Today, we will show how a weak PAC (Probably Approximate Correct) learning algorithm can be boosted to a strong one. This result has far-reaching implications beyond computational learning theory.

1 Introduction

Definition 1 *An algorithm* A *("strongly") PAC learns a concept class* $\mathcal{F}$ *if* $\forall f \in \mathcal{F}, \forall$*distribution* $\mathcal{D}, \forall \epsilon, \delta > 0$*, with probability* $\geq 1 - \delta$*, given examples* $\in \mathcal{D}$ *labelled according to* f*,* A *outputs* h *such that*

$$\Pr_{\mathcal{D}}[h(x) \neq f(x)] \leq \epsilon. \tag{1}$$

Remark

- ϵ is called the accuracy parameter, and δ is called the security parameter or the failure probability.
- Parameter δ is inconsequential here: As long as it is reasonably small, we can drive it down to an arbitrarily small value. (Refer to Question 2 in Homework 2.) For this reason, we shall be omitting this parameter from here onwards.
- Hypothesis h does not necessarily have to be in concept class $\mathcal{F}$. If it does, then the model is called a proper learning model.
- Distribution $\mathcal{D}$ does not have to be uniform either. It can be any distribution, and therefore, the algorithm is distribution-free.

Definition 2 *An algorithm* WL **weakly** *PAC learns a concept class* $\mathcal{F}$ *if* $\forall f \in \mathcal{F}, \forall$*distribution* $\mathcal{D}, \exists \gamma > \mathbf{0}, \forall \delta > 0$*, with probability* $\geq 1 - \delta$*, given examples* $\in \mathcal{D}$ *labelled according to* f*,* WL *outputs* c *such that*

$$\Pr_{\mathcal{D}}[c(x) \neq f(x)] \leq \frac{\mathbf{1}}{\mathbf{2}} - \frac{\gamma}{\mathbf{2}}. \tag{2}$$

Definition 3 *The term* $\frac{\gamma}{2}$ *is called the* advantage *of* WL.

Remark Here, we assume that the concept class $\mathcal{F}$ is Boolean, and so hypothesis c can be just doing slightly better than one of the two constant function. Also, note that WL must be able to output such c *for all distributions*, not just, say, the uniform distribution.

Theorem 1 *If* $\mathcal{F}$ *can be weakly learned, then* $\mathcal{F}$ *can be strongly learned.*

2 A Boosting Algorithm

In this section, we present an algorithm which boosts a weak learner to a strong one, hence proving the above theorem. There are several variants the algorithm, but they revolve around the same idea.

2.1 The Intuition

Suppose a weaker learner is only 51% accurate. We can first learn a weak hypothesis, filter away examples which are correctly classified, and then call the weak learner on the remaining 49% of the data. To increase the collective coverage of the hypotheses, we can repeat alternating between the filtering and the learning steps. A natural question is: Given an unseen example, which hypothesis shall we use? The basic idea of the boosting algorithm is to construct a filtering mechanism so that the majority vote of the collective hypotheses works out.

2.2 The Algorithm

Given a weak learner WL, a distribution $\mathcal{D}$, a concept f, parameters ϵ and γ, the boosting algorithm Boost is the following: (We illustrate the case for the uniform distribution. Note that the algorithm can be easily modified to be distribution-free although we are not showing it here.)

Boost(WL, $\mathcal{D}, f, \epsilon, \gamma$)
    initialize distribution $\mathcal{D}_0 = \mathcal{D} = \mathcal{U}$
      Use weak learner WL to generate weak hypothesis c_1 such that $\Pr_{\mathcal{D}_0}[f(x) = c_1(x)] \geq \frac{1}{2} + \frac{\gamma}{2}$
      Set current hypothesis $h = c_1$
    for $i = 1$ **to** T
      (1) Construct $\mathcal{D}_i$ with the filtering mechanism Filter($\mathcal{D}, h = \mathrm{maj}(c_1, \ldots, c_i), f, \epsilon, \gamma$)
      (2) Run WL on $\mathcal{D}_i$ to get weak hypothesis c_{i+1} such that $\Pr_{\mathcal{D}_i}[f(x) = c_{i+1}(x)] \geq \frac{1}{2} + \frac{\gamma}{2}$
      (3) Update $h = \mathrm{maj}(c_1, \ldots, c_{i+1})$
    return $h = \mathrm{maj}(c_1, \ldots, c_{T+1})$ such that $\Pr_{\mathcal{D}}[f(x) = h(x)] \geq 1 - \epsilon$

Filter($\mathcal{D}, h, f, \epsilon, \gamma$)
    do until we have the desired number of examples
      Draw an example x from $\mathcal{D}$
      if $h = \mathrm{maj}(c_1, \ldots, c_i)$ is wrong on x, **then** keep x
      else if # of c_i's right - # of c_i's wrong $> \frac{1}{\epsilon\gamma}$, **then** throw x away
      else, say # of c_i's right - # of c_i's wrong $= \frac{\alpha}{\epsilon\gamma}$, **then** keep x with probability $1 - \alpha$
    return all retained examples $\mathcal{D}_{i+1}$

The algorithm assumes the weak learner never fails. (Recall that we can easily decrease the probability of failure.) Before giving the bound T on the maximum number of iterations needed, we first introduce some notations.

3 Preliminaries

Here are some notations and their properties:

1. $R_c(x) = \begin{cases} +1 & \text{if } f(x) = c(x) \\ -1 & \text{o.w.} \end{cases}$ gives +1 if (weak) hypothesis c is right on example x
2. $N_i(x) = \sum_{1 \leq j \leq i} R_{c_j}(x)$ is the number of right c's exceeding the wrong ones
3. $M_i(x) = \begin{cases} 1 & \text{if } N_i(x) \leq 0 \\ 0 & \text{if } N_i(x) \geq \frac{1}{\epsilon\gamma} \\ 1 - \epsilon\gamma N_i(x) & \text{o.w.} \end{cases}$
   is a "measure" which upper bounds the error of hypothesis $h = \mathrm{maj}(c_1, \ldots, c_i)$ on example x.
4. $\mu(M) = \frac{1}{2^n} \sum_x M(x) \geq \mathrm{error}(h) \geq \epsilon$ is the "mean" of M. It upper bounds the error of h and therefore also ϵ. (We actually estimate $\mu(M)$ by sampling in each iteration and stop if $\mu(M) < \epsilon$.)
5. $|M| = \sum_x M(x) = 2^n \mu(M)$ is the total "mass" of all examples according to "measure" M.
6. $D_M(x) = \frac{M(x)}{|M|}$ is a distribution over x given M. (Note that we obtain $\mathcal{D}_i$ with c_i, and so $D_{M_i} = \mathcal{D}_i$.)
7. $\mathrm{Adv}_c(M) = \sum_x R_c(x) M(x)$ is the advantage of c on M. (Random guessing gives 0.)
8. $\mathrm{Adv}_c(M) \geq \gamma|M|$ iff $\Pr_{x \in D_M}[c(x) = f(x)] \geq \frac{1}{2} + \frac{\gamma}{2}$
9. If $\Pr_{x \in D_M}[c(x) = f(x)] \geq \frac{1}{2} + \frac{\gamma}{2}$ and $\mu(M) \geq \epsilon$, then $\mathrm{Adv}_c(M) \geq_{(8)} \gamma|M| = \gamma 2^n \mu(M) \geq_{(4)} \gamma 2^n \epsilon$

4 Convergence Proof

Claim 2 $A_i(x) = \sum_{0 \le j \le i-1} R_{c_{j+1}}(x) M_j(x) < \frac{1}{\epsilon\gamma} + 0.5\epsilon\gamma i$

Before proving this claim, we first use it to bound the maximum number of iterations required by the boosting algorithm. Hence, if a concept can be weakly PAC learned, then it can be ("strongly") PAC learned.

Claim 3 *The maximum number of iterations required by the boosting algorithm is* $\le \frac{2}{\gamma^2\epsilon^2}$.

Proof We prove the claim by showing that assuming the algorithm does not stop after $\frac{2}{\gamma^2\epsilon^2}$ iterations leads to a contradiction. Suppose the algorithm continues to run after iteration $i_0 > \frac{2}{(\epsilon\gamma)^2}$ (i.e. $\mu(M_i) \ge \epsilon$), a lower bound can be derived as follows:

$$\begin{aligned} \sum_x A_{i_0+1} &= \sum_x \sum_{0 \le j \le i_0} R_{c_{j+1}}(x) M_j(x) && (3) \\ &= \sum_{0 \le j \le i_0} \underbrace{\sum_x R_{c_{j+1}}(x) M_j(x)}_{Adv_{cj+1}(M_j(x))} && (4) \\ &\ge (i_0+1)\gamma 2^n \epsilon \qquad \text{(using property 9 in section 3)} && (5) \end{aligned}$$

Using Claim 2 leads to an upper bound:

$$\begin{aligned} \sum_x A_{i_0+1} &< \sum_x \left(\frac{1}{\epsilon\gamma} + 0.5\epsilon\gamma i_0\right) && (6) \\ &= 2^n\left(\frac{1}{\epsilon\gamma} + 0.5\epsilon\gamma i_0\right) && (7) \end{aligned}$$

Using both bounds, $(i_0+1)\gamma 2^n \epsilon \le \sum_x A_{i_0+1}(x) < 2^n(\frac{1}{\epsilon\gamma} + 0.5\epsilon\gamma i_0) \Rightarrow i_0 < \frac{2}{\gamma^2\epsilon^2}$, we arrive at a contradiction. So, the algorithm must run for $\frac{2}{\gamma^2\epsilon^2}$ iterations or less. ■

Fact 4 (The Elevator Argument) *If one rides an elevator from the ground floor, then one ascends from the* k*-th to the* $(k+1)$*-th floor at most* 1 *more time than one descends from the* $(k+1)$*-th to the* k*-th floor. (Analogous argument holds when traveling from the ground floor to basements.)*

Proof of Claim 2: The process of adding each term of $N_i(x)$ corresponds to an elevator ride with $R_{c_j}(x)$ dictating the direction and partial sum $N_j(x)$ denoting the current level. The plan is to first match pairs of $R_{cj+1}(x)M_j(x)$ terms and obtain an upper bound of their sum using properties of function $M_j(x)$. As for the unmatched pairs, we can bound the number of them (using the Elevator Argument) and also their sums. And so, an upper bound for $A_i(x)$ can be obtained.

Matched Pairs

For each $k \ge 0$,
  match j such that $N_j(x) = k$ and $N_{j+1}(x) = k+1$
  with j' such that $N_{j'}(x) = k+1$ and $N_{j'+1}(x) = k$

For each matched pair of terms corresponding to indices $a = j, b = j'$, the sum is
$\underbrace{R_{c_{a+1}}(x)}_{+1} \underbrace{M_a(x)}_{N_a(x)=k} + \underbrace{R_{c_{b+1}}(x)}_{-1} \underbrace{M_b(x)}_{N_b(x)=k+1} = M_a(x) - M_b(x)$.

If $0 \leq k \leq \frac{1}{\epsilon\gamma}$ or $0 \leq k+1 \leq \frac{1}{\epsilon\gamma}$, then

$M_a(x) - M_b(x) \leq \epsilon\gamma$ (because $\frac{M_b(x)-M_a(x)}{k+1-k}$ is the slope of $M_i(x)$ which is $\geq -\epsilon\gamma$),

else

$M_a(x) - M_b(x) = 0$.

We can arrive at the same result for $k < 0$. Therefore, the total contribution of matched pairs is $\leq 0.5\epsilon\gamma i$ (because $A_i(x)$ has i terms).

Unmatched Terms Notice that unmatched terms are in the "same direction", i.e. all $R_{c_j}(x)$'s are either negative or positive. Suppose all $R_{c_j}(x)$'s are negative (i.e. -1), then their contribution to the sum is negative (because each term becomes $-M_j(x) \leq 0$). So they do not loosen the upper bound we already derived from matched pairs.

Suppose all $R_{c_j}(x)$'s are positive (i.e. $+1$). Then $N_j(x) \geq 0$, and so each term is $M_j(x) = 1 - \epsilon\gamma N_j(x)$ if $N_j(x) \in [0, \frac{1}{\epsilon\gamma}]$ and 0 otherwise. The Elevator Lemma tells us that there is at most one unmatched $N_j(x)$ for each integer value in the interval $[0, \frac{1}{\epsilon\gamma}]$, and so the total contribution of them (sum of a arithmetic series from 0 to 1 with $\frac{1}{\epsilon\gamma}$ terms) is $\leq \frac{1}{2\epsilon\gamma} < \frac{1}{\epsilon\gamma}$

Summing up the total contribution from both matched and unmatched terms gives $A_i(x) < \frac{1}{\epsilon\gamma} + 0.5\epsilon\gamma i$. ■