Variance Reduction for Stochastic Gradient Descent

Yukang Zhao
AMATH 515

March 8, 2024

Abstract

Stochastic Gradient Descent (SGD) is a powerful and widely-used optimization technique in machine learning. It is popular by its "cheaper" costs in each step by computing less amount of derivatives, in comparison to standard gradient descent. However, this also has a trade-off: because of the variance of random sampling, we have a much slower convergence rate. In this project we will introduce a variance reduction method for SGD which allows a less computation and a fast convergence rate at the same time.

1 Introduction to SGD

Let $f_1, ..., f_n$ be a sequence of vector functions, where $f_i : \mathbb{R}^d \to \mathbb{R}\ \forall i$. Consider the following minimization problem:

$$\min_w \frac{1}{n} \sum_{i=1}^{n} f_i(w)$$

Suppose we are given a set of m data points,

$$(x_1, y_1), ..., (x_m, y_m),\ x_i \in \mathbb{R}^d,\ y_i \in \mathbb{R}$$

and let

$$f_i(w) = \ln(1 + \exp(-w^T x_i y_i)) + \lambda \|w\|^2$$

so the problem becomes regularized logistic regression. We first try to solve this by using Gradient Descent (GD) with step size h_k and update for k-th iteration:

$$w_{k+1} = w_k - \frac{h_k}{n} \sum_{i=1}^{n} \nabla f_i(w_k)$$

However, this requires n times evaluation of derivatives for each iteration, which is super numerically expensive especially for large scale problem. Therefore, we introduce Stochastic Gradient Descent (SGD):

$$w_{k+1} = w_k - h_k \nabla f_{i_k}(w_k)$$

where $i_k \in \{1, ..., n\}$ is randomly chosen for each step. Notice the expectation for each update SGD is the same as GD:

$$\begin{aligned} \mathbb{E}[\nabla f_{i_k}(w_k)] &= \nabla \mathbb{E}[f_{i_k}(w_k)] \\ &= \nabla \sum_{j=1}^{n} P(i_k = j) f_j(w_k) \\ &= \frac{1}{n} \nabla \sum_{j=1}^{n} f_j(w_k) \end{aligned} \tag{1}$$

Thus, we only need to compute one derivative $\nabla f_{i_k}(w_k)$ for each step, which is much cheaper computationally ($\frac{1}{n}$ comparing to GD). But this also has a trade-off: the randomness in choosing i_k introduces variances. Suppose f_i are β-smooth and α-strongly convex, then the standard GD has a linear convergence rate of $O((1 - \frac{\alpha}{\beta})^k)$ Nesterov[2004], but for SGD, due to the variance introduced by randomly chosen f_{i_k}, we need to choose $h_k = O(\frac{1}{k})$ it only obtains a sub-linear convergence rate of $O(\frac{1}{k})$.

In order to improve the convergence rate, we introduce the technique of Variance Reduction that allows us to use a larger step size for each update, so that SGD can achieve the same linear convergence rate as of GD.

2 Variance Reduction

SGD allows us to do significantly less computations for each update, but the trade-off of this is a slower convergence rate. Since SGD approximates the full gradient by using samples, which introduces variances, require the step size decays to zero to ensure convergence.
To reduce the variance in the gradient approximation, so that improve the original sub-linear convergence rate, we will introduce two variance reduction algorithms, SVRG [1] and SAGA [2].
Objective Problem:

$$\min_w \frac{1}{n} \sum_{i=1}^{n} f_i(w)$$

2.1 SVRG

- General Idea:
  Keep a record of w after every m iterations, $m \geq n$, call this $\tilde{w}$, then we compute the average gradient immediately:

$$\tilde{\mu} = \frac{1}{n} \sum_{i=1}^{n} \nabla f_i(\tilde{w})$$

  we can show that $\mathbb{E}[\nabla f_i(\tilde{w}) - \tilde{\mu}] = 0$ by similar steps of (1) in the previous section. Now, for each iteration k, we randomly choose some $i_k \in \{1, ..., n\}$ we update w_{k+1} by:

$$w_{k+1} = w_k - h_k(\nabla f_{i_k}(w_k) - \nabla f_{i_k}(\tilde{w}) + \tilde{\mu}) \qquad \text{(SVRG)}$$

  so by (1), we have:

$$\begin{aligned} \mathbb{E}[w_{k+1}|w_k] &= w_k - h_k \nabla f_{i_k}(w_k) \\ &= w_k - \frac{h_k}{n} \nabla \sum_{j=1}^{n} f_j(w_k) \end{aligned}$$

- Exam the Variance:
  When both $\tilde{w}, w_k$ converge to the same value w^*, then $\tilde{\mu} \to 0$, thus, suppose $\nabla f_i(\tilde{w}) \to \nabla f_i(w^*)$, then the direction vector in the update rule (SVRG):

$$\nabla f_{i_k}(w_k) - \nabla f_{i_k}(\tilde{w}) + \tilde{\mu} \to \nabla f_{i_k}(w_k) - \nabla f_{i_k}(w^*) \to 0$$

- The above SVRG algorithm was developed by Johnson and Zhang in 2013 [1]. They also proved that it enjoys a faster convergence rate by allowing larger step size, assuming f_i are convex and smooth, SVRG has a linear convergence rate.

2.2 SAGA

- Initialization:
  Take some initial value $w_0 \in \mathbb{R}^d$, let $\delta_{i,0} = w_0$, and compute and store the value $\nabla f_i(\delta_{i,0}) \in \mathbb{R}^d \ \forall i \in \{1, ..., n\}$ in a $d \times n$ table, where i-th column of the table stores the value of $\nabla f_i(\delta_{i,k})$.

- Algorithm:

  1. Randomly choose some $j \in \{1, ..., n\}$.

  2. Let $\delta_{j,k+1} = w_k$, update the j-th column of the table by $\nabla f_j(\delta_{j,k+1})$, without changing any other columns.

  3. Update w:

$$w_{k+1} = w_k - h\left(\nabla f_j(\delta_{j,k+1}) - \nabla f_j(\delta_{j,k}) + \frac{1}{n}\sum_{i=1}^{n} \nabla f_i(\delta_{i,k})\right) \qquad \text{(SAGA)}$$

  where h is the step size, and notice that $\nabla f_i(\delta_{i,k})$ are the pre-stored values, so we do not need to compute them every iteration.

- The SAGA algorithm was developed by Defazio, Bach, and Lacoste-Julien in 2014 [2], they also proved that the convergence rates for SAGA is better than SVRG for strongly convex f_i.

3 Experiment

We conducted an experiment for a convex case[1], i.e. Logistic Regression, on rcv1.binary dataset [3] from the LIBSVM site.[2] We choose the changing of tuned w over iterations and accuracy of prediction for every 1000 iterations as our performance evaluation. In Figure 1, on the left is $\|w_{k+1} - w_k\|_1$ for each iteration, on the right is the accuracy improvement for every 1000 iterations. We chose the step size as 1 and maximum iteration as 50000, each time randomly pick 3 gradients for SVRG. From the running time, SVRG takes about 3 minutes to finish 50000 iterations with 95.61% final accuracy, SAGA only takes about 1 minute 10 seconds for 50000 iterations, and with 95.96% final accuracy, even better than SVRG.

From the right plot we can see that SVRG performed better than SAGA previous than about 15000 iterations, then SAGA starts to transcend SVRG. From the smoothness of the curve, we know that SAGA converges more stably than SVRG.

[1]Codes: Click for Google Colab Notebook.
[2]LIBSVM: LIBSVM site.

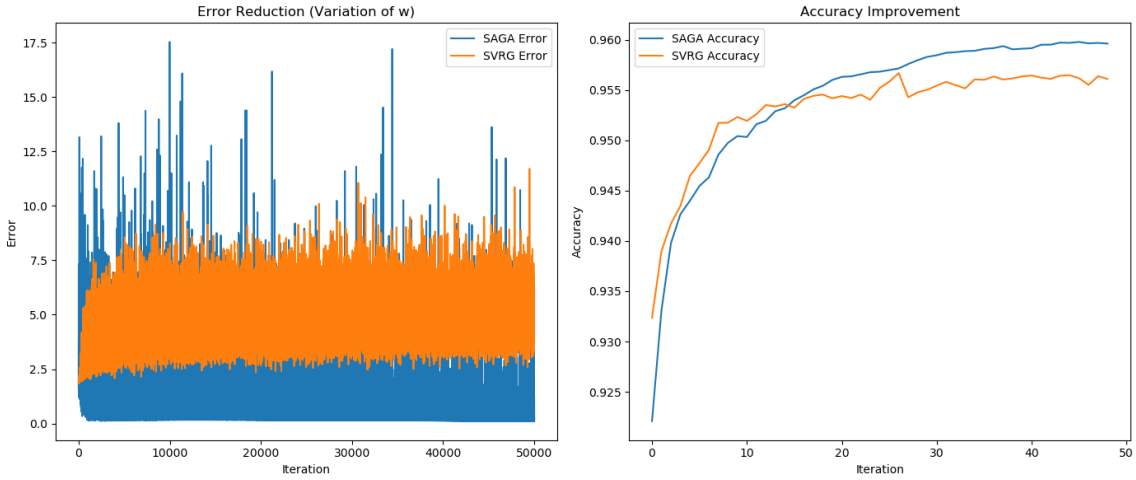

Figure 1: Performance Plot

4 Conclusion

From the experiment we can find that with variance reduction technique, we can achieve high accuracy with fast speed and low computational costs, where both SAGA and SVRG algorithms run 50000 iterations within a few minutes, which may take the standard Gradient Descent Algorithm more than an hour. Comparing SVRG and SAGA, SAGA gives more stable convergence and faster speed than SVRG.

5 References

References

[1] Rie Johnson and Tong Zhang. *Accelerating Stochastic Gradient Descent using Predictive Variance Reduction*. Advances in neural information processing systems 26, 2013.

[2] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. *SAGA: A Fast Incremental Gradient Method With Support for Non-Strongly Convex Composite Objectives*. Advances in neural information processing systems 27, 2014.

[3] David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. *RCV1: A new benchmark collection for text categorization research*. Journal of Machine Learning Research, 5:361–397, 2004.