6.895 Randomness and Computation	February 25, 2008

Lecture 6

Lecturer: Ronitt Rubinfeld *Scribe: Debmalya Panigrahi*

Today, we will describe and analyze the *Low Degree Algorithm.*

1 Preliminaries

We need a few definitions.

Definition 1 *A function $f : \{\pm 1\}^n \to \mathbb{R}$ is said to have $\alpha(\epsilon, n)$ concentration if*

$$\sum_{S \subseteq [n], |S| > \alpha(\epsilon, n)} \hat{f}(S)^2 \leq \epsilon, \ \forall\ 0 < \epsilon < 1. \tag{1}$$

In particular, for Boolean functions $f : \{\pm 1\}^n \to \{\pm 1\}$, the the above relation is equivalent to

$$\sum_{S \subseteq [n], |S| \leq \alpha(\epsilon, n)} \hat{f}(S)^2 \geq 1 - \epsilon, \ \forall\ 0 < \epsilon < 1. \tag{2}$$

This follows from Parseval's theorem, which states that

$$\sum_{S \subseteq [n]} \hat{f}(S)^2 = 1. \tag{3}$$

Intuitively, if α assumes small values, then the concentration of the function is on the low-degree terms of its Fourier decomposition. (Throughout these notes, we will use the terms *Fourier concentration* and *Fourier degree* interchangeably.)

Let us now consider some functions which satisfy this property, i.e. have low Fourier concentration. The first example we consider is the class of *junta* functions. These functions depend on only $k \ll n$ variables. Clearly, if f is such a function which does not depend on a variable i, then

$$\hat{f}(S) = 0, \ \forall\ S \subseteq [n], \ i \in S. \tag{4}$$

Since the junta function depends on only k variables, a direct consequence of the above observation is that

$$\sum_{S \subseteq [n], |S| > k} \hat{f}(S)^2 = 0. \tag{5}$$

Thus, the junta function has Fourier degree at most k for any value of ϵ.

Our next example is the following slightly unusually defined AND function.

$$\mathrm{AND}(x_1, x_2, \ldots, x_k) = \begin{cases} 1 & \text{if } \forall i,\ x_i = -1 \\ -1 & \text{otherwise.} \end{cases} \tag{6}$$

Theorem 1 *The* AND *function defined above has Fourier concentration* $\log(4/\epsilon)$*.*

Proof We consider two cases depending on the value of ϵ.
Case 1: $k \leq \log(4/\epsilon)$. In this case, from our analysis of the junta function above, we directly obtain

$$\sum_{S \subseteq [n], |S| > \log(4/\epsilon)} \hat{f}(S)^2 = 0 < \epsilon. \tag{7}$$

Case 2: $k > \log(4/\epsilon)$. Define

$$g(x) = \begin{cases} 1 & \text{if } \forall i,\ x_i = -1 \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

$$g(x) = (\frac{1-x_1}{2})(\frac{1-x_2}{2})\dots(\frac{1-x_k}{2}) \tag{9}$$

$$= \sum_{S \subseteq [k]} \frac{(-1)^{|S|}}{2^k}\chi_S. \tag{10}$$

Now,

$$\text{AND}(x) = 2g(x) - 1 \tag{11}$$

$$= (-1 + 2^{1-k}) + \sum_{S\subseteq[k],|S|>0} (-1)^{|S|}2^{1-k}\chi_S. \tag{12}$$

Clearly, there are at most 2^k non-zero Fourier co-efficients corresponding to characters S such that $|S| > 0$. Thus,

$$\sum_{S\subseteq[k],|S|>\log(4/\epsilon)} \hat{f}(S)^2 \leq \sum_{S\subseteq[k],|S|>0} \hat{f}(S)^2 \tag{13}$$

$$\leq [(-1)^{|S|}2^{1-k}]^2 2^k \tag{14}$$

$$= \frac{4}{2^k} \tag{15}$$

$$< \epsilon. \tag{16}$$

The last inequality follows from the assumption that $k > \log(4/\epsilon)$. ■

2 The Low Degree Algorithm

The low degree algorithm is a uniform distribution learning algorithm which is efficient for functions with low Fourier degree. The algorithm is presented in Figure 1. The sub-routine SINGLECOEFF is the algorithm we saw in the previous lecture for learning a single Fourier coefficient of a function. Also, as an optimization, note that the result of a single set of m queries to the function f can be re-used in all the iterations of the algorithm since the only property we need to ensure is that the samples in any single iteration are independent and uniformly distributed.

We now analyze the low-degree algorithm. We prove two lemmas which will lead us to the ultimate result.

Lemma 2 *If f has $\alpha(\epsilon, n)$ Fourier concentration, then h computed by the algorithm satisfies*

$$E_x[(f(x) - h(x))^2] \leq \epsilon + \tau, \tag{17}$$

with probability $\geq 1 - \delta$.

To prove this lemma, we need the following lemma.

Lemma 3 *With probability $\geq 1 - \delta$, for each S satisfying $|S| \leq d$,*

$$|C_S - \hat{f}(S)| \leq \gamma, \tag{18}$$

where $\gamma = \sqrt{\frac{\tau}{n^d}}$.

Input: Oracle access to function f under uniform sampling, Fourier degree of f denoted by $d = \alpha(\epsilon, n)$, accuracy parameter τ and confidence or security parameter δ.

1. Let $m = O(\frac{n^d}{\tau} \ln \frac{n^d}{\delta})$.
2. Collect m samples from the function.
3. For each set $S \subseteq [n]$ such that $|S| \leq d$, let C_S be an estimate for $\hat{f}(S)$ obtained using sub-routine SingleCoeff with m independent samples from the function f.
4. Let $h = \sum_{S \subseteq [n], |S| \leq d} C_S \chi_S$.
5. Output sign(h) as the estimate of f.

Figure 1: The low-degree algorithm

Proof Since the $O(\frac{n^d}{\tau} \log \frac{n^d}{\delta}) = O(\frac{1}{\gamma^2} \log \frac{n^d}{\delta})$ samples used to estimate any Fourier coefficient $\hat{f}(S)$ are independent, we can claim, using Chernoff bound, that

$$|C_S - \hat{f}(S)| > \gamma \tag{19}$$

with probability $\leq \frac{\delta}{n^d}$. Since there are $\leq n^d$ sets S such that $|S| \leq d$, the lemma follows using union bound. ■

We now use this lemma to prove Lemma 2.

Proof of Lemma 2 Define

$$g(x) = f(x) - h(x). \tag{20}$$

Since the Fourier decomposition is linear, for each $S \subseteq [n]$,

$$\hat{g}(S) = \hat{f}(S) - \hat{h}(S). \tag{21}$$

From the algorithm, for any S such that $|S| > d$, $\hat{h}(S) = 0$. Thus,

$$\begin{aligned} \sum_{S \subseteq [n], |S| > d} \hat{g}(S)^2 &= \sum_{S \subseteq [n], |S| > d} \hat{f}(S)^2 && (22) \\ &\leq \epsilon. && (23) \end{aligned}$$

The second inequality holds since $d = \alpha(\epsilon, n)$.

On the other hand, for each S such that $|S| \leq d$,

$$\begin{aligned} \hat{g}(S)^2 &= (\hat{f}(S) - \hat{h}(S))^2 && (24) \\ &\leq \gamma^2, && (25) \end{aligned}$$

with probability $\geq 1 - \delta$. The second inequality follows from Lemma 2. Since there are $\leq n^d$ sets S such that $|S| \leq d$,

$$\begin{aligned} \sum_{S \subseteq [n], |S| \leq d} \hat{g}(S)^2 &\leq \gamma^2 n^d && (26) \\ &= \tau. && (27) \end{aligned}$$

We now have

$$E_x[(f(x) - h(x))^2] = E[g(x)^2] \tag{28}$$

$$\begin{aligned}
&= \sum_{S\subseteq[n]} \hat{g}(S)^2 && (29)\\
&= \sum_{S\subseteq[n],|S|\le d} \hat{g}(S)^2 + \sum_{S\subseteq[n],|S|>d} \hat{g}(S)^2 && (30)\\
&\le \tau+\epsilon, && (31)
\end{aligned}$$

with probability $\geq 1-\delta$. The second equation follows from Plancherel's theorem. ■

We now prove another lemma which will be useful in the final result.

Lemma 4 *For any functions $f : \{\pm1\}^n \to \{\pm1\}$ and $h : \{\pm1\}^n \to \mathbb{R}$,*

$$Pr[f(x) \neq \mathrm{sign}(h(x))] \leq E_x[(f(x)-h(x))^2]. \tag{32}$$

Proof We have

$$E_x[(f(x)-h(x))^2] = \frac{1}{2^n}\sum_x (f(x)-h(x))^2, \tag{33}$$

while

$$Pr[f(x) \neq \mathrm{sign}(h(x))] = \frac{1}{2^n}\sum_x \mathbf{1}(x), \tag{34}$$

where $\mathbf{1}$ is defined as

$$\mathbf{1}(x) = \begin{cases} 1 & \text{if } f(x) \neq \mathrm{sign}(h(x)) \\ 0 & \text{otherwise.} \end{cases} \tag{35}$$

Now, for any x, if $f(x) \neq \mathrm{sign}(h(x))$, then

$$(f(x)-h(x))^2 \geq 1 = \mathbf{1}(x), \tag{36}$$

and if $f(x) = \mathrm{sign}(h(x))$, then

$$(f(x)-h(x))^2 \geq 0 = \mathbf{1}(x). \tag{37}$$

Thus,

$$\sum_x (f(x)-h(x))^2 \geq \sum_x \mathbf{1}(x). \tag{38}$$

■

We now arrive at our Main Theorem. However, before stating it, we need another definition.

Definition 2 *The Fourier concentration of a concept class of functions is the maximum Fourier concentration among all the functions in the class.*

Theorem 5 (Main Theorem) *Let concept class $\mathcal{C}$ have Fourier concentration $d = \alpha(\epsilon, n)$. Then, there exists a $q = O(\frac{n^d}{\epsilon}\log\frac{n^d}{\delta})$-sample uniform distribution learning algorithm for $\mathcal{C}$ (i.e. the algorithm uses q uniformly distributed independent samples and with probability $\geq 1-\delta$, outputs a hypothesis h' such that $Pr[f(x) \neq h'(x)] \leq 2\epsilon$).*

Proof The algorithm is to simply run the low-degree algorithm for $\tau = \epsilon$. The error bound follows from Lemmas 2 and 4 given above. ■

What class of functions can we learn using the low-degree algorithm? One such large class of functions that can be learned in quasi-polynomial ($O(n^{\mathrm{polylog}} n)$) time using this algorithm are those which can be represented by constant depth circuits. Recall that a circuit consists of gates (typically AND, OR and NOT gates; we assume that the first two types of gates have unbounded fan-in), constants 0 and

1 and variables $X_1, X_2, \ldots, X_n$. It turns out that any function can be represented by a constant depth circuit, but we would also like to ensure that the size of the circuit is polynomial in the number of input variables n. So, one central question is whether all functions can be computed using constant-depth, polynomial-sized circuits? The answer is no. In fact, it was proved by Furst, Saxe and Sipser that the class of parity functions cannot be computed using constant depth, polynomial sized circuits.

But what about functions which are computable using constant depth, polynomial sized circuits? The following theorem explores the Fourier concentration of such functions.

Theorem 6 (Håstad, Linial Mansour Nisan) *For any function f computable using a size-s depth-d circuit,*

$$\sum_{S \in [n], |S| > t} \hat{f}(S)^2 \leq \alpha, \tag{39}$$

for $t = O((\log \frac{2s}{\alpha})^{d-1})$.

Taking s as a polynomial in n, d as a constant and $\alpha = O(\epsilon)$ gives us $t = O(\log^d \frac{n}{\epsilon})$. Using our main theorem, we can conclude that the number of samples required for learning such a constant depth polynomial sized circuit is $n^{O(\log^d \frac{n}{\epsilon})}$, which is quasipolynomial in n. For DNF formulas, this bound can be improved to $O(n^{O(\log \log n)})$ samples.