1 Recap

Definition 1 Consider function $f : \{-1\}^n \to \mathbb{R}$. We say $f$ has $\alpha(\epsilon, n)$-Fourier Concentration if

$$\sum_{S \subseteq [n], |S| > \alpha(\epsilon, n)} \hat{f}^2(S) \leq \epsilon$$

Theorem 2 If $C$ has Fourier Concentration at most $d = \alpha(\epsilon, n)$, then there is an $O(n^{d/2})$ sample-uniform learning algorithm for $C$.

Definition 3 Let $N_{\epsilon}(x)$ be what we get from randomly flipping each bit of $x$ with probability $\epsilon$ (iid). Then noise sensitivity of function $f$, $ns_{\epsilon}(f)$, is defined as $ns_{\epsilon}(f) = \Pr_{x \in \{-1\}^n} [f(x) \neq f(N_{\epsilon}(x))]$.

Theorem 4 For any linear threshold function $f$, we have $ns_{\epsilon}(f) \leq 8.8\sqrt{\epsilon}$.

2 Today

2.1 Noise Sensitivity vs. Fourier Concentration

Theorem 5 Consider function $f : \{-1\}^n \to \{-1\}$. Then, we have

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

Proof By Definition 3, we have

$$ns_{\epsilon}(f) = \Pr_{x \in \{-1\}^n} [f(x) \neq f(N_{\epsilon}(x))]$$

$$= \Pr_{x \in \{-1\}^n, y = N_{\epsilon}(x)} [f(x) \neq f(y)]$$

$$= \mathbb{E}_{x,y} \left( 1_{f(x) \neq f(y)} \right)$$

$$= \mathbb{E}_{x,y} \left( \frac{(f(x) - f(y))^2}{4} \right)$$

$$= \mathbb{E}_{x,y} \left( 2 - 2f(x)f(y) \right)$$

$$= \frac{1}{2} - \frac{1}{2} \mathbb{E}_{x,y} [f(x)f(y)]$$

$$= \frac{1}{2} - \frac{1}{2} \sum_{s,T \subseteq [n]} \hat{f}(S)\hat{f}(T) \mathbb{E}_{x,y} [\chi_S(x)\chi_T(y)]$$

$$= \frac{1}{2} - \frac{1}{2} \sum_{s,T \subseteq [n]} \hat{f}^2(S) \mathbb{E}_{x,y} [\chi_S(x)\chi_S(y)]$$

Now, we evaluate $\mathbb{E}_{x,y} [\chi_S(x)\chi_S(y)]$. Let $e_{x_i}$ (resp. $e_{y_i}$) be the unit vector that has value $x_i$ (resp $y_i$) at position $i$ and 1 at all the other places. Then, we have

$$\mathbb{E}_{x,y} [\chi_S(x)\chi_S(y)] = \mathbb{E}_{x,y} \left[ \prod_{i=1}^n \chi_S(e_{x_i})\chi_S(e_{y_i}) \right]$$
\begin{align*}
\mathbb{E}_{x,y} \left[ \prod_{i \in S} \chi_S(e_{x_i}) \chi_S(e_{y_i}) \right] &= \prod_{i \in S} \mathbb{E}_{x,y} \left[ \chi_S(e_{x_i}) \chi_S(e_{y_i}) \right] \\
&= \prod_{i \in S} \left( \Pr[\chi_S(e_{x_i}) = \chi_S(e_{y_i})] - \Pr[\chi_S(e_{x_i}) \neq \chi_S(e_{y_i})] \right) \\
&= \prod_{i \in S} \left( \Pr[x_i = y_i] - \Pr[x_i \neq y_i] \right) \\
&= \prod_{i \in S} (1 - 2\epsilon) \\
&= (1 - 2\epsilon)^{|S|} \\
\end{align*}

Hence, substituting this into (1), we get

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

\[ ns_{\epsilon}(f) \leq 2.32 \ ns_{\epsilon}(f) \]

Using this result, we can get the following connection between noise sensitivity of a function and its Fourier Concentration.

**Theorem 6** Consider function \( f : \{-1\}^n \to \{-1\} \). Then, for any \( 0 < \gamma < \frac{1}{2} \), we have

\[ \sum_{S \subseteq [n], |S| \geq \frac{1}{4}} \hat{f}^2(S) \leq 2.32 \ ns_{\gamma}(f) \]

**Proof** Using Theorem 5, we have

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

\[ = \sum_{S \subseteq [n]} \hat{f}^2(S) - \sum_{S \subseteq [n]} (1 - 2\gamma)^{|S|} \hat{f}^2(S) \]

\[ = \sum_{S \subseteq [n]} (1 - (1 - 2\gamma)^{|S|}) \hat{f}^2(S) \]

\[ \geq \sum_{S \subseteq [n], |S| \geq \frac{1}{4}} (1 - (1 - 2\gamma)^{\frac{1}{4}}) \hat{f}^2(S) \]

\[ \geq \sum_{S \subseteq [n], |S| \geq \frac{1}{4}} (1 - e^{-\frac{1}{2}}) \hat{f}^2(S) \]

Hence,

\[ \sum_{S \subseteq [n], |S| \geq \frac{1}{4}} \hat{f}^2(S) \leq \frac{2}{1 - e^{-\frac{1}{2}}} \ ns_{\gamma}(f) \leq 2.32 \ ns_{\epsilon}(f) \]
2.2 Application: Learning Half-Space Functions

The above result gives us a couple of interesting corollaries as follows.

**Corollary 7** Consider function $f : \{\pm 1\}^n \rightarrow \{\pm 1\}$. Also define function $\beta : [0, 0.5] \rightarrow [0, 0.5]$ such that $n_s(f) \leq \beta(\gamma)$ for every $\gamma \in [0, 0.5]$. Then, we have

$$\sum_{S \subseteq [n], |S| \geq \frac{1}{\beta(\gamma)^2}} \hat{f}^2(S) \leq \epsilon$$

**Proof** By Theorem 6, we have

$$\sum_{S \subseteq [n], |S| \geq \frac{1}{\beta(\gamma)^2}} \hat{f}^2(S) \leq 2.32 \ n_s(\frac{1}{\beta(\gamma)^2})(f) \leq 2.32 \ \beta^{-1}(\frac{\epsilon}{2.32}) = \epsilon$$


**Corollary 8** Consider half-space function $h : \{\pm 1\}^n \rightarrow \{\pm 1\}$. Then, we have

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

**Proof** From previous lecture, we know that for a half-space function $h$, we have $n_s(h) \leq 8.8\sqrt{\epsilon}$. Let $\beta(\epsilon) = 8.8\sqrt{\epsilon}$. Thus, using Corollary 7 we have

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

Noting the definition of the $\beta(.)$ function, we have $\beta^{-1}(x) = (\frac{x}{8})^2$, and in particular, $\beta^{-1}(\frac{1}{8}) = (\frac{1}{2\sqrt{2\pi}^2})^2 \geq (\frac{1}{16})^2$. Therefore, $\frac{1}{\beta(\epsilon)^2} \leq \frac{1}{16}$. Hence, we can conclude that

$$\sum_{S \subseteq [n], |S| \geq \frac{1}{16}} \hat{f}^2(S) \leq \sum_{S \subseteq [n], |S| \geq \frac{1}{\beta(\epsilon)^2}} \hat{f}^2(S) \leq \epsilon$$

Note that this corollary, along with our low-degree learning algorithm, gives us a $n^{O(\frac{1}{\epsilon})}$ sample-uniform learning algorithm for half-spaces. As a side note, we add that one can also learn half-space functions in polynomial time, via linear programming.

2.3 Application: Learning Boolean Functions of $k$ Half-Space Functions

**Theorem 9** Consider half-space functions $h_1, h_2, \ldots, h_k$, and boolean functions $f, g : \{\pm 1\}^k \rightarrow \{\pm 1\}$ such that $f(x) = g(h_1(x), h_2(x), \ldots, h_k(x))$. Then, we have $n_s(f) \leq 8.8k\sqrt{\epsilon}$.

**Proof** The main idea of the proof is simple Union bound. We have

$$n_s(f) = \Pr \left[ f(x) \neq f(N_\epsilon(x)) \right]$$

$$= \Pr \left[ g(h_1(x), h_2(x), \ldots, h_k(x)) \neq g(h_1(N_\epsilon(x)), h_2(N_\epsilon(x)), \ldots, h_k(N_\epsilon(x))) \right]$$

$$\leq \Pr \left[ h_1(x) \neq h_1(N_\epsilon(x)) \right] \ or \ h_2(x) \neq h_2(N_\epsilon(x)) \ or \ \ldots \ or \ h_k(x) \neq h_k(N_\epsilon(x))$$

$$\leq \Pr \left[ h_1(x) \neq h_1(N_\epsilon(x)) \right] + \Pr \left[ h_2(x) \neq h_2(N_\epsilon(x)) \right] + \ldots + \Pr \left[ h_k(x) \neq h_k(N_\epsilon(x)) \right]$$

$$\leq 8.8k\sqrt{\epsilon}$$

(4)

where the least inequality follows from Theorem 4. ■
Corollary 10  Consider half-space functions $h_1, h_2, \ldots, h_k$, and boolean functions $f, g : \{\pm 1\}^k \to \{\pm 1\}$ such that $f(x) = g(h_1(x), h_2(x), \ldots, h_k(x))$. Then, we have

$$\sum_{S \subseteq [n], |S| \geq \Theta\left(\frac{k^2}{\epsilon^2}\right)} \hat{f}^2(S) \leq \epsilon$$

**Proof**  Follows immediately from Corollary 7 and Theorem 9.

Again, this corollary, along with our low-degree learning algorithm, gives us a $n^{O\left(\frac{k^2}{\epsilon^2}\right)}$ sample-uniform learning algorithm for any boolean function of $k$ half-space functions. Side note is that, in contrast to learning half-space functions, it is not known how to learn arbitrary functions of $k$ half-space functions in polynomial time via linear programming.

2.4 Learning Parity Functions

In this model, we assume that there is a black box which contains an unknown parity function $f : \{0, 1\}^n \to \{0, 1\}$ and the goal is to learn this function. Speaking differently, the black box calculates some function $f = \chi_S$ and we want to figure out what $S$ is. For this, we have access to samples of this function, i.e., $(x_1, f(x_1)), (x_2, f(x_2)), \ldots, (x_m, f(x_m))$.

**The case without noise:** If there is no noise, i.e., we have access to exact evaluations, then this problem can be solved simply by solving a system of linear equations, where we have $n$ unknowns such that the $i^{th}$ unknown is zero if $i \notin S$ and one otherwise.

**The case with noise:** In the sequel, we focus on the problem of learning parity functions in the presence of noise. This means that for each sample-point $x_i$, we have $\Pr[f(x_i) = \chi_S(x_i)] \geq 0.5 + \delta$ where $\delta$ is some possibly small positive probability. In this scenario, the goal might be either to find $\chi_S(.)$ that is closest to $f(.)$ or find all $\chi_S(.)$ that are close enough to $f(.)$. The former case is just finding the largest Fourier coefficient while the latter is finding all large enough Fourier coefficients.

There are different variants to this problem. When sample-points $x_1, x_2, \ldots, x_m$ are chosen adversarially, the problem is equivalent to “maximum likelihood decoding of linear codes” which is known to be NP-hard. When sample-points $x_1, x_2, \ldots, x_m$ are chosen randomly at uniform, the problem is equivalent to “hardness of decoding linear codes”, or also often called “hardness of parity with noise”. A possibly slightly easier version asks what happens if the noise just flips every bit with some probability. This version is often called “hardness of decoding random linear codes” which is usually used as an assumption in cryptography and coding. A theorem by Blum, Kalai, and Wasserman gives a slightly sub-exponential (2^{\frac{m}{\log m}}) algorithm for this variant. Next time, we will study the question that “what happens if one can make queries into desired sample-points?”.