6.842 Randomness and Computation 2012-04-30

Lecture 21

Lecturer: Ronitt Rubinfeld *Scribe: Michael Forbes*

1 Overview

Today we will finish boosting, and sketch how this implies that every function that is weakly hard on average has a *hardcore set*, that is, a subset of the inputs where the function is strongly hard on average.

2 Boosting

We will first review the algorithm for boosting, as well as the notation and claims surrounding it. We begin with the algorithm.

- $\mathcal{D}_0 \leftarrow \mathcal{D}$
- Run the weak learner (WL) on $\mathcal{D}_0$ to get a hypothesis c_1
- Stage i (given $c_1, \ldots, c_i$) (for $i \leq O(1/(\gamma\epsilon)^2)$ steps)
  - Let $M_i(x) = \begin{cases} 1 & \text{if } Majority(c_1, \ldots, c_i)(x) \neq f(x) \\ 0 & \text{if } |\{j : c_j(x) = f(x)\}| - |\{j : c_j(x) \neq f(x)\}| \geq 1/(\gamma\epsilon) \\ 1-\alpha & \text{if } |\{j : c_j(x) = f(x)\}| - |\{j : c_j(x) \neq f(x)\}| = \alpha/(\gamma\epsilon) \end{cases}$
  - Let c_{i+1} be the result of the weak learner on the distribution $\mathcal{D}_i$

Recall that we defined the distribution $\mathcal{D}_i$ by $\mathcal{D}_{M_i}(x) = M_i(x)/|M_i|$, where $|M_i| = \sum_x M_i(x)$ is a normalizing factor. We also defined the advantage of a hypothesis on a given distribution to be $Adv_c(M) = \sum_x R_c(x)M(x)$, where

$$R_c(x) = \begin{cases} 1 & f(x) = c(x) \\ -1 & \text{else} \end{cases}$$

We also defines $N_i(x) = \sum_{1 \leq j \leq i} R_{c_i}(x)$. With these definitions, we can observe that

$$M_i(x) = \begin{cases} 1 & N_i(x) \leq 0 \\ 0 & N_i(x) \geq 1/(\gamma\epsilon) \\ 1 - \epsilon\gamma N_i(x) & \text{else} \end{cases}$$

We now recall claims proven in previous classes.

Claim 1 $\Pr_{x \in \mathcal{D}_M}[c(x) = f(x)] \geq 1/2 + \gamma/2$ *iff* $Adv_c(M) \geq \gamma|M|$

Claim 2 *If* $\Pr_{x \in \mathcal{D}_M}[c = f] \geq 1/2 + \gamma/2$ *and* $|M| \geq \epsilon 2^n$ *then* $Adv_c(M) \geq \gamma|M| = \gamma\epsilon 2^n$.

The remaining claim to prove is

Claim 3 $A_i(x) \leq 1/(\gamma\epsilon) + i \cdot \epsilon\gamma/2$

where $A_i(x) = \sum_{0 \le j \le i-1} R_{c_{j+1}}(x) M_j(x)$. Recall how we used this claim. We considered the $2^n \times i_0$ sized matrix, where the rows are indexed by $x \in \{0,1\}^n$ and the columns are indexed by stages in our boosting algorithm. The entry in position (x, j) is $R_{c_{j+1}}(x) M_j(x)$. It follows that the column sums are the advantages of the hypotheses we have generated so far. By the above claims, we have that each column sum is at least $\gamma \epsilon 2^n$, as the fact that each $|M_j| \ge \epsilon 2^n$ follows from our termination condition (if $|M_j| < \epsilon 2^n$ then we are done learning).

The row sums of this matrix are the values of $A_{i_0}(x)$. Now consider the total sum of the entries in the matrix. The sum is the same regardless of whether we sum by rows or columns. And, we have an upper bound on the sum of each row, and a lower bound on the sum of each column. As we know the number of rows, it follows that we can derive an upper bound on the number of columns in the matrix. We saw how this occurred last time, and how the above claim is all that remains to be proven.

To prove this claim, we will use an "elevator argument". The main principle is that when travelling in an elevator, the number of times you cross from floor k to floor $k+1$ must be within 1 of the number of times you cross from floor $k+1$ to floor k. This idea, not hard to prove, will be key in the following argument.

Proof [Proof of Claim] Consider the graph of $N_i(x)$ as i increases over time. Each time i increases, the sum $N_i = \sum_{i \le j \le i} R_{c_j}(x)$ changes by ± 1, corresponding to whether the latest hypothesis predicts $f(x)$ correctly. Thus, the plot of $N_j(x)$ will have a slope of ± 1 at each step. We wish to show that the average slope of $A_i(x)$ (which weights the terms with $M_j(x)$) is much less than 1.

To do this, we will pair occurrences of "$N_j(x) = k$ and $N_{j+1}(x) = k+1$" and "$N_{j'}(x) = k+1$ and $N_{j'+1}(x) = k$". Specifically, consider the contribution of the j and j' term to $A_i(x)$, seen by

$$R_{c_{j+1}}(x) M_j(x) + R_{c_{j'+1}}(x) M_{j'}(x)$$

Note that $R_{c_{j+1}}(x) = 1$ and $R_{c_{j'+1}}(x) = -1$ by choice of j — as $R_{c_{j+1}}(x) = N_{j+1}(x) - N_j(x)$. We now have that

$$\begin{aligned} R_{c_{j+1}}(x) M_j(x) + R_{c_{j'+1}}(x) M_{j'}(x) &= M_j(x) - M_{j'}(x) \\ &= \begin{cases} 0 & \text{if } k \le -1, \text{ as } M_j(x) = M_{j+1}(x) = 1 \\ 0 & \text{if } k \ge 1/(\gamma\epsilon), \text{ as } M_j(x) = M_{j+1}(x) = 0 \\ \epsilon\gamma & \text{as } M_j(x) = 1 - \epsilon\gamma N_j(x) = 1 - \epsilon\gamma k \text{ and} \\ & \qquad M_{j'}(x) = 1 - \epsilon\gamma N_{j'}(x) = 1 - \epsilon\gamma(k+1) \end{cases} \end{aligned}$$

So we have established that each such pair contributes at most $\epsilon\gamma$ to $N_i(x)$. As there can be at most $i/2$ such pairs, this gives us the $\epsilon\gamma/2$ term.

However, we note that we may not be able to pair all terms in $N_i(x)$. We can do the pairing such that all unpaired terms occur at the end, and as such all go in the same direction. If all such $R_{c_j}(x)$'s are negative, then as $M_j(x) \ge 0$ always, such terms do not increase are sum and thus we can ignore them for the purpose of an upper bound. If all such $R_{c_j}(x)$'s are 1, then we can observe that $N_i(x)$ is bounded by $1/(\epsilon\gamma)$, for if $N_i(x)$ tries to exceed this, the $M_j(x)$ terms become zero. Thus, in the summation $A_i(x) = \sum_j R_{c_{j+1}}(x) M_j(x)$, there are at most $N_i(x)$ unpaired terms that are nonzero. Each such non-zero unpaired term is at most 1, so we can only have at most $1/(\epsilon\gamma)$ contribution in this unmatched section. Thus, we have covered all terms in the summation of A_i, and thus get the desired bound. ■

So having completed this claim, the work from last lecture shows the correctness of the boosting algorithm.

3 Hardcore Sets

We will now apply the boosting algorithm to understand functions that are hard to compute. Our measure of hardness will be the size of circuit needed to compute the function, where our computation is done in an average-case sense. That is, we consider computing functions such that the probability (over the input) that we error is bounded. In this model, we can say a function is very hard to learn if all circuits of a certain size cannot compute the function with probability better than $1/2 + \epsilon$. A function is slightly hard to compute if this probability is bounded by $1 - \delta$. The goal of this section (and next lecture) is to try to amplify hardness: transform a function slightly hard to compute into one that is very hard to compute.

The relation of this goal to boosting can be seen as follows. To say that a function is weakly learnable means that we can compute a hypothesis that gets the function correct with probability at least $1/2 + \epsilon$. That is, if we also promise that the hypothesis has a small circuit, then we see that weak learning of a function implies that the function is *not* very hard to compute. Correspondingly, to strongly learn a function means to create a hypothesis that agrees with the function on at least $1 - \delta$ fraction of the inputs. Thus, strong learnability implies the function is *not* slightly hard to compute.

Boosting shows that if a function is weakly learnable (on all distributions) then it is strongly learnable. With the above connection to hardness of computation, we can see the contrapositive: if a function is slightly hard to compute then it is very hard to compute (on some distribution). We now make these ideas formal.

Definition 4 *f is δ-**hard on a distribution** $\mathcal{D}$ **for size** g **circuits** if for any boolean circuit $\mathcal{C}$ of size at most g has $\Pr_{x \in \mathcal{D}}[\mathcal{C}(x) = f(x)] \leq 1 - \delta$*

We now define the notion of being very hard to compute.

Definition 5 *Let M be a measure. If for all circuits $\mathcal{C}$ of size at most g, $\Pr_{x \in \mathcal{X}}[\mathcal{C}(x) = f(x)]$ then f is ϵ-**hardcore (ϵ-h.c.) on** M **for size** g **circuits**.*

Ideally, in the above we would have that M is a uniform distribution over a set S, and in that case we would say f is hardcore over that set. We now have the following theorem, due to Impagliazzo.

Theorem 6 *Suppose f is δ-hard for size g circuits on the uniform distribution. Then for any $\epsilon > 0$ there is a measure M such that $|M|/2^n \geq \delta$ and so f is ϵ-hardcore on M for size $(\epsilon\delta)^2 g/4$ circuits.*

Proof We follow the contrapositive of boosting. Thus, if there is no such hardcore distribution then we can weakly learn each distribution with some small circuit, with advantage ϵ. The boosting result then shows that we can strongly learn the function over the uniform distribution by taking a majority of these smaller circuits, and there are $1/(\epsilon\delta)^2$ such circuits. As taking the majority can be done in a linear size circuit, the resulting strong learner would have size $< g$ as a result, contradicting our hypothesis. Thus, it must be that there is a hardcore distribution. ■

The above result shows there is a hardcore distribution, but not that there is a hardcore set. The next theorem gives such a set.

Theorem 7 *Let M be an $\epsilon/2$-hardcore distribution for f for size g circuits. Then there exists a ϵ-hardcore set S for f on size g circuits, and $|S| \geq \delta 2^n$.*

Proof This can be seen as a randomized rounding procedure. Specifically, we randomly take each element in $\{\pm 1\}^n$ with its probability as in M. One can see that the advantage of a circuit on the resulting set is a sum of independent random variables, and by applying the Chernoff bound, is close to its expectation. As the expected hardness is the hardness of M, we get the expected hardness with good probability, and thus such a set exists. ■