ECE 901: Large-scale Machine Learning and Optimization **Fall 2016**

Lecture 6 — 09/22/2016

Lecturer: Dimitris Papailiopoulos *Scribe: Sneha Rudra*

Note: These lecture notes are still rough, and have only have been mildly proofread.

6.1 Stochastic Gradient Descent

6.1.1 Introduction

In the last two lectures we studied the Gradient Descent Method extensively in the context of convex objective functions. We also saw previously that a typical Empirical Risk Minimization objective $f(\mathbf{x})$ can be expressed as follows-

$$f(\mathbf{x}) = \frac{1}{n}\sum_{i}^{n} f_i(\mathbf{x}) \tag{6.1}$$

where the summation is over n training instances, $f_i(\mathbf{x})$ is the convex loss function evaluated at i-th training instance and the decision variable $\mathbf{x}$ represents the parameter vector of the classifier/model belonging to a specific hypothesis class.

We also saw that if we used Gradient Descent to minimize this objective $f(\mathbf{x})$, we would be required to compute $\nabla f(\mathbf{x})$ and hence we would need n gradient computations one corresponding to each training instance i during every iteration of the algorithm.

In the present lecture, we explore the Stochastic Gradient Descent (SGD) algorithm which leverages the fact that performing a full gradient computation during every iteration may often times be unnecessary and thus seeks to reduce the per-iteration computation. SGD is simple to implement, has a small memory footprint and has been shown to perform well on several problems. It is the basis of many algorithms that have been developed to solve large scale optimization problems which appear in various machine learning applications.

6.1.2 Stochastic Gradient Descent Update Steps

The key idea here is that in order to make progress towards the minima, it is enough that on an average our gradient computations are correct [1]. Instead of picking a locally optimal direction in which to update the current iterate, SGD picks an index i uniformly at random from [1,2...n] and evaluates the gradient of f_i at the current iterate to be the update direction

(i.e. at the current iterate, instead of the full gradient, it uses the gradient of one loss function (f_{s_k}) picked randomly from the set of n losses.). SGD update steps are:

- Sample $s_k \sim$ unif{1,n}, s_k are i.i.d
- Update the current iterate according to the following- $x_{k+1} = x_k$ - $\gamma \nabla\ f_{s_k}(x_k)$

x_k is the current iterate and γ is the step size.
Clearly, SGD requires only one gradient computation per iteration (while Gradient Descent requires n). However, as we will see later, since we give up on accuracy, there is usually a tradeoff and while the computational effort per iteration is less, SGD in general needs more iterations than Gradient Descent to reach a specified optimization error.

6.1.3 Expected Convergence Rate of Stochastic Gradient Descent

The following properties and assumptions have been used in proving the Expected Convergence Rate of SGD.

- Stochastic gradients have the property that for any iterate $\mathbf{x}$, in expectation, the stochastic gradient is equal to the true gradient

$$\mathbf{E}_s[\nabla f_s(\mathbf{x})] = \nabla f(\mathbf{x}) \tag{6.2}$$

- Uniform bound on stochastic gradients assumption:

$$\mathbf{E}_s\|\nabla f_s(\mathbf{x})\| \leq M^2 \tag{6.3}$$

  i.e. in expectation the norm of the stochastic gradient is bounded.

- Further f has been assumed to be λ strongly convex. In particular this implies

$$\langle \nabla f(x) - \nabla f(y), x - y \rangle \geq \lambda \|x - y\|^2 \tag{6.4}$$

Theorem 6.1. *For $\gamma = \frac{\epsilon\lambda}{M^2}$ and number of iterations $T = O(\frac{M^2}{\lambda^2}\frac{\log(\frac{\Delta_1}{\epsilon})}{\epsilon})$, after T iterations of SG we get:*

$$\mathbf{E}(\Delta_{k+1}) \leq \epsilon \tag{6.5}$$

Where Δ_k denotes the distance between the k-th iterate and the optimum i.e. $\Delta_k = \|x_k - x*\|^2$. Note that Δ_1 shows up in the estimate of T but in most cases assuming that $\Delta_1 = O(d)$ is reasonable especially when we are operating in a compact domain. M and λ correspond to equations 6.3 and 6.4 above.

Proof: Consider the distance of the $k+1$-th iterate from the optimum and substitute for $x_{k+1} = x_k - \gamma \nabla f_{s_k}(x_k)$ from the update step-

$$\|x_{k+1} - x*\|^2 = \|x_k - \gamma \nabla f_{s_k}(x_k) - x*\|^2 = \|x_k - x*\|^2 - 2\gamma \langle \nabla f_{s_k}(x_k), x_k - x* \rangle + \gamma^2 \|\nabla f_{s_k}(x_k)\|^2 \tag{6.6}$$

Since the directions $s_1, s_2 ... s_k$ were randomly chosen and the current iterate x_k depends on all the previous directions, we take the expectation of the above equation with respect to $s_1, s_2 ... s_k$.

$$\mathbf{E}_{s_1,..s_k}[\|x_{k+1} - x*\|^2] = \mathbf{E}_{s_1,..s_k}[\|x_k - x*\|^2] - 2\gamma \mathbf{E}_{s_1,..s_k}[\langle \nabla f_{s_k}(x_k), x_k - x* \rangle] + \gamma^2 \mathbf{E}_{s_1,..s_k}[\|\nabla f_{s_k}(x_k)\|^2] \tag{6.7}$$

Expanding the second term on the right hand side of the above equation-

$$\mathbf{E}_{s_1,..s_k}[\langle \nabla f_{s_k}(x_k), x_k - x* \rangle] = \mathbf{E}_{s_1..s_{k-1}} \mathbf{E}_{s_k} \langle \nabla f_{s_k}(x_k), x_k - x* \rangle \tag{6.8}$$

$$= \mathbf{E}_{s_1..s_{k-1}} \langle \mathbf{E}_{s_k}(\nabla f_{s_k}(x_k)), x_k - x* \rangle \tag{6.9}$$

Note that equation 6.9 holds because s_k (s_k is a random number between 1 and n) and x_k are independent of each other. Further using equation 6.2,

$$= \mathbf{E}_{s_1..s_k} \langle (\nabla f(x_k)), x_k - x* \rangle \tag{6.10}$$

Plugging equation 6.10 back into 6.7 and using the uniform bound from equation 6.3 we have-

$$\mathbf{E}_{s_1..s_k}[\|x_{k+1} - x*\|^2] \leq \mathbf{E}_{s_1..s_k}(\|x_k - x*\|^2) - 2\gamma \mathbf{E}_{s_1..s_k}[\langle \nabla f(x_k), x_k - x* \rangle] + \gamma^2 M^2 \tag{6.11}$$

Next using strong convexity (equation 6.4), using the fact that $\nabla f(x*) = 0$ and setting $\Delta_k = \|x_k - x*\|^2$ we get

$$\mathbf{E}_{s_1..s_k}(\Delta_{k+1}) \leq (1 - 2\gamma\lambda) \mathbf{E}_{s_1..s_k}(\Delta_k) + \gamma^2 M^2 \leq \epsilon \tag{6.12}$$

We repeat the above for for k, $k-1..1$ and add the inequalities as follows:

$$\mathbf{E}_{s_1..s_k}(\Delta_k) \leq (1 - 2\gamma\lambda) \mathbf{E}_{s_1..s_k}(\Delta_{k-1}) + \gamma^2 M^2 \tag{6.13}$$

$$\mathbf{E}_{s_1..s_k}(\Delta_{k-1}) \leq (1 - 2\gamma\lambda) \mathbf{E}_{s_1..s_k}(\Delta_{k-2}) + \gamma^2 M^2 \tag{6.14}$$

$$\vdots$$

$$+$$

$$\mathbf{E}_{s_1..s_k}(\Delta_2) \leq (1 - 2\gamma\lambda) \mathbf{E}_{s_1..s_k}(\Delta_1) + \gamma^2 M^2 \tag{6.15}$$

$$\downarrow$$

$$\mathbf{E}_{s_1..s_k}(\Delta_{k+1}) \leq (1-2\gamma\lambda)^k \mathbf{E}_{s_1..s_k}(\Delta_1) + \sum_{i=0}^{k-1}(1-2\lambda\gamma)^i(\gamma^2 M^2) \tag{6.16}$$

Next, under the assumption that $0 \leq 2\gamma\lambda \leq 1$, we can use the following property-

$$\sum_{i=0}^{\infty}(1-a)^i \leq \frac{1}{a} \text{ if } 0 \leq a \leq 1 \tag{6.17}$$

Simplifying equation 6.16 using above and setting $\mathbf{E}_{s_1..s_k}(\Delta_1) = \Delta_1$ (the starting point is known and hence Δ_1 is deterministic)

$$\mathbf{E}_{s_1..s_k}(\Delta_{k+1}) \leq (1-2\gamma\lambda)^k \Delta_1 + \frac{\gamma M^2}{2\lambda} \leq \epsilon \tag{6.18}$$

We observe that we have contraction due to the first term as the number of iterations k increases, but we lose some of it due to $\frac{\gamma M^2}{2\lambda}$. Finally, in order to get to an optimization error of ϵ, we have two parameters that can be tuned, γ (the step size) and k (the number of iterations). Assuming that both the terms in equation 6.18 contribute $\epsilon/2$ to the optimization error, we get-

$$\gamma = \frac{\lambda\epsilon}{M^2} \tag{6.19}$$

Now using this value of γ we get

$$k = O(\frac{M^2}{\lambda^2}\frac{\log(\Delta_1/\epsilon)}{\epsilon}) \tag{6.20}$$

Hence theoretically, tuning the parameter γ according to equation 6.19 and performing number of iterations on the order provided by equation 6.20 can hence help SGD obtain an optimization error of ϵ, in expectation. □

6.1.4 Comparison of Gradient Descent and Stochastic Gradient Descent

In summary, the running time of Gradient Descent and Stochastic Gradient Descent can be compared as follows:

SGD : To reach an accuracy of ϵ the number of iterations $T = O(\frac{M^2}{\lambda^2}\frac{\log(\frac{\Delta_1}{\epsilon})}{\epsilon})$

GD : To reach an accuracy of ϵ the number of iterations $T = O(\frac{\beta}{\lambda}\log(\frac{\beta}{\lambda\epsilon}))$

Clearly, if ϵ is the dominating factor, then Gradient Descent requires exponentially smaller number of iterations but the work per iteration is n times more compared to SGD. Note that these results are based on different step sizes for Gradient Descent and SGD. The number of iterations for Gradient Descent is based on our discussion in the previous lecture.

6.1.5 Example: Logistic Regression

$$f(x) = \frac{1}{n}\sum_{i=1}^{n} \log(1 + \exp(-y_i a_i^T x)) + \lambda \|x\|_2^2 \tag{6.21}$$

Consider the following assumptions/ properties hold:

- We are operating in a d dimensional sphere, where d represents the dimensions
- $\|a_i\| = O(\sqrt{d})\ \forall i$ (From the property of vector norm)
- $\Delta_1 = O(d)$ (Reasonable on a d- dimensional sphere)
- $M^2 = O(d)$, $\beta = O(d)$

To reach an accuracy of ϵ

$$\#iter_{GD} = O(\frac{\beta}{\lambda}\log(\frac{\beta}{\lambda\epsilon})) = O(\frac{d}{\lambda}\log(\frac{d}{\lambda\epsilon})) \tag{6.22}$$

$$\#iter_{SGD} = O(\frac{M^2}{\lambda^2}\frac{\log(\frac{\Delta_1}{\epsilon})}{\epsilon}) = O(\frac{d}{\lambda^2}\frac{\log(\frac{d}{\epsilon})}{\epsilon}) \tag{6.23}$$

From the previous lecture, if A matrix has columns given by a_i,

$$\#cost_{GD}/iter = O(nnz(A)) \tag{6.24}$$

Similarly, cost of evaluating a single gradient would be given by

$$\#cost_{SGD}/iter = \frac{1}{n}O(nnz(A)) \tag{6.25}$$

$$\frac{cost_{GD}/iter \times \#iter_{GD}(\epsilon)}{cost_{SGD}/iter \times \#iter_{SGD}(\epsilon)} = \frac{nnz(A) \times \frac{d}{\lambda}\log(\frac{d}{\lambda\epsilon})}{\frac{nnz(A)}{n}\frac{d\epsilon}{\lambda^2\epsilon}\log(\frac{d}{\epsilon})} = O(n\lambda\epsilon) \tag{6.26}$$

This makes sense from the point of view of SGD as O($n\lambda\epsilon$) represents the saving per iteration times the loss in terms of number of iterations.

6.1.6 Conclusion

In the next lecture, we will study Stochastic Variance Reduced Gradient Method also known as SVRG which hopes to achieve the best properties of both GD and SGD - number of iterations proportional to $\log(\frac{1}{\epsilon})$ (ϵ is the optimization error) and small amortized computational complexity per iteration.

[1] Convex Optimization: Algorithms and Complexity, Sebastien Bubeck, Chapter 6.