6.842 Randomness and Computation April 9, 2012

Lecture 16

Lecturer: Ronitt Rubinfeld *Scribe: Will Hasenplaugh*

1 Introduction

Today, we are going to discuss learning noisy Parity functions with queries. If f is linear we can find it efficiently by solving a linear system of equations. However, for an arbitrary f that is "approximately" linear, we would still like to efficiently find all linear functions that are 'close' to f.

Under the general PAC model, this problem is conjectured to be hard, referred to as *hardness of parity with noise*, *hardness of decoding linear codes*, *maximum likelihood decoding of linear codes* or *finding the largest Fourier coefficient*. Sub-exponential algorithms exist for certain relaxations of the problem, however, the general case is believed to be NP-hard.

Since the problem is infeasible in the PAC model, we will allow queries to f instead of random sampling. This problem was first studied by Goldreich and Levin in the context of cryptographic pseudorandom generators and has found applications in error correcting codes (list decoding of Hadamard codes) and learning theory (by Kushilevitz and Mansour).

- Model for Learning with Noise
- Warmup 0 and $0'$ - Brute force algorithms
- Warmup 1 - Learning exactly: zero error case
- Warmup 2 - Learning almost exactly fairly certainly: $\frac{1}{\text{poly}(n)}$ error case
- Warmup 3 - Learning fairly exactly almost certainly: $\approx \frac{1}{4}$ error case

2 Model for Learning with Noise

2.1 Goal

Given a linear Boolean function, $f : \{\pm 1\}^n \to \{\pm 1\}$, and a threshold, θ:

1. Output all $S \subseteq [n]$ such that $|\hat{f}(S)| \geq \theta \to$ "All Important FC's"
2. Do not output any $S \subseteq [n]$ such that $|\hat{f}(S)| < \frac{\theta}{2} \to$ "No Junk"

2.2 Recall

$\Pr_x[f(x) = \chi_S(x)] = \frac{1}{2} + \frac{1}{2}\hat{f}(S)$, so
Case 1 $\Rightarrow \Pr_x[f(x) = \chi_S(x)] \geq \frac{1}{2} + \frac{\theta}{2}$ $\forall S : |\hat{f}(S)| \geq \theta$
Case 2 $\Rightarrow \Pr_x[f(x) = \chi_S(x)] < \frac{1}{2} + \frac{\theta}{4}$ $\forall S : |\hat{f}(S)| < \frac{\theta}{2}$

2.3 Notation

- $e_i = (1, \ldots, 1, -1, 1, \ldots, 1)$ where -1 is in the i^{th} position
- Bitwise product: $x \odot y = (x_1 y_1, x_2 y_2, \ldots, x_n y_n)$

Warmup 0: Brute Force

Suppose we are permitted arbitrary runtime and queries. We could merely evaluate all Fourier Coefficients, $\hat{f}(S)$, exactly.

Warmup 0': (slightly less) Brute Force

If we are instead permitted only polynomially many queries and arbitrary runtime, we could estimate all Fourier Coefficients, $\hat{f}(S)$, using the method from Lecture 12 (Estimating One Fourier Coefficient). However, we can reuse the queries used to estimate the first Fourier Coefficient to estimate all (exponentially) others, requiring only a polynomial number of queries total. This analysis (using Chernoff and Union Bounds) is straightforward, as we saw in Lecture 15.

Warmup 1: Learning exactly

Suppose $f = \chi_S$ for some S (f is linear). This is the "0 error case" and $\theta = 1$.

Algorithm 1
Linear Algebra: merely solve for the coefficients:
Given n linearly independent x_i's, find the S that satisfies $x_i \cdot S = f(x_i)\ \forall i$.

Algorithm 2
$S \leftarrow \emptyset$
for all $i \in [n]$
    if $f(\overrightarrow{1}) \neq f(e_i)$
        $S \leftarrow S \cup \{i\}$
output S

The correctness of this Algorithm 2 is follows directly from the fact that $\chi_S(u) \neq \chi_S(u \odot e_i)\ \forall i \in S$, $u \in \{\pm 1\}^n$.

Warmup 2:

Suppose there exists an S such that f agrees with χ_S "almost everywhere" - that is, on $\geq 1 - \frac{1}{\text{poly}(n)}$ fraction of inputs. It can be efficiently found with probability that depends on the closeness of agreement between f and χ_S.

Algorithm
$S \leftarrow \emptyset$
choose $r \in_R \{\pm 1\}^n$
for all $i \in [n]$
    if $f(r) \neq f(r \odot e_i)$
        $S \leftarrow S \cup \{i\}$
output S

Why does this work? (Sketch)

First, notice that r and $r \odot e_i$ are both uniformly distributed and that $f(r) = \chi_S(r)$ and $f(r \odot e_i) = \chi_S(r \odot e_i)$ for almost all r. So,

$$\Pr[S \text{ not correct}] \leq 2n \frac{1}{\text{poly}(n)}$$

by the Union Bound. Here, the 2 arises from the 2 queries, the n arises from the union across all $i \in [n]$ and the $\frac{1}{\text{poly}(n)}$ arises from the closesness of agreement between f and χ_S. So, as f and χ_S more closely agree, the more the third term will dominate the second, driving down the probability of error.

Warmup 3:

Now suppose that we require only that there exists an S such that f agrees with χ_S on at least $\frac{3}{4} + \frac{\theta}{2}$ of the inputs. Notice that there can be only one such S. Otherwise, if there were two such S's, S_1 and S_2, then they would have to agree with each other on at least $\frac{1}{2} + \theta$ of the inputs, contradicting the fact that unique χ_S's differ on exactly half of the inputs. The algorithm from Warmup 2 would not suffice to find an S differing so much with f, so we employ a new technique:

Algorithm
$S \leftarrow \emptyset$
choose $r_1, \cdots, r_t \in_R \{\pm 1\}^n$
for all $i \in [n]$
  if $\sum_{j=1}^{t} f(r_j) f(r_j \odot e_i) \leq 0$ ... *a majority of r_j's yield $f(r_j) \neq f(r_j \odot e_i)$*
    $S \leftarrow S \cup \{i\}$
output S

Why does this work? (Sketch)

Let $S^\star \subseteq [n]$ be the unique set such that $\chi_{S^\star}$ agrees with f on $\geq \frac{3}{4} + \frac{\theta}{2}$ of the inputs. We seek to bound the probability that the r_j's collectively select some $S \neq S^\star$.

$$\Pr[\text{"wrong" answer for } r_j \text{ on } i] = \Pr[f(r_j) f(r_j \odot e_i)(-1)^{1_{i \in S^\star}} = -1]$$

but, $\chi_S(r_j)\chi_S(r_j \odot e_i)(-1)^{1_{i \in S}} = 1 \ \forall S$. So,

$$\begin{aligned} \Pr[\text{"wrong" answer for } r_j \text{ on } i] &= \Pr[f(r_j) \neq \chi_{S^\star}(r_j) \cup f(r_j \odot e_i) \neq \chi_{S^\star}(r_j \odot e_i)] \\ &\leq (\frac{1}{4} - \frac{\theta}{2}) + (\frac{1}{4} - \frac{\theta}{2}) = \frac{1}{2} - \theta \end{aligned}$$

using the Union Bound and the observation that the r_j's (and $r_j \odot e_i$'s) are uniformly distributed. So, using the Chernoff Bounds, we can set $t = \Theta(\frac{1}{\theta^2} \log \frac{n}{\delta})$ to boost the probability of outputting $S^\star$ to $1 - \delta$:

$$\begin{aligned} \Pr[\text{most } r_j \text{ are "correct" on } i] &\geq 1 - \frac{\delta}{n} \\ \Pr[\text{most } r_j \text{ are "correct" } \forall i] &\geq 1 - \delta \\ \Pr[\text{Algorithm outputs } S^\star] &\geq 1 - \delta \end{aligned}$$

So, if we are lucky enough to learn a function which highly agrees with a single χ_S (i.e. on more than $\frac{3}{4} + \frac{\theta}{2}$ of the inputs), then we can find it with arbitrarily high probability, $1 - \delta$, with a number of samples that is polynomial in $\log n$, $\log \frac{1}{\delta}$ and $\frac{1}{\theta}$.