CS 6817: Special Topics in Complexity Theory **Spring 2025**

Lecture 10: PAC learning from Fourier concentration

Lecturer: Eshan Chattopadhyay *Scribe: Kevin Hua*

1 Low degree algorithm

Recall that there are two models of *PAC learning* that we are considering in this class.

And those two are $\begin{cases} \text{Random Example Model: we are given } \{(x_i, f(x_i))\}_{i=1}^t, x \sim \{-1,1\}^n \\ \text{Query Model: we are given an oracle that answers any query of } f \end{cases}$

Our goal is to learn any $f \in \mathcal{F}$, where $\mathcal{F}$ is a *concept class.*

In other words, in any of these two models, we want to design a learning algorithm $\mathcal{A}$ that will output $h : \{-1,1\}^n \to \{-1,1\}$, such that, with high probability, $\text{dist}(h, f) = \Pr[h \neq f] \leq \epsilon$.

Theorem 1.1. *Suppose for any* $f \in \mathcal{F}$, $\underbrace{\boldsymbol{W}^{\geq k}[f]}_{= \sum_{|s|\geq k} \hat{f}(s)^2} \leq \underbrace{\eta}_{\uparrow \text{ assume } \eta \leq \epsilon/2}$. *Then,* $\mathcal{F}$ *is PAC-learnable with* $poly(n^k, 1/\epsilon)$ *samples.*

The first step of proving this theorem is to approximate the low-degree fourier coefficients. To do this, we need the subroutine below.

Subroutine: Estimate Fourier coefficients using $\underset{f,\epsilon_1,\delta_1}{FOURIER(s)}$

Algorithm 1: $FOURIER(S)_{f,\epsilon_1,\delta_1}$

1 Output $\overset{\text{as defined in last lecture}\downarrow}{\widetilde{\hat{f}(S)}} \in \mathbb{R}$, such that, with probability at least $1 - \delta_1$, $|\widetilde{\hat{f}}(S) - \hat{f}(S)| < \epsilon_1$.

Note that we have independent samples $\{(x_i, f(x_i))\}_{i=1}^t$. And we supply $FOURIER(s)_{f,\epsilon_1,\delta_1}$ with these samples to generate $\widetilde{\hat{f}(S)}$.

With a handy algorithm estimating any Fourier coefficients, our final algorithm is as follows. The intuition is simple: we don't have the leisure to estimate every Fourier coefficient as there are exponentially many of them. However, given we know the high-degree Fourier coefficients are tiny, we might well get by without estimating them.

Algorithm 2: $\mathcal{A}$[low degree algorithm]

1 Estimate $\hat{f}(S)$, $\forall S \subseteq [n]$, $|S| \leq k$ using $\underset{f,\epsilon_1,\delta_1}{FOURIER(S)}$

2 output $\text{sign}(h(x))$, where $h(x) = \sum_{S \subseteq [n], |S| \leq k} \widetilde{\hat{f}(S)} \chi_S(x)$.

Analysis:

Assume all estimates in step 1 are "good" $--|\widetilde{\hat{f}(S)} - \hat{f}(S)| \leq \epsilon_1, \forall S \subseteq [n]$

Note that by Union Bound, $\Pr[\exists \text{a bad estimate}] \leq \delta_1 \binom{n}{\leq k}$. At the end, we are going to pick δ_1 so that this probability is some constant strictly less than 1.

Goal: we want to show $\text{dist}(\text{sign}(h), f) = \Pr[\text{sign}(h(x)) \neq f(x)]$ is small.

Let $g(x) = f(x) - h(x)$. Note that when $\text{sign}(h(x)) \neq f(x)$, $|g(x)| = |f(x) - h(x)| > 1$

Thus, we have that $\text{dist}(\text{sign}(h), f) = \Pr[\text{sign}(h(x)) \neq f(x)] \leq ||g||_2 = \mathbf{E}[(f-h)^2]$.

By Parseval's Theorem,

$$\begin{aligned} ||g||_2^2 &= \sum_{s \subseteq [n]} (\hat{g}(S))^2 \\ &= \sum_{s:|S| \leq k} \hat{g}(S)^2 + \mathbf{W}^{\geq k+1}[g] \end{aligned}$$

As $g(x) = f(x) - h(x)$, $\hat{g}(S) = \hat{f}(S) - \hat{h}(S)$.

For $|S| \leq k$, by definition,

$$\begin{aligned} |\hat{g}(S)| &= |\hat{f}(S) - \hat{h}(S)| \\ &= |\hat{f}(S) - \widetilde{\hat{f}(S)}| \\ &\leq \epsilon_1 \quad \text{(by assumption that the estimate } \widetilde{\hat{f}(S)} \text{ is "good")} \end{aligned}$$

For $\mathbf{W}^{\geq k+1}[g]$, as $h(S) = 0$ for $|S| \geq k+1$, we have $\mathbf{W}^{k+1}[g] = \mathbf{W}^{k+1}[f] \leq \eta \leq \epsilon/2$.

Summing up the upper bounds for $\sum_{|S| \leq k} \hat{g}(S)^2$ and the one for $\mathbf{W}^{k+1}[g]$, we get

$$\begin{aligned} \text{dist}(\text{sign}(h), f) \leq ||g||_2^2 &\leq \binom{n}{\leq k} \epsilon_1^2 + \epsilon/2 \\ &\leq \binom{n}{\leq k} \epsilon_1 + \epsilon/2 \quad (\text{for } \epsilon_1 \leq 1) \text{ with probability } \geq 1 - \delta_1 \binom{n}{\leq k}. \end{aligned}$$

Now, setting $\epsilon_1 = \frac{\epsilon}{2\binom{n}{\leq k}} \sim O(\frac{\epsilon}{n^k})$, $\delta_1 = O(\frac{1}{n^k})$, we get that with some positive constant probability, $\mathcal{A}$ outputs a h such that $\text{dist}(h, f) \leq \epsilon$. □

Note that sampling complexity of FOURIER is $O(\frac{kn^{2k}}{\epsilon^2} \log(n))$ from last time.

As we repeat FOURIER $O(\binom{n}{\leq k})$ times in $\mathcal{A}$, we get the sampling complexity of $\mathcal{A}$ is $O(\frac{k \cdot n^{3k} \log n}{\epsilon^2})$.

2 Kushilevitz-Mansour Algorithm

Theorem 2.1. *Suppose $\mathcal{F}$ is η-concentrated as follows: for any $f \in \mathcal{F}$, $\exists L_f = \{s_1, \cdots, s_M\}$, s.t. $\sum_{s \in L_f} \hat{f}(s)^2 \geq 1 - \eta$. Then, $\mathcal{F}$ is ϵ-PAC learnable in query model with sample complexity* $\text{poly}(n, \frac{1}{\epsilon}, M)$*, with $\eta \leq \frac{\epsilon}{2}$.*

First, suppose L_f is given to $\mathcal{A}$.

Then we can estimate $\hat{f}(S), \forall S \in L_f$ by outputting $\text{sign}(\sum_{s \in L_f} \widetilde{\hat{f}(s)} \chi_s(x))$.

Repeating the same analysis as for the low-degree algorithm, we can easily see that with $\text{poly}(n, \frac{1}{\epsilon}, M)$ samples, $\text{dist}(\text{sign}(\sum_{s \in L_f} \widehat{f(s)} \chi_s(x)), f) \leq \epsilon$. Therefore, we are done if we can find L_f(efficiently). GOOD NEWS: *Goldreich-Levin* algorithm find the L_p for us!

---

3 Goldreich-Levin

Theorem 3.1. *With query access to f and parameter δ, Goldreich-Levin outputs a set $\tilde{L}_f$. With high probability, $\tilde{L}_f$ satisfies the following two constraints:*
(1) if $S \in \tilde{L}_f$, $|\hat{f}(S)| \geq \frac{\delta}{2}$.
(2) if $|\hat{f}(S)| \geq \delta$, $S \in \tilde{L}_f$.

Notation: For any $0 \leq k \leq n$, $S \subseteq [k]$

$$B_{k,S} := \{T \subseteq [n] : T \cap [k] = S\}$$

$$W(B_{k,S}) = \sum_{T \in B_{k,S}} \hat{f}(T)^2$$

Note that $|B_{k,S}| = 2^{n-k}$ and $B_{0,\emptyset} = 2^{[n]}$.

Algorithm 3: Goldreich-Levin Algorithm

Set $\underset{\substack{\uparrow \\ \text{collection of buckets}}}{\mathcal{B}} = \{B_{0,\emptyset}\}$

while $\exists B \in \mathcal{B}$ *such that B contains at least 2 sets* **do**

  Suppose $B = B_{k,S}$. Remove B from $\mathcal{B}$.

  Let $B_0 = B_{k+1,S}$, $B_1 = B_{k+1,S \cup \{i+1\}}$.

  Estimate $W(B_0)$ and $W(B_1)$ to accuracy $\frac{\delta^2}{4}$.

  Add B_i to $\mathcal{B}$ if B_i is estimated with weight $\geq \frac{\delta^2}{2}$ (We call B_i "heavy" in this case).

end

Output $\mathcal{B}$.

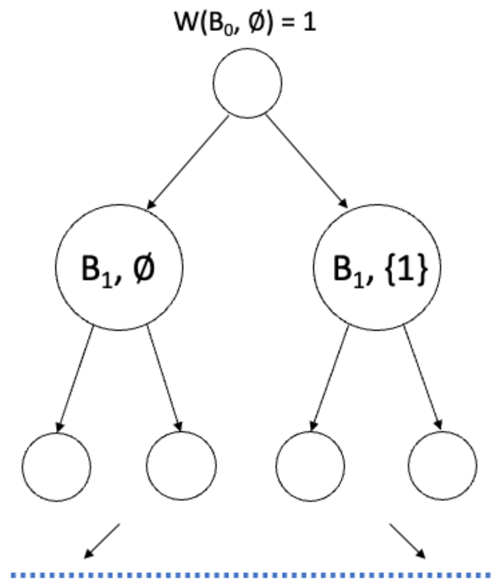

Figure 1: representation of buckets as a complete binary tree

Question: How many heavy leaves are there in the trees that can be included as part of the output for *Goldreich-Levin*, as shown in Figure 1?

3Answer: $\frac{4}{\delta^2}$. Note that the leaves are disjoint buckets. Therefore, given the accuracy of the algorithm is $\frac{\delta^2}{4}$, there are at most $\frac{4}{\delta^2}$ leaves buckets that can be included in the output.