E0 370 Statistical Learning Theory **Lecture 11 (Sep 22, 2011)**

Learnability: Basic Definitions, Examples, Results

Lecturer: Shivani Agarwal *Scribe: Naresh Manwani*

1 Introduction

The notion of learnability plays a central role in learning theory and continues to be an active topic of study. As noted in the last lecture, it is related to the notion of statistical consistency, with origins in theoretical computer science. Over the next few lectures, we will study various models of learnability related to the *probably approximately correct* (PAC) learning model introduced in the seminal work of Valiant [7]. We will begin in this lecture with some basic definitions and results, and illustrate with more results and examples in the next few lectures.

2 Basic Definitions

2.1 Learnability

Definition (Learnability)**.** Let $\mathcal{H} \subseteq \{\pm 1\}^{\mathcal{X}}$. $\mathcal{H}$ is said to be *(properly) learnable* if there exists an algorithm[1] $\mathcal{A} : \cup_{m=1}^{\infty}(\mathcal{X} \times \{\pm 1\})^m \rightarrow \mathcal{H}$, which given a training sample S outputs a function $h_S \in \mathcal{H}$, such that $\forall \epsilon, \delta \in (0,1]$, $\exists m_0(\epsilon, \delta) \in \mathbb{N}$ such that for all probability distributions D on $\mathcal{X} \times \{\pm 1\}$ and $m \geq m_0(\epsilon, \delta)$,

$$\mathbf{P}_{S \sim D^m}\Big(\mathrm{er}_D[h_S] - \mathrm{er}_D[\mathcal{H}] \geq \epsilon\Big) \leq \delta \,. \tag{1}$$

For each ϵ, δ, let $m_{\mathcal{A}}(\epsilon, \delta)$ be the smallest integer for which $\mathcal{A}$ satisfies the above; then $m_{\mathcal{A}}(\cdot, \cdot)$ is called the *sample complexity* of $\mathcal{A}$. The smallest sample complexity $m_{\mathcal{A}}(\epsilon, \delta)$ over all algorithms $\mathcal{A}$ satisfying the above is called the *sample complexity of learning* $\mathcal{H}$, denoted $m_{\mathcal{H}}(\epsilon, \delta)$. $\mathcal{H}$ is said to be *learnable from a polynomial number of examples* if $m_{\mathcal{H}}(\epsilon, \delta) = \mathrm{poly}(\frac{1}{\epsilon}, \frac{1}{\delta})$.

It is worth noting that learnability of $\mathcal{H}$ requires the existence of an algorithm $\mathcal{A} : \cup_{m=1}^{\infty}(\mathcal{X} \times \{\pm 1\})^m \rightarrow \mathcal{H}$ that is universally consistent in $\mathcal{H}$, and moreover, for which the following property is satisfied: for all $m \in \mathbb{N}$ and $\epsilon \in (0,1]$, there is a *fixed* $\delta_0(m, \epsilon)$ such that for *all* probability distributions D on $\mathcal{X} \times \{\pm 1\}$,

$$\mathbf{P}_{S \sim D^m}\Big(\mathrm{er}_D[h_S] - \mathrm{er}_D[\mathcal{H}] \geq \epsilon\Big) \leq \delta_0(m, \epsilon) \,, \tag{2}$$

with $\delta_0(m, \epsilon) \rightarrow 0$ as $m \rightarrow \infty$ for all ϵ.

2.2 Learnability in Target Function Setting

Often, it is of interest to study learnability in a more restricted setting where there is a 'target function' in the function class of interest according to which all examples are labeled (in fact this is the setting originally studied by Valiant). To this end, for any $\mathcal{H} \subseteq \{\pm 1\}^{\mathcal{X}}$ and $t \in \mathcal{H}$, let $\mathcal{X} \times t = \{(x, t(x)) \mid x \in \mathcal{X}\}$, and for any probability distribution μ on $\mathcal{X}$, denote by $\mu \times t$ the joint distribution on $\mathcal{X} \times t$ that simply samples a random $x \in \mathcal{X}$ according to μ, and then returns the example $(x, t(x))$.

[1] Note that an algorithm here is simply a function; we do not consider here computational complexity issues (in fact the function is not even required to be computable).

Definition (Learnability in target function setting)**.** Let $\mathcal{H} \subseteq \{\pm 1\}^{\mathcal{X}}$. $\mathcal{H}$ is said to be *learnable in the target function setting* if there exists an algorithm $\mathcal{A} : \cup_{t\in\mathcal{H}} \cup_{m=1}^{\infty} (\mathcal{X} \times t)^m \rightarrow \mathcal{H}$, which given a training sample $S \in \cup_{t\in\mathcal{H}} \cup_{m=1}^{\infty} (\mathcal{X} \times t)^m$ outputs a function $h_S \in \mathcal{H}$, such that $\forall \epsilon, \delta \in (0, 1]$, $\exists m_0(\epsilon, \delta) \in \mathbb{N}$ such that for all probability distributions μ on $\mathcal{X}$ and all target functions $t \in \mathcal{H}$, for all $m \geq m_0(\epsilon, \delta)$,

$$\mathbf{P}_{S\sim(\mu\times t)^m}\Big(\mathrm{er}_{\mu\times t}[h_S] \geq \epsilon\Big) \leq \delta\,. \tag{3}$$

Note that in this setting, $\mathrm{er}_{\mu\times t}[\mathcal{H}] = 0$. The sample complexity of $\mathcal{A}$ and of learning $\mathcal{H}$ in the target function setting can be defined as before. Again, $\mathcal{H}$ is said to be *learnable in the target function setting from a polynomial number of examples* if $m_0(\epsilon, \delta) = \mathrm{poly}(\frac{1}{\epsilon}, \frac{1}{\delta})$.

2.3 Learnability with respect to Domain Dimension

In practice, one is often interested in function classes defined over domains of varying dimension, such as linear functions defined over $\mathbb{R}^n$ for varying n; it is then of interest to study learnability of such classes taking into account the dependence on the domain dimension.

Definition (Learnability w.r.t. domain dimension)**.** Let $\mathcal{X} = \cup_{n=1}^{\infty}\mathcal{X}_n$, where n can be viewed as the 'dimension' of $\mathcal{X}_n$ (e.g. $\mathcal{X}_n = \mathbb{R}^n$ or $\{0,1\}^n$). Let $\mathcal{H}_n \subseteq \{\pm 1\}^{\mathcal{X}_n}$. We say the 'dimension-graded' function class $\mathcal{H} = \cup_{n=1}^{\infty}\mathcal{H}_n$ is *learnable w.r.t. the domain dimension* n if there exists an algorithm $\mathcal{A}$: $\cup_{n=1}^{\infty} \cup_{m=1}^{\infty} (\mathcal{X}_n \times \{\pm 1\})^m \rightarrow \mathcal{H}$, which given a training sample $S \in \cup_{m=1}^{\infty}(\mathcal{X}_n \times \{\pm 1\})^m$ outputs a function $h_S \in \mathcal{H}_n$, such that $\forall \epsilon, \delta \in (0, 1]$ and $n \in \mathbb{N}$, $\exists m_0(\epsilon, \delta, n) \in \mathbb{N}$ such that for all probability distributions D_n on $\mathcal{X}_n \times \{\pm 1\}$ and $m \geq m_0(\epsilon, \delta, n)$,

$$\mathbf{P}_{S\sim D_n^m}\Big(\mathrm{er}_{D_n}[h_S] - \mathrm{er}_{D_n}[\mathcal{H}_n] \geq \epsilon\Big) \leq \delta\,. \tag{4}$$

$\mathcal{H}$ is said to be *learnable w.r.t. domain dimension from a polynomial number of examples* if $m_0(\epsilon, \delta, n) = \mathrm{poly}(\frac{1}{\epsilon}, \frac{1}{\delta}, n)$.

2.4 Learnability with respect to Target Complexity

Another parameter of interest in studying learnability in the target function setting is the complexity or size of the target function:

Definition. Let $\mathcal{H} \subseteq \{\pm 1\}^{\mathcal{X}}$, and let $\mathrm{size} : \mathcal{H}\rightarrow\mathbb{N}$ be such that $\mathrm{size}(t)$ measures the 'complexity' or 'size' of any function $t \in \mathcal{H}$. We say $\mathcal{H}$ is *learnable w.r.t. the complexity function* 'size' if there exists an algorithm $\mathcal{A} : \cup_{t\in\mathcal{H}} \cup_{m=1}^{\infty} (\mathcal{X} \times t)^m \rightarrow \mathcal{H}$, which given a training sample $S \in \cup_{t\in\mathcal{H}} \cup_{m=1}^{\infty} (\mathcal{X} \times t)^m$ outputs a function $h_S \in \mathcal{H}$, such that $\forall \epsilon, \delta \in (0, 1]$ and $s \in \mathbb{N}$, $\exists m_0(\epsilon, \delta, s) \in \mathbb{N}$ such that for all probability distributions μ on $\mathcal{X}$ and all $t \in \mathcal{H}$ with $\mathrm{size}(t) \leq s$, for all $m \geq m_0(\epsilon, \delta, s)$,

$$\mathbf{P}_{S\sim(\mu\times t)^m}\Big(\mathrm{er}_{\mu\times t}[h_S] \geq \epsilon\Big) \leq \delta\,. \tag{5}$$

$\mathcal{H}$ is said to be *learnable w.r.t. target complexity* 'size' *from a polynomial number of examples* if $m_0(\epsilon, \delta, s) = \mathrm{poly}(\frac{1}{\epsilon}, \frac{1}{\delta}, s)$.

Note: One can also have a dimension-graded function class $\mathcal{H} = \cup_{n=1}^{\infty}\mathcal{H}_n$, where $\mathcal{H}_n \subseteq \{\pm 1\}^{\mathcal{X}_n}$, with a complexity measure $\mathrm{size} : \mathcal{H}\rightarrow\mathbb{N}$ defined for all functions in $\mathcal{H}$. In this case, one allows the sample complexity to depend on $\frac{1}{\epsilon}, \frac{1}{\delta}$, dimension n, and (an upper bound on) complexity s of the target function.

2.5 Polynomial-Time/Efficient Learnability

In each of the above settings, a class $\mathcal{H}$ is said to be *polynomial-time/efficiently learnable* if it is learnable from a polynomial number of examples (polynomial in the relevant parameters), by an algorithm whose running time is polynomial in the size of its input, i.e. polynomial in the sample size m, and where relevant, in the domain dimension n (assuming that instances from $\mathcal{X}_n$ are encoded using a representation of size polynomial in n, which is the case for example for $\mathcal{X}_n = \mathbb{R}^n$ (with finite precision) or $\mathcal{X}_n = \{0,1\}^n$).

It is worth noting that the original probably approximately correctly (PAC) model of learnability proposed by Valiant [7] was defined in the target function setting, included both dimension and complexity parameters, and required both polynomial sample complexity and polynomial running time as integral parts of its definition. The general model defined in Definition 1 was studied as a generalization by Haussler [4], Kearns et al. [5] and others; it is sometimes referred to as the 'agnostic' model of learnability [5]. Clearly, learnability in the general model implies learnability in the target function setting (since the target function setting considers a restricted subset of the set of all joint probability distributions on $\mathcal{X} \times \{\pm 1\}$). Details and further generalizations can be found for example in [6, 1].

2.6 Learnability of $\mathcal{H}$ by a Different Function Class $\mathcal{H}'$

Often, it is desirable to let the algorithm learn functions from a larger function class $\mathcal{H}' \supseteq \mathcal{H}$, but evaluate its performance w.r.t. the optimal performance in $\mathcal{H}$ as before. This leads to the following:

Definition (Learnability by $\mathcal{H}'$)**.** Let $\mathcal{H} \subseteq \mathcal{H}' \subseteq \{\pm 1\}^{\mathcal{X}}$. $\mathcal{H}$ is said to be *learnable by* $\mathcal{H}'$ if there exists an algorithm $\mathcal{A} : \cup_{m=1}^{\infty}(\mathcal{X} \times \{\pm 1\})^m \to \mathcal{H}'$, which given a training sample $S \in (\mathcal{X} \times \{\pm 1\})^m$ outputs a function $h_S \in \mathcal{H}'$, such that $\forall \epsilon, \delta \in (0,1]$, $\exists m_0(\epsilon,\delta) \in \mathbb{N}$ such that for all probability distributions D on $\mathcal{X} \times \{\pm 1\}$ and $m \geq m_0(\epsilon,\delta)$,

$$\mathbf{P}_{S \sim D^m}\Big(\mathrm{er}_D[h_S] - \mathrm{er}_D[\mathcal{H}] \geq \epsilon\Big) \leq \delta\,. \tag{6}$$

Often, function classes $\mathcal{H}$ that cannot be learned efficiently (or are not known to be efficiently learnable) by algorithms using $\mathcal{H}$ can be learned efficiently by an algorithm using a larger class $\mathcal{H}'$.

We note here that sometimes, $\mathcal{H}$ is said to be learnable if $\mathcal{H}$ is learnable by *any* class $\mathcal{H}'$, and *properly* learnable if it is learnable by $\mathcal{H}$. In this course, unless stated otherwise, learnability will refer to proper learnability (when we need to consider learnability by a different function class, we will make this explicit).

3 Some Learnability Results

We showed last time that ERM in any class $\mathcal{H} \subseteq \{\pm 1\}^{\mathcal{X}}$ with finite VC-dimension is universally consistent in $\mathcal{H}$. In fact, we showed something stronger: we showed that a fixed function $\delta_0(m,\epsilon)$ works for all distributions D on $\mathcal{X} \times \{\pm 1\}$. This gives us our first learnability result:

Theorem 3.1. Let $\mathcal{H} \subseteq \{\pm 1\}^{\mathcal{X}}$, with $\mathrm{VCdim}(\mathcal{H}) = d < \infty$. Then $\mathcal{H}$ is learnable, with sample complexity

$$m_{\mathcal{H}}(\epsilon,\delta) \leq \frac{64}{\epsilon^2}\Big(2d\ln\Big(\frac{12}{\epsilon}\Big) + \ln\Big(\frac{4}{\delta}\Big)\Big)\,. \tag{7}$$

Proof. Consider any ERM algorithm in $\mathcal{H}$. We showed last time that for all distributions D on $\mathcal{X} \times \{\pm 1\}$, all $\epsilon \in (0,1]$, and all $m \in \mathbb{N}$, such an algorithm satisfies

$$\mathbf{P}_{S \sim D^m}\Big(\mathrm{er}_D[h_S] - \mathrm{er}_D[\mathcal{H}] \geq \epsilon\Big) \leq 4\left(\frac{2em}{d}\right)^d e^{-m\epsilon^2/32}\,. \tag{8}$$

It can be verified that for $m \geq \frac{64}{\epsilon^2}\Big(2d\ln\Big(\frac{12}{\epsilon}\Big) + \ln\Big(\frac{4}{\delta}\Big)\Big)$, the RHS above is at most δ. □

The above clearly implies learnability of any $\mathcal{H}$ that has finite VC-dimension in the target function setting as well. In this case, however, it is possible to obtain a better bound on the sample complexity:

Theorem 3.2. Let $\mathcal{H} \subseteq \{\pm 1\}^{\mathcal{X}}$, with $\mathrm{VCdim}(\mathcal{H}) = d < \infty$. Then $\mathcal{H}$ is learnable in the target function setting with sample complexity

$$m_{\mathcal{H}}(\epsilon,\delta) \leq \frac{4}{\epsilon}\Big(d\ln\Big(\frac{12}{\epsilon}\Big) + \ln\Big(\frac{2}{\delta}\Big)\Big)\,. \tag{9}$$

Proof. In this case, an ERM algorithm returns a function $h_S \in \mathcal{H}$ with $\mathrm{er}_S[h_S] = 0$. The result follows from an improved uniform convergence result for such functions: for all μ on $\mathcal{X}$, $t \in \mathcal{H}$, $\epsilon \in (0,1]$, and $m \in \mathbb{N}$:

$$\mathbf{P}_{S\sim(\mu\times t)^m}\Big(\exists h \in \mathcal{H} : \mathrm{er}_S[h] = 0, \mathrm{er}_{\mu\times t}[h] \geq \epsilon\Big) \leq 2\,\Pi_{\mathcal{H}}(2m)e^{-m\epsilon/2}\,. \tag{10}$$

Details can be found in [2, 3]. □

The above results do not say anything about the computational complexity. Indeed, as we have discussed before, for many function classes $\mathcal{H}$ of interest (such as linear classifiers over $\mathbb{R}^n$), finding an empirical risk minimizer can be NP-hard. Below we give some examples of efficient learning algorithms, all in the target function setting. Note that, in this setting, ERM reduces to finding any function in $\mathcal{H}$ that achieves zero error on the training sample. This is what the algorithms below do.[2]

Example 1 (Learnability of axis-parallel rectangles in $\mathbb{R}^2$)**.** Let $\mathcal{X} = \mathbb{R}^2$, and let $\mathcal{H} = \{h : \mathcal{X}\to\{\pm1\} \mid h(x) = 1 \text{ if } x \in [a,b] \times [c,d] \text{ and } -1 \text{ otherwise, for some } a \leq b,\ c \leq d\}$. Recall from Lecture 4 that $\mathrm{VCdim}(\mathcal{H}) = 4 < \infty$. Therefore, by Theorem 3.2 above, we have that $\mathcal{H}$ is learnable in the target function setting (from $\mathrm{poly}(\frac{1}{\epsilon}, \frac{1}{\delta})$ examples) by any algorithm that given a training sample S labeled according to a target function $t \in \mathcal{H}$, returns a function h_S in $\mathcal{H}$ that classifies all examples in S correctly. As an example of such an algorithm, consider returning as h_S the smallest rectangle containing all the positive points in S (since the examples in S are labeled according to a rectangle in $\mathcal{H}$, the resulting rectangle h_S will contain all positive instances in S but no negative instances, and will therefore classify all points in S correctly). This involves computing simply 4 numbers l, r, t, b representing the minimum and maximum values of the x and y coordinates of the positive examples seen in S, which can clearly be done in $O(m)$ time (where $m = |S|$). Therefore, the class $\mathcal{H}$ of axis-parallel rectangles in $\mathbb{R}^2$ is efficiently learnable in the target function setting.

Example 2 (Learnability of k-DNF over $\{0,1\}^n$)**.** Let $X_n = \{0,1\}^n$. Fix $k \in \mathbb{N}$, and for each n, let $\mathcal{H}_n$ be the class of binary-valued functions represented by a k-DNF formula over $\{0,1\}^n$, i.e. by a Boolean formula in disjunctive normal form, consisting of a disjunction of terms, with at most k literals per term (each term is a conjunction of literals; each literal is either a variable x_i for some $i \in [n]$ or its negation $\bar{x}_i$). For example, for $n = 4$ and $k = 2$, the following is an example of a function that can be represented as a k-DNF forumla over $\{0,1\}^n$ and is therefore a member of $\mathcal{H}_n$:

$$h(\mathbf{x}) \quad = \quad \begin{cases} \ \ 1 & \text{if } (x_1 \wedge \bar{x}_3) \vee (\bar{x}_1 \wedge x_4) \vee (\bar{x}_2 \wedge \bar{x}_4) = 1 \\ -1 & \text{otherwise.} \end{cases}$$

Here $|\mathcal{H}_n| \leq 2^{(2n)^{k+1}}$, since there are at most $\binom{2n}{1}+\binom{2n}{2}+\ldots+\binom{2n}{k} < (2n)^{k+1}$ terms of at most k literals, each of which can be present or absent in a specific k-DNF formula. Therefore $\mathrm{VCdim}(\mathcal{H}_n) \leq \log_2|\mathcal{H}_n| = (2n)^{k+1}$, which for fixed k is polynomial in n. Thus, by Theorem 3.2, $\mathcal{H}_n$ is learnable in the target function setting (from $\mathrm{poly}(\frac{1}{\epsilon}, \frac{1}{\delta}, n)$ examples) by any algorithm that given a training sample S labeled according to a target function $t \in \mathcal{H}_n$, returns a function $h_S \in \mathcal{H}_n$ that classifies all examples in S correctly. As an example of such an algorithm, consider starting with an initial function $h \in \mathcal{H}_n$ that consists of a disjunction of *all* possible terms of size $\leq k$, i.e. at most $(2n)^{k+1}$ terms; for each negative example in S, remove from h all terms that are satisfied by that example, and then return the remaining function as h_S. This clearly classifies all examples in S correctly. Moreover, the running time of the algorithm can be verified to be $O(mk(2n)^{k+1})$, which for fixed k is polynomial in both m and n; therefore the dimension-graded function class $\mathcal{H} = \cup_{n=1}^{\infty}\mathcal{H}_n$ in this case is efficiently learnable w.r.t. the domain dimension n in the target function setting.

4 Next Lecture

The next lecture will discuss further examples and results related to learnability.

[2] In the PAC learning literature, a function that achieves zero error on a training sample S ($\mathrm{er}_S[h] = 0$) is said to be consistent with S, and an algorithm that returns such a function is sometimes said to be a 'consistent' algorithm. It is important to note that this usage of the term 'consistent' is completely different from its usage in 'statistically consistent' (see previous lecture). To avoid confusion, in this course we will simply refer to such algorithms as achieving zero error on the training sample S, or as classifying all training examples correctly.

References

[1] Martin Anthony and Peter L. Bartlett. *Learning in Neural Networks: Theoretical Foundations.* Cambridge University Press, 1999.

[2] Anselm Blumer, Andrzej Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM*, 36(4):929–965, 1989.

[3] Luc Devroye, Laszlo Gyorfi, and Gabor Lugosi. *A Probabilistic Theory of Pattern Recognition.* Springer, 1996.

[4] David Haussler. Decision theoretic generalizations of the PAC model for neural net and other learning applications. *Information and Computation*, 100(1):78–150, 1992.

[5] Michael J. Kearns, Robert E. Schapire, and Linda Sellie. Toward efficient agnostic learning. *Machine Learning*, 17(2-3):115–141, 1994.

[6] Michael J. Kearns and Umesh V. Vazirani. *An Introduction to Computational Learning Theory.* MIT Press, Cambridge, MA, 1994.

[7] Leslie G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.