COS-511: Learning Theory **Spring 2017**

Lecture 5: Proof of Fundemental Theorem & Deep Networks (take 1)

Lecturer: Roi Livni

In previous lecture we defined the notion of VC-dimension. We stated the fundemental theorem which roughly says that the following are equivalent Learnability = finite VC dimension= Uniform Convergence = learnable through ERM. In this lecture we will prove that learnability implies finite VC dimension and that finite VC dimension implies uniform convergence.

5.1 Proof that finite VC dimension implies uniform convergence (Agnostic)

We will make use of the following two events:

1. Let A be the event that given a sample $S \sim \mathcal{D}^m$, there exists some $h \in \mathcal{H}$ such that the error on the sample $\text{err}_S(h)$ is 0 and the generalization error $\text{err}(h)$ is greater than some $\epsilon > 0$. In other words:

$$\mathbb{P}[A] \triangleq \mathbb{P}[\exists h \in \mathcal{H} \text{ s.t. } |\text{err}_S(h) - \text{err}(h)| > \epsilon \mid S \sim \mathcal{D}^m]$$

   We will prove that $\mathbb{P}[A] = O\left(\tau_{\mathcal{H}}(m) e^{-c \cdot m \epsilon^2}\right)$. for some constatn $c \geq 0$.

2. Let B be the event that given two samples, $S, S' \sim \mathcal{D}^m$, there exists some $h \in \mathcal{H}$ such that the error on the first sample, S is smaller the $\epsilon/2$ over the error of S' $\epsilon > 0$. In other words:

$$\mathbb{P}[B] \triangleq \mathbb{P}\left[\exists h \in \mathcal{H} \text{ s.t. } |\text{err}_S(h) - \text{err}_{S'}(h)| > \frac{\epsilon}{2} \mid S, S' \sim \mathcal{D}^m\right]$$

Claim 5.1. $\mathbb{P}[A] \leq 2\mathbb{P}[B]$.

Proof: By the law of total probability, we can write

$$\begin{aligned}\mathbb{P}[B] &= \mathbb{P}[B|A]\mathbb{P}[A] + \mathbb{P}[B|\neg A]\mathbb{P}[\neg A] \\ &\geq \mathbb{P}[B|A]\mathbb{P}[A]\end{aligned}$$

To prove the claim, it is sufficient to show $\mathbb{P}[B|A] \geq 1/2$. Let $S' = \{x_1, \ldots, x_m\} \sim \mathcal{D}^m$. Using the hypothesis $h \in \mathcal{H}$ defined for event A, we know that $|\text{err}(h) - \text{err}_S(h)| > \epsilon$ by definition. Then, with

$$z_i \triangleq \begin{cases} 1 & \ell_{0,1}(h(x_i), y_i) = 1 \\ 0 & o/w \end{cases}$$

and

$$\begin{aligned} Y &\triangleq \frac{1}{m}\sum_{i=1}^{m} z_i = \text{err}_{S'}(h) \\ \mathbb{E}Y &= \mathbb{E}\left[\frac{1}{m}\sum_{i=1}^{m} z_i\right] = \mathbb{E}[\text{err}_{S'}(h)] && \text{by definition of sample error} \\ &= \text{err}(h) > \text{err}_S(h) - \epsilon \end{aligned}$$

we see that $1 - \mathbb{P}[B|A] \leq \mathbb{P}[|Y - \mathbb{E}Y| > \frac{\epsilon}{2}]$ because in order for $|Y - \mathbb{E}Y| > \frac{\epsilon}{2}$, Y must deviate from its mean, the generalization error, by more than $\epsilon/2$. Denote $e = \text{err}(h)$ then $m \cdot Y$ is a binomial random variable with expectation e, hence variance $e(1-e)m < em$, and Y has variance smaller then $\frac{e}{m}$. Applying Chebysheff's inequality we observe that:

$$\mathbb{P}\left[|Y - \mathbb{E}Y| > \frac{\epsilon}{2}\right] \leq \frac{4e}{m \cdot \epsilon^2} \ll \frac{1}{2}$$

for any m we might choose (note that $m \sim \mathcal{O}(1/\epsilon^2)$), thus completing the proof of Thm. 5.1. □

Continuing our proof that $\mathbb{P}[A] = O\left(\tau_{\mathcal{H}}(m)e^{-c\cdot m\epsilon^2}\right)$, we now seek to show $\mathbb{P}[B] \leq \tau_{\mathcal{H}}(2m)e^{-c\cdot m\epsilon^2}$. We employ the following symmetry argument that states that the probability of two samples S and S' being so different that some hypothesis performs much better on one versus the other is small.

Construct two new sets T and T' by randomly partitioning $S \cup S'$ into equal sets (note, this proof still works if $S \cap S' \neq \emptyset$, but need to use multisets; in practice, however, S and S' are nearly always disjoint because the domain is very large). Now, define a distribution $\mathcal{T}$ over choices of T and T' and B_T as the event B, but with T and T' instead of S and S':

$$\mathbb{P}[B_T] \triangleq \mathbb{P}\left[\exists h \in \mathcal{H} \text{ s.t. } |\text{err}_T(h) - \text{err}_{T'}(h)| > \frac{\epsilon}{2} \mid T, T' \sim \mathcal{T}\right]$$

We claim that $\mathbb{P}_{S,S'}[B] = \mathop{\mathbb{E}}_{S,S'}\left[\mathop{\mathbb{P}}_{T,T'}[B_T|S,S']\right]$. We take this detour because it is much easier to analyze $\mathop{\mathbb{P}}_{T,T'}[B_T|S,S']$ (i.e., what is the probability that one set has all errors and the other has none).

$$\begin{aligned} \mathop{\mathbb{P}}_{T,T'}[B_T|S,S'] &= \mathop{\mathbb{P}}_{T,T'}\left[\exists h \in \mathcal{H} \text{ s.t. } |\text{err}_T(h) - \text{err}_{T'}(h)| > \frac{\epsilon}{2} \mid S, S' \sim \mathcal{D}^m\right] \\ &\leq |\mathcal{H}_{S\cup S'}| \max_h \mathop{\mathbb{P}}_{T,T'}\left[|\text{err}_T(h) - \text{err}_{T'}(h)| > \frac{\epsilon}{2}\right] && \text{by union bound} \\ &\leq \tau_{\mathcal{H}}(2m) \max_h \mathop{\mathbb{P}}_{T,T'}\left[|\text{err}_T(h) - \text{err}_{T'}(h)| > \frac{\epsilon}{2}\right] \\ &\leq \tau_{\mathcal{H}}(2m) \max_h \mathop{\mathbb{P}}_{T,T'}\left[|\text{err}_T(h) - \text{err}(h)| + |\text{err}_{T'}(h) - \text{err}(h)| > \frac{\epsilon}{2}\right] && \text{Traingle Inequality} \\ &\leq 2\tau_{\mathcal{H}}(2m) \max_h \mathop{\mathbb{P}}_{T}\left[|\text{err}_T(h) - \text{err}(h)| > \frac{\epsilon}{4}\right] \end{aligned}$$

Finally we apply Hoeffding's inequality to achieve that: $\mathbb{P}(B_T|S,S') \leq 4\tau_{\mathcal{H}}(2m)e^{-8m\epsilon^2}$. Taken together we've shown that for some $c \geq 0$:

$$\mathbb{P}(A) \leq 4 \cdot \tau_{\mathcal{H}}(2m)e^{-c\cdot m\epsilon^2}$$

Thus if we take m, such that $m \gg \frac{8}{\epsilon^2} \log \frac{4\tau_H(m)}{\delta}$ then we achieve that $\mathbb{P}(A) < \delta$.

VC-dim$(\mathcal{H}) = d$ by Sauer's Lemma (Lemma 4.2) we have that $\tau_{\mathcal{H}}(m) = m^d$ hence $m = O(\frac{d \log 1/\delta}{\epsilon^2})$ suffices.

5.2 The class of deep networks – The Free Lunch

We next give our first analysis to the class of hypotheses named "Neural Networks". There are many different architectures to consider when we consider neural networks and certain parameters need to be described: These define the *architecture*.

Roughly, a neural network hypothesis class is described by an acyclic directed graph (V, E). that can be decomposed to *layers*. Namely $V_{t=1}^{d} = \cup_{t=1} V^{(t)}$. The vertices at each layer are referred to as *neurons* or *nodes*. A node at layer t is connected only towards neurons at layer $t+1$. We will call such a graph a d-layer feed forward graph.

Definition 5.2 (d-depth feed–forward graph)**.** *A directed graph (V, E) is said to be a d-depth feed froward graph (dFFD) if the set of vertices can be decomposed into disjoint sets $V = \cup_{t=1} V^{(t)}$ such that $(v_i, v_j) \in E$ if and only if $v_i \in V^{(i)}$ and $v_j \in V^{(j)}$ and $j - i = 1$.*

We will assume for simplicity that $|V^{(d)}| = 1$ (i.e. there is only one neuron). The layer $V^{(0)}$ is called the input layer, the neurons at layer $V^{(d)}$ are call the output layer (or output neurons). All other neurons are called hidden units *or hidden layers.*

An architecture of neural network is defined by a d-depth feed forward graph and an activation function σ. For now we concentrate on $\sigma(x) = \text{sgn}(x)$.

We next define the hypothesis class of neural network (for some fixed architecture) which we will denote by $\mathcal{N}_{(V,E),\sigma}$. Each target function can be consider as a weighted graph with (V, E) vertices and edges. Namely a hypothesis $f_{\omega,b} \in \mathcal{N}_{(V,E),\sigma}$ is parametrized by a weight function over the edges $\omega : E \to \mathbb{R}$ and bias term for each node $b : V \to \mathbb{R}$. To describe the output of the network, we next describe the output of each neuron:

The bottom layer, $V^{(0)}$ is the called the input layer. It contains n neurons where n is the dimensionality of the domain χ. Given $\mathbf{x} \in \chi$ For every neuron in the input layer $v_i^{(0)}$ we define its output to be simply x_i.

The output of neurons $v_i^{(t)}(\mathbf{x})$ is then defined by induction, given the output of neurons $v_1^{(t-1)}, \ldots v_m^{(t-1)}$ and the weight/bias functions, the output of neuron $v_i^{(t)}$ is defined by

$$v_j^{(t)}(\mathbf{x}) = \sigma_{\text{sgn}} \left(\sum_{i=1}^{m} \omega_{j,i}^{(t)} v_i^{(t-1)}(\mathbf{x}) + b_j^{(t)} \right)$$

Where $\omega_{j,i}^{(t)} = \omega(v_i^{(t-1)}, v_j^{(t)})$ and $b_i^{(t)} = b(v_i^{(t)})$ are defined by the weight function ω and bias b. Finally the output of the network $f_{\mathbf{w},b}$ is given by the output of the final neuron $v^{(d)}$ in the output layer (here we assume there's only one neuron).

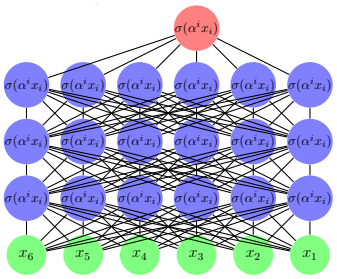

Figure 5.1: A fully connected neural network with 3 hidden layer and an output neuron.