Notes for Lecture 15

Notes written 12/07/04

Learning Decision Trees

In these notes it will be convenient to denote bits using the set $\{-1, 1\}$ instead of $\{0, 1\}$.

1 Decision Trees

A decision tree accepts or rejects input strings $x_1, \ldots, x_n$ and can be described as a directed tree where every internal node has two children. The computational path begins at the root. Each non-final vertex is labeled with a variable x_i and has two outgoing edges, one labeled 1 and the other -1. The path follows edge b such that $x_i = b$, and ends at a leaf. Each leaf is labeled -1 or 1. Such a label is called the *label* of the leaf. The output of a decision tree T on input x is the value of the leaf reached in the computational path of T given x.

The *size* of a decision tree is the number of nodes. Note that in a tree where every internal node has exactly two children it is always true that the number of internal vertices is equal to the number of leaves minus 1. Therefore measuring size in terms of number of internal vertices, number of leaves, or total number of vertices are all equivalent measures to within a constant factor.

2 Fourier Analysis

For every $S \subseteq \{1, \ldots, n\}$, define the function $u_S : \{-1, 1\}^n \to \mathbb{R}$ as

$$u_S(x_1, \ldots, x_n) = \prod_{i \in S} x_i$$

If $S = \emptyset$, then we define $u_\emptyset(x) = 1$ for every x.

Given two functions $f, g : \{-1, 1\}^n \to \mathbb{R}$, define their inner product as

$$f \cdot g = \frac{1}{2^n} f(x)g(x) = \mathbf{E}_x[f(x)g(x)]$$

Then we have that for every set S

$$u_S \cdot u_S = 1$$

In fact, more generally, $f \cdot f = 1$ for every function $f : \{-1, 1\}^n \to \{-1, 1\}$.

Furthermore, if $S \neq T$, we have

$$u_S \cdot u_T = 0$$

Which implies that the functions u_S are linearly independent over the reals. Indeed, if there were a linear combination

$$u_S = \sum_{T \neq S} \alpha_T u_T$$

then we would also have

$$1 = u_S \cdot u_S = (\sum_{T \neq S} \alpha_T u_T) \cdot u_S = \sum_{T \neq S} \alpha_T u_T \cdot u_S = 0$$

The set of functions $f : \{-1,1\}^n \to \mathbb{R}$ is a 2^n-dimensional vector space over the reals, and since there are 2^n functions u_S, all linearly independent, they must form a basis. Indeed, with respect to the inner product we have defined, the u_S form an orthonormal basis.

Every function $f : \{-1,1\}^n \to \{-1,1\}$ can then be written as a linear combination

$$f(x) = \sum_S \hat{f}_S u_S(x)$$

of the functions u_S. The coefficients $\hat{f}_S$ are called the Fourier coefficients of f.

We will need a few properties of the coefficients. First, we note that

$$\hat{f}_S = f \cdot u_S \tag{1}$$

Which is a standard property of inner-product spaces with an orthonormal basis.

Next, we have

Lemma 1 (Parseval's Equality) *For every function $f : \{-1,1\}^n \to \{-1,1\}$,*

$$\sum_S \hat{f}_S^2 = \mathbf{E}_x[f^2(x)] \ .$$

In particular, if $f : \{-1,1\}^n \to \{-1,1\}$ is boolean, then $\sum_S \hat{f}_S^2 = 1$.

Finally, if $f : \{-1,1\}^n \to \{-1,1\}$ is a boolean function, then

$$\hat{f}_S = \mathbf{E}[f(x)u_S(x)] = \mathbf{Pr}[f(x) = u_S(x)] - \mathbf{Pr}[f(x) \neq u_S(x)] = 2\mathbf{Pr}[f(x) = u_S(x)] - 1 \tag{2}$$

Note also that $u_S(x_1, \ldots, x_n) = \bigoplus_{i \in S} x_i$ if we adopt the convention that 1 is $False$ and -1 is $True$. Therefore, $\hat{f}_S \geq \epsilon$ if and only if f has agreement at least $1/2 + \epsilon/2$ with the "linear" function $\bigoplus_{i \in S} x_i$ (where we mean *linear* with respect to operations over bits, not linear over the reals).

This observation, together with Parseval's equality, implies that there are at most $1/\epsilon^2$ linear functions that have agreement at least $1/2 + \epsilon/2$ with a given boolean function.

We can then formulate the following version of the Goldreich-Levin theorem [GL89].

Lemma 2 (Goldreich-Levin, revised form) *There is a probabilistic algorithm GL that given oracle access to a function $f : \{0,1\}^n \to \{0,1\}$ and given a threshold parameter $\tau > 0$, an accuracy parameter $\gamma > 0$ and a confidence parameter $\delta > 0$, runs in time polynomial in n, $1/\tau$, $1/\gamma$ and $\log 1/\delta$, and outputs a list of sets $S_1, \ldots, S_t$ and of numbers $\bar{f}_{S_1}, \ldots, \bar{f}_{S_t}$, such that, with probability at least $1 - \delta$ the following conditions hold:*

- *Every set S such that $|\hat{f}_S| \geq \tau$ is in the list;*
- *For every set S in the list, $|\hat{f}_S - \bar{f}_S| \leq \gamma$.*

PROOF: In the standard version, the Goldreich-Levin algorithm is given a parameter ϵ and it produces a list of size $O(1/\epsilon^2)$ such that, for each S such that $\hat{f}_S \geq 2\epsilon$, there is a probability at least $3/4$ that S is in the list.

We first run the Goldreich-Levin algorithm with parameter $\tau/2$ independently $O(\log 1/\tau\delta)$ times, and take the union of the lists. Now, for each of the $\leq 1/\tau^2$ sets S such that $\hat{f}_S \geq \tau$, the set S has a probability at most $1/4$ of being missed in each iteration, and a probability at most, say, $\tau^2\delta/4$ of being missed every time. By a union bound, there is a probability at least $1 - \delta/4$ that the final list contains every set S such that $\hat{f}_S \geq \tau$.

We then repeat the same operations, but using $-f$ as an oracle. This gives another list of size $O(\tau^{-2} \log \tau^{-1}\delta^{-1}$ that contains all the sets S such that $\hat{f}_S \leq -\tau$, except with probability at most $\delta/4$.

We take the union of the two lists, and define it to be L. Except with probability at most $\delta/2$, the list satisfies the first condition.

For every set $S \in L$, we pick $t = O(\gamma^{-2} \log L/\delta)$ sample points $x^1, \ldots, x^t$ in $\{-1, 1\}^n$, and define

$$\bar{f}_S = \frac{1}{t} \sum_i f(x^i) u_S(x^i) \tag{3}$$

By Chernoff bounds, there is a probability at most $\delta/2L$ that $\bar{f}_S$ differs from $\hat{f}_S$ by more than γ. Taking a union bound over the sets, we see that the numbers $\bar{f}_S$ satisfy the second condition, except with probability at most $\delta/2$. □

3 Overview of the Proof

Our main result is the following.

Theorem 3 *There is a probabilistic learning algorithm A that given oracle access to a function $f : \{0,1\}^n \to \{0,1\}$ that can be computed by a decsion tree of size S, and given S and parameters $\epsilon, \delta > 0$, runs in time polynomial in $n, S, 1/\epsilon, \log 1/\delta$ and outputs a circuit C that, with probability at least $1 - \delta$, is ϵ-close to f.*

We prove it in two steps. We show that every function whose Fourier coefficients satisfy a certain condition can be efficiently learned, and then we show that functions computed by small decision trees satisfy the condition.

Lemma 4 *There is a probabilistic learning algorithm A that given oracle access to a function $f : \{0,1\}^n \to \{0,1\}$, given a number m such that*

$$\sum_S |\hat{f}_S| \leq m$$

and given $\epsilon, \delta > 0$, runs in time polynomial in $n, m, 1/\epsilon, \log 1/\delta$ and outputs a circuit C that, with probability at least $1 - \delta$, is ϵ-close to f.

The number $\sum_S |\hat{f}_S|$ is called the ℓ_1-norm of the function f, and is also denoted by $||f||_1$. Lemma 4 says that functions of polynomial ℓ_1-norm can be learned in polynomial time.

Lemma 5 *If f can be computed by a decision tree of size S, then the ℓ_1 norm of f is at most S.*

4 Proof of Lemma 4

We fix $\tau = \epsilon/2L$. If $\ell = O(\tau^{-2}\log(\tau^{-1}\delta^{-1})$ is an upper bound to the size of the list returned by the Goldreich-Levin algorithm with threshold τ and confidence δ, then we fix $\gamma = \sqrt{\epsilon/2\ell}$ and we run the Goldreich-Levin algorithm with threshold τ, confidence δ and accuracy γ. We find a list L of sets and values $\bar{f}_S$ for each set in the list such that, with probability $\geq 1-\delta$ over the internal coin tosses of the algorithm, we have:

- Every set S such that $|\hat{f}_S| \geq \tau$ is in the list;
- For every set S in the list, $|\hat{f}_S - \bar{f}_S| \leq \gamma$.

Then we define the function $h(x) = \sum_{S\in L} \bar{f}_S u_S$. The Fourier coefficients of the difference $d(x) := f(x) - h(x)$ are as follows.

- If $S \notin L$, then $\hat{h}_S = 0$ and so $\hat{d}_S = \hat{f}_S$, and also $|\hat{d}_S| = |\hat{f}_S| \leq \tau$.
- If $S \in L$, then $|\hat{h}_S| \leq \gamma$.

We now want to estimate $\mathbf{E}[(f(x)-h(x))^2]$, which is a measure of how good is h as an approximation to f. We have

$$\begin{aligned}
\mathbf{E}[(f(x)-h(x))^2] &= \mathbf{E}[d^2(x)] \\
&= \sum_S \hat{d}_S^2 \\
&= \sum_{S\notin L} \hat{d}_S^2 + \sum_{S\in L} \hat{d}_S^2 \\
&\leq \tau \sum_{S\notin L} |\hat{d}_S| + \sum_{S\in L} \gamma^2 \\
&\leq \tau m + |L|\gamma^2 \\
&\leq \epsilon
\end{aligned}$$

Define $g : \{-1,1\}^n \to \{-1,1\}$ such that $g(x) = 1$ if $h(x) \geq 0$ and $g(x) = -1$ if $h(x) < 0$. We see that

$$\mathbf{Pr}[g(x) \neq f(x)] \leq \mathbf{E}_x[(f(x)-h(x))^2] \leq \alpha + \epsilon$$

because every x such that $g(x) \neq f(x)$ must also be such that $|f(x)-h(x)| \geq 1$ and so $(f(x)-h(x))^2 \geq 1$.

We output the circuit that computes g.

5 Proof of Lemma 5

First note that if $f, g, h : \{-1,1\}^n \to \mathbb{R}$ are functions such that $f = g + h$, then $\hat{f}_S = \hat{g}_S + \hat{h}_S$ for every set S, and we have

$$||f||_1 \leq ||g||_1 + ||h||_1 \tag{4}$$

Let f be a function computed by a decision tree T with m leaves. We assume that tests are never repeated in a computational path of T, otherwise we can get a smaller tree T for f satisfying such a property.

For every leaf v of the decision tree, let $val(v)$ be the output associated with that leaf, $d(v)$ the depth of v and I_v the set of inputs x such that computation of T on input x ends at v. The set I_v clearly contains a $1/2^{d(v)}$ fraction of $\{-1,1\}^n$ (this is where we are using the assumption that in each computational path we test distinct inputs). We also define $var(v) \subseteq \{1,\ldots,n\}$ the be the set of indices i such that a test for x_i is in the path from the root to v. Note that $d(v) = |var(v)|$. Finally, we define $f^v(x)$ to be $val(v)$ if the computation of T on input x ends at v, and zero otherwise.

By definition:

$$f(x) = \sum_{v \text{leaf}} f^v(x)$$

For a set S and a leave v,

$$\hat{f}^v_S = \mathbf{E}_x f^v(x) u_S(x) = \frac{1}{2^{d(v)}} \mathbf{E}_{x \in I_v} f^v(x) u_S(x) = \frac{1}{2^{d(v)}} val(v) \mathbf{E}_{x \in I_v} u_S(x)$$

From which we see that $\hat{f}^v_S = 0$ if $S \not\subseteq var(v)$, and $|\hat{f}^v_S| = 1$ if $S \subseteq var(v)$. Since there are $2^{d(v)}$ subsets of $var(v)$, we get

$$||f^v||_1 = \sum_S |\hat{f}^v_S| = \sum_{S \subseteq var(v)} \frac{1}{2^{d(v)}} = 1$$

And so

$$||f||_1 \leq \sum_{v \text{leaf}} ||f^v||_1 \leq m$$

6 References

The results of these notes are due to Kushilevitz and Mansour [KM93].

References

[GL89] O. Goldreich and L. Levin. A hard-core predicate for all one-way functions. In *Proceedings of the 21st ACM Symposium on Theory of Computing*, pages 25–32, 1989. 2

[KM93] Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the fourier spectrum. *SIAM Journal on Computing*, 22(6):1331–1348, 1993. 5