

Lecture 9: March 22, 2007

Lecturer: Rocco Servedio

Scribe: Timothy Teräväinen

Last lecture covered the low-degree algorithm (LDA), which estimates low-degree Fourier coefficients to learn a function. Of course, we need to know somehow that the function doesn't have too much high-degree structure, in the sense of having Fourier concentration. We showed that $\text{poly}(n)$ -size, depth- d circuits have such concentration and can thus be learned in time $n^{O(\log^d(n/\epsilon))}$.

This lecture shows some limitations on learning constant-depth circuits due to the richness of this class, and introduces the notion of *noise sensitivity*, or the effect of random perturbations of input strings on the output of the function. We will see that an upper bound on noise sensitivity implies an upper bound on the Fourier concentration. This method will be used immediately to give a result for learning single halfspaces and will be used in the next lecture to easily give results for learning arbitrary boolean functions of k halfspaces.

1 Learning Constant-Depth Circuits (AC^0)

Can we improve on the run-time of the result for learning constant-depth circuits? Not very much, by the results of Kharitonov, who uses a cryptographic reduction from the hardness of factoring to obtain hardness of learning constant-depth circuits. Specifically, assuming that random n -bit Blum integers require time at least 2^{n^α} to factor on average (where $\alpha > 0$ is some fixed absolute constant) fixed, then, for all $d > d_0 \approx \alpha^{-1}$, any algorithm to learn $\text{poly}(n)$ -size, depth- d circuits requires $n^{\log(n)^{\Omega(d)}}$ time.

We note that this hardness result relies on a strong hardness assumption (the average-case lower bound is assumed to be exponential), but it is still reasonable by current standards; it would be very remarkable to beat it.

While Kharitonov's result indicates that we may not be able to learn depth- d , $\text{poly}(n)$ -size AND/OR/NOT circuits much faster than the LMN result, in fact there is a different algorithm which can learn a richer class of circuits called MAJ-OF- AC^0 in the same runtime. This is the class of circuits with the top gate computing the MAJ function of at most $\text{poly}(n)$ boolean circuits of the previous type, i.e. of size $\text{poly}(n)$ and depth- d .

The result requires more than just Fourier concentration so we do not pursue it here. (In fact, just MAJ itself does not have Fourier concentration.) The algorithm instead uses boosting methods in addition to Fourier techniques.

2 Noise Sensitivity

The remainder of the class was spent in making progress toward the following theorem:

Theorem 1 *Let $h(x) = \text{sign}(w \cdot x - \theta)$ be any halfspace. Then*

$$\sum_{S: |S| \geq \frac{441}{\epsilon^2}} \hat{f}(S)^2 \leq \epsilon.$$

This means that h has $(441/\epsilon^2)$ -Fourier concentration, which means that we can use the LDA to learn a halfspace in time n^{441/ϵ^2} . This is not a very impressive runtime, recalling that we can learn arbitrary halfspaces in the OLMB model in time roughly n^5 . However, as we will see next time, a trivial change to this proof will give essentially the same runtime for learning arbitrary boolean functions of any constant number of halfspaces. (In contrast, recall that in the OLMB model, even for learning an AND of two $\text{poly}(n)$ -weight halfspaces, the fastest algorithm known takes $n^{O(\log n)}$ time.)

The key idea here is noise sensitivity and its connection to Fourier concentration.

Definition 1 *Given a noise rate $\epsilon \in (0, \frac{1}{2})$, the noise operator N_ϵ acts on bit-strings by flipping each bit, independently of other bits, with probability ϵ .*

Definition 2 *The noise sensitivity of a function f (at a fixed length of input, n) is given by*

$$ns_\epsilon(f) = \mathbb{P}_{x \sim \text{Unif}(\{0,1\}^n)} [f(x) \neq f(N_\epsilon(x))]$$

As a first easy observation, the noise operator N_ϵ leaves x unchanged with probability $(1 - \epsilon)^n$, and consequently we have:

$$ns_\epsilon(f) \leq 1 - (1 - \epsilon)^n.$$

We will see later in the lecture that $ns_\epsilon(f) < \frac{1}{2}$ for any function f and any $0 < \epsilon < \frac{1}{2}$. We will consider a few examples:

- i. If f is such that $f(x) = x_1$, it is obvious that $ns_\epsilon(f) = \epsilon$; the function output is changed only if x_1 is changed.
- ii. Look at $f(x) = x_1 \wedge x_2 \wedge \cdots \wedge x_k$. Since the noise operator acts on a uniformly-distributed string x to generate another uniformly-distributed string, the original string x is interchangeable with the corresponding noisy string $N_\epsilon(x)$ and we have

$$ns_\epsilon(f) = 2 \mathbb{P}[\{f(x) = \text{T}\} \cap \{f(N_\epsilon(x)) = \text{F}\}] = 2\left(\frac{1}{2^k}(1 - (1 - \epsilon)^k)\right).$$

- iii. Finally, let's look at the easiest example of a "real" linear threshold function, the MAJ function.

Let $f(x) = \text{MAJ}(x) = \text{sign}(\sum_{i=1}^n x_i)$. It turns out that the noise sensitivity is $\Theta(\sqrt{\epsilon})$. Note that this is independent of n . The upper bound $O(\sqrt{\epsilon})$ holds in fact for all linear threshold functions. However, the proof is tough and unintuitive so we will just hash out some intuition for this easier case.

Consider the random walk given by the sum of n uniform independent random variables with range $\{-1, 1\}$. The expected displacement of this n -step walk is

$$\mathbb{E}[|x_1 + x_2 + \cdots + x_n|] = \sqrt{n}.$$

Now suppose we have a random walk corresponding to the original string at such a location and we "correct" the location after-the-fact to account for the effect of the noise. Note that the output of MAJ changes iff the walk crosses the origin as a result of this correction. Also note that the correction itself is an ϵ -lazy random walk with step size 2, giving an expected displacement of $2\sqrt{\epsilon n}$. This means that in the "typical" case where the initial random walk had displacement $\approx \sqrt{n}$, there is at most a $\sqrt{\epsilon}$ probability of walking from the *expected* pre-noise displacement past the origin (and flipping the label assigned by MAJ). Thus, intuitively we should have $ns_\epsilon(f) \leq O(\sqrt{\epsilon})$.

3 Fourier Concentration from Noise Sensitivity

This last example was just an intuitive argument, but it is toward a correct result:

Theorem 2 (Peres) *Every LTF $h(x) = \text{sign}(w \cdot x - \theta)$ has noise sensitivity*

$$ns_\epsilon(h) \leq 8.8\sqrt{\epsilon}.$$

Let us look at one more example, the noise sensitivity of parity.

Claim: $ns_\epsilon(\text{PAR}_k) = \frac{1}{2}(1 - (1 - 2\epsilon)^k)$

Proof: Note that the noise operator changes the output of the k -parity function iff an odd number of input bits are changed (with probability ϵ each). Thus, we have

$$ns_\epsilon(\text{PAR}_k) = \mathbb{P}[\text{there are an odd \# of flips}] = \binom{k}{1}(1 - \epsilon)^{k-1}\epsilon + \binom{k}{3}(1 - \epsilon)^{k-3}\epsilon^3 + \dots$$

We can show the claim by rewriting this using a few tricks, specifically by writing $1 = 1^k$ and $(1 - 2\epsilon)^k$ the right way:

$$1^k = ((1 - \epsilon) + \epsilon)^k = \binom{k}{0}(1 - \epsilon)^k + \binom{k}{1}(1 - \epsilon)^{k-1}\epsilon + \binom{k}{2}(1 - \epsilon)^{k-2}\epsilon^2 + \dots$$

$$(1 - 2\epsilon)^k = ((1 - \epsilon) - \epsilon)^k = \binom{k}{0}(1 - \epsilon)^k - \binom{k}{1}(1 - \epsilon)^{k-1}\epsilon + \binom{k}{2}(1 - \epsilon)^{k-2}\epsilon^2 - \dots$$

Thus, by subtracting the second expression from the first, and dividing by two:

$$\frac{1^k - (1 - 2\epsilon)^k}{2} = \binom{k}{1}(1 - \epsilon)^{k-1}\epsilon + \binom{k}{3}(1 - \epsilon)^{k-3}\epsilon^3 + \dots,$$

which is in fact equal to the probability that an odd number of flips happens under operator N_ϵ .

Note that $\frac{1}{2} - \frac{(1-2\epsilon)^k}{2}$ is not a small number, and tends to $\frac{1}{2}$ exponentially quickly in k . In fact, PAR_k has maximal noise sensitivity among boolean functions of k variables, as one can see by greedily maximizing the right-hand side of the expression in Theorem 3 below. (One may ask: What is the most noise sensitive monotone function? There is not a concise characterization, but see [BKS99, MO03].)

Now, since we have the noise sensitivity figured out for parity, we would expect to have a result for all functions, by looking at the representation with respect to the parity basis. In fact:

Theorem 3 *For any $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$, we have*

$$ns_\epsilon(f) = \frac{1}{2} - \frac{1}{2} \sum_{S \subseteq [n]} (1 - 2\epsilon)^{|S|} \hat{f}(S)^2.$$

Note briefly that if we wanted to maximize $ns_\epsilon(f)$ as a function of f , since the total Fourier energy is constrained, $\sum \hat{f}(S)^2 = 1$, we would put as much energy as possible onto the maximum-size set ($S = \{1, \dots, n\}$) since the weight $(1 - 2\epsilon)^{|S|}$ is the smallest here. Since the parity function and its negation are the only two Boolean functions that have all their Fourier weight on the coefficient $S = \{1, \dots, n\}$, this gives us that the noise sensitivity of parity is maximal over all Boolean functions.

Now we prove Theorem 3. Let x be selected at uniform random from $\{-1, 1\}^n$, and say $y = N_\epsilon(x)$. Then,

$$\begin{aligned}
ns_\epsilon(f) &= \mathbb{P}[f(x) \neq f(y)] \\
&= \frac{1}{4} \mathbb{E}[(f(x) - f(y))^2] \\
&= \frac{1}{4} \mathbb{E}[f(x)^2 + f(y)^2 - 2f(x)f(y)] \\
&= \frac{1}{2} (1 - \mathbb{E}[f(x)f(y)]) \\
&= \frac{1}{2} \left(1 - \mathbb{E} \left[\sum_S \hat{f}(S)x_S \sum_T \hat{f}(T)y_T \right] \right) \\
&= \frac{1}{2} - \frac{1}{2} \sum_{S,T} \hat{f}(S)\hat{f}(T)\mathbb{E}[x_S y_T] \tag{1}
\end{aligned}$$

First consider pairs S, T where $S \neq T$. Then wlog suppose i is an index with $i \in T$, $i \notin S$. Then:

$$\begin{aligned}
\mathbb{E}[x_S y_T] &= \frac{1}{2} \mathbb{E}[x_S y_T | y_i = 1] + \frac{1}{2} \mathbb{E}[x_S y_T | y_i = -1] \\
&= \frac{1}{2} \mathbb{E}[x_S y_{T-\{i\}}] - \frac{1}{2} \mathbb{E}[x_S y_{T-\{i\}}] \\
&= 0.
\end{aligned}$$

On the other hand, if $S = T$ we have

$$\begin{aligned}
\mathbb{E}[x_S y_T] &= \mathbb{E}[x_S y_S] \\
&= \mathbb{P}[x_S = y_S] - \mathbb{P}[x_S \neq y_S] \\
&= 1 - \mathbb{P}[x_S \neq y_S] - \mathbb{P}[x_S \neq y_S] \\
&= 1 - 2\mathbb{P}[x_S \neq y_S] \\
&= 1 - 2ns_\epsilon(\text{PAR}_S) = 1 - 1 + (1 - 2\epsilon)^{|S|} = (1 - 2\epsilon)^{|S|}
\end{aligned}$$

Here we used the previously derived result for noise sensitivity of a parity function. Pulling these cases together obtain:

$$\mathbb{E}[x_S y_T] = \begin{cases} 0 & S \neq T \\ (1 - 2\epsilon)^{|S|} & S = T \end{cases}$$

Plugging this into (1), we get that (1) equals

$$\frac{1}{2} - \frac{1}{2} \sum_{S \subseteq [n]} (1 - 2\epsilon)^{|S|} \hat{f}(S)^2$$

and the theorem is proved.

If $ns_\epsilon(f)$ is small, we have good Fourier concentration; note that a large $|S|$ means more “discounting” in the sum. If we waste $\hat{f}(S)^2$ -energy on large sets, $ns_\epsilon(f)$ remains close to $1/2$. That is, we can engineer an f with a large $ns_\epsilon(f)$ only by loading $\hat{f}(S)^2$ for small $|S|$.

Lemma 1 *For any boolean function $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$, and $\gamma \in (0, \frac{1}{2})$,*

$$\sum_{S: |S| \geq \gamma^{-1}} \hat{f}(S)^2 \leq 2.32ns_\gamma(f).$$

Proof:

$$\begin{aligned}
2ns_\gamma(f) &= 1 - \sum_S (1 - 2\gamma)^{|S|} \hat{f}(S)^2 \\
&= \sum_S \hat{f}(S)^2 - \sum_S (1 - 2\gamma)^{|S|} \hat{f}(S)^2 \\
&= \sum_S (1 - (1 - 2\gamma)^{|S|}) \hat{f}(S)^2 \\
&\geq \sum_{S:|S|\geq\gamma^{-1}} (1 - (1 - 2\gamma)^{|S|}) \hat{f}(S)^2 \\
&\geq \sum_{S:|S|\geq\gamma^{-1}} (1 - (1 - 2\gamma)^{1/\gamma}) \hat{f}(S)^2 \\
&\approx \sum_{S:|S|\geq\gamma^{-1}} (1 - e^{-2}) \hat{f}(S)^2.
\end{aligned}$$

Thus,

$$\frac{2}{(1 - \frac{1}{e^2})} ns_\gamma(f) \geq \sum_{|S|\geq\gamma^{-1}} \hat{f}(S)^2,$$

and note that

$$\frac{2}{(1 - \frac{1}{e^2})} \approx 2.32.$$

By inspecting the lemma and inverting the condition for γ , we obtain the following result:

Corollary 1 *Let $f : \{-1, 1\}^n \rightarrow \{-1, 1\}$, any boolean function and let $\beta(x)$ be such that $ns_\gamma(f) \leq \beta(\gamma)$. Then*

$$\sum_{S:|S|\geq\frac{1}{\beta^{-1}(\frac{1}{2.32})}} \hat{f}(S)^2 \leq \epsilon.$$

We'll apply this to the case of one halfspace and generalize next class.

Fact 1 *Any halfspace h has $(\frac{8.8 \cdot 2.32}{\epsilon})^2$ -Fourier concentration and is thus learnable in polynomial time with the low-degree algorithm.*

Proof: Recall (Theorem 2) that any halfspace h has $ns_\gamma(h) \leq 8.8\sqrt{\gamma}$. That is, solving for $\beta^{-1}(\gamma)|_{\gamma=\epsilon/2.32}$ we obtain

$$\begin{aligned}\beta(\gamma) &= 8.8\sqrt{\gamma} = \frac{\epsilon}{2.32} \\ \Rightarrow \beta^{-1}\left(\frac{\epsilon}{2.32}\right) &= \left(\frac{\epsilon}{8.8 \cdot 2.32}\right)^2\end{aligned}$$

And using this as the lower bound for $|S|$ in Corollary 1 above,

$$\sum_{S:|S|\geq\left(\frac{8.8\cdot 2.32}{\epsilon}\right)^2} \hat{f}(S)^2 \leq \epsilon.$$

References

- [BKS99] Benjamini, Itai; Kalai, Gil; Schramm, Oded. *Noise sensitivity of boolean functions and applications to percolation*. Publ. IHES, 1999.
- [MO03] Mossel, Elchanan; O'Donnell, Ryan. *On the noise sensitivity of monotone functions*. Random Structures and Algorithms, 2003.