Machine Learning, 14, 271-294 (1994)

The Power of Self-Directed Learning

SALLY A. GOLDMAN (SG@CS.WUSTL.EDU)
Department of Computer Science, Washington University, St. Louis, MO 63130

ROBERT H. SLOAN (SLOAN@TURING.EECS.UIC.EDU)
Department of Electrical Engineering and Computer Science, University of Illinois at Chicago, Chicago, IL 60680

Editor: David Haussler

Abstract. This article studies self-directed learning, a variant of the on-line (or incremental) learning model in which the learner selects the presentation order for the instances. Alternatively, one can view this model as a variation of learning with membership queries in which the learner is only "charged" for membership queries for which it could not predict the outcome. We give tight bounds on the complexity of self-directed learning for the concept classes of monomials, monotone DNF formulas, and axis-parallel rectangles in $\{0, 1, \ldots, n-1\}^d$. These results demonstrate that the number of mistakes under self-directed learning can be surprisingly small. We then show that learning complexity in the model of self-directed learning is less than that of all other commonly studied on-line and query learning models. Next we explore the relationship between the complexity of self-directed learning and the Vapnik-Chervonenkis (VC-)dimension. We show that, in general, the VC-dimension and the self-directed learning complexity are incomparable. However, for some special cases, we show that the VC-dimension gives a lower bound for the self-directed learning complexity. Finally, we explore a relationship between Mitchell's version space algorithm and the existence of self-directed learning algorithms that make few mistakes.

Keywords. computational learning theory, on-line learning, incremental learning, self-directed learning

1. Introduction

This article studies self-directed learning, a variant of the on-line (or incremental) learning model in which the learner selects the presentation order for the instances. As in the standard on-line model, the learner answers a sequence of yes/no questions with immediate feedback provided after each question. The learner strives to discover the correct classification rule while making as few mistakes as possible. Clearly, the performance of the learner depends on the order in which the questions are presented. Although typically the assumption made is that an adversary selects the order of the questions, here we allow the learner to select this order. Alternatively, one can view this model as a variation of learning with membership queries in which the learner is only "charged" for membership queries for which it could not predict the outcome. We apply this variant of the on-line learning model to problems from the area of concept learning. That is, when the learner chooses the instance sequence, how many incorrect predictions are made before the target concept is uniquely specified?

This article is organized as follows. In the next section, we motivate the self-directed learning model. In section 3, we give some preliminary definitions that set the framework

for the self-directed learning model. Then in section 4 we formally describe the model of self-directed learning (as originally defined by Goldman, Rivest, & Schapire, 1993). In section 5, we briefly discuss some related work. In section 6, we give tight bounds on the complexity of self-directed learning for the concept classes of monomials, monotone DNF formulas, and axis-parallel rectangles in $\{0, 1, \cdots, n-1\}^d$. These bounds demonstrate that the number of mistakes can be surprisingly small. Then in section 7 we prove that the model of self-directed learning is more powerful than all other commonly used on-line and query learning models. In particular, we show that this model is more powerful than all the models considered by Maass and Turán (1992). Next, in section 8, we study the relationship between the optimal mistake bound under self-directed learning and the Vapnik-Chervonenkis (VC-)dimension. We first show that the VC-dimension can be arbitrarily larger than the complexity of self-directed learning. We then give a family of concept classes for which the complexity of self-directed learning is larger than the VC-dimension. Next we show that for concept classes of VC-dimension 1, the self-directed learning complexity is 1. We also show that for *any* maximum[1] class, the VC-dimension provides a lower bound for the self-directed learning complexity. In section 9, we explore a relationship between Mitchell's version-space algorithm (1982) and the existence of self-directed learning algorithms that make few mistakes. Finally, in sections 10 and 11, we consider some alternative models, summarize the results of this article, and suggest some directions for future research, including a succinct combinatorial characterization of one of our open problems.

2. Motivation

In this section, we explain why we feel the self-directed learning model is an interesting one to study.

2.1. Example: the allergist

We begin by reviewing the allergist example from the original paper on self-directed learning (Goldman, Rivest, & Schapire, 1993).

Consider an allergist with a set of patients to be tested for a given set of allergens. Each patient is either *highly allergic*, *mildly allergic*, or *not allergic* to any given allergen. The allergist may use either an *epicutaneous* (scratch) test in which the patient is given a fairly low dose of the allergen, or an *intradermal* (under the skin) test in which the patient is given a larger dose of the allergen. The patient's reaction to the test is classified as *strong positive*, *weak positive*, or *negative*. Figure 1 describes the reaction that occurs for each combination of allergy level and dosage level. Finally, we assume a strong positive reaction is extremely uncomfortable to the patient, but not dangerous.

What options does the allergist have in testing a patient for a given allergen? He could just perform the intradermal test (option 0). Another option (option 1) is to perform an epicutaneous test, and if it is not conclusive, then perform an intradermal test. (See figure 2 for decision trees describing these two testing options.) Which testing option is best?

	Intradermal (Under the Skin)	Epicutaneous (Scratch)
Not Allergic	negative	negative
Mildly Allergic	weak positive	negative
Highly Allergic	strong positive	weak positive

Figure 1. Summary of testing reactions for allergy testing example.

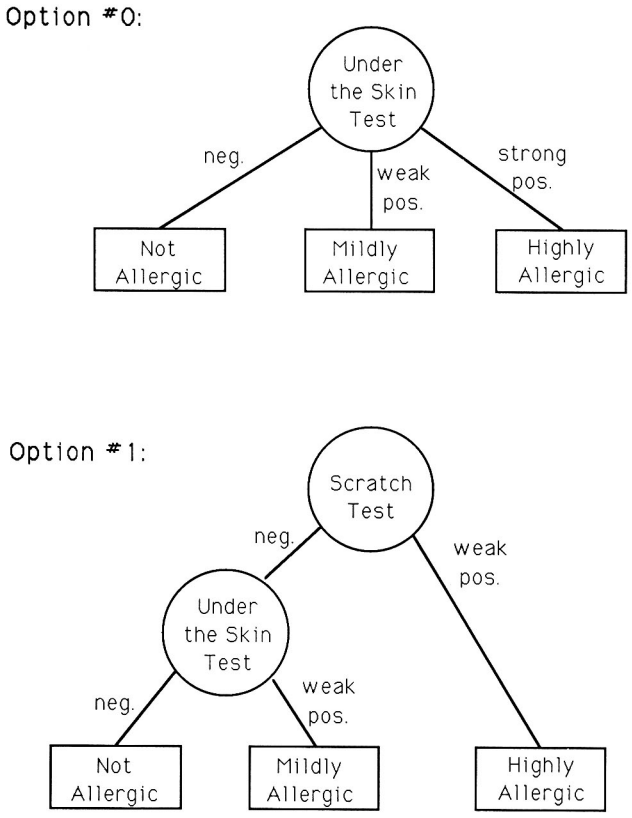

Figure 2. The testing options available to the allergist.

If the patient has no allergy or a mild allergy to the given allergen, then testing option 0 is best, since the patient need not return for the second test. However, if the patient is highly allergic to the given allergen, then testing option 1 is best, since the patient does not experience a bad reaction. Let us assume that the inconvenience of going to the allergist twice is approximately the same as having a bad reaction. That is, the allergist has no

preference for one sort of error over the other. While the allergist's final goal is to determine each patient's allergies, we consider the problem of learning the optimal testing option for each combination of patient and allergen.

The allergist interacts with the environment as follows. In each "trial," the allergist is asked to predict the best testing option for a given patient/allergen pair. He is then told the testing results, thus learning whether the patient is not allergic, mildly allergic, or highly allergic to the given allergen. In other words, the allergist receives feedback as to the correct testing option.

Observe that to model this situation, we want an on-line model. There is no training phase here; the allergist wants to predict the correct testing option for each patient/allergen pair. How should we judge the performance of the learning algorithm? For each wrong prediction made, a patient is inconvenienced with either making a second trip or having a bad reaction. Since the learner wants to give all patients the best possible service, he strives to minimize the number of incorrect predictions made. Thus we want to judge the performance of the learning algorithm by the number of incorrect predictions made during a learning session in which the learner must eventually test each patient for each allergen.

Since the allergist has no control over the target relation (i.e., the allergies of his patients), it makes sense to view the feedback as coming from an adversary. However, do we really want an adversary to select the presentation order for the instances? It could be that the allergist is working for a cosmetic company and, due to restrictions of the Food and Drug Administration and the cosmetic company, the allergist is essentially told when to test each person for each allergen. In this case, it is appropriate to have an adversary select the presentation order. Typically, however, the allergist chooses the order in which to perform the testing so that he can make the best predictions possible. In this case, we want to allow the learner to select the presentation order.

Clearly, the model of self-directed learning is also appropriate for other similar testing situations. The important features are a need to perform a set of tests, in which the outcome of previously performed tests can be used to help select the best method for performing the remaining test. Furthermore, since all tests must eventually be performed, there is no need to minimize the total number of queries (or tests) performed; rather, the goal is to minimize the number of incorrectly predicted instances.

2.2. Self-directed learning as a variation of standard models

Another way to view the self-directed learning model is as a modification of the standard model of learning with membership queries (as introduced by Angluin, 1988), in which the learner aims to achieve exact identification using membership queries, except that in our case the learner is only "charged" for a membership query if it incorrectly predicts the classification. Thus, it is not at all surprising that the self-directed learning complexity will always be less than or equal to the number of membership queries needed to obtain exact identification. The interesting question is *how much less* the self-directed learning complexity can be than the minimum number of membership queries needed for exact identification. While the self-directed learning model provides a measure of the minimum number

of *mistakes* that the learner makes (versus a measure of the minimum number of *queries* made), it is important to understand how this new learning model relates to the standard on-line learning model and the various models of learning with queries.

We will consider this question right after providing a crisp formal definition of self-directed learning.

3. Preliminary definitions

We now formally describe the on-line (or incremental) learning model. The basic goal of the learner is to accurately predict whether each of a given set of objects (or *instances*) is a positive or negative instance of the *target concept*. Let the instance space X denote the set of instances to be classified, and let the concept class be $C \subseteq 2^X$. (Typically, X and C are parameterized according to some size measure n. In this case, we will write $C = (C_n, X_n)$. For a concept $c \in C$ and instance $x \in X$, $c(x)$ denotes the classification of c on instance x. That is, $c(x) = 1$ if and only if $x \in c$. We say that x is a positive instance of c if $c(x) = 1$ and x is a negative instance of c if $c(x) = 0$. Finally, a *hypothesis* h for C is a rule that given any $x \in X$ outputs in polynomial time a prediction for $c(x)$.

We are now ready to define the absolute mistake-bound variant of the on-line learning model (Haussler, Littlestone, & Warmuth, 1988; Littlestone, 1988). An *on-line algorithm* (or *incremental algorithm*) for C is an algorithm that runs under the following scenario.[2] A *learning session* consists of a set of *trials*. In each trial, an *adversary*[3] presents the learner with an unlabeled instance $x \in X$. The learner uses its current hypothesis to predict whether x is a positive or negative instance of the *target concept* $c_* \in C$, and then the learner is told the correct classification of x. If the prediction is incorrect, the learner has made a *mistake*. The goal of the learner is to minimize the number of mistakes made over the learning session.

We now define the Vapnik-Chervonenkis dimension (Vapnik & Chervonenkis, 1971). Let X be any instance space, and let C be a concept class over X. A finite set $Y \subseteq X$ is *shattered* by C if $\{c \cap Y \mid c \in C\} = 2^Y$. In other words, $Y \subseteq X$ is shattered by C if for each subset $Y' \subseteq Y$, there is a concept $c \in C$ that contains all of Y', but none of the instances in $Y - Y'$. The *Vapnik-Chervonenkis dimension* of C, denoted $\text{VCD}(C)$, is defined to be the smallest d for which no set of $d + 1$ points is shattered by C. Blumer et al. (1989) have shown that this combinatorial measure of a concept class characterizes the number of examples required for learning any concept in the class under the distribution-free or PAC model of Valiant (1984).

Related to the VC-dimension are the notions of *maximal* and *maximum* concept classes (Floyd, 1989; Weizl, 1987). A concept classis *maximal* if adding any concept to the class increases the VC-dimension of the class. Define

$$\Phi_d(m) = \begin{cases} \Sigma_{i=0}^{d} \binom{m}{i} & \text{for } m \geq d \\ 2^m & \text{for } m < d. \end{cases}$$

If C is a concept class of VC-dimension d on a finite set X with $|X| = m$, then the cardinality of C is at most $\Phi_d(m)$ (Sauer, 1972; Vapnik & Chervonenkis, 1971). A concept class C over X is *maximum* if for every finite subset $Y \subseteq X$, the class C, when restricted to be a class over Y, contains $\Phi_d(|Y|)$ concepts.

Finally, we describe membership and equivalence queries as originally defined by Angluin (1988).

- A *membership query* is a call to an oracle that on input x for any $x \in X$ classifies x as either a positive or negative instance according to the target concept $c_* \in C$.
- An *equivalence query* is a call to an oracle that on input $c \in C$ either replies that the target concept c_* is equivalent to c or provides an instance $x \in X$ such that c and c_* classify x differently. That is, the oracle either replies that the conjectured concept is correct or provides a counterexample to it.
- A *generalized equivalence query* is just like an equivalence query except that the conjectured hypothesis can be any element of 2^X.

4. The self-directed learning model

An important contribution of the on-line learning model is that it applies to problems in which the learner must predict the classification of unseen instances. Sometimes this learning model appropriately captures the interaction between the learner and its environment; yet as indicated by the allergist example, sometimes it does not. In studying the problem of learning binary relations, Goldman, Rivest, and Schapire (1993) introduced the model of *self-directed learning*, in which the learner selects the order in which the instances are presented to the learner.

The focus of this article is studying this learning model when applied to the standard problems of concept learning. That is, we shall apply the model of self-directed learning to some commonly studied concept classes and compare this learning model to other on-line and query learning models that have been studied.

The self-directed learning model is defined as follows. Note that these definitions only apply to finite instance spaces (and, of course, to countable sequences of finite instance spaces). We define a *query sequence* to be a permutation $\pi = \langle x_1, x_2, \ldots, x_{|X|} \rangle$ of the instance space X, where x_t is the instance the learner will predict at the t^{th} trial. The learner may build its query sequence in an on-line manner. Namely, for the t^{th} trial, x_t may be selected by any polynomial time algorithm that takes as input the set of labeled examples obtained in the first $t - 1$ trials. Note that by the definition of a query sequence, x_t must be an instance that the learner has not yet considered. Furthermore, if, after the completion of the t^{th} trial, the learner knows with certainty the classification of all instances from X that have not yet been queried (i.e., $x_{t+1}, \ldots, x_{|X|}$), then we say the learning session is completed. The *mistake bound* for learning concept c with a given self-directed learning algorithm is the number of incorrect predictions made during the learning session. In other words, it is the number of incorrect predictions that are made by the self-directed learning algorithm until the point at which the target concept is uniquely specified. For any nonempty concept class C and any self-directed learning algorithm, we define the mistake bound of the learning algorithm for C to be the maximum of the mistake bound for each concept $c \in C$.

The *optimal mistake bound* for the self-directed learning of concept class C, denoted $opt(C)$, is the minimum over all self-directed learning algorithms for C of the mistake bound. We define the *self-directed learning complexity* of a concept class C, written as $\text{SDC}(C)$, to be $opt(C)$.

5. Previous work

We now briefly discuss some relevant previous work. As we have mentioned, Goldman, Rivest, and Schapire (1993) use the self-directed learning model (described in section 4) to study the problems of learning binary relations and total orders. In particular, for a binary relation over n objects with m attributes per object, they describe an efficient self-directed learning algorithm that makes at most $km + (n - k) \lfloor \lg k \rfloor$ mistakes, where k is the number of distinct objects.[4] Furthermore, this bound is shown to be asymptotically tight. For the concept class of a total order over n objects, they give an efficient self-directed learning algorithm that achieves an optimal mistake bound of $n - 1$ mistakes.

Maass and Turán's (1992) work comparing the complexity of learning under the commonly studied on-line and query learning models is quite useful. In addition to comparing previously defined models, they defined a new learning model of partial equivalence queries, in which for the instance space X the learner can present a hypothesis $h : X \to \{0, 1, *\}$, and is then either told that all specified instances are correct or is given an $x \in X$ such that $h(x) \in \{0, 1\}$ and x is misclassified by h. We will add to their results by showing how the model of self-directed learning fits into their hierarchy relating various on-line and query learning models.

6. Self-directed learning complexity for various concept classes

In this section, we compute bounds on the self-directed learning complexity for the concept classes of monotone monomials, monotone DNF formulas, axis-parallel rectangles in $\{0, 1, \cdots, n - 1\}^d$, and multiples in $\mathbf{N}$. As we shall show, the learner can perform extremely well for these concept classes.

6.1. Monomials

We first compute the self-directed learning complexity of monotone monomials, and then we generalize our results to arbitrary monomials.

Theorem 1. $\text{SDC}(\text{monotone monomials}) = 1$.

Proof. The learner uses the following query sequence, always predicting that the instances are negative, and stopping when the first mistake occurs. First, consider the instance in which all variables are 0. Next, consider the n instances in which a single variable is 1. Then consider the $\binom{n}{2}$ instances in which two variables are 1, and so on. (See figure 3.)

$$
\begin{array}{cccccccc}
0 & 0 & 0 & \cdots & 0 & 0 & 0 & , \; - \\
1 & 0 & 0 & \cdots & 0 & 0 & 0 & , \; - \\
0 & 1 & 0 & \cdots & 0 & 0 & 0 & , \; - \\
 & & & \vdots & & & & \\
0 & 0 & 0 & \cdots & 0 & 1 & 0 & , \; - \\
0 & 0 & 0 & \cdots & 0 & 0 & 1 & , \; - \\
1 & 1 & 0 & \cdots & 0 & 0 & 0 & , \; - \\
1 & 0 & 1 & \cdots & 0 & 0 & 0 & , \; - \\
 & & & \vdots & & & & \\
0 & 0 & 0 & \cdots & 1 & 0 & 1 & , \; - \\
0 & 0 & 0 & \cdots & 0 & 1 & 1 & , \; - \\
1 & 1 & 1 & \cdots & 0 & 0 & 0 & , \; -
\end{array}
$$

Figure 3. Learner-selected query sequence for learning the monotone monomial $x_1x_2x_3$. Note that the last instance is the first one incorrectly predicted by the learner.

Let c_* be the target monomial, and let c be the monomial consisting of the variables that are 1 in the incorrectly predicted instance x. We now prove that $c = c_*$. Clearly, any variable in c_* must also be in c—if variable v is in c_*, then v must be 1 in any positive example. Suppose that some irrelevant variable v is 1 in the incorrectly predicted instances, yet it is not in c_*. Consider the positive instance x', which is the same as x except that v is 0. Clearly, x' must precede x in the query sequence defined above. Since the learner reaches the instance x, it follows that x' must be a negative instance, giving the desired contradiction. ■

We now modify these ideas to handle the situation in which some variables in the monomial may be negated.

Theorem 2. SDC(monomials) = 2.

Proof. The algorithm used here is a simple modification of the algorithm for learning monotone monomials. Suppose that the learner knew the sign of each variable. Then the learner can use the algorithm for learning monotone monomials where setting a variable to 0 (respectively, 1) is interpreted as setting the variable so that the corresponding literal is false (respectively, true).

We then use a standard trick to learn the sign of each variable at a cost of only one mistake (Littlestone, 1988). For arbitrarily chosen instances, predict that each one is negative until a mistake is made. Let x be the positive instance obtained on the first mistake. The sign of each relevant variable is given by its assignment in x.

Finally, observe that the adversary can force the learner to make two mistakes—intuitively, one to learn the sign of the variables and one to determine which variables are relevant. ■

6.2. *Monotone DNF formulas*

In this section, we consider learning monotone DNF formulas under self-directed learning. We obtain our algorithm for learning monotone DNF formulas of m terms by extending the algorithm of theorem 1 to handle the conjunction of m monotone monomials.

Theorem 3. SDC(monotone DNF) $\leq m$ *where* m *is the number of terms in the target DNF formula.*

Proof sketch: The algorithm used here is a modification of that described in theorem 1. The query sequence selected is like the one shown in figure 3, except that an instance x is predicted as positive if the monomial corresponding to any incorrectly predicted instance predicts that x is positive. Using the same technique as in the proof of theorem 1, one can show that the target formula is just the disjunction of the monomials corresponding to the incorrectly predicted instances. ■

6.3. *Axis-parallel rectangles in* $\{0, 1, \cdots, n-1\}^d$

Finally, in this section we consider the concept class BOX_n^d of axis-parallel rectangles in $\{0, 1, \cdots, n-1\}^d$. (This class has previously been studied by Maass and Turán (1992) for other learning models.)

Theorem 4. $\text{SDC}(\text{BOX}_n^d) = 2$.

Proof. We begin by describing a self-directed learning algorithm for BOX_n^d that makes only two mistakes. Select two opposing corners of the space $\{0, 1, \cdots, n-1\}^d$. Let L be the line through these two opposing corners. Our query sequence finds each corner of the target box by approaching it with a hyperplane perpendicular to L. That is, for each opposing corner, query the following set of instances, predicting that each is negative. (See figure 4.) So dovetailing is used to find each corner. See figure 5 for an example of the portion of the query sequence for learning one corner of a box in two-dimensional space.

```
Learn-box-corner(corner pt)
        Query the corner point, predicting it is negative
        While no mistake has occurred
                Choose any unseen point of minimum Manhattan distance (L1 norm
                  distance) from the corner point.
                Query this unseen point, predicting it is negative.
```

Figure 4. The algorithm for finding a corner of the box.

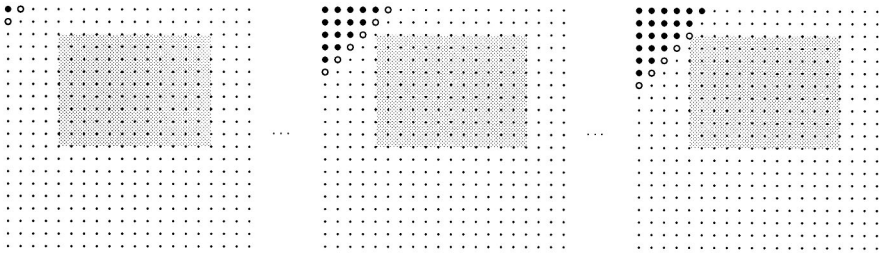

Figure 5. The query sequence for learning a concept from BOX_n^2. The filled circles in the figure represent correctly predicted negative instances. The unfilled circles represent the instances on the frontier. Finally, note that in the last figure the querying stops when a mistake is made, predicting that the corner of the box is a negative instance.

At the first mistake, a corner point of the box has been found. Since the target box is approached with a hyperplane perpendicular to the axis between the two corner points of the space, the first point of the box queried must be the corner point. Then, the second corner is found in the same manner.

Finally, we argue that the adversary can force the learner to make at least two mistakes. Clearly, the learner cannot exactly determine the target concept until two opposing corners of the target box have been found. Furthermore, the learner can be forced to make a mistake in finding each corner. ■

6.4. Multiples of integers

As one more example, consider a concept class $C = (X_n, C_n)$ where the self-directed learning complexity is 1 for each C_n, but the VC-dimension grows monotonically with n. Let the instances space X_n be the natural numbers less than or equal to n, and let the concept class be all multiples of i for each $i \in \mathbf{N}$. This class has VC-dimension of roughly $\ln n / \ln \ln n$ (Helmbold, Sloan & Warmuth, 1992)[5]. Nevertheless, we now show that $\text{SDC}(\text{MULTIPLES}) = 1$. The learner predicts the instance x (initially 1) is a negative instance. If a mistake is made, then the target is just all multiples of x. Otherwise, increment x and repeat the above procedure. Clearly, this procedure finds the target concept while making only a single mistake. Thus, we have the following.[6]

Theorem 5. *The concept class of multiples (of natural numbers bounded by n) has a constant self-directed learning complexity of 1.*

7. Relation to other on-line and query learning models

In this section, we build on the results of Maass and Turán (1992) by showing that the model of self-directed learning is more powerful than all the on-line and query learning models that they considered. This result is not so surprising, since the model of self-directed

learning uses a different system of charging for errors than all other models. We first informally describe the models that they studied. (See their paper for formal definitions.)

LC	The model of learning with standard equivalence queries that must come from the concept class. (Or, equivalently, the standard on-line learning model in which the adversary selects the instances and the learner predicts according to some concept in the concept class.)
LC-ARB	The model of learning from generalized equivalence queries. (Or, equivalently, the standard on-line learning model in which the adversary selects the instances.)
LC-PARTIAL	The model of partial equivalence queries in which the learner can present a hypothesis $h : X \to \{0, 1, *\}$, and is then either told that all specified instances are correct or is given an $x \in X$ such that $h(x) \in \{0, 1\}$ and x is misclassified by h.
MEMB	The model of learning from membership queries.
LC-MEMB	The model of learning from membership queries and equivalence queries that must come from the concept class.
LC-ARB-MEMB	The model of learning from membership queries and generalized equivalence queries.

Mass and Turán (1992) showed that the LC-PARTIAL learning model is more powerful than all of the other learning models. That is, for all concept classes, the complexity under LC-PARTIAL is strictly less than the complexity under any of the other models. Furthermore, there exists a concept class such that the complexity of LC-PARTIAL is exponentially less than that of any other model. We now show that the model of self-directed learning is even more powerful than the model of learning with partial equivalence queries.

Theorem 6. *For any concept class C,* $\text{SDC}(C) \leq \text{LC-PARTIAL}(C)$. *Furthermore, there is a concept class C' such that* $\text{SDC}(C') < \text{LC-PARTIAL}(C')$.

Proof. Let $\text{LC-PARTIAL}(C) = m$, and let algorithm A be an algorithm using partial equivalence queries that achieves this bound. We now use algorithm A to create a self-directed learning algorithm A' that demonstrates that $\text{SDC}(C) \leq m$. For each partial equivalence query made by algorithm A, algorithm A' simulates it as shown in the algorithm in figure 6.

```
Simulate-partial-equivalence-query(h)
    For each x ∈ X for which h(x) ∈ {0,1}
        Make a prediction (x, h(x))
    Until a mistake occurs
```

Figure 6. Algorithm to simulate a partial equivalence query under the model of self-directed learning.

If the answer to the partial equivalence query is "yes," then no mistakes will be made by A'. However, if there is a counterexample to the partial equivalence query, then A' will find one while only making a single mistake. Thus, the number of mistakes made by A' is at most the number of partial equivalence queries made by A.

Finally, note that we showed that $\text{SDC}(\text{BOX}_n^d) = 2$, whereas $\text{LC-PARTIAL}(\text{BOX}_n^d) = \Theta(d \log n)$ (Maass & Turán, 1989). Combined with the results of Maass and Turán (1992), this proves that the model of self-directed learning is strictly more powerful than all of the learning models mentioned above. ■

8. Relation to the Vapnik-Chervonenkis dimension

In this section, we study the relationship between the VC-dimension and the self-directed learning complexity. We have already seen three concept classes for which the VC-dimension can be arbitrarily larger than the self-directed learning complexity. In particular, we have seen that SDC(monotone monomials) = 1, yet VCD(monotone monomials) = n where n is the number of variables, and $\text{SDC}(\text{BOX}_n^d) = 2$, whereas $\text{VCD}(\text{BOX}_n^d) = 2d$ for all large enough n, and SDC(MULTIPLES) = 1, whereas VCD(MULTIPLES) grows logarithmically in n, where n is the largest integer allowed. Observe that for all concept classes considered so far, $\text{SDC}(C) \leq \text{VCD}(C)$. We now describe a concept class for which the optimal mistake bound under self-directed learning is in fact greater than the VC-dimension.

Theorem 7. *There exists a family of concept classes* $C = (X_n, C_n)$ *with* $\text{SDC}(C_n) = 3$ *and* $\text{VCD}(C_n) = 2$.

Proof sketch. We begin by describing concept classes induced by simple planar arrangements as defined in Floyd's thesis (1989). A *simple planar arrangement* G is the dissection of the plane into cells by a finite set of lines with the property that no two lines may be parallel and no three lines can intersect at a single point. A simple planar arrangement can be seen as a concept class C with $\text{VCD}(C) = 2$. The m lines $x_1, \ldots, x_m$ in the arrangement G correspond to the m instances, and each cell in the arrangement corresponds to a concept in C. The classification for an instance is obtained as follows: each line x_i defines two halfspaces in the plane for which all cells on one side of the halfspace classify x_i as positive, and all cells on the other side classify x_i as negative. We now define the concept class C_n of an n-gon. For any odd $n \geq 3$, the concept of an n-gon is defined by taking the linear arrangement obtained by extending the line segments forming a regular n-gon. Assume that the halfspaces are oriented so that all instances are negative in the interior of the n-gon. See figures 7 and 8 for geometric and tabular representations of a 5-gon. Observe that the number of cells in the linear arrangement is $\Phi_d(m)$, and thus C_n is a maximum class of VC-dimension 2.

We now sketch the proof that for odd $n \geq 5$, $\text{SDC}(n\text{-gon}) = 3$. First, we will describe a self-directed learning algorithm that makes only three mistakes. The learner predicts according to the concept corresponding to the interior of the n-gon (all instances negative) until the first mistake occurs. Let x_i be the instance for which the first mistake occurs. (So without loss of generality, assume that a mistake is made on the first prediction.) We

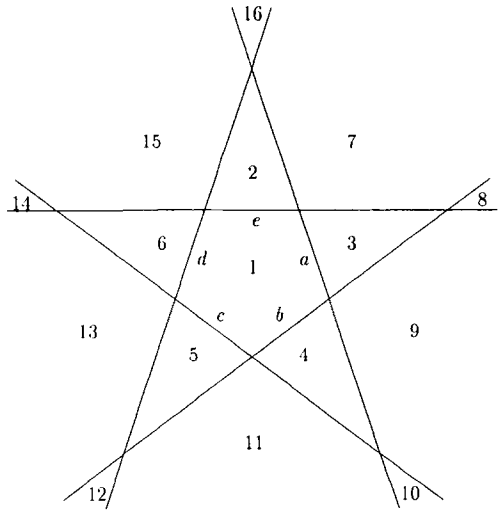

Figure 7. The concept class of a 5-gon drawn geometrically.

	a	b	c	d	e
c_1	0	0	0	0	0
c_2	0	0	0	0	1
c_3	1	0	0	0	0
c_4	0	1	0	0	0
c_5	0	0	1	0	0
c_6	0	0	0	1	0
c_7	1	0	0	0	1
c_8	1	1	0	0	1
c_9	1	1	0	0	0
c_{10}	1	1	1	0	0
c_{11}	0	1	1	0	0
c_{12}	0	1	1	1	0
c_{13}	0	0	1	1	0
c_{14}	0	0	1	1	1
c_{15}	0	0	0	1	1
c_{16}	1	0	0	1	1

Figure 8. The tabular representation of a 5-gon. Concept c_i corresponds to region i in figure 7.

now number the remaining lines according to the order in which they cross x_i. (For ease of exposition, orient the planar arrangement so that x_i lies on the x-axis and the n-gon lies

below the x-axis. Thus, after the first mistake, we can restrict our attention to above the x-axis.) Let $\ell_1, \ldots, \ell_{(n-1)/2}$ be the first $(n-1)/2$ lines that cross x_i moving right from $-\infty$. Thus ℓ_1 forms the leftmost crossing. Likewise, let $r_1, \ldots, r_{(n-1)/2}$ be the first $(n-1)/2$ lines that cross x_i moving left from $+\infty$. The learner now performs the predictions according to the algorithm given in figure 9.

Let $C' \subset C$ be the hypotheses that remain after this second mistake has occurred. (If the above procedure makes no mistakes, then only a single concept remains.) Without loss of generality, suppose this mistake occurred when predicting $(l_i, 0)$. Note that the hypotheses in C' correspond to the regions in the portion R of the arrangement between l_{i-1} and l_i that are above the x-axis. (See figure 10.) The key observation needed here is that there are *no* intersections in the *interior* of R. There may be lines passing through R (in fact, the lines passing through R are what define the regions corresponding to the concepts in C') but these lines do not intersect in the interior of R. Thus by considering the concepts in C' according to the order in which they occur when moving through R starting at the x-axis, the learner can ensure that a single concept remains after the next prediction mistake.

Now we must argue that SDC(n-gon) ≥ 3. We will show how an adversary can force the learner to make at least three mistakes. Assume without loss of generality that the learner's first prediction is for x_1. Regardless of the learner's prediction, the adversary informs the learner that it has made a mistake.

If the learner predicted x_1 negative, then let ℓ_i and r_i be as in the part of the proof showing that SDC(n-gon) ≤ 3. The adversary selects $\ell_j = 0$ and $r_j = 0$ for $0 \leq j \leq (n-1)/2 - 1$ (i.e., the learner is billed for a mistake on those lines if and only if it predicts positive). This allows all four possible combinations of the regions defined by $\ell_{(n-1)/2}$ and $r_{(n-1)/2}$, so whatever the learner predicts for each of those two lines will be called a mistake by the adversary.

If the learner predicted that x_1 was positive, then find the line x_j such that x_1 is ℓ_1 with respect to x_j. Now the same argument as before holds, with x_j serving the place of x_1, and the initial mistake being predicting ℓ_1 positive. ■

While, in general, the VC-dimension and the self-directed learning complexity are incomparable, there are two special cases in which VC-dimension is a lower bound for the self-directed learning complexity. We begin by considering the special case in which the VC-dimension is 1.

For $i = 1, \ldots, (n-1)/2$
  Predict $(\ell_i, 0)$
  If $(\ell_i = 0)$ **Then** Predict $(r_i, 0)$
Until a mistake occurs.

Figure 9. The second phase of the self-directed learning algorithm for learning n-gons. The algorithm for finding a corner of the box.

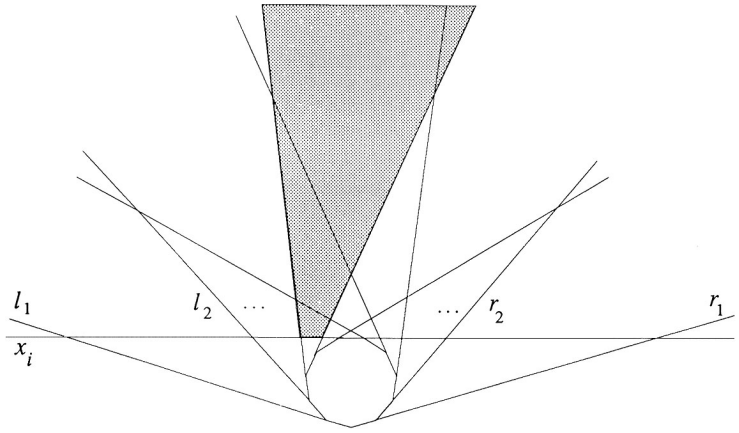

Figure 10. The shaded region corresponds to the portion of the arrangement that remains after prediction mistakes are made on $(x_i, 0)$ and $(\ell_4, 0)$.

Theorem 8. *For any concept class C over a finite instance space for which* $\text{VCD}(C) = 1$, $\text{SDC}(C) = 1$.

Proof. Since $\text{VCD}(C) = 1$, the concept class C must have at least two distinct concepts, and thus it immediately follows that $\text{SDC}(C) \geq 1$. We now show that $\text{SDC}(C) \leq 1$. Any class C of VC-dimension 1 can be embedded in a maximum class C' such that $\text{VCD}(C') = 1$.[7] Consider the 1-inclusion graph for C' that consists of a node for each concept in C' and an edge (with label x) between two nodes whose corresponding concepts differ only in the classification of x. For a maximum concept class of VC-dimension 1 over a finite instance space, the 1-inclusion graph is a tree, and every instance x appears exactly once as a label on an edge (Floyd, 1989; Wenzl & Woeginger, 1986). The learning algorithm will select the next instance to predict as follows. Select an example x that is a label to an edge adjacent to a leaf node. Let l be the classification of x by the concept c associated with the leaf node. The learner makes the prediction $(x, \bar{l})$. If this prediction is wrong, then the target concept is c. Otherwise, c and its edge are removed from the tree, and this process can be repeated. ■

We now show that for any maximum class, the VC-dimension is a lower bound for the self-directed learning complexity. This result reveals an interesting connection between the self-directed learning model and Floyd's (1989) work on space-bounded learning. Before giving this result, we briefly discuss data compression schemes. A *data compression* scheme of size k for concept class C consists of a *compression function* f and a *reconstruction function* g. The compression function f maps every set of $m \geq k$ labeled examples to a subset of at most k labeled examples. The reconstruction function g maps every possible subset of at most k labeled examples to a hypothesis c on X. (This hypothesis is not required to be in C.) Finally, for a data compression scheme, it is required that for any set S_m of labeled examples, the hypothesis $g(f(S_m))$ must be consistent with S_m.

Theorem 9. *For any maximum class C over a finite instance space,* $\text{SDC}(C) \geq \text{VCD}(C)$.

Proof. Let $d = \text{VCD}(C)$. If C is a maximum class of VC-dimension d on the set X, then there is a data compression scheme of size d for C (Floyd, 1989). For any concept $c \in C$, let S_c be the set of labeled examples obtained by applying the compression function to the entire instance space labeled according to c. For ease of exposition, let X_c denote the unlabeled sample obtained by removing the labels from S_c. Now, since C is a maximum class, it follows from theorem 3.6 of Floyd (1989) that for any labeling ℓ of the 2^d possible labelings of X_c, there is a concept in C with the instances in X_c labeled with ℓ and all instances in $X - X_c$ labeled as in c. Thus, any learning algorithm can be forced to make d mistakes in identifying this concept. ■

Remark. If we extend our definition of self-directed learning to allow infinite instance spaces, then theorem 9 still holds for maximum concept classes over infinite instance spaces. (For this case, the proof would use theorem 3.7 of Floyd's (1989) thesis.)

9. Relation to Mitchell's version space algorithm

We have demonstrated that the number of mistakes made under self-directed learning may be quite small. Is there some characterization for the situations in which the learner can perform so well? As a partial answer to this question, we describe a relation between this work and Mitchell's (1982) version space algorithm.

We begin by describing Mitchell's version space algorithm. The *version space* with respect to a given sample and concept class C is the set of *all* concepts $c \in C$ that are consistent with the sample.[8] The hope is that the version space will shrink over time until it contains only one concept, which must be the target concept.

The set G contains all the *most general* (maximal) concepts in the version space. The set S contains all the *most specific* (minimal) concepts in the version space. Hence, S and G together delimit the "borders" of the version space in the lattice of all concepts with respect to the subset relation (see figure 11.) The main idea of the version space algorithm

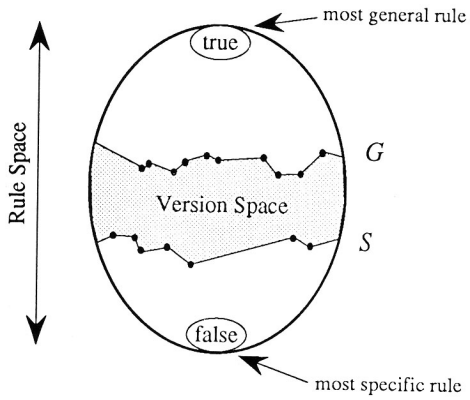

Figure 11. The rule space.

is as follows. Initially, let G contain only the concept containing all instances, and let S contain only the empty concept. Then, for each example, both G and S are appropriately updated. See Haussler (1987, 1988), and also Rivest and Sloan (in press) for a discussion of connections between Mitchell's version space algorithm and the distribution-free learning model.

We now describe a relation between the version space algorithm and situations in which the learner can perform well under self-directed learning. Consider a concept class C such that, given any set of instances labeled according to some concept in C, there is a unique most specific concept (i.e., $|S| = 1$). We now define the notion of a *spanning set*. A spanning set of a concept $c \in C$ with respect to the class C is a set $I \subseteq c$ having the property that c is the unique most specific concept consistent with the instances in I. (This generalizes the notation of a spanning set of an intersection-closed class given by Helmbold, Sloan, and Warmuth (1990)). Finally, we define $I(C)$ for concept class C as follows:

$$I(C) = \max_{c \in C} \{|I_c| : I_c \text{ is a minimal spanning set for } c \text{ with respect to } C\}.$$

To provide some intuition for our more general result, we first consider the simple case for which $I(C) = 1$ and the concept class C has a unique most specific concept consistent with any sample. For example, the class of monotone monomials[9] has the property that the set S never contains more than one hypothesis (Bundy, Silver, & Plummer, 1985). Furthermore, for any monomial c, any minimal spanning set I_c is just the single instance for which all the variables in c are 1 and the rest are 0. Thus, for the class of monotone monomials, $I(C) = 1$.

We now describe a self-directed learning algorithm that makes a single mistake. The algorithm goes through all the concepts from most general to most specific (i.e., goes through the layers of the lattice from top to bottom), and for each concept c it predicts that the instance I_c is negative. We claim that when the first mistake is made, the target concept is the single concept in S. Clearly, if the first mistake is made on the prediction $(I_c, 0)$, then the target is c or a generalization of c. However, since the concepts are considered from most general to most specific, all concepts that are generalizations of c must already have been eliminated from the version space. Thus, the target concept must be c, which is the concept in S. Observe that this general technique when applied to the class of monotone monomials yields exactly the algorithm described in theorem 1.

Figure 12 describes an algorithm that generalizes the above algorithm for the case in which $I(C) > 1$. We now prove the correctness of this algorithm. We first argue that if a mistake occurs on all predictions made in step 2b in figure 12, then it follows that all instances in I_c are positive. Since $c \in G$, it follows that c is consistent with all previously seen instances, and thus all instances in I_c that have previously been queried are positive. Furthermore, since a mistake occurs on all predictions made in step 2b, the remaining instances in I_c are also positive.

Now by the definition of a spanning set, it follows that the single most specific concept consistent with the previously seen examples must be the target or a generalization of the target. Finally, since the concepts are considered from most general to most specific, all concepts that are generalizations of the target must have already been eliminated, and thus the algorithm is correct.

1. Let G be the set of most general concepts in C.
2. For each $c \in G$
   (a) Let I_c be a minimal spanning set for c with respect to C.
   (b) For all $x \in I_c$ such that x has not yet been queried
      - Make the prediction $(x, 0)$.
      - If the prediction is correct remove from G all concepts for which x is positive and return to step 2.
   (c) If all predictions made in step (2b) were incorrect then output S.
   (d) Else update G (some new concepts may need to be added) and return to step 2.

Figure 12. Self-directed learning algorithm for the case in which $|S| = 1$.

Without any further constraints on the selection of the minimal spanning sets, this algorithm could potentially make $|C|(I(C) - 1) + 1$ mistakes, since $I(C) - 1$ mistakes could occur on the concepts that are generalizations of the target and $I(C)$ mistakes could be made on the target. (Observe that the learner makes this many mistakes only if every concept has a spanning tree of size $I(C)$.) Let P be the set of previously seen positive instances during the execution of the algorithm described in figure 12. A sufficient property to obtain a good mistake bound is that for each concept c in G, there exists a minimal spanning set for c that contains all instances in P. Given that this additional condition can be met, we obtain the following result.

Theorem 10. *If for concept class C there is always a unique most specific concept consistent with any set of examples (i.e., we always have* $|S| = 1$*), and for each concept* $c \in G$ *there exists a minimal spanning set for c with respect to C that contains all previously seen positive instances, then there exists a self-directed learning algorithm that makes at most* $I(C)$ *mistakes.*

Proof. The self-directed learning algorithm that achieves the $I(C)$ mistake bound is the algorithm shown in figure 12, where the spanning set selected in step 2a meets the conditions of the theorem. We have already argued that the algorithm is correct. Thus, we need just argue that it makes at most $I(C)$ mistakes. Observe that when the learner correctly predicts an instance, the current concept being considered is eliminated from the version space and the learner then considers some concept $c_G \in G$. By the condition placed on the selection of the minimal spanning sets, if any mistakes were previously made, then those instances must be in I_{c_G}. Thus it can easily be shown by induction that this algorithm makes at most $I(C)$ mistakes.

As an application, observe that for axis-parallel rectangles in $\{0, 1, \cdots, n-1\}^d$ this theorem applies where, for all $c \in C$, I_c contains the pair of positive instances from two opposing corners. (Note that it is essential that the same two opposing corners are used for each concept.) This algorithm is like the algorithm described in theorem 4, except that it interleaves the steps described for learning the two corners.

We conjecture that the condition required in theorem 10 holds for all intersection-closed concept classes. Combined with the result from Helmbold, Sloan, and Warmuth (1990), the fact that $I(C) \leq \text{VCD}(C)$ would give the following.

Conjecture 11. *For all intersection-closed concept classes,* $\text{SDC}(C) \leq \text{VCD}(C)$.

We now extend theorem 10 to concept classes that are made of the disjunction of concepts from a concept class for which it currently applies.

Theorem 12. *Let C be a concept class for which theorem 10 applies. Then a concept of the form* $c_1 \vee c_2 \vee \cdots \vee$ *for* $c_1, \ldots, c_k \in C$ *can be learned with at most* $k \cdot I(C)$ mistakes under self-directed learning.

Proof sketch. The algorithm used here is a modification of the algorithm given in figure 12. Let the learner's hypothesis h initially be empty. The algorithm used here proceeds just as the above algorithm except that if for instance x the hypothesis h classifies x as positive, then the learner (correctly) predicts that x is positive. Finally, in step 2c, if a mistake is made on all predictions, then the concept in S is added to h and then it returns to the start of the second step. Now using the same technique as in the proof of theorem 10, we can show that the target formula is just the disjunction of the concepts corresponding to the incorrectly predicted instances. Thus, at most $I(C)$ mistakes are made when placing any new concept in h. Furthermore, at most k, concepts are put in h. Thus, we get the desired mistake bound. ■

Applying the result of Bundy, Silver, and Plummer (1985) to this theorem, we get the result of theorem 3.

Finally, all the results we have described relating the number of mistakes made under a learner-selected query sequence to Mitchell's version space algorithm can be modified to give the dual results for when $|G| = 1$.

10. Alternate definitions and variations on the model

Before concluding, we would like to discuss briefly some of the choices we made in defining our model, and some variations that we believe are worthy of further study.

10.1. Extension to infinite instance spaces

By modifying the definition of query sequence, one could, of course, define the self-directed learning complexity for any concept class over an arbitrary instance space. We had several

reasons for not doing so. First of all, this seems contrary to the spirit of the model. When first introducing this model, Goldman, Rivest, and Schapire (1993) applied it to a concept class with a polynomial-sized instance space. Furthermore, the learner was expected to eventually query each instance, so there was no concern about the number of correctly predicted queries. (Although we do allow the learner to stop the learning session once the learner knows the classification of all instances that have not yet been queried.) Our goal in this article has been to apply this learning model to the more usual case of concept classes in which the instance space is super-polynomial. This may be a stretch, since the learner has to have some interaction with all of the instances, but this question still seems interesting if we are at least limited to a finite number of instances.

It is also the case that we would get odd results from allowing infinite instance spaces. Were we to go all the way and allow uncountable instance spaces, then there would be many geometry-based instance spaces with infinite self-directed learning complexity. For example, let X be the interval [0, 1] and let C be the class of all closed initial segments, i.e., $C = \{[0, r] : 0 \leq r \leq 1\}$. It is easily seen that for this class, $\text{SDC}(C) = \infty$, since there is no way to make the queries "in the right order" to avoid making infinitely many mistakes in the worst case.

While at first it may appear that restricting X to be countable would correct this difficulty, observe that even if we let X be the rationals in [0, 1], the self-directed learning complexity is still infinite.[10]

10.2. Restricting the learner's queries

Given that the learner is restricted to use polynomial time in selecting the next query for the query sequence, is it reasonable to allow the learner to make a super-polynomial number of correctly predicted queries? This question suggests an interesting variation of the self-directed learning model that we do not explore here, namely, how does the self-directed learning complexity change if we restrict the learner to use only polynomial time and make only a polynomial number of queries? In this variation of the model, the learning complexity would still be the number of incorrectly predicted queries, but learning algorithms must be designed to limit the total number of queries to be polynomial.

10.3. A subset-based model

Another interesting variation on the basic model is to allow an all-powerful adversary to select some finite subset S of X. Then the learner must choose queries from the subset S, on-line, in such a manner that all instances in S are eventually queried (or at least enough are queried so the learner knows the correct classification of all the instances in S). Here too, the learner will be charged only for incorrectly predicted queries. The learning complexity would then be the worst-case taken over all target concepts and all finite subsets S of X. The key differences here are that the learner cannot make any queries on instances not in S, and the learner need not obtain exact identification, but should only minimize

the number of mistakes on the given subset of instances. Observe that under this variation of the model, the VC-dimension of the concept class is now a lower bound for the learning complexity, since the adversary can choose S to be some shattered set. As we have seen, such behavior does not occur in the original model of self-directed learning—there are concept classes for which the self-directed learning complexity is much less than the VC-dimension. While this is an interesting variation to study, restricting the learner to only query instances in S greatly reduces the learner's power by taking away a key feature of both the self-directed learning model and membership query model—the learner no longer has the capability of learning the classification of some "important" instance. Nevertheless, this is an interesting variation to study and fits in well with the allergist example in the situation in which the allergist cannot test some patients for some allergens.

11. Conclusions and open problems

We have demonstrated that the model of self-directed learning is quite powerful. In particular, we have shown the self-directed learning complexity is less than the learning complexities of most other commonly studied models. Given that the learner is only charged for incorrectly predicted queries, the "power" of this model is really no surprise. However, as we have shown for several well-studied concept classes, a surprisingly small number of mistakes can be made under self-directed learning. While the complexity of self-directed learning can be arbitrarily smaller than the VC-dimension, in general, the VC-dimension is not an upper bound for the self-directed learning complexity. Namely, for the family of concept classes of n-gons for odd $n \geq 5$, we have shown that $\text{VCD}(n\text{-gon}) = 2$, whereas the $\text{SDC}(n\text{-gon}) = 3$. In contrast to that example, we showed that all concept classes of VC-dimension 1 over finite instance spaces have self-directed learning complexity 1. Also, for any maximum concept class, we showed that the VC-dimension gives a lower bound for the self-directed learning complexity. Finally, we explored a relationship between Mitchell's version space algorithm and the existence of self-directed learning algorithms that make few mistakes.

There are many interesting directions for future research. In section 10, we described two variations of the self-directed learning model that raise interesting questions for future research. It would be interesting to understand how these variations of the self-directed learning model relate to the model studied here and the standard on-line and query learning models.

Another very intriguing open problem is either to prove conjecture 11 or to give a counterexample to it. Along these lines, we are very interested in finding an answer to the following question: Is there a concept class C over some finite set[11] for which $\text{SDC}(C) = \omega(\text{VCD}(C))$, or one can prove that for all C, $\text{SDC}(C) \leq \alpha \cdot \text{VCD}(C)$ for some constant α? Or even better, can it be shown that $\text{SDC}(C) \leq \text{VCD}(C) + \beta$ for some constant β? (As far as we know, this may hold for $\beta = 1$.)

We now describe a nice combinatorial characterization of this open problem, as observed by David Haussler (personal communication). Littlestone (1988) defined a *mistake tree* for target class C over an instance space X as a decision tree in which each node is a nonempty subset of C and each internal node is labeled with a point of X, and that satisfies the following:

1. The root of the tree is C.
2. Given any internal node C' labeled with x, the left child of C', if present, is the subset of C' for which x is 0, and the right child, if present, is the subset of C' for which x is 1.

Ehrenfeucht and Haussler (1989) gave the following definition of the *rank* of a decision tree Q, denoted $r(Q)$.

1. If Q contains only one node, then $r(Q) = 0$.
2. Else, if r_0 is th rank of the left subtree of Q, and r_1 is the rank of the right subtree of Q, then

$$r(q) = \begin{cases} \max(r_0, r_1) & \text{if } r_0 \neq r_1 \\ r_0 + 1 & \text{otherwise} \end{cases}$$

Littlestone (1988) showed that for any concept class C, the worst-case mistake bound in the standard on-line learning model is equal to the largest integer k such that there is a mistake tree for C such that every leaf has depth at least k. Observe that the rank of a mistake tree corresponds to the minimum depth of a leaf. Thus, Littlestone's result can be restated as follows:

$$M_A(C) = \max_{c \in C} \{\max_{t \in T_c} r(t)\}$$

where $M_A(C)$ denotes the optimal-mistake bound for target class C (in the standard on-line learning model), and T_c denotes the set of mistake trees for target concept c. As part of the proof of this result, Littlestone introduces an algorithm he calls the standard optimal algorithm. Observe that every target concept and instance sequence define a mistake tree for C. Littlestone shows that by selecting a mistake tree with rank r, the adversary can force the learner to make r mistakes; furthermore, he shows that the standard optimal algorithm will make at most r mistakes. Recall that the standard on-line learning model and the self-directed learning model are distinguished by whether the adversary or learner selects the order in which the instances are presented. Thus, Littlestone's proof can be easily modified to show

$$\text{SDC}(C) = \max_{c \in C} \{\min_{t \in T_c} r(t)\}.$$

Thus, the only difference is that now the learner is selecting the query sequence and thus can select one that generates a mistake tree of minimum rank. Thus we ask: How does the VC-dimension of C relate to $\max_{c \in C}$ {minimum rank of a mistake tree for c}?

Acknowledgments

Sally A. Goldman is supported in part by a GE Foundation Junior Faculty Grant and NSF Grant CCR-9110108. Part of this research was conducted while the author was at the M.I.T. Laboratory for Computer Science and supported by NSF grant DCR-8607494 and a grant from the Siemens Corporation.

Robert H. Sloan is supported in part by NSF grant CCR-9108753. Portions of this research were performed while the author was at the Harvard Aiken Computation Laboratory, supported by ARO grant DAAL 03-86-K-0171.

We are grateful to David Haussler and an anonymous referee for their extremely insightful comments about both the presentation and technical content of an earlier draft of this article. Both variations of the self-directed learning models discussed in section 10 and the results of theorems 8 and 9 are due to them. We thank Bernd Gaertner for suggesting we use the concept class of a 5-gon as an example for which the VC-dimension is less than the self-directed learning complexity. We also thank Sally Floyd for useful discussions on this matter. Finally, we thank Ron Rivest for many useful conversations on the material presented here.

Notes

1. A concept class of VC-dimension d over a finite instance space is *maximum* (or complete) if its size is maximum among concept classes of VC-dimension d over the given sized instance space.
2. Such algorithms have also been referred to in the learning theory literature as *prediction algorithms*.
3. The adversary, who tries to maximize the learner's mistakes, knows the learner's algorithm and has unlimited computing power.
4. Two objects are distinct if there exists an attribute that they classify differently.
5. To see that the VC-dimension grows with N, consider for any d all products of $d - 1$ of the first d primes. This set is shattered.
6. Note that if we did not restrict the definition of self-directed learning to finite instance spaces, we would have a concept class with infinite VC-dimension and self-directed learning complexity of 1. That choice of definition would lead to other difficulties, however. See the discussion below in section 10.
7. As discussed in section 6.2 of Floyd's thesis (1989), this is not true of all classes of VC-dimension greater than 1.
8. In the version space literature, the concept class is usually called the *rule space*, and is thought of as being the learner's hypothesis space, although it is usually implicitly assumed that the target concept does in fact come from the rule space.
9. Actually, this is true for the more general class of pure conjunctive concepts over a finite set of tree-structured attributes.
10. One might try to limit self-directed learning to countably infinite instance spaces where the "natural" order on the instance space is not dense.
11. We are interested in finite versus countable instance spaces, for the reasons discussed in section 10.1.

References

Angluin, Dana. (1988). Queries and concept learning. *Machine Learning, 2(4)*, 319–342.

Blumer, Anselm, Ehrenfeucht, Andrzeg, Haussler, David, & Warmuth, Manfred K. (1989). Learnability and the Vapnik-Chervonenkis dimension. *Journal of the Association for Computing Machinery, 36(4)*, 929–965.

Bundy, A., Silver, B., & Plummer, D. (1985). An analytical comparison of some rule-learning programs. *Artificial Intelligence, 27,* 137-181.

Ehrenfeucht, Andrzej, & Haussler, David. (1989). Learning decision trees from random examples. *Information and Computation, 82(3),* 231-246.

Floyd, Sally. (1989). *On space-bounded learning and the Vapnik-Chervonenkis dimension.* Ph.D. thesis, International Computer Science Institute, Berkeley, CA.

Goldman, Sally A., Rivest, Ronald L., & Schapire, Robert E. (1993). Learning binary relations and total orders. *SIAM Journal on Computing, 22(5),* 1006-1034.

Haussler, David. (1987). Learning conjunctive concepts in structural domains (Technical Report UCSC-CRL-87-1). Santa Cruz, CA: Computer Research Laboratory, University of Santa Cruz.

Haussler, David. (1988). Quantifying inductive bias: AI learning algorithms and Valiant's learning framework. *Artificial Intelligence, 36,* 177-221.

Haussler, David, Littlestone, Nick, & Warmuth, Manfred K. (1988). Predicting {0, 1}-functions on randomly drawn points. In *29th Annual Symposium on Foundations of Computer Science,* White Plains, NY (pp. 100-109), IEEE Computer Society Press: Los Alamitos, CA.

Helmbold, David, Sloan, Robert, & Warmuth, Manfred K. (1990). Learning nested differences of intersection-closed concept classes. *Machine Learning, 5,* 165-196. Special issue for COLT 89.

Helmbold, David, Sloan, Robert, & Warmuth, Manfred K. (1992). Learning integer lattices. *SIAM Journal on Computing, 21(2),* 240-266.

Littlestone, Nick. (1988). Learning when irrelevant attributes abound: A new linear-threshold algorithm. *Machine Learning, 2,* 285-318.

Maass, Wolfgang, & Turán, György. (1989). On the complexity of learning from counterexamples. In *30th Annual Symposium on Foundations of Computer Science,* Research Triangle Park, NC (pp. 262-267), IEEE Computer Society Press: Los Alamitos, CA.

Maass, Wolfgang, & Turán, György. (1992). Lower bound methods and separation results for on-line learning models. *Machine Learning, 9,* 107-145.

Mitchell, Thomas M. (1982). Generalization as search. *Artificial Intelligence, 18,* 203-226.

Rivest, Ronald L., & Sloan, Robert (In press). A formal model of hierarchical concept learning. *Information and Computation.*

Sauer, A. (1972). On the density of families of sets. *Journal of Combinatorial Theory (A), 13,* 145-147.

Valiant, Leslie. (1984). A theory of the learnable. *Communications of the ACM, 27(11),* 1134-1142.

Vapnik, V.N., & Chervonenkis, A. Ya. (1971). On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and Its Applications, XVI(2),* 264-280.

Welzl. E. (1987). Complete range spaces. Unpublished manuscript.

Welzl, E., & Woeginger, G. (1986). On Vapnik-Chervonenkis dimension one. Unpublished manuscript.

Received May 14, 1992
Accepted December 8, 1992
Final Manuscript February 18, 1993