Lecture 5: Selected Topics — PAC Learning & Boosting

In this lecture we study one popular approach — Boosting technique — for designing efficient learning algorithms in the PAC learning framework. We studied in the previous lecture the notion of "Occam Razor", a standard goal for designing reasonable learning algorithms; that is, for a given set of examples, compute a "succinct" hypothesis that is consistent with these examples. Unfortunately, when we try for this goal, we would often end up with a serious computational problem; that is, this goal is often NP-complete, one of the hardest problems in NP, and we may not hope a fast algorithm that achieves this goal perfectly.

> **An Important Message from O.W:** Even in this case, you want/need to achieve your goal and design somewhat efficient algorithms. In fact, this may not be hopeless. Note that the NP-completeness notion is for the worst-case complexity measure; even if the problem is hard in the worst-case, it may still be easier for many instances and it may be good enough for your purpose! But then how can we push our formal investigation? The success story of AdaBoost would give us a good hint for breaking such an algorithmic barrier when you face a similar situation.

1 Preliminaries for Boosting Technique

First recall that the following is the learning goal of a PAC learning algorithm A for some concept $\mathcal{C}$.

$$\begin{array}{l}
\forall \epsilon, \delta,\ 0 < \epsilon, \delta < 1, \quad \forall n \geq 1, \\
\exists m \geq 0 \text{ (which is determined by } A \text{ from } \epsilon, \delta, n), \\
\forall D_* \text{ (distribution over } \{+1, -1\}^n), \quad \forall f_* \in \mathcal{C} \\
\Pr_{S:D_*^m}\left[\begin{array}{c} A \text{ given } S \text{ yields some } h \text{ satisfying} \\ (*) \quad \Pr_{\boldsymbol{x}:D_*}[\ f_*(\boldsymbol{x}) \neq h(\boldsymbol{x})\ ] \ \leq\ \epsilon \end{array}\right] \ \geq\ 1 - \delta.
\end{array}$$

If we can design an algorithm achieving this goal whose time complexity is bounded by polynomial in $1/\epsilon$, $1/\delta$, and n, then we say that a concept class $\mathcal{C}$ is *polynomial-time PAC learnable*.

A standard way to design such a learning algorithm is to construct A so that it yields a "succinct" hypothesis that is *consistent* with all examples in a given sample S. Then, based on the degree of "succinctness", we estimate the number m of examples that is sufficient for achieving the goal for given parameters ϵ and δ. (It is often the case in practice, m is much larger than the number of data we have. In this case, we would simply use all available examples.)

Valiant, the founder of the PAC learning framework, somehow considered the following weaker learning goal.

$$\begin{array}{l}
\exists \gamma, 0 < \gamma < 1/2,\ \delta << 1, \quad \forall n \geq 1, \\
\exists m \geq 0 \text{ (which is determined by } B \text{ from } n), \\
\forall D_* \text{ (distribution over } \{+1,-1\}^n), \quad \forall f_* \in \mathcal{C} \\
\Pr_{S:D_*^m}\left[\begin{array}{c} B \text{ given } S \text{ yields some } h \text{ satisfying} \\ (*) \quad \Pr_{\boldsymbol{x}:D_*}[\ f_*(\boldsymbol{x}) = h(\boldsymbol{x})\ ] \ \geq\ \dfrac{1}{2} + \gamma \end{array}\right] \ \geq\ 1 - \delta.
\end{array}$$

That is, B (almost always) produces a hypothesis that is slightly better than the "random guess." Here γ is called an *advantage* and an algorithm like this B is called a *weak learner*. We say that a concept class $\mathcal{C}$ is *weak learnable* if we have a weak learner for $\mathcal{C}$ whose time complexity is bounded by some polynomial in n.

Valiant asked (I guess from his theoretical interest) whether the weak learnability implies the polynomial-time PAC learnability. It turned out that this theoretical question lead to one of the important learning algorithms in Machine Learning — AdaBoost.

2 AdaBoost

A *boosting technique* or a boosting algorithm is an algorithm that make use of a given weak learner B to design some learning algorithm with much higher accuracy. Below is the general outline of boosting algorithms.

target concept: a Boolean function f_* over n Boolean attributes in $\mathcal{C}$;
distribution: a distribution D_* over $\{0,1\}^n$;
input: parameters ϵ, δ, and n;
output:some hypothesis n approximating f_*;
begin
  $t = 1$; $D_1 = D_*$;
  repeat {
    $h_t =$ a hypothesis for f_* that B yields under D_t;
    $\gamma_t =$ the advantage of h_t under D_t;
    $\alpha_t =$ the degree of importance of h_t;
    define the (current) *combined hypothesis* f_t as follows:

$$f_t(\boldsymbol{x}) \ =\ \text{sign}\left(\sum_{1 \leq i \leq t} \alpha_i h_i(\boldsymbol{x})\right);$$

    if $\Pr_{\boldsymbol{a}:D_*}[\, f_*(\boldsymbol{a}) \neq f_t(\boldsymbol{a})\,] \ <\ \epsilon$ **then break**;
    $D_{t+1} =$ a new distribution made from D_t and h_t;
    $t = t + 1$;
  }
  output the obtained f_t;
end.

Each repeat iteration of the above outline is called a *boosting step*. A combined hypothesis f_t is often called a *weighted majority vote hypothesis*.

An actual boosting algorithm is determined by defining α_t and D_t appropriately. Here we introduce one of the famous boosting techniques — AdaBoost — which is defined as follows:

$$\begin{aligned} &\alpha_t \;=\; \ln \beta_t^{-1}, \quad w_t(\boldsymbol{x}) \;=\; D_*(\boldsymbol{x}) \cdot \prod_{1 \le i \le t-1} \beta_i^{h_i(\boldsymbol{x}) \cdot f_*(\boldsymbol{x})}, \quad D_t(\boldsymbol{x}) \;=\; \frac{w_t(\boldsymbol{x})}{W_t}. \\ &\text{Where} \;\; \beta_t \;=\; \sqrt{\frac{1-2\gamma_t}{1+2\gamma_t}}, \quad W_t \;=\; \sum_{\boldsymbol{x} \in \{+1,-1\}^n} w_t(\boldsymbol{x}). \end{aligned} \tag{1}$$

Formally speaking, a boosting technique is called a *boosting algorithm* if we prove its convergence. For example, the key property of AdaBoost is the following convergence theorem.

Theorem 1. *For any concept class $\mathcal{C}$, suppose that we have a weak learner B for $\mathcal{C}$. For any target concept $f_* \in \mathcal{C}$ and any target distribution D_*, consider the execution of AdaBoost by using B for t repeat iterations, and for any i, $1 \le i \le t$, let γ_i denote the advantage of the weak hypothesis h_i that B produces at each boosting step. Then we have*

$$\Pr_{\boldsymbol{x}:D_*}[\, f_*(\boldsymbol{x}) \neq f_t(\boldsymbol{x}) \,] \;<\; \prod_{1 \le i \le t} \sqrt{1-4\gamma_i^2}. \tag{2}$$

Example 1. To appreciate this convergence property, let us assume that $\gamma_i \ge 1/10$ at all iterations; that is, every weak hypothesis is just 10% better than the random guess (i.e., 60% accuracy). In this case, we have

$$\sqrt{1-4\gamma_i^2} \;\le\; \sqrt{1-\frac{1}{25}} \;=\; \frac{\sqrt{24}}{5} \;\le\; \frac{49}{50},$$

and since $(49/50)^{200} \le 0.02$, we can guarantee that the error of f_{200} is less than 2%. □

We can use this AdaBoost for obtaining a hypothesis that is consistent with a given sample S. Let m denote the number of examples in S. We can simply use AdaBoost with $D_*(\boldsymbol{x}) = 1/m$ for all examples $\boldsymbol{x}$ in S and $D_*(\boldsymbol{x}) = 0$ for the other $\boldsymbol{x}$'s to obtain a combined hypothesis f_t whose error probability is less than $1/m$ w.r.t. D_*; this f_t must be consistent with S.

For this application, the following update formula for w_t would be useful to speed up each boosting step.

$$w_{t+1}(\boldsymbol{x}) \;=\; \begin{cases} w_t(\boldsymbol{x}) \cdot \beta_t, & \text{if } h_t(\boldsymbol{x}) = f_*(\boldsymbol{x}), \text{ and} \\ w_t(\boldsymbol{x}) \cdot (1/\beta_t), & \text{if } h_t(\boldsymbol{x}) \neq f_*(\boldsymbol{x}). \end{cases} \tag{3}$$

Then what about a weak learner? How can we design a week learner? In practice we would use some simple hypothesis class for weak learner's hypothesis class. That is, we consider a weak learner that produces a very simple hypothesis, most typically, a

hypothesis that gives a decision based on a single attribute. In this case it is easy to select the best hypothesis (under the current distribution) by going through all possible hypotheses. Since the weak learning goal is much easier, we may be able to expect that even such a simple weak learner can achieve the goal.

> **An Important Message from O.W:** Why does AdaBoost work? When have we gone beyond the algorithmic barrier? Well, we assumed that a good weak learner exists! It is quite optimistic to assume some weak learning algorithm (in particular, the simple one stated in the above) can always give us a sufficient hypothesis (even for the weak learning condition). But maybe we can assume this up to a certain point. Boosting techniques are designed under such assumption and they are formally analyzed and guaranteed to work so long as this assumption holds.
>
> We may be able to use this approach! Even though you cannot prove/solve your target problem completely, by separating some part that is hard to analyze as a reasonable assumption, you may be able to analyze things precisely, which may lead to a new and very useful technique!

Homework assignment from this lecture

Solve one of the following problems. (Q3.1 was given in the previous lecture.)

Q3.2. Implement AdaBoost and obtain a good hypothesis for the mushroom data. The data and a sample program can be obtained from

`http://www.is.titech.ac.jp/~watanabe/class/boost/`

Use only `mushroomB5000.txt` for creating your hypothesis and check its performance with `mushroomB3000.txt`.

Q3.3. The proof of Theorem 1 is stated below. From this proof, it is not so difficult to see the reason why β_t is defined as (1). Explain this reason.

Appendix: Proof of Theorem 1

We first show that the error probability of f_t is at most W_{t+1}. For any instance $\boldsymbol{x}$, we have

$$
\begin{aligned}
& f_*(\boldsymbol{x}) \neq f_t(\boldsymbol{x}) \\
\iff & \sum_{i:f_*(\boldsymbol{x})=h_i(\boldsymbol{x})} \log \beta_i^{-1} \;\leq\; \sum_{i:f_*(\boldsymbol{x})\neq h_i(\boldsymbol{x})} \log \beta_i^{-1} \\
\iff & \prod_{i:f_*(\boldsymbol{x})=h_i(\boldsymbol{x})} \beta_i^{-1} \;\leq\; \prod_{i:f_*(\boldsymbol{x})\neq h_i(\boldsymbol{x})} \beta_i^{-1} \\
\iff & 1 \;\leq\; \frac{\prod_{i:f_*(\boldsymbol{x})=h_i(\boldsymbol{x})} \beta_i}{\prod_{i:f_*(\boldsymbol{x})\neq h_i(\boldsymbol{x})} \beta_i} \;=\; \frac{\prod_{i:f_*(\boldsymbol{x})\cdot h_i(\boldsymbol{x})=+1} \beta_i}{\prod_{i:f_*(\boldsymbol{x})\cdot h_i(\boldsymbol{x})=-1} \beta_i} \;=\; \prod_{1\leq i\leq t} \beta_i^{f_*(\boldsymbol{x})\cdot h_i(\boldsymbol{x})}.
\end{aligned}
$$

By using this characterization, we can bound the error probability of f_t as follows:

$$\begin{aligned} W_{t+1} &= \sum_{\boldsymbol{x}} w_{t+1}(\boldsymbol{x}) \;\geq\; \sum_{\boldsymbol{x}:f_*(\boldsymbol{x})\neq f_t(\boldsymbol{x})} w_{t+1}(\boldsymbol{x}) \\ &= \sum_{\boldsymbol{x}:f_*(\boldsymbol{x})\neq f_t(\boldsymbol{x})} D_*(\boldsymbol{x}) \cdot \prod_{1\leq i\leq t} \beta_i^{h_i(\boldsymbol{x})\cdot f_*(\boldsymbol{x})} \\ &\geq \sum_{\boldsymbol{x}:f_*(\boldsymbol{x})\neq f_t(\boldsymbol{x})} D_*(\boldsymbol{x}) \;=\; \Pr_{\boldsymbol{x}:D_*}[\, f_*(\boldsymbol{x}) \neq f_t(\boldsymbol{x})\,]. \end{aligned}$$

Next we analyze how W_{t+1} gets decreased. For this we modify the definition of W_{t+1} by

$$\begin{aligned} W_{t+1} &= \sum_{\boldsymbol{x}} w_{t+1}(\boldsymbol{x}) \;=\; \sum_{\boldsymbol{x}} w_t(\boldsymbol{x}) \cdot \beta_t^{h_t(\boldsymbol{x})\cdot f_*(\boldsymbol{x})} \\ &= \sum_{\boldsymbol{x}:f_*(\boldsymbol{x})\neq h_t(\boldsymbol{x})} w_t(\boldsymbol{x}) \cdot \beta_t + \sum_{\boldsymbol{x}:f_*(\boldsymbol{x})\neq h_t(\boldsymbol{x})} w_t(\boldsymbol{x}) \cdot \beta_t^{-1} \\ &= \sum_{\boldsymbol{x}:f_*(\boldsymbol{x})\neq h_t(\boldsymbol{x})} (W_t D_t(\boldsymbol{x})) \cdot \beta_t + \sum_{\boldsymbol{x}:f_*(\boldsymbol{x})\neq h_t(\boldsymbol{x})} (W_t D_t(\boldsymbol{x})) \cdot \beta_t^{-1} \\ &= W_t\beta_t \cdot \left(\sum_{\boldsymbol{x}:f_*(\boldsymbol{x})\neq h_t(\boldsymbol{x})} D_t(\boldsymbol{x}) \right) + W_t\beta_t^{-1} \cdot \left(\sum_{\boldsymbol{x}:f_*(\boldsymbol{x})\neq h_t(\boldsymbol{x})} D_t(\boldsymbol{x}) \right) \\ &= W_t\beta_t \cdot \Pr_{\boldsymbol{x}:D_t}[f_*(\boldsymbol{x}) \neq h_t(\boldsymbol{x})] + W_t\beta_t^{-1} \cdot \Pr_{\boldsymbol{x}:D_t}[f_*(\boldsymbol{x}) = h_t(\boldsymbol{x})]. \end{aligned}$$

where we used the recurrence formula (3).

Then by definition, we have $\Pr_{\boldsymbol{x}:D_t}[f_*(\boldsymbol{x}) = h_t(\boldsymbol{x})] = 1/2 + \gamma_t$; hence, the error probability is $\Pr_{\boldsymbol{x}:D_t}[f_*(\boldsymbol{x}) \neq h_t(\boldsymbol{x})] = 1 - (1/2 + \gamma_t) = 1/2 - \gamma_t$. Therefore,

$$\begin{aligned} W_{t+1} &= W_t\beta_t \cdot \left(\frac{1}{2} - \gamma_t\right) + W_t\beta_t^{-1} \cdot \left(\frac{1}{2} + \gamma_t\right) \\ &= W_t \cdot \frac{\beta_t \cdot (1 - 2\gamma_t) + \beta_t^{-1}(1 + 2\gamma_t)}{2} \\ &= W_t \cdot \frac{2\sqrt{(1 - 2\gamma_t)(1 + 2\gamma_t)}}{2} \;=\; W_t \cdot \sqrt{1 - 4\gamma_t^2}. \end{aligned}$$

From the discussion above, the theorem is immediate from this bound.