6.858/18.428 Machine Learning	Lecture 21 : November 28, 1994
Lecturer: Ron Rivest	*Scribe: Andreas Argiriou*

Outline

- Learning decision trees with Fourier theory

In this lecture we see how we can apply Fourier techniques to learn decision trees, as described by Kushilevitz and Mansour in [1]. Such techniques have also been used in other problems, like DNF formulas.

21.1 Introduction

The type of learning game we need to consider now is special in the following ways:

1. The distribution is *uniform* and not arbitrary.
2. We want to learn *decision trees*.
3. There is no source of random examples.
4. The algorithm uses *membership queries*.

In the model of membership queries, the learner makes up the questions so that it may ask for the label of an example. With *equivalence queries*, on the other hand, the learner presents a hypothesis h and asks if h is the same as the target concept f. If $h \neq f$, the learner receives a counterexample as an answer. In this problem we essentially need statistical queries, but we can use membership queries to get information of the type a statistical query would give.

We will consider boolean functions with a slight modification: a boolean function now maps $\{0,1\}^n$ to $\{\pm 1\}$ instead of $\{0,1\}$ in the following way

$$x \in \{0,1\} \mapsto (-1)^x.$$

	x			
z	00	01	10	11
00	1	1	1	1
01	1	-1	1	-1
10	1	1	-1	-1
11	1	-1	-1	1

Table 21.1: The χ functions for $n = 2$.

In other words, negative examples map to $+1$, and positive examples map to -1.

The concept class being learned is the class $\mathcal{C}$ of boolean functions computed by *decision trees with linear operations*. In this model each node computes a sum (modulo 2) of a subset of the n variables and if the result is 1 branches to the right subtree, else branches to the left subtree. The output is the value of the leaf where we end up.

Learning this class in a passive model of learning is an open problem. In our new model, when given ϵ, δ we need to produce a hypothesis h such that $\Pr[h(x) \neq f(x)] \leq \epsilon$ with probability at least $1-\delta$ (where x is drawn according to the uniform distribution and f is the target function).

21.2 The orthonormal basis

The basis functions for the Fourier transform we will use are given by the following definition.

Definition 1 *For each $z \in \{0,1\}^n$, define the function $\chi_z : \{0,1\}^n \to \{\pm 1\}$ by*

$$\chi_z(x) = (-1)^{z \cdot x \bmod 2}$$

The value of the function depends on the exclusive or of those variables "selected" by z in the dot product. If $\sum_{i=1}^{n} z_i x_i \equiv 0 \bmod 2$ the output is $+1$, else -1. Table 21.1 shows the values of the χ functions if $n = 2$.

Our objective now is to define an inner product of functions such that

$$\langle \chi_{z_1}, \chi_{z_2} \rangle = \begin{cases} 0 & \text{if } z_1 \neq z_2 \\ 1 & \text{if } z_1 = z_2 \end{cases} \tag{21.1}$$

and such that a function f on $\{0,1\}^n$ can be uniquely expressed as

$$f = \sum_{z \in \{0,1\}^n} \hat{f}(z)\chi_z. \tag{21.2}$$

where $\hat{f}(z)$ are real constants.

The Fourier transform of f is the expansion of f as a linear combination of the χ_z's. Equations (21.1) and (21.2) together state that the χ functions form an **orthonormal basis**. Then the Fourier coefficients are given by

$$\hat{f}(z) = \langle f, \chi_z \rangle.$$

The method of using Fourier coefficients is helpful because now the objective will be constructing an algorithm that approximates a function f as a linear combination of the basis functions. The algorithm should find enough of the biggest Fourier coefficients to meet restrictions on the error.

To define the inner product, we use the concept of the expectation of f (over uniform distribution)

$$\mathrm{E}[f] = \frac{1}{2^n} \sum_{x \in \{0,1\}^n} f(x).$$

Definition 2 *For any functions $f, g : \{0,1\}^n \to \Re$, define the* inner product *of f, g as*

$$\langle f, g \rangle = E[fg] = \frac{1}{2^n} \sum_{x \in \{0,1\}^n} f(x)g(x)$$

and the $\mathcal{L}_2$ norm *of f as*

$$\|f\|_2 = \sqrt{\langle f, f \rangle}.$$

Note that for *boolean functions* $\|f\|_2 = 1$.

It is also clear from definition 1 that for all $z_1, z_2 \in \{0,1\}^n$,

$$\chi_{z_1}\chi_{z_2} = \chi_{z_1 \oplus z_2}$$

where $\oplus$ denotes exclusive or.

Using the above, one can see that definition 2 verifies equations (21.1), so that indeed the χ functions form an orthonormal basis.

Another useful relation is *Parseval's identity*:

$$\|f\|_2^2 = \sum_{z \in \{0,1\}^n} \hat{f}^2(z) \tag{21.3}$$

or for boolean functions

$$\sum_{z \in \{0,1\}^n} \hat{f}^2(z) = 1. \tag{21.4}$$

Finally, we can define the $\mathcal{L}_1$ *norm* of f, as follows:

Definition 3 *The* $\mathcal{L}_1$ norm *of* f *is*

$$\mathcal{L}_1(f) = \sum_{z \in \{0,1\}^n} |\hat{f}(z)|.$$

The $\mathcal{L}_1$ norm is a measure of how spread out the coefficients are and ranges from 1 to $2^{\frac{n}{2}}$.

21.3 Approximation by sparse functions

The following theorem will be proven later:

Theorem 1 *If the function* f *is represented by a decision tree with* m *nodes, then* $\mathcal{L}_1(f) \leq m$.

We will thus try to learn decision trees with a small number of nodes, since it is easier to approximate a function with a small $\mathcal{L}_1$ norm (intuitively, weight is not spread a lot and there are just a few big coefficients).

The approximation of the target function f will be another function g, real valued and not boolean. However

$$\text{sign}(g(x)) = \begin{cases} +1 & \text{if } g(x) \geq 0 \\ -1 & \text{otherwise} \end{cases}$$

is a boolean function. So we want $\text{sign}(g(x))$ to be a good approximation of f. We see now that this happens when g ϵ*-approximates* f *in the* $\mathcal{L}_2$ *norm.*

Definition 4 *We say that a function g ϵ-approximates f in $\mathcal{L}_2$ norm, when*

$$E[(f-g)^2] \leq \epsilon$$

.

Notice that when $\text{sign}(g(x)) \neq f(x)$, then $(f(x)-g(x))^2 \geq 1$ and hence $\Pr[f \neq \text{sign}(g)] \leq \text{E}[(f-g)^2]$. Thus we have the following lemma:

Lemma 1 *If g ϵ-approximates f in $\mathcal{L}_2$ norm, then $Pr[f \neq sign(g)] \leq \epsilon$.*

So, now the question is how many of the Fourier coefficients are needed to ϵ-approximate the target function. For this reason it is helpful to define the concept of a *t-sparse* function.

Definition 5 *The function g is* t-sparse *if it has at most t non-zero Fourier coefficients.*

As we mentioned earlier, we prefer approximations with large coefficients and the next lemma shows that this can be achieved.

Lemma 2 *If f can be ϵ-approximated (in $\mathcal{L}_2$ norm) by a t-sparse function g then f can be $\epsilon + O(\epsilon^2)$-approximated by a t-sparse function h whose non-zero coefficients are at least ϵ/t.*

Proof: Write $g(x) = \sum_{i=1}^{t} \hat{g}(z_i)\chi_{z_i}(x)$. Parseval's identity (21.3) gives $\text{E}[(f-g)^2] = \sum_z (\hat{f}(z) - \hat{g}(z))^2$ and this has a minimum when $\hat{g}(z_i) = \hat{f}(z_i), i = 1, \ldots, t$. Let $g'(x) = \sum_{i=1}^{t} \hat{f}(z_i)\chi_{z_i}(x)$. We can show that

$$h(x) = \sum_{\hat{f}(z_i) \geq \frac{\epsilon}{t}} \hat{f}(z_i)\chi_{z_i}(x)$$

is as desired. Indeed, $\text{E}[(f-h)^2] - \text{E}[(f-g)^2] \leq \text{E}[(f-h)^2] - \text{E}[(f-g')^2] = \sum_{\hat{f}(z_i) < \frac{\epsilon}{t}} \hat{f}^2(z_i) < t(\frac{\epsilon}{t})^2 = \text{O}(\epsilon^2)$. Since $\text{E}[(f-g)^2] \leq \epsilon$ we are done. □

A consequence of lemma 2 is that our algorithm can look only for the larger Fourier coefficients. However, the space is huge and this is not an easy task. What we can do is use a recursive procedure to implement tree search over the values of z and find the large coefficients (see Figure 21.1). The problem is solved if at each node we can say whether there is a large coefficient in the subtree. With a *perfect* test like that we would need $\text{O}(tn)$ work to find the t largest coefficients in a tree of depth n. We need only to figure out

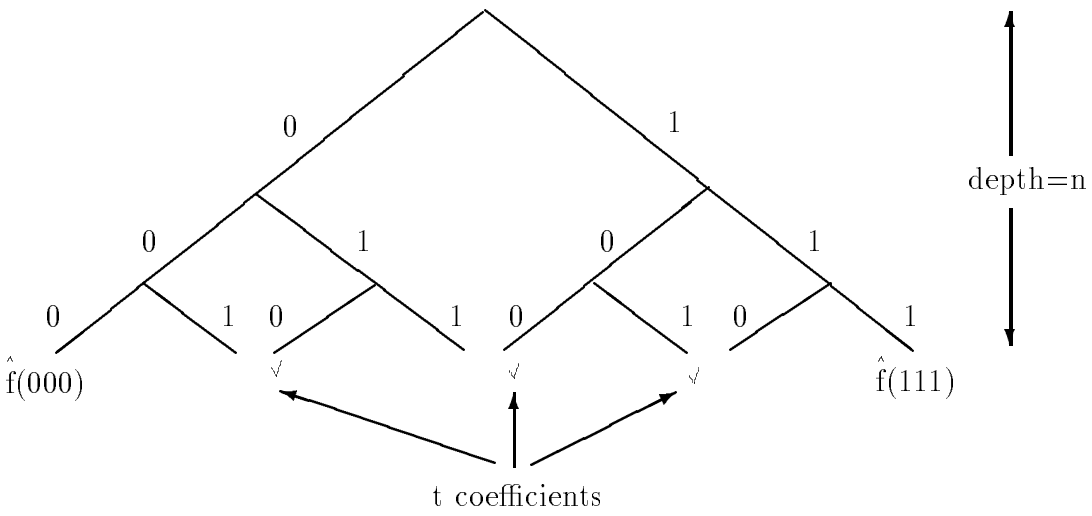

Figure 21.1: Search for large coefficients on the n-dimensional cube

- how to do the test
- how many coefficients to find.

The idea is to define a function that includes coefficients under a certain node.

Definition 6 *For each $\alpha \in \{0,1\}^k$, define the function $f_\alpha : \{0,1\}^{n-k} \to \Re$ by*

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

By estimating $\mathrm{E}[f_\alpha^2]$, which equals $\sum_{\beta \in \{0,1\}^{n-k}} \hat{f}^2(\alpha\beta)$ (Parseval), it will be possible to make a prediction about the size of the coefficients $\hat{f}(\alpha\beta)$. The following lemma will be necessary for the approximation of $\mathrm{E}[f_\alpha^2]$:

Lemma 3 *If $1 \leq k < n$ and $\alpha \in \{0,1\}^k$, then*

$$f_\alpha(x) = E_{y \in \{0,1\}^k}[f(yx)\chi_\alpha(y)].$$

Proof: It is easy to see from the definitions that if $z = z_1 z_2$ and $|z_1| = k$, then $\chi_z(yx) = \chi_{z_1}(y)\chi_{z_2}(x)$. So, we have

$$\mathrm{E}_y[f(yx)\chi_\alpha(y)] = \mathrm{E}_y[\sum_z \hat{f}(z)\chi_z(yx)\chi_\alpha(y)] = \mathrm{E}_y[\sum_{z_1}\sum_{z_2} \hat{f}(z_1 z_2)\chi_{z_1}(y)\chi_{z_2}(x)\chi_\alpha(y)]$$

$$= \sum_{z_1}\sum_{z_2} \hat{f}(z_1 z_2)\chi_{z_2}(x)\mathrm{E}_y[\chi_{z_1}(y)\chi_\alpha(y)].$$

Now from equations (21.1) we see that

$$\mathrm{E}_y[\chi_{z_1}(y)\chi_\alpha(y)] = \begin{cases} 0 & \text{if } z_1 \neq \alpha \\ 1 & \text{if } z_1 = \alpha \end{cases}$$

and therefore

$$\mathrm{E}_y[f(yx)\chi_\alpha(y)] = \sum_{z_2} \hat{f}(\alpha z_2)\chi_{z_2}(x) = f_\alpha(x)$$

□

The remaining issues are now the following:

- How to use an approximation technique to pick out large Fourier coefficients
- Showing the approximation is good enough
- Showing the algorithm applies to decision trees.

References

[1] Eyal Kushilevitz and Yishay Mansour. Learning Decision Trees using the Fourier Spectrum. In *Proceedings of the 23rd Annual ACM Symposium on Theory of Computing*, pages 455-464, 1991.