CS 395T Computational Complexity of Machine Learning

Lecture 13: March 3, 2005 Lecturer: Adam Klivans

Scribe: YF Sit

Approximating Decision Trees with Queries

In this lecture, we will show how polynomial sized decision trees can be approximated with high probability in polynomial time using queries. We first reduce the problem of finding a sparse function that closely approximates the target function f to finding all the large Fourier coefficients of f. We then show an algorithm to find these coefficients in polynomial time with probability $1-\delta$ of success. Then we show that if a function has a small L_1 norm, it can be approximated by sparse function and polynomial sized decision tree is such a function. Therefore, we can learn a function that approximates a decision tree in polynomial time with $1-\delta$ probability.

13.1 Background

All the functions in this note are Boolean and have a range of {-1, 1} unless specified otherwise. The input to the functions, $x \in \{0,1\}^n$, is usually omitted for convenience. Expectation E is taken over all $x \in \{0,1\}^n$ if not specified.

We say a Boolean function $f(x) : \{0,1\}^n \rightarrow \{-1,1\}$ is ϵ-approximated by another Boolean function $g(x)$ if the squared expected error $E[(f(x)-g(x))^2]$ is $\leq \epsilon$.

A function f is t-sparse if it has at most t non-zero Fourier coefficients.

Theorem 1 *If f is ϵ-approximated by a t-sparse function g, then there exists a t-sparse function h such that $|\hat{h}(s)| \geq \epsilon/t$ for all the Fourier coefficients of h and f is $\epsilon + \epsilon^2/t$ approximated by h.*

Proof: First, without loss of generality, we assume $\hat{g}(s_i) = \hat{f}(s_i)$, for $1 \leq i \leq t$. Let $A = \{s | \hat{g}(s) \neq 0\}$ and $A' = A \bigcap \{s | \hat{f}(s) \geq \epsilon/t\}$. Hence, h can be written as $\sum_{x \in A'} \hat{f}(s)\chi_s$. Now consider the squared expected error between f and g: $E\left[|f-g|^2\right]$. It can be written as $\sum_{s \notin A} \hat{f}(s)^2 = \epsilon$, since f is ϵ-approximated by g.

Each element in the set $A - A'$ has a magnitude less than ϵ/t and because g is t-sparse, $|A - A'|$ is at most t. Therefore, $\sum_{s \in A - A'} \hat{f}(s)^2 < t(\epsilon/t)^2$. Now the squared expected error between f and h is:

$$E[|f-h|^2] = \sum_{s \notin A} \hat{f}(s)^2 + \sum_{s \in A - A'} \hat{f}(s)^2$$

$$= \epsilon + t\left(\frac{\epsilon}{t}\right)^2$$
$$= \epsilon + \frac{\epsilon^2}{t}$$

■

The above theorem has reduced the problem of finding a t-sparse function to ϵ-approximate f into the problem of finding all the coefficients of f that are greater than a threshold such that the t-sparse function $O(\epsilon)$-approximates f. Let the required threshold be θ.

Now we will look at some properties of the Fourier spectrum if we fix the prefix of s. For a string α with length k, we define

$$f_\alpha(x) = \sum_{z\in\{0,1\}^{n-k}} \hat{f}(\alpha z)\chi_z(x),$$

which is essentially the spectrum that has α as the prefix.

Theorem 2 *For any function f, $k \in \{1,\ldots,n-1\}, \alpha \in \{0,1\}^k, x \in \{0,1\}^{n-k}$, and $y \in \{0,1\}^k$,*

$$f_\alpha(x) = E_y[f(yx)\chi_\alpha(y)],$$

and $|f_\alpha(x)| \le 1$.

Proof: Using Fourier transform, $f(yx) = \sum_z \hat{f}(z)\chi_z(yx)$. Let $z = z_1 z_2, z_1 \in \{0,1\}^k, z_2 \in \{0,1\}^{n-k}$, then $\chi_z(yx) = \chi_{z_1}(y)\chi_{z_2}(x)$. Therefore,

$$\begin{aligned} E_y\left[f(yx)\chi_\alpha(y)\right] &= E_y\left[\left(\sum_{z_1}\sum_{z_2}\hat{f}(z_1z_2)\chi_{z_1}(y)\chi_{z_2}(x)\right)\chi_\alpha(y)\right] \\ &= \sum_{z_1}\sum_{z_2}\hat{f}(z_1z_2)\chi_{z_2}(x)E_y[\chi_{z_1}(y)\chi_\alpha(y)] \end{aligned}$$

By the orthonormality of the basis, $E_y[\chi_{z_1}(y)\chi_\alpha(y)] = 0$ when $z_1 \ne \alpha$ and is 1 when $z_1 = \alpha$. Therefore, we only have to add the terms with $z_1 = \alpha$ in the above equation, which gives

$$E_y\left[f(yx)\chi_\alpha(y)\right] = \sum_{z_2}\hat{f}(\alpha z_2)\chi_{z_2}(x) = f_\alpha(x).$$

Since $|f(yx)| = 1$ and $|\chi_\alpha(y)| = 1$, from the above equality, $|f_\alpha(x)| \le 1$. ■

It is not too hard to see the above theorem can be applied to find the coefficients $\hat{f}(z)$ such that $|\hat{f}(z)| \ge \theta$. But before describing the actual algorithm, we will first prove the following two lemmas that claim the number of such coefficients is limited and the property $E[f_\alpha^2] \ge \theta^2$ only holds for a limited number of α's.

Lemma 1 *At most $1/\theta^2$ values of z satisfy $|\hat{f}(z)| \geq \theta$, where $0 < \theta < 1$.*

Proof: Using Parseval's equality, we have $\sum_z \hat{f}^2(z) = \sum_z |\hat{f}(z)|^2 = E[f^2] = 1$. Therefore, there are at most $1/\theta^2$ values of z such that $|\hat{f}(z)| \geq \theta$. ∎

Lemma 2 *At most $1/\theta^2$ f_α satisfy $E[f_\alpha^2] \geq \theta^2$, for $\alpha \in \{0,1\}^k$ and $1 \leq k < n$.*

Proof: From the definition of f_α, we have

$$\begin{aligned} E[f_\alpha(x)^2] &= E\left[\sum_{z_1 \in \{0,1\}^{n-k}} \hat{f}(\alpha z_1)\chi_{z_1}(x) \sum_{z_2 \in \{0,1\}^{n-k}} \hat{f}(\alpha z_2)\chi_{z_2}(x)\right] \\ &= \sum_{z_1 \in \{0,1\}^{n-k}} \sum_{z_2 \in \{0,1\}^{n-k}} \hat{f}(\alpha z_1)\hat{f}(\alpha z_2) E[\chi_{z_1}(x)\chi_{z_2}(x)] \\ &= \sum_{z_1 \in \{0,1\}^{n-k}} \hat{f}(\alpha z_1)^2 \end{aligned}$$

Therefore, if $|\hat{f}(\alpha z_1)| \geq \theta$ for some $z_1 \in \{0,1\}^{n-k}$, then $E[f_\alpha^2] \geq \theta^2$. Since

$$\sum_{\alpha \in \{0,1\}^k} E[f_\alpha^2] = E[f^2] = 1,$$

there are hence at most $1/\theta^2$ of f_α that have $E[f_\alpha^2] \geq \theta^2$. ∎

13.2 The KM Algorithm

Using the above theorem and lemma, we can find the coefficients $\hat{f}(z)$ such that $|\hat{f}(z)| \geq \theta$ by the following algorithm (by Kushilevitz and Mansour [1]):

$KM(\alpha)$ {
  if *length(α)* $= n$ **then**
    output α
  else if $E[f_\alpha^2] \geq \theta^2$ **then**
    $KM(\alpha 0)$
    $KM(\alpha 1)$
}

We start finding the coefficients by calling $KM(\phi)$, where ϕ is an empty string. The KM algorithm works by increasing the number of known bits ($|\alpha|$) at the prefix of the strings s such that $|\hat{f}(s)| \geq \theta$.

From the argument in Lemma 2, if $|\hat{f}(\alpha z)| \geq \theta$ for some $z \in \{0,1\}^{n-k}$, then $E[f_\alpha^2] \geq \theta^2$. Therefore, the condition $E[f_\alpha^2] \geq \theta^2$ in the algorithm ensures that all the required coefficients will be output.

From Lemma 2, we know that there are at most $1/\theta^2$ f_α's that satisfy $E[f_\alpha^2] \geq \theta^2$ for each length of α. Therefore, the algorithm makes at most $2n/\theta^2$ recursive calls to find all the required coefficients.

13.3 Approximating $E[f_\alpha^2]$

The above algorithm assumes we know the true value of $E[f_\alpha^2]$, but we cannot compute this value with only polynomial number of examples. However, we can approximate $f_\alpha = E_y[f(yx)\chi_\alpha(y)]$ and $E[f_\alpha^2]$ by double sampling. It turns out that the number of samples needed to maintain the error bound δ is polynomial in $1/\theta, \log n$, and $\log \frac{1}{\delta}$. The algorithm and complete prove can be found in section 3 of [1].

Here we just quote the result:

Theorem 3 *There exists a randomized algorithm that outputs a list of strings $\alpha_i \in \{0,1\}^n$ for any Boolean function $f : \{0,1\}^n \rightarrow \{-1,1\}$, any δ and θ, where $0 < \delta, \theta \leq 1$, such that*

- *with probability $1-\delta$, $|\hat{f}(\alpha_i)| \geq \theta$ for every string α_i in the output list and there is no strings α for which $|\hat{f}(\alpha)| \leq \theta/2$. The maximum length of the list is $4/\theta$.*
- *the algorithm runs in time polynomial in $n, 1/\theta$, and $\log 1/\delta$.*

Applying Theorem 1, if f can be ϵ-approximated by a t-sparse function, then we need to find all the coefficients larger than $\theta = \epsilon/t$ to $O(\epsilon)$-approximate f. The running time of the algorithm will then a polynomial in $n, t, 1/\epsilon$, and $\log 1/\delta$.

13.4 Functions with Small L_1 Norm

Define the L_1 norm of a Boolean function f (also written as $L_1(f)$) to be $\sum_s |\hat{f}(s)|$. The following lemma shows that functions with small L_1 norm can be approximated by sparse functions.

Lemma 3 *For any Boolean function f, there exists a Boolean function h such that h is $\frac{L_1(f)^2}{\epsilon}$-sparse and $E[(f-h)^2] \leq \epsilon$.*

Proof: Consider the set $A = \{s \,\Big|\, |\hat{f}(s)| \geq \epsilon/L_1(f)\}$. There are at most $L_1(f)/(\frac{\epsilon}{L_1(f)}) = L_1(f)^2/\epsilon$ elements in A. Let $h = \sum_{s\in A} \hat{f}(s)\chi_s(x)$, which is $\frac{L_1(f)^2}{\epsilon}$-sparse. Then,

$$E[(f-h)^2] \quad = \quad \sum_{s \notin A} \hat{f}(s)^2$$

$$\begin{aligned} &\le \max_{s \notin A} |\hat{f}(s)| \cdot \sum_s |\hat{f}(s)| \\ &\le \frac{\epsilon}{L_1(f)} L_1(f) \\ &= \epsilon \end{aligned}$$

■

From this lemma and Theorem 3, we reach the following corollary:

Corollary 1 *If a Boolean function f has L_1 norm that is polynomial in n, then a function h that ϵ-approximates f is learnable in polynomial time (in $n, 1/\epsilon$, and $\log 1/\delta$) with queries with probability $1 - \delta$.*

13.5 Approximating Decision Trees with Queries

If we can bound the L_1 norm of decision trees to some polynomial of n, then from the previous corollary, we can learn a function that approximates the decision tree in polynomial time $1 - \delta$ of the time.

Here we consider decision tree with m leaves, where m is polynomial in n. It follows that the size of the tree is also polynomial in n. Define the function $AND_{b_1,\ldots,b_d} : \{0,1\}^d \to \{0,1\}$ to be:

$$AND_{b_1,\ldots,b_d}(x_1, \ldots, x_d) = \begin{cases} 1 & b_i = x_i, \quad \forall i, 1 \le i \le d \\ 0 & \textit{otherwise.} \end{cases}$$

We are going to find $L_1(AND_{b_1,\ldots,b_d})$, and before that, we make the following claims for bounding the L_1 norm of functions:

Claim 1 $L_1(f + g) \le L_1(f) + L_1(g)$.

Claim 2 $L_1(f \cdot g) \le L_1(f) \cdot L_1(g)$.

Note that $AND_{b_1,\ldots,b_d}$ can written as

$$AND_{b_1,\ldots,b_d}(x_1, \ldots, x_d) = \prod_{i=1}^{d} \left(\frac{1 + (-1)^{b_i} \chi_i(x)}{2} \right)$$

where $\chi_i(x) = 1$ if $x_i = 0$ and -1 if $x_i = 1$. Using Claim 2,

$$L_1(AND_{b_1,\ldots,b_d}) \le \prod_{i=1}^{d} L_1 \left(\frac{1 + (-1)^{b_i} \chi_i(x)}{2} \right) \le 1.$$

On the other hand, since $AND_{b_1,\ldots,b_d}(b_1,\ldots,b_d) = 1$, $L_1(AND) \geq 1$. Therefore $L_1(AND) = 1$.

Now consider a decision tree. A path of length $d-1$ from the root node to a leaf can be expressed as $b = (b_1,\ldots,b_d)$, where $b_1,\ldots,b_d$ are the assignment of the variables along the path. Therefore, if $AND_{b_1,\ldots,b_d}(x_1,\ldots,x_d) = 1$, the decision tree will output the value of the leaf. A decision tree $DT : \{0,1\}^d \to \{0,1\}$ can therefore be expressed as the sum of all the paths that returns 1 at the leaf. Let this set of paths be P. Since there are m leaves, $|P| \leq m$. By Claim 1 and $L_1(AND) = 1$, we have

$$L_1(DT) = L_1(\sum_{b \in P} AND_b) \leq m.$$

Hence the L_1 norm of a decision tree with m leaves is at most m, which is polynomial in n. By Corollary 1, polynomial sized decision tree can be approximated in polynomial time with queries with probability $1-\delta$.

References

[1] E. Kushilevitz and Y. Mansour. Learning decision trees using the Fourier spectrum. In *Proceedings of the twenty-third annual ACM symposium on Theory of computing*, pages 455–464, 1991.

[2] Y. Mansour. Learning boolean functions via the fourier transform. In V. Roychowdhury, K.-Y. Siu, and A. Orlitsky, editors, *Theoretical Advances in Neural Computation and Learning*. Kluwer, 1994.