3

The Vapnik-Chervonenkis Dimension

3.1 When Can Infinite Classes Be Learned with a Finite Sample?

In this chapter, we consider the following question: How many random examples does a learning algorithm need to draw before it has sufficient information to learn an unknown target concept chosen from the concept class $\mathcal{C}$? We should emphasize that we will temporarily ignore issues of computational efficiency while studying this question (or equivalently, we assume that the learning algorithm has infinite computing power to process the finite random sample it has drawn). We first note that the results of the previous chapter can be used to give such a bound in the case that $\mathcal{C}$ is a concept class of finite cardinality. If the learning algorithm simply draws a random sample of $O((1/\epsilon)\log(|\mathcal{C}|/\delta))$ examples, and finds any $h \in \mathcal{C}$ consistent with these examples (say, by exhaustive search), then Theorem 2.2 guarantees that h will meet the PAC model criteria. Notice that this bound is not meaningful if $\mathcal{C}$ has infinite cardinality. Are there any non-trivial infinite concept classes that are learnable from a finite sample?

Actually, our PAC learning algorithm for axis-aligned rectangles in the Euclidean plane given in Section 1.1 is an example of such a class. In the analysis of that PAC learning algorithm, we made critical use of the fact that axis-aligned rectangles have simple boundaries: the target rectangle is always completely specified by four real numbers that indicate the locations of the four bounding edges, and this allowed us to partition the error of the tightest-fit hypothesis into four simple rectilinear regions. It is tempting to say that the "complexity" of this concept class is four, because the boundary of any concept in the class can be described by four real numbers.

In this chapter, we are interested in a general measure of complexity for concept classes of infinite cardinality. We would like this measure to play the same role in the sample complexity of PAC learning infinite classes that the quantity $\log|\mathcal{C}|$ (which we saw in Chapter 2 was closely related to the size of representations) plays in the sample complexity of PAC learning finite classes. We will define a purely combinatorial measure of concept class complexity known as the *Vapnik-Chervonenkis dimension*, a measure that assigns to each concept class $\mathcal{C}$ a single number that characterizes the sample size needed to PAC learn $\mathcal{C}$.

3.2 The Vapnik-Chervonenkis Dimension

For the remainder of this chapter, $\mathcal{C}$ will be a concept class over instance space X, and both $\mathcal{C}$ and X may be infinite. The first thing we will need is a way to discuss the behavior of $\mathcal{C}$ when attention is restricted to a finite set of points $S \subseteq X$.

Definition 7 *For any concept class $\mathcal{C}$ over X, and any $S \subseteq X$,*

$$\Pi_{\mathcal{C}}(S) = \{c \cap S : c \in \mathcal{C}\}.$$

Equivalently, if $S = \{x_1, \ldots, x_m\}$ *then we can think of* $\Pi_C(S)$ *as the set of vectors* $\Pi_C(S) \subseteq \{0,1\}^m$ *defined by*

$$\Pi_C(S) = \{(c(x_1), \ldots, c(x_m)) : c \in C\}.$$

Thus, $\Pi_C(S)$ is the set of all the **behaviors** or **dichotomies** on S that are induced or **realized** by C. We will use the descriptions of $\Pi_C(S)$ as a collection of subsets of S and as a set of vectors interchangeably.

Definition 8 *If* $\Pi_C(S) = \{0,1\}^m$ *(where* $m = |S|$*), then we say that* S *is* **shattered** *by* C*. Thus,* S *is shattered by* C *if* C *realizes all possible dichotomies of* S*.*

Now we are ready for our key definition.

Definition 9 *The* **Vapnik-Chervonenkis (VC) dimension** *of* C*, denoted as* $VCD(C)$*, is the cardinality* d *of the largest set* S *shattered by* C*. If arbitrarily large finite sets can be shattered by* C*, then* $VCD(C) = \infty$*.*

3.3 Examples of the VC Dimension

Let us consider a few natural geometric concept classes, and informally calculate their VC dimension. It is important to emphasize the nature of the existential and universal quantifiers in the definition of VC dimension: in order to show that the VC dimension of a class is at least d, we must simply find some shattered set of size d. In order to show that the VC dimension is at most d, we must show that no set of size $d+1$ is shattered. For this reason, proving upper bounds on the VC dimension is usually considerably more difficult than proving lower bounds. The following examples are not meant to be precise proofs of the stated bounds on

Figure 3.1: *A dichotomy unrealizable by intervals.*

the VC dimension, but are simply illustrative exercises to provide some practice thinking about the VC dimension.

Intervals of the real line. For this concept class, any set of two points can be shattered, so the VC Dimension is at least two, but no set of three points can be shattered: label the three points as shown in Figure 3.1, a labeling which cannot be induced by any interval. Thus the VC dimension for this class is two.

Linear halfspaces in the plane. For this concept class, any three points that are not collinear can be shattered. Figure 3.2(a) shows how one dichotomy out of the possible 8 dichotomies can be realized by a halfspace; the reader can easily verify that the remaining 7 dichotomies can be realized by halfspaces. To see that no set of four points can be shattered, we consider two cases. In the first case (shown in Figure 3.2(b)), all four points lie on the convex hull defined by the four points. In this case, if we label one "diagonal" pair positive and the other "diagonal" pair negative as shown in Figure 3.2(b), no halfspace can induce this labeling. In the second case (shown in Figure 3.2(c)), three of the four points define the convex hull of the four points, and if we label the interior point negative and the hull points positive, again no halfspace can induce the dichotomy. Thus the VC dimension here is three. In general, for halfspaces in $\Re^d$, the VC dimension is $d + 1$.

Axis-aligned rectangles in the plane. For this concept class, we can shatter the four points shown in Figure 3.3(a), where we have again indicated how a single dichotomy can be realized and left the remainder to the reader. However, not *all* sets of four points can be shattered, as indicated by the unrealizable dichotomy shown in Figure 3.3(b). Still, the existence of a single shattered set of size four is sufficient to lower bound

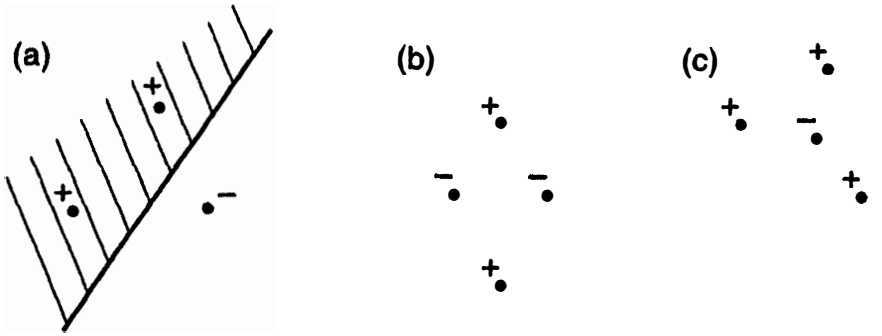

Figure 3.2: (a) *A dichotomy and its realization by a halfspace, with the shaded region indicating the positive side.* (b) and (c) *Dichotomies unrealizable by halfspaces.*

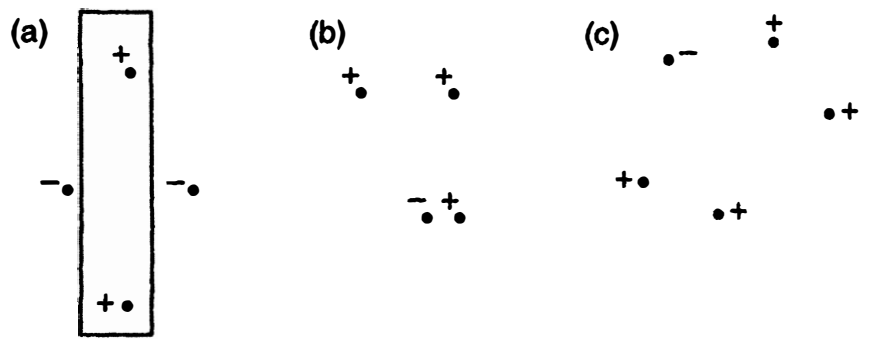

Figure 3.3: (a) *A dichotomy and its realization by an axis-aligned rectangle.* (b) and (c) *Dichotomies unrealizable by axis-aligned rectangles.*

the VC dimension. Now for any set of five points in the plane, there must be some point that is neither the extreme left, right, top or bottom point of the five (see Figure 3.3(c)). If we label this non-extremal point negative and the remaining four extremal point positive, no rectangle can realize the dichotomy. Thus the VC dimension is four.

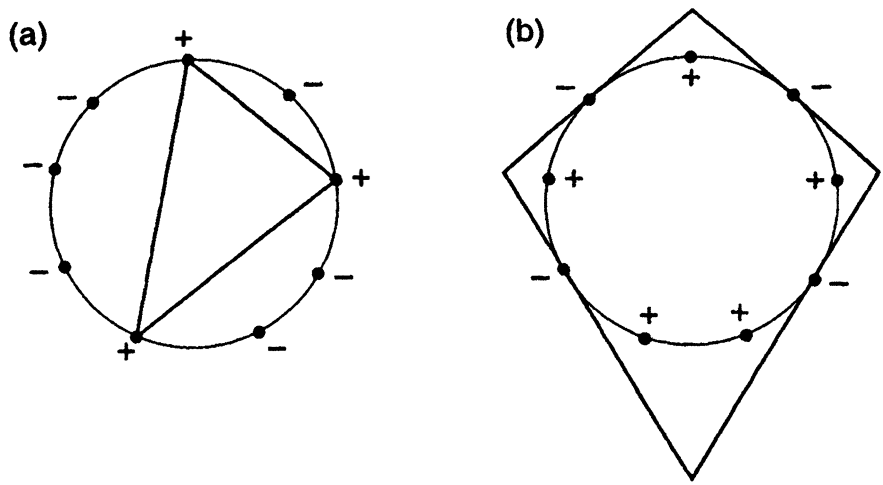

Figure 3.4: (a) *Realizing a dichotomy with a polygon when there are fewer positive labels.* (b) *When there are fewer negative labels.*

Convex polygons in the plane. For convex d-gons in the plane, the VC dimension is $2d + 1$. For the lower bound, we can induce any labeling of any $2d + 1$ points on a circle using a d-gon as follows: if there are more negative labels than positive labels, use the positive points as the vertices as shown in Figure 3.4(a). Otherwise, use tangents to the negative points as edges as shown in Figure 3.4(b). For the upper bound, it can be shown that choosing the points to lie on a circle does in fact maximize the number of points that can be shattered, and we can force $d + 1$ sides using $2d + 2$ points on a circle by alternating positive and negative labels.

3.4 A Polynomial Bound on $|\Pi_{\mathcal{C}}(S)|$

Definition 10 *For any natural number m we define*

$$\Pi_{\mathcal{C}}(m) = \max\{|\Pi_{\mathcal{C}}(S)| : |S| = m\}.$$

The function $\Pi_{\mathcal{C}}(m)$ can be thought of as a measure of the complexity of $\mathcal{C}$: the faster this function grows, the more behaviors on sets of m points that can be realized by $\mathcal{C}$ as m increases. Now clearly, if $\mathcal{C}$ does not have finite VC dimension, then $\Pi_{\mathcal{C}}(m) = 2^m$ for all m since we can shatter arbitrarily large finite sets. In this section, we prove a surprising and beautiful result, namely that despite the fact that we might naively expect $\Pi_{\mathcal{C}}(m)$ to grow as rapidly as an exponential function of m, it is actually bounded by a polynomial in m of degree d, where d is the VC dimension of $\mathcal{C}$. In other words, depending on whether the VC dimension is finite or infinite, the function $\Pi_{\mathcal{C}}(m)$ is either eventually polynomial or forever exponential. For the more interesting and typical case of finite VC dimension, we shall eventually translate the polynomial upper bound on $\Pi_{\mathcal{C}}(m)$ into an upper bound on the sample complexity of PAC learning that is linear in d.

We begin by proving that $\Pi_{\mathcal{C}}(m)$ is bounded by the function $\Phi_d(m)$ defined below. We then show a polynomial bound on $\Phi_d(m)$.

Definition 11 *For any natural numbers m and d, the function $\Phi_d(m)$ is defined inductively by*

$$\Phi_d(m) = \Phi_d(m-1) + \Phi_{d-1}(m-1)$$

with initial conditions $\Phi_d(0) = \Phi_0(m) = 1$.

Lemma 3.1 *If $VCD(\mathcal{C}) = d$, then for any m, $\Pi_{\mathcal{C}}(m) \leq \Phi_d(m)$.*

Proof: By induction on both d and m. For the base cases, the lemma is easily established when $d = 0$ and m is arbitrary, and when $m = 0$ and d is arbitrary. We assume for induction that for all m', d' such that $m' \leq m$ and $d' \leq d$ and at least one of the two inequalities is strict, we have $\Pi_{\mathcal{C}}(m') \leq \Phi_{d'}(m')$. We now show that this inductive assumption establishes the desired statement for d and m.

Given any set S of size m, let $x \in S$ be a distinguished point. Let us first compute $|\Pi_C(S - \{x\})|$. This is easy since by induction (note that $S - \{x\}$ is a set of size $m - 1$) we have $|\Pi_C(S - \{x\})| \leq \Phi_d(m - 1)$.

The difference between $\Pi_C(S)$ and $\Pi_C(S - \{x\})$ is that pairs of distinct sets in $\Pi_C(S)$ that differ only on their labeling of x are identified (that is, merged) in $\Pi_C(S - \{x\})$. Thus let us define

$$C' = \{c \in \Pi_C(S) : x \notin c, c \cup \{x\} \in \Pi_C(S)\}.$$

Then $|C'|$ counts the number of pairs of sets in $\Pi_C(S)$ that are collapsed to a single representative in $\Pi_C(S - \{x\})$. Note that $C' = \Pi_{C'}(S - \{x\})$ because C' consists only of subsets of $S - \{x\}$. This yields the simple equality

$$|\Pi_C(S)| = |\Pi_C(S - \{x\})| + |\Pi_{C'}(S - \{x\})|.$$

We now show that $VCD(C') \leq d - 1$. To see this, let $S' \subseteq S - \{x\}$ be shattered by C'. Then $S' \cup \{x\}$ is shattered by C. Thus we must have $|S'| \leq d - 1$. Now by induction we have $|C'| = |\Pi_{C'}(S - \{x\})| \leq \Phi_{d-1}(m - 1)$.

Our total count is thus bounded by $\Phi_d(m-1)+\Phi_{d-1}(m-1) = \Phi_d(m)$, as desired. □(Lemma 3.1)

Lemma 3.2 $\Phi_d(m) = \sum_{i=0}^{d} \binom{m}{i}$.

Proof: By induction; the base cases are easy to check. For the induction step, we have:

$$\begin{aligned}
\Phi_d(m) &= \Phi_d(m-1) + \Phi_{d-1}(m-1) \\
&= \sum_{i=0}^{d} \binom{m-1}{i} + \sum_{i=0}^{d-1} \binom{m-1}{i} \\
&= \sum_{i=0}^{d} \left[\binom{m-1}{i} + \binom{m-1}{i-1} \right] \\
&= \sum_{i=0}^{d} \binom{m}{i}
\end{aligned}$$

where the second equality is by induction and we define $\binom{m-1}{-1} = 0$ for the third equality. □(Lemma 3.2)

Now for $m \leq d$, $\Phi_d(m) = 2^m$. For $m > d$, since $0 \leq d/m \leq 1$, we may write:

$$\left(\frac{d}{m}\right)^d \sum_{i=0}^{d} \binom{m}{i} \leq \sum_{i=0}^{d} \left(\frac{d}{m}\right)^i \binom{m}{i} \leq \sum_{i=0}^{m} \left(\frac{d}{m}\right)^i \binom{m}{i} = \left(1 + \frac{d}{m}\right)^m \leq e^d.$$

Dividing both sides by $\left(\frac{d}{m}\right)^d$ yields

$$\Phi_d(m) = \sum_{i=0}^{d} \binom{m}{i} \leq \left(\frac{em}{d}\right)^d = O(m^d)$$

which is polynomial in m for fixed d, giving us the promised polynomial bound for the case $m > d$.

3.5 A Polynomial Bound on the Sample Size for PAC Learning

3.5.1 The Importance of ϵ-Nets

Let us now fix the target concept $c \in \mathcal{C}$, and define the class of **error regions** with respect to c and $\mathcal{C}$ by $\Delta(c) = \{c \Delta c' : c' \in \mathcal{C}\}$. It is easy to show that $VCD(\mathcal{C}) = VCD(\Delta(c))$. To see this, for any set S we can map each element $c' \in \Pi_{\mathcal{C}}(S)$ to $c'\Delta(c \cap S) \in \Pi_{\Delta(c)}(S)$. Since this is a bijective mapping of $\Pi_{\mathcal{C}}(S)$ to $\Pi_{\Delta(c)}(S)$, $|\Pi_{\Delta(c)}(S)| = |\Pi_{\mathcal{C}}(S)|$. Since this holds for any set S, $VCD(\mathcal{C}) = VCD(\Delta(c))$ follows.

We may further refine the definition of $\Delta(c)$ to consider only those error regions with weight at least ϵ under the fixed target distribution $\mathcal{D}$. Thus, let $\Delta_\epsilon(c) = \{r \in \Delta(c) : \mathbf{Pr}_{x \in \mathcal{D}}[x \in r] \geq \epsilon\}$. We can now make the following important definition:

Definition 12 *For any $\epsilon > 0$, we say that a set S is an* ϵ-net *for $\Delta(c)$ if every region in $\Delta_\epsilon(c)$ is "hit" by a point in S, that is, if for every $r \in \Delta_\epsilon(c)$ we have $S \cap r \neq \emptyset$.*

An ϵ-net for $\Delta(c)$ is thus a set that hits all of the ϵ-heavy regions of $\Delta(c)$. As an example, suppose X is the closed interval $[0,1]$ and let $\mathcal{D}$ be the uniform density on X. Suppose that $\mathcal{C}$ consists of all closed intervals on $[0,1]$ as well as the empty set $\emptyset$, and that the target concept $c = \emptyset$. Then $\Delta(c)$ is again the set of all closed intervals on $[0,1]$. For any interval I under the uniform density, $\mathbf{Pr}_{x \in \mathcal{D}}[x \in I]$ is just the length of I. Any interval whose probability is greater than ϵ will have length greater than ϵ, so the set of all points $k\epsilon$, for natural numbers $1 \leq k \leq \lceil 1/\epsilon \rceil$, is an ϵ-net for $\Delta(c)$.

The notion of ϵ-nets has actually been implicit in some of our earlier analyses, in particular those of Occam's Razor in Chapter 2. The important property of ϵ-nets is that if the sample S drawn by a learning algorithm forms an ϵ-net for $\Delta(c)$, and the learning algorithm outputs a hypothesis $h \in \mathcal{C}$ that is consistent with S, then this hypothesis must have error less than ϵ: since $c\Delta h \in \Delta(c)$ was not hit by S (otherwise h would not be consistent with S), and S is an ϵ-net for $\Delta(c)$, we must have $c\Delta h \notin \Delta_\epsilon(c)$ and therefore $error(h) \leq \epsilon$.

Thus if we can bound the probability that the random sample S fails to form an ϵ-net for $\Delta(c)$, then we have bounded the probability that a hypothesis consistent with S has error greater than ϵ. For the case of finite $\mathcal{C}$, the analysis of Occam's Razor obtained such a bound by a simple counting argument that we sketch again here in our new notation: for any fixed error region $c\Delta h \in \Delta_\epsilon(c)$, the probability that we fail to hit $c\Delta h$ in m random examples is at most $(1-\epsilon)^m$. Thus the probability that we fail to hit some $c\Delta h \in \Delta_\epsilon(c)$ is bounded above by $|\Delta(c)|(1-\epsilon)^m$, which in turn is bounded by $|\mathcal{C}|(1-\epsilon)^m$.

Alternatively, we can carry out the above analysis replacing $|\mathcal{C}|$ by $\Phi_d(|X|)$. This follows immediately from the fact that $\mathcal{C} = \Pi_{\mathcal{C}}(X)$ and

Lemma 3.1. This gives us a bound of $\Phi_d(|X|)(1-\epsilon)^m$ on the probability of failing to draw an ϵ-net for $\Delta(c)$. However, this does not represent any progress over the state of affairs in which we began this chapter, since if X is infinite then $\Phi_d(|X|)$ is infinite as well. Ideally, we would like to carry out a similar analysis that instead of considering the entire domain X considers only the small random subset S observed by the learning algorithm.

3.5.2 A Small ϵ-Net from Random Sampling

We now show that if we draw a small set of examples from the oracle $EX(c, \mathcal{D})$, then they form an ϵ-net with high probability. The important property is that the size of the required sample depends on the VC dimension d and ϵ and δ, but is independent of $|\mathcal{C}|$ and $|X|$. From the preceding discussion, this will immediately lead to an upper bound on the number of examples required for PAC learning that depends only on these same quantities.

Suppose that we draw a multiset S_1 of m random examples from $\mathcal{D}$, and let A denote the event that the elements of S_1 fail to form an ϵ-net for $\Delta(c)$. Clearly, our goal is to upper bound the probability of event A. If event A occurs, then by the definition of ϵ-nets, S_1 misses some region $r \in \Delta_\epsilon(c)$. Let us fix this missed region r, and suppose we now draw an additional multiset S_2 of m random examples from $\mathcal{D}$. Since each element of S_2 has probability at least ϵ of hitting r, if $m = O(1/\epsilon)$ the probability S_2 hits r at least $\epsilon m/2$ times is at least $1/2$ by Markov's inequality (see the Appendix in Chapter 9).

If we let B be the combined event over the random draws of S_1 and S_2 that A occurs on the draw of S_1 (so S_1 is not an ϵ-net) *and* S_2 has at least $\epsilon m/2$ hits in a region of $\Delta_\epsilon(c)$ that is missed by S_1, then we have argued that $\mathbf{Pr}[B|A] \geq 1/2$. Since the definition of event B already requires that event A occurs on S_1, we also have $\mathbf{Pr}[B] = \mathbf{Pr}[B|A]\mathbf{Pr}[A]$, so $2\mathbf{Pr}[B] \geq \mathbf{Pr}[A]$.

Thus, we can upper bound the probability of event A by upper bounding the probability of event B. The principal advantage of event B over event A for the purposes of our analysis can be described as follows. To directly analyze the probability of event A, we must consider all regions of the uncountably infinite class $\Delta_\epsilon(c)$ that S_1 might miss. To analyze the probability of event B, we need only consider the regions of $\Pi_{\Delta_\epsilon(c)}(S_1 \cup S_2)$. This is because the occurrence of event B is equivalent to saying that there is some $r \in \Pi_{\Delta_\epsilon(c)}(S_1 \cup S_2)$ such that $|r| \geq \epsilon m/2$ and $r \cap S_1 = \emptyset$.

To bound the probability that such an r exists, rather than drawing S_1 at random and then drawing S_2 at random, we can instead first draw a multiset S of $2m$ instances at random, and then randomly divide S into S_1 and S_2. The resulting distribution of S_1 and S_2 is the same in both experiments, since each draw from $\mathcal{D}$ is independent and identically distributed. Now once S is drawn and fixed (but before it is divided randomly into S_1 and S_2), we may also fix a region $r \in \Pi_{\Delta_\epsilon(c)}(S)$ satisfying $|r| \geq \epsilon m/2$. For this fixed S and fixed r, we now analyze the probability (with respect only to the random partitioning of S into S_1 and S_2) that $r \cap S_1 = \emptyset$. We will then obtain a bound on the probability of event B by summing over all possible fixed $r \in \Pi_{\Delta_\epsilon(c)}(S)$ and applying the union bound.

Our problem is now reduced to the following simple combinatorial experiment: we have $2m$ balls (the multiset S), each colored red or blue, with exactly $\ell \geq \epsilon m/2$ red balls (these are the instances of S that fall in r). We divide these balls randomly into two groups of equal size S_1 and S_2, and we are interested in bounding the probability that all ℓ of the red balls fall in S_2 (that is, the probability that $r \cap S_1 = \emptyset$).

Equivalently, we can first divide $2m$ uncolored balls into S_1 and S_2, and then randomly choose ℓ of the balls to be marked red, the rest being marked blue. Then the probability that all ℓ of the red marks fall on balls S_2 is exactly $\binom{m}{\ell}/\binom{2m}{\ell}$ — this is simply the number of ways we can choose the ℓ red marks in S_2 divided by the number of ways the ℓ red

marks can be chosen without constraints. But $\binom{m}{\ell}/\binom{2m}{\ell} \leq 1/2^{\ell}$. This is because

$$\frac{\binom{m}{\ell}}{\binom{2m}{\ell}} = \Pi_{i=0}^{\ell-1} \frac{(m-i)}{(2m-i)} \leq \Pi_{i=0}^{\ell-1} \left(\frac{1}{2}\right) = \frac{1}{2^{\ell}}.$$

Thus, for any fixed S and $r \in \Pi_{\Delta_\epsilon(c)}(S)$ satisfying $|r| \geq \epsilon m/2$, the probability that the random partitioning of S results in $r \cap S_1 = \emptyset$ is at most $2^{-\epsilon m/2}$. The probability that this occurs for *some* $r \in \Pi_{\Delta_\epsilon(c)}(S)$ satisfying $|r| \geq \epsilon m/2$ (and thus $\mathbf{Pr}[B]$) is at most

$$|\Pi_{\Delta_\epsilon(c)}(S)|2^{-\frac{\epsilon m}{2}} \leq |\Pi_{\Delta(c)}(S)|2^{-\frac{\epsilon m}{2}} \leq |\Pi_{\mathcal{C}}(S)|2^{-\frac{\epsilon m}{2}}$$

$$\leq \Phi_d(2m)2^{-\frac{\epsilon m}{2}} \leq \left(\frac{2em}{d}\right)^d 2^{-\frac{\epsilon m}{2}}.$$

Finally, $\mathbf{Pr}[A] \leq 2\mathbf{Pr}[B] \leq 2(2em/d)^d 2^{-\epsilon m/2}$, which is less than δ for

$$m = O\left(\frac{1}{\epsilon}\log\frac{1}{\delta} + \frac{d}{\epsilon}\log\frac{1}{\epsilon}\right).$$

We have proved the main result of this chapter:

Theorem 3.3 *Let $\mathcal{C}$ be any concept class of VC dimension d. Let L be any algorithm that takes as input a set S of m labeled examples of a concept in $\mathcal{C}$, and produces as output a concept $h \in \mathcal{C}$ that is consistent with S. Then L is a PAC learning algorithm for $\mathcal{C}$ provided it is given a random sample of m examples from $EX(c, \mathcal{D})$, where m obeys*

$$m \geq c_0\left(\frac{1}{\epsilon}\log\frac{1}{\delta} + \frac{d}{\epsilon}\log\frac{1}{\epsilon}\right)$$

for some constant $c_0 > 0$.

Recall that in Chapter 1, we saw that for computational reasons there may sometimes be a great advantage in using a hypothesis class $\mathcal{H}$ that is more powerful than the class $\mathcal{C}$ from which the target is chosen. The reader can verify that the same proof used to establish Theorem 3.3 can be used to prove the following analogue:

Theorem 3.4 *Let $\mathcal{C}$ be any concept class. Let $\mathcal{H}$ be any representation class of VC dimension d. Let L be any algorithm that takes as input a set S of m labeled examples of a concept in $\mathcal{C}$, and produces as output a concept $h \in \mathcal{H}$ that is consistent with S. Then L is a PAC learning algorithm for $\mathcal{C}$ using $\mathcal{H}$ provided it is given a random sample of m examples from $EX(c, \mathcal{D})$, where m obeys*

$$m \geq c_0 \left(\frac{1}{\epsilon} \log \frac{1}{\delta} + \frac{d}{\epsilon} \log \frac{1}{\epsilon} \right)$$

for some constant $c_0 > 0$.

Thus, to obtain an algorithm for PAC learning $\mathcal{C}$ using $\mathcal{H}$, we take a number of examples on the order of the VC dimension of $\mathcal{H}$ (which is at least as large as the VC dimension of $\mathcal{C}$ if $\mathcal{H} \supset \mathcal{C}$). This shows that while we may reduce our computation time by choosing a more powerful hypothesis representation, we may also increase the number of examples required.

3.6 Sample Size Lower Bounds

We now show that the upper bound on the sample complexity of PAC learning given by Theorem 3.3 is tight within a factor of $O(\log 1/\epsilon)$ (ignoring the dependence on δ). First we show a lower bound of $\Omega(d)$ on the number of examples required for PAC learning using a fairly simple argument, then we present a refined argument that improves the bound to $\Omega(d/\epsilon)$.

Theorem 3.5 *Any algorithm for PAC learning a concept class of Vapnik-Chervonenkis dimension d must use $\Omega(d/\epsilon)$ examples in the worst case.*

Proof: Consider a concept class $\mathcal{C}$ such that $VCD(\mathcal{C}) = d$. Let $S = \{x_1, \ldots, x_d\}$ be shattered by $\mathcal{C}$. To show a lower bound, we construct a

particular distribution that forces any PAC learning algorithm to take many examples. Thus, let $\mathcal{D}$ give probability $1/d$ to each point in S, and probability 0 to points not in S. For this distribution, we can assume without loss of generality that $\mathcal{C} = \Pi_{\mathcal{C}}(S)$ (that is, $X = S$), so $\mathcal{C}$ is a finite class and $|\mathcal{C}| = 2^d$.

Note that we have arranged things so that for all of the 2^d possible binary labelings of the points in S, there is exactly one concept in $\mathcal{C}$ that induces this labeling. Thus, choosing the target concept c randomly from $\mathcal{C}$ is equivalent to flipping a fair coin d times to determine the labeling induced by c on S.

Now let L be any PAC learning algorithm for $\mathcal{C}$. Set the error parameter $\epsilon \leq 1/8$, and consider running L when the target concept $c \in \mathcal{C}$ is chosen randomly and the input distribution is $\mathcal{D}$. Suppose that after drawing $m < d$ examples from $EX(c, \mathcal{D})$, L has drawn $m' \leq m$ different instances; without loss of generality, let these be $x_1, \ldots, x_{m'}$. Then from the above observations, it is clear that the problem of predicting the correct label of any unseen instance x_j for $j > m'$ is equivalent to predicting the outcome of a fair coin, since each label of c on S is determined by an independent coin flip. Thus the expected error (over the random choice of c and the sample of points) of L's hypothesis is $(d - m')/2d$, and by Markov's inequality (see the Appendix in Chapter 9) is at least $(d - m')/4d$ with probability at least $1/2$. For $m = d/2$ we obtain that the error of L's hypothesis is at least $1/8$ with probability at least $1/2$ (over the random choice of c and the sample). Since this shows that L must fail when c is chosen randomly, there must certainly be some fixed target concept on which L fails, thus giving the $\Omega(d)$ sample complexity lower bound.

To refine this argument to get a lower bound that incorporates ϵ, we simply scale the above coin flipping construction to a region of the distribution that is small but still too large to be "ignored" by the algorithm. Thus, we modify $\mathcal{D}$ to let the distinguished instance x_1 have probability $1 - 8\epsilon$ under $\mathcal{D}$ (we are essentially "giving" this instance along with its correct label to L), and let $x_2, \ldots, x_d$ each have probability $8\epsilon/(d-1)$

under $\mathcal{D}$ (this is the coin flipping region). Now by simply scaling our previous calculation to the coin flipping region, the expected error of L after seeing at most $d/2$ different instances is at least $(1/8)8\epsilon = \epsilon$ with probability at least $1/2$. But it is not difficult to show that now drawing $d/2$ different points requires $\Omega(d/\epsilon)$ examples, because our problem is reduced to obtaining $d/2$ "successes" in independent trials, each with probability of success only 4ϵ. □(Theorem 3.5)

3.7 An Application to Neural Networks

We conclude this chapter by giving a useful general lemma that bounds $VCD(\mathcal{C})$ when each concept in the class $\mathcal{C}$ is actually a **composition** of simpler concepts. Such classes arise frequently — for instance, a DNF formulae is simply a (very constrained) composition of boolean conjunctions (the constraint being that we can only compute disjunctions of conjunctions). After giving this lemma, we then apply it to obtain upper bounds on the sample size required for PAC learning neural networks.

To formalize a general notion of concept composition, let G be a **layered** directed acyclic graph. By this we mean that the nodes of G can be partitioned into layers, and the directed edges of G go only from a node at layer ℓ to a node at layer $\ell + 1$. We let n be the number of nodes at layer 0, and we assume that all of these have indegree 0. We think of these n layer 0 nodes as being the inputs to the graph. We also assume that there is only a single node of outdegree 0 at the highest level of the graph, and we think of this node as being the output node of the graph. All internal (that is, non-input) nodes have the same indegree r, and we let s denote the number of internal nodes. Figure 3.5 shows an example of such a layered graph with $n = 8$, $s = 8$ and $r = 3$.

Now let $\mathcal{C}$ be a concept class over r-dimensional Euclidean space $\Re^r$. Suppose we take such a layered graph G, and we label each internal (that is, non-input) node N_i with a concept $c_i \in \mathcal{C}$. Then such a labeled

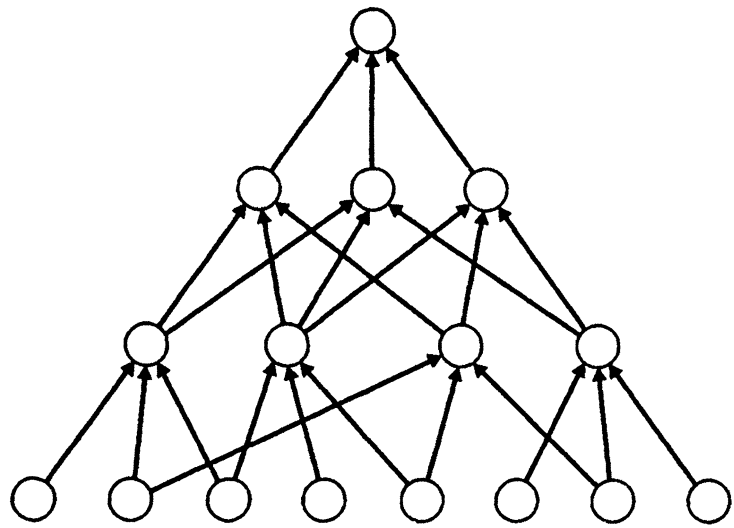

Figure 3.5: *A layered directed acyclic graph.*

graph represents a concept over n-dimensional Euclidean space $\Re^n$ in the obvious way: if we label each of the n input nodes at layer 0 with a real number, then starting with layer 1 we can compute the value at each node N_i by applying the concept c_i labeling node N_i to the values computed at the nodes feeding N_i. (Note that although concepts in $\mathcal{C}$ are defined over $\Re^r$, the input values feeding nodes at level 2 and higher will actually only be from $\{0,1\}^r$.) The output of the entire labeled graph is the binary value computed at the output node. We will call the class of all concepts over $\Re^n$ that can be obtained by labeling G with concepts from $\mathcal{C}$ the **G-composition of $\mathcal{C}$**, which we denote $\mathcal{C}_G$.

Theorem 3.6 *Let G be a layered directed acyclic graph with n input nodes and $s \geq 2$ internal nodes, each of indegree r. Let $\mathcal{C}$ be a concept class over $\Re^r$ of VC dimension d, and let $\mathcal{C}_G$ be the G-composition of $\mathcal{C}$. Then $VCD(\mathcal{C}_G) \leq 2ds\log(es)$.*

Proof: The idea is to first bound the function $\Pi_{\mathcal{C}_G}(m)$. Let us fix any

set S of m input vectors $\vec{x}_1, \ldots, \vec{x}_m \in \Re^n$ to the graph G (thus, each $\vec{x}_i$ determines a complete setting of the n input nodes of G). For this fixed input set S, if we now also label each node in G with a concept from $\mathcal{C}$, then for each $\vec{x}_i$ we have completely determined the binary values that will be computed at every node of G when the input is $\vec{x}_i$. Let us call the collection of all the values computed at each node, for each $\vec{x}_i \in S$, a *computation* of G on S. Thus, a computation can be represented by labeling each internal node with the vector in $\{0,1\}^m$ of the values computed at that node on the m vectors in S. Then the set of all possible computations of G on S is obtained by ranging over all possible choices of labels from $\mathcal{C}$ for the nodes of G. Note that two computations of G on S differ if and only if the value computed at some node on some input from S differs in the two computations. Clearly, $|\Pi_{\mathcal{C}_G}(S)|$ is bounded by the total number of possible computations of G on S, which we shall denote $T_{\mathcal{C}_G}(S)$.

To bound $T_{\mathcal{C}_G}(S)$, let G' be the subgraph obtained by removing the output node N_o from G. Let $T_{\mathcal{C}_{G'}}(S)$ denote the total number of computations of G' on S. Each fixed computation of G' can be extended to at most $\Pi_{\mathcal{C}}(m)$ computations of G, because fixing the computation of G' determines for each $1 \leq i \leq m$ the input $\vec{y}_i \in \{0,1\}^r$ that is fed to N_o when $\vec{x}_i$ is fed to G, and at most $\Pi_{\mathcal{C}}(m)$ labelings of $\vec{y}_1, \ldots, \vec{y}_m$ can be obtained at N_o by varying the choice of concept from $\mathcal{C}$ placed at N_o. Thus we obtain that for any S, $T_{\mathcal{C}_G}(S) \leq T_{\mathcal{C}_{G'}}(S) \times \Pi_{\mathcal{C}}(m)$, and a simple inductive argument establishes

$$|\Pi_{\mathcal{C}_G}(S)| \leq T_{\mathcal{C}_G}(S) \leq (\Pi_{\mathcal{C}}(m))^s \leq \left(\frac{em}{d}\right)^{ds}$$

where the second inequality comes from the polynomial bound on the $\Pi_{\mathcal{C}}(m)$ given in Section 3.4. Since S was arbitrary, this bound in fact holds for $\Pi_{\mathcal{C}_G}(m)$.

Thus in order for $\mathcal{C}_G$ to shatter m points, the inequality $(em/d)^{ds} \geq 2^m$ must hold. Conversely, if $(em/d)^{ds} < 2^m$ for some m, then m is an upper bound on $VCD(\mathcal{C}_G)$. It is easy to verify that this latter inequality holds for $m = 2ds\log(es)$ provided $s \geq 2$. □(Theorem 3.6)

To apply Theorem 3.6 to the problem of PAC learning neural networks, we simply let the function at each node in the graph G be a **linear threshold function**. If the indegree is r, such a function is defined by real **weights** $w_1, \ldots, w_r \in \Re$ and a **threshold** $\Theta \in \Re$. On inputs $x_1, \ldots, x_r \in \Re$ the function outputs 1 if $\sum_{i=1}^{r} w_i x_i \geq \Theta$, and outputs 0 otherwise. We call G the underlying **architecture** of the neural network.

Now as we mentioned in Section 3.3, it is known that the VC dimension of the class of linear threshold on r inputs is $r+1$. By Theorem 3.6 we find that the Vapnik-Chervonenkis of the class of neural networks with architecture G is at most $2(rs+s)\log(es)$, and combined with Theorem 3.3, we obtain:

Theorem 3.7 *Let G be any directed acyclic graph, and let C_G be the class of neural networks on an architecture G with indegree r and s internal nodes. Then the number of examples required to learn C_G is*

$$O\left(\frac{1}{\epsilon}\log\frac{1}{\delta} + \frac{(rs+s)\log s}{\epsilon}\log\frac{1}{\epsilon}\right).$$

3.8 Exercises

3.1. Compute the VC dimension of the class of boolean conjunctions of literals over $\{0,1\}^n$.

3.2. Consider the concept class over the Euclidean plane $\Re^2$ consisting of the interior regions of circles; thus, the positive examples of each concept form a disk in the plane. Compute the VC dimension of this class. Compute the VC dimension of the class of interiors of triangles in the plane.

3.3. Show that there is no 1-decision list over $\{0,1\}^n$ computing the exclusive-or function $x_1 \oplus x_2$. Then show that the VC dimension of

1-decision lists over $\{0,1\}^n$ is $\Theta(n)$, and that the VC dimension of k-decision lists is $\Theta(n^k)$. Hint: show that 1-decision lists over $\{0,1\}^n$ compute linearly separable functions (halfspaces). You may use the fact that the VC dimension of halfspaces over $\Re^n$ is linear in n.

3.4. Let $P_{d,k}$ be the class of concepts over $\Re^d$ defined by convex polytopes with k sides; thus, each the positive examples of each concept in $P_{d,k}$ are defined by the convex intersection of k halfspaces in $\Re^d$. Give the best upper and lower bounds that you can on $VCD(P_{d,k})$. You may use the fact that the VC dimension of halfspaces over $\Re^d$ is linear in d.

3.5. Let $\mathcal{C}$ be any concept class of VC dimension d over X, and let $\mathcal{D}$ be any distribution over X. Suppose we are given access to a source of random (unlabeled) instances drawn according to $\mathcal{D}$, and also access to an oracle that for any labeled sample of points will return "Yes" if there is a concept in $\mathcal{C}$ that is consistent with the labeled sample, and will return "No" otherwise. Describe an algorithm that on input any finite set of instances $S \subseteq X$ and any $\epsilon, \delta > 0$ will output either the answer "Yes, S in an ϵ-net for $\mathcal{C}$ with respect to $\mathcal{D}$", or the answer "No, S is not an $\epsilon/4$-net for $\mathcal{C}$ with respect to $\mathcal{D}$". Moreover, the algorithm must give a correct answer with probability at least $1-\delta$. The algorithm need not be efficient. (The quantity $\epsilon/4$ in the "No" condition can in fact be replaced by $\alpha\epsilon$ for any fixed constant $\alpha < 1$, giving an arbitrarily refined test.)

3.6. Prove that the bound of $\Phi_d(m)$ on $\Pi_{\mathcal{C}}(m)$ is tight: that is, for any concept class $\mathcal{C}$ of VC dimension d and any m, there exists a set S of m points such that $|\Pi_{\mathcal{C}}(S)| = \Phi_d(m)$.

3.7. In this exercise we consider the two-oracle model of PAC learning defined in Exercise 1.3 of Chapter 1. We say that a concept class $\mathcal{C}$ is **PAC learnable from positive examples alone** if it is PAC learnable by an algorithm that only draws from the oracle $EX(c, \mathcal{D}_c^+)$ when learning target concept $c \in \mathcal{C}$ (the hypothesis must still meet the two-sided error criterion). We have already seen in Chapter 1 that boolean conjunctions are efficiently PAC learnable from positive examples alone. This exercise

ignores computational considerations, and concentrates on the number of examples required for learning from positive examples alone.

We say that a subclass $\mathcal{C}' \subseteq \mathcal{C}$ has **unique negative examples** if for every $c \in \mathcal{C}'$, there is an instance $x_c \in X$ such that $x_c \notin c$ but $x_c \in c'$ for every other $c' \in \mathcal{C}'$. We define the **unique negative dimension** of the class $\mathcal{C}$, $UND(\mathcal{C})$, to be the cardinality of the largest subclass $\mathcal{C}'$ that has unique negative examples.

Prove that any algorithm learning $\mathcal{C}$ from positive examples alone (regardless of computation time or the hypothesis class used) requires $\Omega(UND(\mathcal{C})/\epsilon)$ positive examples.

Then prove that $O(UND(\mathcal{C})/\epsilon)$ positive examples are sufficient for learning from positive examples alone by the following steps. Consider the algorithm that takes a sample S of positive examples of the target concept and returns the hypothesis

$$h = \min_{\mathcal{C}}(S) = \bigcap_{c \in \mathcal{C}: S \subseteq c} c.$$

Note that h may not be contained in $\mathcal{C}$, and also that this algorithm will never err on a negative example of the target concept.

First show that if on random samples S of size d/ϵ (where $d = UND(\mathcal{C})$) from $EX(c, \mathcal{D}_c^+)$, the expected error of $\min_{\mathcal{C}}(S)$ with respect to $\mathcal{D}_c^+$ exceeds ϵ, then there must exist a set $S^* \subseteq c$ of size $d/\epsilon + 1$ with the property that for a fraction at least ϵ of the $x \in S^*$, $x \notin \min_{\mathcal{C}}(S^* - \{x\})$. Then show that this implies that $UND(\mathcal{C}) > d$, a contradiction.

Thus, $\Theta(UND(\mathcal{C})/\epsilon)$ positive examples are necessary and sufficient for learning from positive examples alone, and the unique negative dimension plays a role analogous to the Vapnik-Chervonenkis dimension for this model of learning.

3.9 Bibliographic Notes

The classic paper on the VC dimension, and the one in which the main elements of the proof of Theorem 3.3 are first introduced, is by Vapnik and Chervonenkis [95]. These ideas were introduced into the computational learning theory literature and elaborated upon in the influential work of Blumer, Ehrenfeucht, Haussler and Warmuth [22]. Vapnik has also written an excellent book [94] that greatly extends the original ideas into a theory known as structural risk minimization.

The VC dimension and its attendant theorems have been influential in the neural network and artificial intelligence machine learning communities. The calculation of the VC dimension of neural networks is due to Baum and Haussler [13], and Abu-Mostafa [1] and Tesauro and Cohn [89] examine VC dimension issues from a neural network perspective. Haussler [45] examines the VC dimension as a form of inductive bias from an artificial intelligence viewpoint.

The value of the VC dimension as a measure of the sample complexity of learning transcends the PAC model; many authors have shown that the VC dimension provides upper or lower bounds on the resources required for learning in many models. These include on-line models of learning (Haussler, Littlestone and Warmuth [51]; Maass and Turán [69]; Littlestone [66]), models of query learning (Maass and Turán [69]); and many others.

The VC dimension has also been generalized to give combinatorial complexity measures that characterize the sample complexity of learning in various extensions of the PAC model. Perhaps the most general work along these lines in the computational learning theory literature has been undertaken by Haussler [48], who draws on work in statistics, notably the work of Pollard [74] and of Dudley [31]. Haussler's general framework is examined carefully in the context of learning probabilistic concept by Kearns and Schapire [61], who prove that a certain generalization of the VC dimension provides a lower bound on sample size for learning in this

model, and by Alon et al. [4], who give an upper bound.

The VC dimension and its generalizations are only one of the many ways that computational learning theory and statistics attempt to quantify the behavior of *learning curves*, that is, the error of the hypothesis as a function of the number of examples seen. For instance, among the many alternative methods of analysis are theories based on tools from information theory and statistical physics [50, 86].

The $\Omega(d/\epsilon)$ sample size lower bound is due to Ehrenfeucht et al. [33], who also give the solution to Exercise 3.3. Exercise 3.7 is due to Gereb-Graus [39].