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

Lecture 14 — 10/20

Lecturer: Dimitris Papailiopoulos *Scribe: Ronak Mehta*

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

14.1 A Brief Recap

Previously we discussed algorithmic stability, where if an algorithm's output w is defined as a function of the data $A(S)$, then the empirical risk minimizer (ERM) is

$$A(S) = \arg\min_w \frac{1}{n} \sum_{i=1}^{n} l(w, x_i) \tag{14.1}$$

We saw that if the loss functions $l(w, x_i)$ are convex and Lipschitz smooth, or strongly convex, then if you add a regularizer $\lambda||w||$ you automatically get algorithmic stability. If the objective is λ-strongly convex, then the stability of the algorithm is approximately $\mathrm{O}(\frac{1}{\lambda n})$, where n is the number of samples. This is intuitive, as a large number of samples will inherently be invariant to a single one, and a stronger regularization will only lead to the model learned being more agnostic to *all* sample points.

Note. This is a statement about an algorithm which is completely static. In other words, the only randomness introduced into our model comes from the underlying distribution from which the data was sampled.

14.2 Randomized Stability

What if we are using a *randomized* algorithm? If now our randomness not only comes from the sampling distribution but also from our learning algorithm, can we conclude something about stability?

To develop this idea, let's look at the most common randomized algorithm in use, which also fits nicely into our ERM setup above. Stochastic gradient descent (SGD) is the method most commonly used to solve objectives such as the ERM 14.1. The minimization procedes in steps, during which at each step the subproblem to be minimized is the loss function evaluated at a random sample from the training set. In this particular setup, let us assume that we run SGD for T iterations, where T may be greater than the number of samples n.

To further clarify, instead of looking at our generalization error as solely a function of the sample S,

$$\epsilon_{gen} = E_S \left[\hat{R}(A(S)) - R(A(S)) \right] \tag{14.2}$$

we are also interested in accounting for randomness in the algorithm:

$$\epsilon_{gen} = E_{S,A}\left[\hat{R}(A(S)) - R(A(S))\right] \tag{14.3}$$

Where $\hat{R}$ and R represent the empirical and generalization risk respectively.

14.2.1 Some Definitions

Let the sample set $S = \{z_1, \ldots, z_n\}$, where $z = (\vec{x}, y) \sim D$. Also define the set $S^i = \{z_1, \ldots, z_{i-1}, z_i', z_{i+1}, \ldots, z_n\}$ with the same samples as S except differing on z_i, z_i'.

Recall from the previous lecture and from [1] that an algorithm A is ϵ-uniformly stable if

$$\sup_{z,S,z_i} E_A\left[l(A(S), z) - l(A(S^i), z)\right] \leq \epsilon \tag{14.4}$$

And that ϵ-uniform stability implies $\epsilon_gen \leq \epsilon$ generalization error.

Consider running SGD for T iterations, and let w_T and w_T' denote the corresponding outputs of the stochastic gradient models whose update takes the following form:

$$w_{t+1} = w_t - \alpha_t \nabla l(w_t; z_{st}) \quad , \quad st \sim Unif(1, \ldots, n) \tag{14.5}$$

14.2.2 Convex Optimization Stochastic Stability

The following result comes from [2].

Theorem 14.1. *Assume that the loss function $f(,\dot{}z)$ is β-smooth, convex, and L-Lipschitz for every z. Suppose that we run SGD with step sizes $\alpha_t \leq 2/\beta$ for T steps. Then SGD satisfies uniform stability with*

$$\epsilon_{stability} \leq \frac{2L^2}{n} \sum_{t=1}^{T} \alpha_t \tag{14.6}$$

Proof: Let $G_1, \ldots, G_T$ and $G_1', \ldots, G_T'$ be the gradient updates induced by the algorithm run on S and S', and the corresponding models by w and w' as defined above. Define $\delta_T = ||w_T - w_T'||$. Now fix the example $z \in Z$ as a random sample from the distribution. Applying the Lipschitz condition of the loss function over z, we have

$$E|l(w_T; z) - l(w_T'; z)| \leq LE||w_T - L_T'|| = LE||\delta_t|| \tag{14.7}$$

We also have that:

$$\delta_{t+1} = ||w_{t+1} - w'_{t+1}|| = ||w_t - \alpha_t \nabla l(w_t; z_{st}) - w'_t + \alpha_t \nabla l(w'_t; z'_{st})|| \tag{14.8}$$
$$\leq ||w_t - w'_t|| + \alpha_t ||\nabla l(w_t; z_{st}) - \nabla l(w'_t; z_{st})|| \tag{14.9}$$
$$\leq ||w_t - w'_t|| + \alpha_t (||\nabla l(w_t; z_{st})|| - ||\nabla l(w'_t; z_{st})||) \tag{14.10}$$
$$\leq ||w_t - w'_t|| + 2\alpha_t L \tag{14.11}$$
$$\tag{14.12}$$

The first and second inequalities are just the result of splitting up the norm, and the third follows from the fact that regardless of what the update is, both models have Lipschitz-bounded gradients.

At any specific step t, the probability that we have sampled the one sample that the two models differ on is $1/n$, and in this case we have the bound on the difference from 14.8. On the other hand, the probability that the models are exactly the same, $G_t = G'_t$, is $1 - 1/n$, and the difference is exactly equal to the difference from the previous iteration. We can then describe the expected-norm difference between the models directly based on these probabilities:

$$E[\delta_{t+1}] \leq \left(1 - \frac{1}{n}\right) E[\delta_t] + \frac{1}{n}(E[\delta_t] + 2\alpha_t L) = E[\delta_t] + \frac{2\alpha_t L}{n} \tag{14.13}$$

We can unravel the recursion as follows:

$$E[\delta_T] \leq E[\delta_{T-1}] + \frac{2\alpha_{T-1} L}{n} \tag{14.14}$$
$$\leq E[\delta_0] + \sum_{t=1}^{T-1} \frac{2\alpha_t L}{n} \tag{14.15}$$
$$\leq \sum_{t=0}^{T-1} \frac{2\alpha_t L}{n} \tag{14.16}$$
$$\tag{14.17}$$

and so

$$E[\delta_{T+1}] \leq E[\delta_t] + \frac{2\alpha_t L}{n} \leq \frac{2L}{n} \sum_{t=1}^{T} \alpha_t \tag{14.18}$$

Plugging this back in to 14.7, we have

$$E|l(w_T; z) - l(w'_T; z)| \leq \frac{2L^2}{n} \sum_{t=1}^{T} \alpha_t \tag{14.19}$$

While we fixed z here, we did not choose it to be anything specific, and similarly with S and S'. With these conditions, this bound indeed holds for uniform stability.

□

Main Takeaway. The important idea this result is that if we know for certain that the function is convex, β-smooth, and L-Lipschitz, then as long as our step sizes are reasonable and we do not run for *too* many iterations, our generalization error is guaranteed to be small, regardless of the underlying data distribution.

14.2.3 Other function types

We have even stronger bounds for strongly convex loss functions, where we find that $\epsilon_{stability} \leq \frac{2L^2}{\lambda n}$. For nonconvex objectives, unsurprisingly, we find that the stability is only exponentially bounded, and requires the step size α_t to be non-increasing with order $1/Lt$. The proof and more details can again be found in [2].

14.2.4 Inducing stability

In many cases we may have a formulation for a problem which is not inherently stable, but which we may want to make stable. Towards this end there are many operations we can apply to do so. Some existing methods that lead to stability include weight decay, gradient clipping, dropout, and projections with proximal steps.

14.3 Open Problems

While these stability results have significant impact through their direct relationship with generalization error, there are still many open problems that have yet to be answered. In particular, could these ideas be extended to a concentration result, even if limited to the convex or strongly convex case? Is there a *family* of algorithms where the step sizes can be guaranteed to be not too small? Are gradient descent, randomized gradient descent, SVRG, etc. stable? If you do not know the underlying algorithm, can you test stability?

Bibliography

[1] Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of Machine Learning Research*, 2(Mar):499–526, 2002.

[2] Moritz Hardt, Benjamin Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. *arXiv preprint arXiv:1509.01240*, 2015.