Lecture #: Regularization and Stability

Instructor: Aditya Bhaskara *Scribe: Anurag Gupta*

CS 5966/6966: Theory of Machine Learning

March 3rd, 2022

Abstract

In this lecture we go over the concepts of regularization and stability, and how stability is related to generalization and concentration bounds. We discuss when do we call a function to be stable, and what effect changing a few input points can have on the stability of the loss function.

1 Improvements and Generalization

Heavy Ball Method (Momentum): On a high level to add to the update equations an additional term which depends on the previous movement.

$$w_{t+1} = w_t - \eta \nabla f(w_t) - \partial(w_t - w_{t-1})$$

As seen in the previous lecture, for strongly convex functions our error after T steps was $exp(-\frac{\mu}{M}.T)$ or $exp(-\frac{1}{\kappa}.T)$, but when we use the above update method (momentum) for strongly convex functions, we observe that we can achieve a better bound like $exp(-\frac{1}{\sqrt{\kappa}}.T)$.

Second order methods, and first order proxies: Vanilla gradient descent is a first order update method, since it involves only the first order derivative. Adagrad is a first order approximation to the second order methods. Newton method is a type of a second order method wherein instead having a fixed η value we choose a η that depends on the Hessian.

We notice that in the above methods the general theme is to avoid the slow convergence which is what happens in gradient descent, the above methods suggest to move fast in the regions where it is possible for the convergence function to move fast by taking larger steps.

We see how this theme of taking large steps when possible can help us in different ways:-

Helpful when dealing with non-convex functions : It is possible for us to get stuck at a local minima, but if we had some momentum we could keep moving further along and get out of it.

Perturbed Gradient Descent: Since the amount of movement is dependent on the gradient of the function, in cases the gradient is tiny the movement is also very little, and so to tackle this after a little while of not moving we make a random jump to a uniformly random point in the certain ball around of that point and then continue again.

Can also help us prove formally that this method allows for us to get bad saddle points.

Finally, we see that when this process converges, we end up at an "approximately" local minima.
We define local minima as the point where the value of the Hessian is positive semi-definite ($\nabla^2 f \geq 0$), "approximately" local minima is a point where the Hessian is not positive semi-definite but greater than some small constant s times I ($\nabla^2 f \geq s.I$).

2 Choosing Loss Functions

The motivation for choosing loss functions: If we are able to choose a nice or smooth loss function then we know that the optimization can be done faster.

Consider data points $(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)$, with the loss function,

$$Loss(w) = \frac{1}{m}.\sum_{i=1}^{m} l(w, x_i, y_i)$$

If we wish for the above loss function to be smooth we can add an additional term, to make it strongly convex, so our new loss function becomes,v

$$Loss(w) = \frac{1}{m}.\sum_{i=1}^{m} l(w, x_i, y_i) + \theta\|w\|^2$$

this can also be achieved by replacing function l with another function l', where,

$$l'(w, x_i, y_i) = l(w, x_i, y_i) + \theta\|w\|^2$$

But it's important to note that when θ, gets a very large value, the second term in the above equation starts to dominate the first term (which is our objective term), so we end up minimizing the second term instead of our objective.

The benefit of having a nice or a smooth loss function is not only that the loss function optimizes well, but also that it generalizes well and also provides stability to input changes.

In plain terms, we can think of stability as when we a have bunch of inputs and we minimize it's loss and find the optimal value of w, now we change some data point in our input and again repeat the above procedure of minimizing the loss and calculating the optimal w', when w and w' are very close to each other we call the function to be stable.

If a function is stable it also generalizes well and stability is also related to concentration inequalities.

3 Stability of a Loss Minimization Algorithm

Let us consider an example, where we are given input points $(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)$, with the loss function

$$L(w) = \frac{1}{m}.\sum_{i=1}^{m} l(w, x_i, y_i)$$

And after minimizing the loss, we find the $argmin_w L(w) = w^*$.
We can consider a map from the input examples to the parameter value w, so we can say, for an input,

$$S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$$

we get $w^*(S)$, which is the best function when we are in S. We think of this as deterministic training procedure, which means that, given some fixed input we only get a fixed output.
Here $w*$ is a map that takes some input and produces the best hypothesis for an S.

Now how does changing a single (x_i, y_i) change the w^*?
Consider S',

$$S' = \{(x_1, y_1), (x_2, y_2), \ldots, (x_{m-1}, y_{m-1}), (x'_m, y'_m)\}$$

for which we get $w * (S')$.
Let's formally define stability here first. Loss function is said to be $\delta(m)$-stable (for training sets of size m), if $\forall\ x, y, x'_m, y'_m$ as above $\|w * (S') - w * (S')\| \leq \delta(m)$.

Now let's say our loss functions are bounded between $[0, 1]$ always, and we change one of the points, what should be the difference between the objective values at both the stages? It should definitely not be more than $\frac{1}{m}$, since we can think of it as, that there are m examples, and we are changing only 1 value, so the objective value should not change by more than $\frac{1}{m}$. So we can say,

$$\|L(w^*(S')) - L(w^*(S))\| \leq \frac{1}{m}$$

4 Understanding Stability - Linear Functions

Suppose we are optimizing over $w \in [-1, 1]$, and we have the loss function values for all our training examples, which are of the fashion, w for the first training example ($l_1(w)$), $-2w$ for the second training example ($l_2(w)$), $2w$ for the third training example ($l_3(w)$), $-2w$, $2w$, $\ldots$ for the following training examples.

Now if we examine,

$$L(w) = \frac{1}{m}.\sum_{i=1}^{m} l_i(w)$$

we see that,

$$argmin_w L(w) = \begin{cases} -1 & \text{is } m \text{ is even} \\ 1 & \text{is } m \text{ is odd} \end{cases}$$

And when we change any of the $2w$ loss function's value to a $-2w$ value, in that case the $argmin_w L(w)$ flips from $+1$ to -1.

So we can say that, in general changing one of the l_i's can significantly change the w^*.

4.1 theorem. *If $L(w)$ is α-strongly convex and we replace $l_i(w)$ with $l_i'(w)$ such that $\|\nabla(l_i' - l_i)\| \leq G$ say, then $\|w^*(S') - w^*(S)\| \leq \frac{G}{m\alpha}$.*

It is saying that, if we change one of the input terms the loss functions gets changed by a value roughly $\frac{1}{m}$ and also the optimizer gets changed by value $\frac{1}{m}$, if the value G is not too big and α is not too small.

So, we can say that if the loss function is chosen such that the objective becomes strongly convex then we automatically have stability along with fast convergence and other good properties, like generalization.

5 Stability implies Generalization

Generalization Gap is defined as $|\text{Loss on test} - \text{Loss on training}|$, where the data is independent and identically distributed and $\in$ to a distribution D.

Suppose our loss function $l(w, x_i, y_i)$ is $\delta(m)$-stable then, the generalization error is $\leq \delta(m)$.

Loss on test of $w^*(S) = \mathbb{E}_{x_m' \ D} l(w^*(S), x_m')$
Loss on training of $w^*(S) = \mathbb{E}_{i \ [m]} l(w^*(S), x_i)$

The Loss on training can also be written as $\mathbb{E}_{i'} \mathbb{E}_{i \ [m]} l(w^*(S \backslash x_i + x_i'), x_i)$.

Regularization has two benefits which are, firstly, it helps improve the generalization gap, which can at times be a "fake win" sometimes since it is possible that we might have high error before regularization. And secondly, it helps improve convergence.

6 Concentration Bounds and Stability

Consider the Chernoff bound, where $X_1, X_2, \ldots, X_n$ are independent and identically distributed variables from distribution D, with support $[a, b]$, and a mean μ,

$$Pr[|\frac{1}{n}(X_1 + X_2 + \cdots + X_n - \mu)| > t] \leq e^{-\frac{t^2 n}{a-b}}$$

Now suppose we change a random variable, X'_n, then it can maximum differ in value by $\frac{a-b}{n}$.

This can be shown by a similar result by Talagrand,

6.1 THEOREM. *Let f be any function which is δ-stable, that is,*

$$|f(X_1, X_2, \ldots, X_n) - f(X_1, X_2, \ldots, X_{i-1}, X'_i, \ldots, X_n)| \leq \delta$$

then, if $X_1, X_2, \ldots, X_n$ are independent and identically distributed random variables then,

$$Pr[|f(X_1, X_2, \ldots, X_n) - \textit{median value of the function}| \geq t] \leq e^{-\frac{t^2 n}{\delta^2}}$$

It basically says that, the probability that the function value for the IID variable $X_1, X_2, \ldots, X_n$, differs from the median function value depends on the smoothness parameter δ.
So if we take any function which does not change too much we can say that it is very well concentrated.