1 Prototypical Concept Learning

1.1 Basic set up

- **The hypothesis space:** consider our training instances to be $S \times \{0, 1\}$ – including both positive and negative examples of the target concept – such that training instances are generated by a fixed unknown probability distribution D over X (D is unknown over $X \times Y$).
- **A (data, label) tuple as training samples**:
  $S = [(x_1, f(x_1)), (x_2, f(x_2)), \cdots, (x_n, f(x_n))]$, where each is drawn independently from D.
- **One global assumption:** both training and testing instances are drawn from the same distribution D.
- **Goal:** determine a hypothesis $h \in H$ that estimates f, evaluated by its performance on subsequent instances $x \in X$ drawn according to D.

1.2 Motivation

- How much do the results really tell you?
- Can we be certain about how the learning algorithm generalizes?
- We would have to see all the examples. (Not practical)

Insight: Introduce probabilities to measure degree of certainty and correctness. (Valiant 1984)

1.3 Proposal of 'Error'

Consider the error as the probability of an example having different labels according to the hypothesis and the target function, given by

$$Error_D = Pr_{x \in D}(f(x) \notin h(x)).$$

Example 1.1 (Intuitive example)**.** *showing the space predicted by target function f and hypothesis function h, where points inside the circle are positive, points outside are negative, and the functions are given by*

$$h = x_1 \vee x_2 \vee x_3 \vee x_4 \vee x_5, \quad f = x_2 \vee x_3 \vee x_4 \vee x_5$$

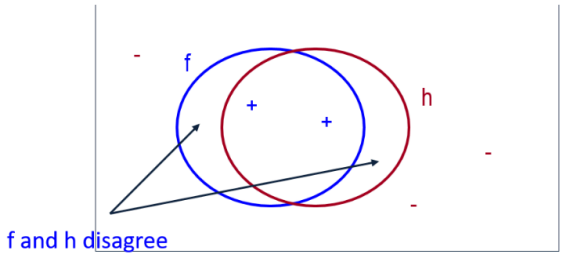

Figure 1: f not equal h

In this example, x_1 in all positive training instances. Therefore, it is very likely that it will be active in future positive examples. If not, it is active in only a small percentage of examples, so the error should be small.

2 Conjunction

2.1 Error Bounds

Claim : $Error(h) \leq \sum_{z \in h} p(z)$.

- $p(z)$: the probability that a D-sampled example is positive and z is false in it (like x_1 in eg1).(type1-error? I in FDP?)

Claim: It there are no bad literals, then $Error(h) < \epsilon$.

2.2 Example (just examble... to be deleted)

Example 2.1. *The core idea of PAC-learnability is easy to understand, and we'll start with a simple example to explain it. Imagine a game between two players. Player 1 generates numbers x at random in some fixed way, and in Player 1's mind he has an*

interval $[a, b]$. Whenever Player 1 gives out an x, he must also say whether it's in the interval (that is, whether $a \leq x \leq b$). Let's say that Player 1 gives reports a 1 if x is in the interval, and a 0 otherwise. We'll call this number the label of z, and call the pair of (x, label) a sample, or an example. We recognize that the zero and one correspond to "yes" and "no" answers to some question (Is this email spam? Does the user click on my ad? etc.), and so sometimes the labels are instead ± 1, and referred to as "positive" or "negative" examples. We'll use the positive/negative terminology here, so positive is a 1 and negative is a 0.

Player 2 (we're on her side) sees a bunch of samples and her goal is to determine a and b. Of course Player 2 can't guess the interval exactly if the endpoints are real numbers, because Player 1 only gives out finitely many samples. But whatever interval Player 2 does guess at the end can be tested against Player 1's number-producing scheme. That is, we can compute the probability that Player 2's interval will give an incorrect label if Player 1 were to continue giving out numbers indefinitely. If this error is small (taking into account how many samples were given), then Player 2 has "learned" the interval. And if Player 2 plays this game over and over and usually wins (no matter what strategy or interval Player 1 decides to use!), then we say this problem is PAC-learnable.

PAC stands for Probably Approximately Correct, and our number guessing game makes it clear what this means. Approximately correct means the interval is close enough to the true interval that the error will be small on new samples, and Probably means that if we play the game over and over we'll usually be able to get a good approximation. That is, we'll find an approximately good interval with high probability.

Indeed, one might already have a good algorithm in mind to learn intervals. Simply take the largest and smallest positive examples and use those as the endpoints of your interval. It's not hard to see why this works, but if we want to prove it (or anything) is PAC-learnable, then we need to solidify these ideas with mathematical definitions.

3 Formulating Prediction Theory

- This notion relies on the Consistent Distribution Assumption: there is one probability distribution D that governs both training and testing examples.

(what's the meaning of 'governs'? like 'dominate'? how does it been defined?)

3.1 PAC introduction

Definitions and Notation:

- X: set of all possible instances or examples, e.g., the set of all men and women characterized by their height and weight.
- $c : X \to \{0, 1\}$: the target concept to learn; can be identified with its support $\{x \in X : c(X) = 1\}$.
- C: concept class, a set of target concepts c.
- D: target distribution, a fixed probability distribution over X. Training and test examples are drawn according to D.
- H: training sample.
- S: set of concept hypotheses, e.g., the set of all linear classifiers.

The learning algorithm receives sample and selects a hypothesis h_S from H approximating. *Any (efficient) algorithm that returns hypotheses that are PAC is called a PAC-learning algorithm. (Formal definition to be introduced later) ? Explainaiton in slides-W14 Wu Peiyuan*

Definition 3.1. *True error or generalization error of h with respect to the target concept c and distribution D:*

$$Error(h) = \Pr_{x \sim D}[h(x) \neq c(x)] = \mathop{\mathrm{E}}_{x \sim D}\left[1_{h(x) \neq c(x)}\right].$$

Definition 3.2. *Empirical error: average error of h on the training sample S drawn according to distribution D,*

$$\widehat{Error}_S(h) = \Pr_{x\sim\widehat{D}}[h(x) \neq c(x)] = \mathrm{E}_{x\sim\widehat{D}}\left[1_{h(x)\neq c(x)}\right] = \frac{1}{m}\sum_{i=1}^{m} 1_{h(x_i)\neq c(x_i)}.$$

Note: $Error(h) = \underset{S\sim D^m}{\mathrm{E}}\left[\widehat{R}_S(h)\right]$.

A few comments on notation. ϵ is called the accuracy parameter, and we call h "ϵ-good" if $Error(h) \leq \epsilon$, where $Error(h)$ is called the true error or the generalization error. δ is the confidence parameter.

The name "Probably Approximately Correct" comes from the fact that we want a hypothesis that is approximately correct (ϵ-good) with high probability (namely $1-\delta$).

3.2 PAC Learnability

Consider a concept class C defined over an instance space X, and a learner L using a hypothesis space H. Denote as $Error(h)$ as $\mathcal{R}(h)$.

Definition 3.3. *C is PAC learnable by L using H if: $\forall f \in C$, $\forall D$ over X, and fixed $0 < \epsilon, \delta < 1$, L – given a collection of m examples sampled independently according to D – produces with probability at least $1-\delta$ a hypothesis $h \in H$ with error at most ϵ where m is polynomial in $\frac{1}{\epsilon}, \frac{1}{\delta}$ and $|H|$.*

$$\Pr_{S\sim D^m}[\mathcal{R}(h(S)) \leq \epsilon] \geq 1-\delta,$$

where $h(S) \in H$ is the hypothesis learned by $\mathbb{A}$ from sample S. We say $\mathbb{A}$ is a PAC-learning algorithm for C.

Another definition of PAC learning:

Definition 3.4. *Concept class C is PAC-learnable if there exists a learning algorithm such that:*

- *for all $c \in C, \epsilon > 0, \delta > 0$ and all distributions D, $\underset{S\sim D^m}{\Pr}[R_S(h) \leq \epsilon] \geq 1-\delta$*
- *for samples S of size $m = poly(1/\epsilon, 1/\delta)$ for a fixed polynomial.*

Theorem 3.5. *Sample complexity for finite hypothesis sets - consistent case: Let H be a finite set of binary classifiers on* X*. Let A be an algorithm such that for any target concept $c \in H$ and i.i.d. sample S of size m returns a consistent hypothesis $\mathbb{A}(S) \in H$ such that $\widehat{\mathcal{R}}_S(\mathbb{A}(S)) = 0$. Then*

$$\Pr_{x \sim \widehat{D}}[\mathcal{R}(\mathbb{A}(S)) \leq \epsilon] \geq 1 - |H| e^{-m\epsilon}$$