sum_{j=1}^m \alpha_j^{(0)}$

For $k = 0, 1, 2, ..., T-1$:

sample $i_k \in \{1, 2, \cdots, m\}$ uniformly randomly

update $g_k = \nabla F_{i_k}(x_k) - \nabla F_{i_k}(\alpha_{i_k}^{(k)}) + u_k$

update $x_{k+1} = x_k - \gamma g_k$

update $\alpha_{i_k}^{(k+1)} = x_k$ and $\alpha_j^{(k+1)} = \alpha_j^{(k)}$ if $j \neq i_k$

update $u_{k+1} = u_k + \frac{1}{m}\left(\nabla F_{i_k}(x_k) - \nabla F_{i_k}(\alpha_{i_k}^{(k)})\right)$

Output x_T.

Compared to vanilla SGD, SAGA incurs additional $O(md)$ memory cost.

SAGA algorithm

- In special scenario, SAGA only incurs $O(m)$ memory cost

- In linear regression, the finite-sum minimization problem is

$$\min_{x \in \mathbb{R}^d} \quad \frac{1}{2m} \sum_{i=1}^{m} (a_i x - b_i)^2$$

- The stochastic gradient $\nabla F_{i_k}(x)$ for linear regression is

$$\nabla F_{i_k}(x) = (a_{i_k} x - b_{i_k}) a_{i_k}$$

- Since $\{a_1, \cdots, a_m\}$ has been stored in memory as data samples, it is enough to let α_j track $a_j^\top x - b_j$ (which is a constant); reduce the memory to $O(m)$

SAGA convergence

Lemma 1

Assume each $F_i(x)$ is L-smooth and index i_k is sampled uniformly randomly. If the learning rate $\gamma \le \frac{1}{L}$, the iterate x_k generated by SAGA satisfies

$$\mathbb{E}[f(x_{k+1})] \le \mathbb{E}[f(x_k)] - \frac{\gamma}{2}\mathbb{E}\|\nabla f(x_k)\|^2 + \frac{L^3\gamma^2}{2m}\sum_{j=1}^{m}\mathbb{E}\|x_k - \alpha_j^{(k)}\|^2.$$

In comparison, the iterate x_k in SGD satisfies

$$\mathbb{E}[f(x_{k+1})] \le \mathbb{E}[f(x_k)] - \frac{\gamma}{2}\mathbb{E}\|\nabla f(x_k)\|^2 + \frac{L\gamma^2\sigma^2}{2}.$$

in which $\frac{L\gamma^2\sigma^2}{2}$ cannot vanish over time

SAGA convergence

Lemma 2

Assume each $F_i(x)$ is L-smooth and index $i_t \in [m]$ is sampled uniformly randomly. Let x_k and $\{\alpha_j^{(k)}\}_{j=1}^m$ be generated by the SAGA algorithm. It holds for any $k = 0, 1, \cdots, K-1$ that

$$\frac{1}{m}\sum_{j=1}^{m}\mathbb{E}\|x_{(k+1)} - \alpha_i^{(k+1)}\|^2 \le \Big(1 - \frac{1}{m} + \gamma\beta + \gamma^2 L^2\Big)\frac{1}{m}\sum_{j=1}^{m}\mathbb{E}\|x_k - \alpha_i^{(k)}\|^2 + \big(\gamma^2 + \frac{\gamma}{\beta}\big)\mathbb{E}\|\nabla f(x_k)\|^2$$

where $\beta > 0$ is a constant to be determined later.

SAGA convergence

Theorem 1

Assume each $F_i(x)$ is L-smooth and index i_k is sampled uniformly randomly. If the learning rate $\gamma = (3Lm^{2/3})^{-1}$, the iterate x_k generated by SAGA satisfies

$$\frac{1}{K}\sum_{k=0}^{K-1} \mathbb{E}\|\nabla f(x_k)\|^2 \le \frac{10Lm^{2/3}\Delta_0}{K}$$

where $\Delta_0 = f(x_0) - f^\star$.

In comparison, the iterate x_k in SGD satisfies

$$\frac{1}{K}\sum_{k=0}^{K-1} \mathbb{E}\|\nabla f(x_k)\|^2 = O(\frac{\sigma}{\sqrt{K}} + \frac{1}{K})$$

Variance reduction improves the order of convergence!

SAGA sample complexity in difference scenarios

SAGA achieves the best sample complexity compared to GD and SGD

Table: Comparison in terms of the number of sampled data

Scenario	**GD**	**SGD**	**SAGA**
strongly-convex	$m\kappa \ln(\frac{1}{\epsilon})$	$\frac{L}{\epsilon}$	$(m+\kappa)\ln(\frac{1}{\epsilon})$
non-convex	$\frac{mL}{\epsilon}$	$\frac{L}{\epsilon^2}$	$\frac{m^{2/3}L}{\epsilon}$

A brief summary

- SAGA is an extended version of SGD with a variance-reduced gradient

$$g_k(x) = \nabla F_{i_k}(x) - \nabla F_{i_k}(\alpha_{i_k}) + u_k, \text{ where } u_k = \frac{1}{m}\sum_{j=1}^{m} \nabla F_j(\alpha_j)$$

- Pro: SAGA has a faster convergence rate $O(1/K)$
- Con: SAGA incurs massive memory $O(md)$ cost to store $\{\alpha_j\}_{j=1}^m$
- Can we maintain the $O(1/K)$ rate with far less memory cost?

SVRG algorithm

- Recall the variance-reduced gradient $g_k(x)$ satisfies

$$\mathbb{E}\|g_k - \nabla f(x)\|^2 \le \frac{L}{m} \sum_{j=1}^{m} \|x - \alpha_j\|^2$$

- To achieve variance reduction, we need $\alpha_j \to x$

- To save storage, we should not maintain $\{\alpha_j\}_{j=1}^m$ in memory

- Idea: let $\alpha_j = \tilde{x},\ \forall j \in [n]$ to save memory; let $\tilde{x} \to x$ to reduce variance (Johnson and Zhang, 2013)

SVRG algorithm

- SVRG constructs the following variance-reduced gradient

$$g_k(x) = \nabla F_{i_k}(x) - \nabla F_{i_k}(\tilde{x}) + u_k, \text{ where } u_k = \frac{1}{m}\sum_{j=1}^{m} \nabla F_j(\tilde{x})$$

- Only need to maintain a single $\tilde{x}$, not $\{\alpha_j\}_{j=1}^m$.
- However, u_k is expensive to calculate; incurs full-batch computation
- The rule to update $\tilde{x}$ is critical; cannot be too slow or fast
  - Too slow: $\tilde{x} \to x$ slowly; affects variance-reduction
  - Too fast: u_k has to be calculated for each new $\tilde{x}$

SVRG algorithm

An arbitrary initialization $x_S^0 = x^0$; let $R = K/S$.

For $r = 0, 1, 2, ..., R-1$:

$x_0^{r+1} = x_S^r$

$\tilde{x}^{r+1} = x_0^{r+1}$

$u^{r+1} = \frac{1}{m}\sum_{i=1}^m \nabla F_i(\tilde{x}^{r+1})$

For $s = 0, 1, 2, ..., S-1$:

sample $i_s \in [m]$ randomly and uniformly

update $g_s = \nabla F_{i_s}(x_s^{r+1}) - \nabla F_{i_s}(\tilde{x}^{r+1}) + u^{r+1}$

update $x_{s+1}^{r+1} = x_s^{r+1} - \gamma g_s$

Output x_S^R.

SVRG convergence

Theorem 1

Assume each $F_i(x)$ is L-smooth and index i_k is sampled uniformly randomly. If learning rate $\gamma = \mu_0(Lm^\alpha)^{-1}$, inner loop $S = m^{\frac{3\alpha}{2}}/(2\mu_0)$, constant $\mu_0 = [5(e-1)]^{-1}$ where e is the Euler's number and constant $\alpha \in (0,1)$, then x_s^{r+1} generated by SVRG will converge as follows:

$$\frac{1}{T}\sum_{r=0}^{R-1}\sum_{s=0}^{S-1}\mathbb{E}\|\nabla f(x_s^{r+1})\|^2 \leq \frac{200Lm^\alpha\Delta_0}{T}.$$

where $\Delta_0 = f(x^0) - f^\star$.

Numerical performanc

Stochastic variance-reduced methods are as cheap to update as SGD, and also have a fast exponential convergence like full gradient descent.

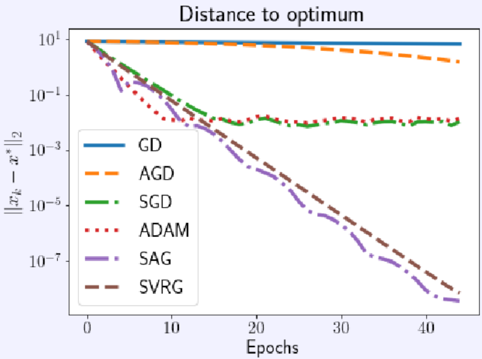

Figure: SGD, SVRG, and SAGA[1]

[1]This figure is from (Gower et al., 2020)

References I

A. Defazio, F. Bach, and S. Lacoste-Julien, "Saga: A fast incremental gradient method with support for non-strongly convex composite objectives," *Advances in neural information processing systems*, vol. 27, 2014.

R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," *Advances in neural information processing systems*, vol. 26, 2013.

R. M. Gower, M. Schmidt, F. Bach, and P. Richtárik, "Variance-reduced methods for machine learning," *Proceedings of the IEEE*, vol. 108, no. 11, pp. 1968–1983, 2020.