18.218 Topics in Combinatorics Spring 2021 – Lecture 3

Dor Minzer

In this lecture, we will present a learning algorithm for the class of Boolean functions that are close to sparse functions.

1 What is PAC learning?

In the PAC learning model (probably approximately correct), one is required to learn a function g from a certain concept class $\mathcal{C} \subseteq \{f\colon \{-1,1\}^n \to \{-1,1\}\}$. The input of the learner is a sequence of random input-output tuples, $(x_i, f(x_i))$ for $i = 1, \ldots, q$, and the learner is supposed to product a hypothesis function $h\colon \{-1,1\}^n \to \{-1,1\}$ such that h is likely to be close to g, say $\|f - g\|_2 \leqslant \varepsilon$ with probability $\geqslant 1 - \delta$ (the probability here is over the randomness of the samples and the learner). The learner is called proper if $h \in \mathcal{C}$ (i.e. the hypothesis itself is from the concept class), and otherwise it is called improper.

The advantage of PAC learning is that it is something that is potentially applicable in practice. In real life we often face the problem of getting random samples (say, occurrences of a virus) without the ability to produce something ourselves, and we wish to extract information using these samples. In theory, however, PAC learning model turns out to be fairly weak and does not allow for efficient learning algorithm for many problems. One thing it does allow to do is estimate Fourier coefficients.

For that, we introduce the basic and powerful concentration inequality known as Chernoff-Hoeffding bound.

Fact 1.1. *Suppose $Y_1, \ldots, Y_n$ are independent random variables such that $|Y_i| \leqslant 1$ almost surely. Then for every $\varepsilon > 0$,*

$$\Pr\left[\sum_{i=1}^{n} Y_i - \sum_{i=1}^{n} \mathbb{E}\left[Y_i\right] \ \geqslant \varepsilon n\right] \leqslant 2e^{-\frac{\varepsilon^2}{2+\varepsilon}n}.$$

Claim 1.2. *For all $\varepsilon, \delta > 0$, there is $q = O\left(\frac{\log(1/\delta)}{\varepsilon^2}\right)$ and an algorithm performing the following task. Given a sequence of random input-output pairs of $f\colon \{-1,1\}^n \to [-1,1]$ and a character $S \subseteq [n]$, and algorithm produces an estimate a_S of $\widehat{f}(S)$ such that*

$$\Pr\left[\ \widehat{f}(S) - a_S \ \geqslant \varepsilon\right] \leqslant \delta.$$

Proof. By definition, $\widehat{f}(S) = \mathbb{E}_x\left[f(x)\chi_S(x)\right]$. Thus, given the sequence $(x_i, f(x_i))_{i=1,\ldots,q}$ of input-output pairs, our estimator would be $a_S = \frac{1}{q}\sum_i Y_i = \frac{1}{q}\sum_{i=1}^{q} f(x_i)\chi_S(x_i)$. The Chernoff bound applied on Y_i (which are independent, bounded and have mean $\widehat{f}(S)$ gives the claim. □

Thus, for example, one may come up with a PAC learning algorithm for functions that are concentrated on degrees up to k, which has $\mathrm{poly}(n^k, 1/\varepsilon, \log(1/\delta))$ queries (check that!). This running time is not very impressive however, so one is often led to consider stronger learning algorithms.

2 Learning using membership queries

The membership query model is a vast strengthening of the PAC learning model (and in so is less "realistic"). Here, one again wishes to learn a function f from a concept class $\mathcal{C}$. The difference is that the learner is allowed to choose any point $x \in \{-1,1\}^n$, and upon doing so it gets the value $f(x)$.

As before, there are proper and improper learners, and the main complexity measures of a learner are again the precision parameter ε, the confidence parameter δ and the query complexity as well as running time of the algorithm (which often times are the same) and are denoted by q and t respectively.

It turns out that using membership queries, one may approximate significantly more complex expressions involving Fourier coefficients. An important example is given by the following claim.

Claim 2.1. *For all $\varepsilon, \delta > 0$, there is $q = O\left(\frac{\log(1/\delta)}{\varepsilon^2}\right)$ and an algorithm performing the following task. Given membership queries to $f\colon \{-1,1\}^n \to [-1,1]$, and sets $T \subseteq J \subseteq [n]$, the algorithm outputs a number $b_{T,J}$ such that*

$$\Pr\left[b_{T,J} - \sum_{S : S \cap J = T} \widehat{f}(S)^2 \geqslant \varepsilon \right] \leqslant \delta.$$

Proof. We recall from the last lecture that

$$\begin{aligned}\sum_{S : S \cap J = T} \widehat{f}(S)^2 = \mathop{\mathbb{E}}_{z \in \{-1,1\}^{\bar{J}}}\left[\widehat{f_{\bar{J} \to z}}(T)^2\right] &= \mathop{\mathbb{E}}_{z \in \{-1,1\}^{\bar{J}}}\left[\mathop{\mathbb{E}}_{x \in \{-1,1\}^J}[f(z,x)\chi_T(x)]^2\right] \\ &= \mathop{\mathbb{E}}_{\substack{z \in \{-1,1\}^{\bar{J}} \\ x,y \in \{-1,1\}^J}}[f(z,x)\chi_T(x)f(z,y)\chi_T(y)].\end{aligned}$$

Thus, we now have an algorithm: we sample $z^i \in \{-1,1\}^{\bar{J}}$, $x^i, y^i \in \{-1,1\}^J$ for $i = 1, \ldots, q$ independently, calculate $A_i = f(z^i, x^i)\chi_T(x^i)f(z^i, y^i)\chi_T(y^i)$, and output $\frac{1}{q}\sum_{i=1}^{q} A_i$. The result now follows from Chernoff's bound. □

3 Learning sparse functions using membership queries

Recall that $g\colon \{-1,1\}^n \to \mathbb{R}$ is said to be t-sparse if its Fourier spectrum is supported on at most t characters, and $f\colon \{-1,1\}^n \to \{-1,1\}$ is said to be (t,ε) sparse if there is a t-sparse function g such that $\|f - g\|_2^2 \leqslant \varepsilon$. Our main goal is to prove the following result.

Theorem 3.1. *For all $t, \varepsilon, \delta > 0$ there exists an algorithm whose runtime is $\mathsf{poly}(n, t, 1/\varepsilon, 1/\delta)$ such that the following holds. Given an oracle access to a (t,ε)-sparse function $f\colon \{-1,1\}^n \to \{-1,1\}$, the algorithm produces an hypothesis function $H\colon \{-1,1\}^n \to \{-1,1\}$ such that $\|f - H\|_2^2 \leqslant 4\varepsilon + \delta$.*

Proof. The proof has several steps.

Heavy coefficients are what matters. Let $\xi = \frac{\delta}{8t}$, and let g be the t-sparse function close to f. Let $\mathcal{S}$ be the set of characters such that $\widehat{f}(S) \geqslant \sqrt{\xi}$. Then

$$\sum_{S \notin \mathcal{S}} \widehat{f}(S)^2 = \sum_{S \notin \mathcal{S}} (\widehat{f}(S) - \widehat{g}(S))^2 1_{\widehat{g}(S)=0} + \widehat{f}(S)^2 1_{\widehat{g}(S)\neq 0} \leqslant \|f - g\|_2^2 + t\xi^2 \leqslant \varepsilon + \frac{\delta}{8},$$

so all but $\varepsilon + \frac{\delta}{8}$ of the mass of f lies in $\mathcal{S}$. Our goal will be to find a set L containing $\mathcal{S}$, and then approximate $\widehat{f}(S)$ for each $S \in L$ within a small error, say by a number a_S. Once we do that our hypothesis function will be $h(x) = \sum_S a_S \chi_S(x)$, and if we want a Boolean function we will take $H(x) = \mathsf{sign}(h(x))$.

Note that $|\mathcal{S}| \leqslant 1/\xi^2$; our set L may be slightly larger, but its size will be of the same order of magnitude

Locating the heavy coefficients. Our algorithm will work in steps. At each step k, we will keep a set of live subsets of $[k]$, which are subsets $A \subseteq [k]$ such that we suspect

$$\sum_{S : S \cap [k] = A} \widehat{f}(S)^2$$

to be larger than ξ^2. Note that:

- at each step, we expect there to be at most $O(1/\xi^2)$ such subsets A, since all of these sums together sum up to $\sum_S \widehat{f}(S)^2 = \|f\|_2^2 = 1$;
- if there is a coefficient S in $\mathcal{S}$ such that $S \cap [k] = A$, then we expect A to be alive.

Thus, we design the following algorithm. It will use an estimator to the above sum as a black-box. Throughout the algorithm, we maintain k, starting with $k = 0$ as well as a list L of live subsets of $[k]$, starting with $L = \{\emptyset\}$

1. For each $A \in L$, using the algorithm from Claim 2.1:
   (a) Estimate $\sum_{S : S \cap [k+1] = A} \widehat{f}(S)^2$ within precision $\xi/10$ and certainty $\frac{\delta\xi^2}{n}$, and if it is larger than $\xi/2$, add A to L'.
   (b) Estimate $\sum_{S : S \cap [k+1] = A \cup \{k+1\}} \widehat{f}(S)^2$ within precision $\xi/2$ and certainty $\frac{\delta\xi^2}{n}$, and i it is larger than $\xi/10$, add $A \cup \{k+1\}$ to L'.
2. Set $L \leftarrow L'$; if $k = n$, halt, otherwise $k \leftarrow k + 1$ and go to step 2.

We first argue that with probability $1 - o(1)$, this algorithm terminates with $L \supseteq \mathcal{S}$ such that $|L| = O(1/\xi^2)$. Indeed, by induction on k starting with $k = 0$, for each $S \in \mathcal{S}$ the probability that $S \cap [k] \notin L$ is at most $\frac{\delta\xi^2}{n}$ (since the sum corresponding to $A = S \cap [k]$ is at least $\widehat{f}(S)^2 \geqslant \xi^2$, and the parameters of the approximator). Thus, by the union bound the probability that $S \cap [k]$ will not be in L for some k is at most $\delta\xi^2$, by a union bound over $\mathcal{S}$ the probability that there is $S \in \mathcal{S}$ and k such that $S \cap [k] \notin L$ is at most δ. Thus, with probability $1 - \delta$ we have $\mathcal{S} \subseteq L$.

By similar arguments, with probability $1 - O(\delta)$ we have $|L| \leqslant O(1/\xi^2)$ at each point of the algorithm, so the running time is at most $O(n/\xi^2)$ times the running time of the approximator, hence $\mathsf{poly}(n, t, 1/\varepsilon, 1/\delta)$ in total.

Finishing the proof. Assume that the algorithm terminated with $L \supseteq \mathcal{S}$ with size $R \leqslant O(1/\xi^2)$. By the algorithm from Claim 1.2, we may estimate $\widehat{f}(S)$ for each $S \in L$ with in precision $\frac{\delta}{8R}$ and probability of error at most δ/R. Thus, by the union bound we get numbers $(a_S)_{S \in L}$ such that $a_S - \widehat{f}(S) \leqslant \sqrt{\frac{\delta}{8R}}$ for

all $S \in L$ with probability at least $1 - \delta$. Define the function $h(x) = \sum_{S \subseteq [n]} a_S \chi_S(x)$, and then $H(x) = \mathsf{sign}(h(x))$. Then

$$\|f - H\|_2^2 \leqslant 4\|f - h\|_2^2 = 4\left(\sum_{S \in L} (\widehat{f}(S) - \widehat{h}(S))^2 + \sum_{S \notin L} \widehat{f}(S)^2\right) \leqslant 4\,|L|\,\frac{\delta}{8R} + 4\sum_{S \notin \mathcal{S}} \widehat{f}(S)^2$$
$$\leqslant \frac{\delta}{2} + 4\left(\varepsilon + \frac{\delta}{8}\right) = 4\varepsilon + \delta.$$

The first inequality holds since $|f(x) - H(x)| \leqslant 2\,|f(x) - h(x)|$ for all x, since if $f(x) \neq H(x)$, then this difference is 2 in absolute value and $f(x)$, $h(x)$ have different signs so the second different is at least 1 in absolute value. □

Remark 3.2. *This algorithm has its origin from the field of cryptography, where it is known as the Goldreich-Levin hardcore bit. This algorithm has several interesting extensions to other settings.*

18.218 Topics in Combinatorics: Analysis of Boolean Functions

Spring 2021