COS 597G: Toward Theoretical Understanding of Deep Learning **Fall 2018**

Lecture 7: October 9

Lecturer: Sanjeev Arora *Scribe: Revisions: Charlie Hou, Original ver: Hrishikesh Khandeparkar*

7.1 Preliminaries

Generalization theory is concerned with the ability of a set of candidate functions to perform on data. In these notes, we will be primarily concerned with worst-case bounds where we don't really exploit the structure of the functions that we are considering. Such bounds typically depend on how "big" the set of functions we consider as candidates and how many samples we have to train with.

A quick disclaimer: generalization theory has so far had a difficult time explaining the success of deep learning, which may require analysis that doesn't focus on analyzing worst-case scenarios. Some progress has been made in studying

7.1.1 Empirical Risk Minimization (ERM)

Define

$\mathbf{D}$: Distribution on labelled data

$\mathbf{S_m}$: m i.i.d samples of data points z_i drawn from $\mathbf{D}$

$\mathcal{H}$: The set of classifiers h which we consider to explain the data

$l(h, z)$: Loss of classifier $h \in \mathcal{H}$ on point $z = (x, y)$

$L_{S_m}(h)$: Average loss of classifier h on the sample S_m

Then, for a hypothesis $h \in \mathcal{H}$ we have that $L_{S_m}(h) = \mathrm{E}_{z \sim S_m}[l(h, z)]$. Here, $\mathrm{E}_{z \sim S_m}$ means expectation over z drawn uniformly at random from $\mathbf{S_m}$. We drop the m and say S from now on, with the m being implied. This loss $L_S(h)$ models the steps of measuring how well our theory (here, hypothesis h) can explain the data. A simple loss function is binary —loss 0 for correct classification and loss 1 for incorrect. In addition it may have a term $complexity(h)$ that measures the complexity of h. One simple such measure is the sum of the absolute values of the numbers that describe h.

Now we seek to find a

$$h_S = argmin_h L_S(h)$$

which best explains the dataset. We denote $L_S = L_S(h_S)$ and call it the empirical loss. Thus, we call this method **Empirical Risk Minimization** as it minimizes the empirical error.

However the empirical error isn't the true error/loss that we are actually interested in.

In order to say that a classifier h is actually good, we need it to perform well on real world data. Specifically, we are seeking to minimize $L_D(h) = \mathrm{E}_{z \sim D}[l(h, z)]$. We call this $L_D(h)$ the true loss of the hypothesis h. How is $L_D(h)$ related to $L_S(h)$ for a given h?

We define the generalization error as

$$\Delta_S(h) = L_D(h) - L_S(h)$$

This measures how well hypothesis h generalizes to a distribution D when its performance on a sample S drawn from D is known.

Intuitively, if the generalization error is large then the hypothesis's performance on sample S does not accurately reflect the performance on the full distribution of examples, so we say it *overfitted* to the sample S.

A trivial example of this is the hypothesis class that assigns the known label to all seen examples, and the label 0 to all unseen examples. Clearly, this hypothesis class can achieve 0 loss on any dataset but won't perform well in the real world. Another example from folklore is how conspiracy theorist can join seemingly random facts to explain an outcome with a theory but they are clearly not to be relied on for making good predictions about future.

In the opposite direction, if we had a hypothesis that assigned 0's to everything, we would have very low generalization error, as the hypothesis would perform roughly the same on testing data as on training data (due to both being drawn from the same distribution D). This "set" of hypotheses would have "0 complexity", and is incredibly inflexible; but it does achieve low generalization error.

Typically, when selecting a class of hypotheses $\mathcal{H}$, there's a tradeoff between how complex we want it to be, which will drive the empirical error down, and how simple we want it to be, which will decrease the generalization error. In practical applications, this loosely translates to having more parameters if we want low empirical error and having fewer if we want lower generalization error.

However, in deep learning, we don't seem to have this tradeoff. As we increase the number of parameters, the training loss AND the testing loss (the loss on out-of-sample data) is driven lower. This brings us to one of the fundamental mysteries of the field:

Why does overfitting not happen in deep learning?

7.2 Generalization Theory

Generalization theory tries to upper bound this error $\Delta_S(h)$. Sanjeev has a different way of phrasing this: *If overfitting occured and $\Delta_S(h)$ was high, then the hypothesis class was complex in some way.* Generalization theory formalizes what it means for classes to be complex. Sanjeev emphasizes that it is primarily a *descriptive* theory that gives a name for the type of complexity. But it is not a *prescriptive* theory, in that it gives no insight into how estimate this complexity.

7.2.1 Finding the generalization error

Start off by fixing a hypothesis h. Then we know from basic concentration bounds that $\mathrm{E}_D[l(h,z)]$ will be close to $\mathrm{E}_S[l(h,z)] = \frac{1}{m}\sum_{z_i \in S_m} l(h,z_i)$. In fact, we know that if $\|S\| = m$ and $\|loss\| \leq 1$, we have loosely that

$$\sqrt{m}\Delta_S(h) \sim N(0,1)$$

Therefore, under some fixed hypothesis, we know that the generalization loss will be low. However, we know that in reality, h is actually dependent on S. Now what do we do?

We use union bound. First assume that the set of models is bounded in some way (we will devise a way to extend this to a non-finite model class later): for example, assume there are N models. We know that

$$P(\Delta_S(h) > \epsilon) \leq e^{-\epsilon^2 m}$$

for some fixed h. Then we know that by union bound, for all hypotheses to have generalization error less than ϵ with probability greater than $1 - \delta$, we must have

$$Ne^{-\epsilon^2 m} \leq \delta$$

which gives us

$$m > \frac{logN - log(\delta)}{\epsilon^2}$$

We typically view δ as fixed. Therefore, we have that our samples must scale logarithmically with the cardinality of the set of models.

Now, we will try to deal with a case where the set of models is not bounded. Suppose that the model is the set of unit vectors in R^d in a ball of radius 1, and assume that loss(h) is Lipschitz with respect to these vectors on a fixed sample x. Then we know that

$$Loss(h_1(x)) - Loss(h_2(x)) \leq L\|h_1(x) - h_2(x)\|$$

.

This means that If we find a minimal covering of the space using balls of ϵ radius, and h_1 and h_2 are in the same ball (or more precisely, the vectors representing them are in the same ball), then we know that

$$\|\Delta_S(h_1) - \Delta_S(h_2)\| \leq \epsilon L$$

(up to constants, as truthfully we should be talking about balls of $\epsilon/2$ radius, but we keep things simple here for illustration). Then from high dimensional geometry, we know that the number of balls needed to cover the space of models is roughly $(\frac{1}{\epsilon})^d$.

Let's return to our original bounds in the finite case. We have a finite number of balls: $c(\frac{1}{\epsilon})^d$; select the centers of each of them. We know that for any function h and the function in the center of the ball that h is in (call the center h_b and the ball B),

$$P(\exists h \in B \ s.t. \ \Delta_S(h) > t) \leq P(\Delta_S(h_b) + \epsilon L > t) = P(\Delta_S(h_b) > t - \epsilon L) \leq e^{-(t-\epsilon L)^2 m}$$

Then, with this we can use the union bound strategy from the finite case:

$$P(\exists h \ s.t. \Delta_S(h) > t) \leq (\frac{1}{\epsilon})^d P(\exists h \in B \ s.t. \ \Delta_S(h) > t) \leq (\frac{1}{\epsilon})^d e^{-(t-\epsilon L)^2 m}$$

Solving for m, we get

$$m > \frac{dlog\frac{1}{\epsilon} + log\frac{1}{\delta}}{(t - \epsilon L)^2}$$

Which you can solve for some optimum ϵ (the radius of the balls in this case, we abused notation a little bit here). The optimal radius will be in terms of L. We find again that the number of samples required to have low error w.h.p is again logarithmic in the number of functions in consideration, except this time the "number of functions" depends on the Lipschitz constant (the higher the Lipschitz constant, the "more" functions we have) and the dimension of the space.

Most complexity measures derive themselves from an idea like the one above, of discretizing the function space (for example, VC dimension, Fat-shattering dimension, covering numbers, etc) What distinguishes these measures from each other is (1): how useful they are and (2): how easy they are to use. However, next we will introduce one of the most widely used measures of complexity in the next section: Rademacher Complexity, which doesn't use such a discretization idea.

7.2.2 Rademacher Complexity OR "ability to correlate with random labels"

Now we turn to Rademacher complexity. Sanjeev cautions that often this topic confuses students, or falsely impresses them. Possibly because standard accounts use the wrong definition and don't clarify that the basic point is rather trivial.

We first formalize the idea of the "complexity" of a hypothesis class. To this this we use the notion of Rademacher complexity inspired by our intuition of classifying random labels. First, let

$$\begin{aligned}
&\mathcal{H} : \text{Hypothesis class} \\
&S : 2m \text{ i.i.d samples from D} \\
&\sigma_i = \begin{cases} 1 & i \in \{1, ..m\} \\ -1 & i \in \{m+1, ..2m\} \end{cases}
\end{aligned}$$

Now

$$\mathcal{R}_{m,D}(\mathcal{H}) = \mathrm{E}_{S \sim D^m} \left[\frac{1}{2m} \sup_{h \in \mathcal{H}} \left| \sum \sigma_i h(z_i) \right| \right]$$

We call $\mathcal{R}_{m,D}(H)$ the *Rademacher Complexity* of H on a distribution D.

Note that flipping the sign in front of the loss function turns high loss into low and vice versa, so it is effectively like flipping the label of the underlying datapoint. Thus effectively we are flipping the labels of half the datapoints randomly and retaining the labels of the other half. The definition requires finding classifier h in the class that correlates well with this random relabeling; this is the usual interpretation of Rademacher complexity.

(Sanjeev's definition is different from the one used in literature where σ_i is picked randomly for each i, but you can convince yourself that picking exactly half -1s and half +1s isn't too different.)

Claim 7.1: For a given loss function,$\forall \delta > 0$, with probability $> 1 - \delta$, we have that the generalization error of all hypothesis $h \in \mathcal{H}$, on a sample S of m i.i.d. samples drawn from a distribution D, is

$$\Delta_S(h) \leq 2\mathcal{R}_{m,D}(\mathcal{H}) \quad \left(+ O\left(\frac{1}{m} ln\left(\frac{1}{\delta} \right) \right) \right)$$

The main takeaway of this claim is that generalization error can be upper bounded by the Rademacher complexity.

(The part in the square brackets comes from concentration bounds from the sample S being "representative" of the distribution and having the generalization bound hold for all $h \in \mathcal{H}$)

Suppose for a random sample S the generalization error is high.consider the following thought experiment

- Split S_m into sets S_1 and S_2 randomly, with the sets being of equal size.
- For a given h (**picked independently of S**), consider $L_{S_1}(h)$ and $L_{S_2}(h)$
- For large enough m, we have that $L_{S_2}(h) \approx L_D(h)$ and thus $L_D(h) - L_{S_1}(h) \approx L_{S_2}(h) - L_{S_1}(h)$ Here S_2 is like the "test set" and S_1 is like the "training set". Thus,

$$\Delta_S(h) \approx L_{S_1}(h) - L_{S_2}(h)$$

- But since S_1 and S_2 are randomly picked, we can just consider S_1 as the first half of the sample S and then the difference reduces to

$$\mathrm{E}_{S\sim D^m}\left[\mathrm{E}_{z\sim S_2}[L(h,z)] - \mathrm{E}_{z\sim S_1}[L(h,z)]\right] \leq \mathrm{E}_{S\sim D^m}\left[\frac{1}{m}\left|\sum \sigma_i h(z_i)\right|\right] \leq \sup_{h\in\mathcal{H}} \mathrm{E}_{S\sim D^m}\left[\frac{1}{m}\left|\sum \sigma_i h(z_i)\right|\right]$$

- Thus we have

$$\Delta_S(h) \leq \sup_{h\in\mathcal{H}} \mathrm{E}_{S\sim D^m}\left[\frac{1}{m}\left|\sum \sigma_i h(z_i)\right|\right] \leq \mathrm{E}_{S\sim D^m}\left[\sup_{h\in\mathcal{H}} \frac{1}{m}\left|\sum \sigma_i h(z_i)\right|\right] = 2\mathcal{R}_{m,D}(\mathcal{H})$$

(The 2 in the end is simply because we defined Rademacher Complexity with a set of size 2m. We also leave out the concentration term that arrives due to the approximation of the generalization error using a training and test set. For a more formal treatment of this topic refer to the chapter in Understanding Machine Learning: From Theory to Algorithms, Shalev-Shwartz, Shai and Ben-David, Shai)

7.2.2.1 Rademacher Complexity as practitioner intuition

Rademacher complexity seems a little mysterious at first, but if we think about it a little bit, it's essentially just a formalization of a practitioner's intuition of what "overfitting" means. When does a practitioner believe overfitting has happened? It happens when the learner gets great training error but also has bad testing error. Say we have a sample S_1, which we trained on, and S_2, which we held out, and say we have good training error and bad testing error. Then

$$\mathrm{E}_{S\sim D^m}\left[\mathrm{E}_{z\sim S_2}[L(h,z)] - \mathrm{E}_{z\sim S_1}[L(h,z)]\right]$$

Will be large, since the second term is small and the first term is large. This leads to a higher Rademacher Complexity. So we can see Rademacher Complexity, although initially seemingly arcane, is a concept we all basically understand.