A survey of some aspects of computational learning theory
(Extended abstract)

GYÖRGY TURÁN*
Department of Mathematics, Statistics and Computer Science
University of Illinois at Chicago, Chicago
Automata Theory Research Group of the Hungarian Academy of Sciences, Szeged

1. Introduction

The goal of computational learning theory is to establish formal models of the process of learning, and to understand what can be and what cannot be learned efficiently in these models. In this way one hopes to obtain results useful for the growing number of computer applications of learning, and perhaps to gain insight into the human learning process as well. Among the many different aspects of learning it is the task of learning a concept which has received greatest attention from the theoretical point of view. We say that we learned a concept if we are able to distinguish between its positive and negative instances. It is assumed that the distinction is made on the basis of a rule or a definition specifying which instances are positive and which ones are negative. Concept learning is usually illustrated by the example of learning the concept of an elephant by obtaining a definition like "has four legs and a trunk". This rule is selected from a large set of potential rules which are conjunctions of Boolean predicates such as "has wings", "is red", etc. In general, it is assumed that the concept to be learned, called the target concept, is selected from a class of possible concepts called the concept class, which is fixed in advance, for example by assuming as above, that it has a representation of a prespecified form. Thus formally, learning a concept is equivalent to identifying (exactly or approximately) a set from a given class of possibilities. A formal model of concept learning is further specified by determining what are the means of identification, i.e. what is a learning algorithm, and what are the criteria of successful identification. Typically the learning algorithm is provided with a sample of the target concept and it may also have the opportunity to present hypotheses and to query an oracle about the target concept. What distinguishes computational learning theory from other related fields such as inductive inference, pattern recognition, machine learning and neural computing is that it tries to establish formal models of learning and puts the emphasis on the efficiency of learning algorithms by determining upper and lower bounds for the complexity of a learning problem. Another distinguishing feature is that, as noted above, its favorite animal appears to be the elephant, as opposed to the penguin.

Among the early developments we mention Rosenblatt's perceptron convergence theorem (Rosenblatt (1962)), the influential book of Minsky and Papert (1988) and Gold's work on inductive inference (Gold (1967)). Computational learning theory as a separate

---

*Partially supported by OTKA-501. E-mail: U11557 @ UICVM.BITNET

field was started several years later by Valiant's paper on probably approximately correct (PAC) learning (Valiant (1984)), which initiated an impressive amount of research in a few years. In this paper we try to give a brief overview of some of the results obtained. For many others we refer to the proceedings of the annual COLT conferences. Among the important topics not discussed here we mention the learning of functions (Haussler (1989)) and probabilistic concepts (Kearns and Schapire (1990)), learning in the presence of errors (see e.g. Sloane (1988)), distribution dependent results (see e.g. Linial, Mansour and Nisan (1989), Faigle and Kern (1990)) and space bounded learning (Floyd (1989)).

2. Learning problems

A *learning problem* is specified by a set X of possible *instances,* a set $\mathcal{C} \subseteq 2^X$ of *concepts,* called the *concept class* and a set $\mathcal{H} \subseteq 2^X$ of *hypotheses,* called the *hypothesis space.* The goal is to learn an unknown *target concept* $C \in \mathcal{C}$ by using hypotheses from $\mathcal{H}$ (the details of the models are given in the next section). We usually (but not always, see the examples below) write $X = \bigcup_{n\geq 1} X_n$, $\mathcal{C} = \bigcup_{n\geq 1} \mathcal{C}_n$, $\mathcal{H} = \bigcup_{n\geq 1} \mathcal{H}_n$, where $X_n = A^n$ for some set A such as $\{0,1\}$ or $\mathbf{R}$, and $\mathcal{C}_n$ (resp. $\mathcal{H}_n$) is the restriction of $\mathcal{C}$ (resp. $\mathcal{H}$) to X_n.

As one actually deals with *representations* of concepts, it is assumed that instances, concepts and hypotheses are encoded as words over some finite or infinite alphabet in such a way that it can be decided in polynomial time whether an instance belongs to a concept or hypothesis. A concept or a hypothesis may have several different representations. Each concept and hypothesis has a *size,* which is typically the length of a shortest representation. The computational models are standard, sometimes one considers machines with oracles. We note that for an infinite alphabet such as $\mathbf{R}$, one can consider either the uniform cost or the logarithmic cost model of computation (see Blumer, Ehrenfeucht, Haussler and Warmuth (1989) for a discussion of these options).

We mention some typical learning problems. In the first three examples we consider the Boolean domain $X_n = \{0,1\}^n$ for every $n \geq 1$. Unless mentioned otherwise, the hypothesis space is the same as the concept class. The representations are standard.

k-term DNF: concepts are defined by a disjunction of at most k conjunctions of literals from $\{x_1, \ldots, x_n\}$. (Thus formally $\mathcal{C}_n = \{C \subseteq X_n :$ for some DNF $\phi = c_1 \vee \ldots \vee c_\ell, \ell \leq k$ over the variables $x_1, \ldots, x_n$ it holds that $C = \{\mathbf{x} \in \{0,1\}^n : \phi$ is true for $\mathbf{x}\}\}$.)

k-term DNF with k-CNF as hypotheses: the same as above, but the hypotheses can be arbitrary conjunctions of disjunctions of at most k literals from $\{x_1, \ldots, x_n\}$.

Halfspaces over $\{0,1\}^n$ (or *Boolean threshold functions*): concepts are defined by a linear inequality $\alpha_1 x_1 + \ldots + \alpha_n x_n \geq t$, where $\alpha_1, \ldots, \alpha_n, t \in \mathbf{Z}$.

d-dimensional boxes over a discrete space: $X_n^d = \{0, \ldots, n-1\}^d$, concepts are axis-parallel d-dimensional rectangles. Note that here the parameter n plays a different role as above.

Unions of boxes in a fixed Euclidean space: $X = \mathbf{R}^d$, concepts are arbitrary finite unions of axis-parallel d-dimensional rectangles.

Deterministic finite automata (DFA): $X = \{0,1\}^*$, the concepts are the regular languages. The representations of the concepts are standard encodings of DFA.

3. Models

In this section we introduce several formal learning models. The models to be presented are of two different types. In the PAC models approximate learning is achieved with high probability by random sampling. In the on-line models the learning algorithm may ask queries and it is required to identify the target concept exactly. We formalize the basic models and some of their modifications.

3.1 Probably approximately correct (PAC) learning

A positive (resp. negative) *example* of a concept C is a pair $(x,+)$ (resp. $(x,-)$) for some $x \in C$ (resp. $x \notin C$). An example of C is a positive or a negative example of C. A *sample* of size m of C is a sequence of m examples of C.

The *error* of a hypothesis H with respect to a concept C, given a probability distribution on the domain, is the probability of the symmetric difference $C \Delta H$. This is the probability of the event that H misclassifies a random example of C.

The following definition for efficient learning was introduced by Valiant (1984).

A concept class $\mathcal{C}$ is *PAC learnable* by the hypothesis space $\mathcal{H}$ if there is a polynomial algorithm A and a polynomial p such that for every $n \geq 1$, every target concept $C \in \mathcal{C}_n$, every probability distribution D_n on X_n and every ϵ, δ $(0 < \epsilon, \delta < 1)$ the following holds: if A is a given a sample of size $p\left(n, \frac{1}{\epsilon}, \frac{1}{\delta}\right)$ of C selected from X_n according to D_n, then it outputs a hypothesis $H \in \mathcal{H}_n$ such that with probability at least $1 - \delta$, the error of H is at most ϵ.

For simplicity, the later discussion in Section 4.1. will be restricted to the case when the hypothesis space is identical to the concept class. A concept class $\mathcal{C}$ which is PAC learnable by $\mathcal{C}$ is called *properly PAC learnable.*

Thus informally, an efficient learning algorithm is required to find a good hypothesis fast for a typical sample of small size. The definition of PAC learnability is quite robust in the sense that it has several equivalent versions (see Haussler, Kearns, Littlestone and Warmuth (1988)). For example, the learning algorithm can be allowed to have access to an oracle which provides a random example of the target concept if requested.

The PAC models are also called *distribution-free* learning models as the learning algorithms have to perform well independently of the underlying distribution.

For some learning problems one has to modify the above definition by introducing a new parameter corresponding to the size of the target concept. This is necessary in such cases as learning DNF, Boolean formulas or Boolean circuits in general. Here $\mathcal{C}_n$ consists of all subsets of X_n, thus if the sample size and the size of hypothesis is polynomial in n (as implicitly required by the definition), one cannot hope for successful learning. (Indeed, it follows from results mentioned later on, that this is not possible.)

Thus a concept class $\mathcal{C}$ is *s-PAC learnable* by a hypothesis space $\mathcal{H}$ if there is a polynomial algorithm A and a polynomial p such that for every $n, s \geq 1$, every target concept $C \in \mathcal{C}_n$ of size at most s, every probability distribution D_n on X_n and every ϵ, δ $(0 < \epsilon, \delta < 1)$ the following holds: if A is given a sample of size $p\left(n, s, \frac{1}{\epsilon}, \frac{1}{\delta}\right)$ of C selected according to D_n, then it outputs a hypothesis $H \in \mathcal{H}_n$ such that with probability at least $1 - \delta$, the error of H is at most ϵ.

Another important modification, the prediction model introduced by Haussler, Littlestone and Warmuth (1988) does not require the learning algorithm to output the representation of a hypothesis at all. Here the learning algorithm gets a random sample of the target concept and a random element of the domain, and it has to predict the classification of this element with respect to the target concept.

A concept class C is called *predictable* if there is a polynomial algorithm A and a polynomial p such that for every $n \geq 1$, every target concept $C \in \mathcal{C}_n$, every probability distribution D_n on X_n and every ϵ $(0 < \epsilon < 1)$ the following holds: if A is given a sample of size $p(n, \frac{1}{\epsilon})$ of C selected from X_n according to D_n and an element x selected from X_n according to D_n, then it outputs 0 or 1, such that with probability at least ϵ, the output is 1 iff $x \in C$.

It can be shown (see Haussler, Kearns, Littlestone and Warmuth (1988)) that a concept class $\mathcal{C}$ is polynomially predictable iff there is some hypothesis space $\mathcal{H}$ such that $\mathcal{C}$ is PAC learnable by $\mathcal{H}$. (Note that by the assumptions given in Section 2. membership in the hypotheses of $\mathcal{H}$ must be testable in polynomial time.)

3.2 On-line learning

In this class of models learning is thought of as an interaction between the learning algorithm, asking queries about the target concept, and the environment, responding to these queries. Thus the role of the environment is quite different from that of the oracle providing random examples in the version PAC model mentioned in the previous section. As we are interested in exact identification here, the concept classes considered are always finite. Typically $X_n = \{0,1\}^n$, a different example is provided by the domain $\{0, \ldots, n-1\}^d$ for boxes (see Section 2.).

In the basic model of *learning with equivalence queries* due to Angluin (1988), a query of the learning algorithm is an equivalence query H from the hypothesis space $\mathcal{H}$, or with other words a hypothesis from $\mathcal{H}$. The response to the query is "yes", if H is equivalent to the target concept. Otherwise the response is a *counterexample* to the hypothesis, i.e. an element x from the symmetric difference $C \triangle H$. Note that given a hypothesis, the environment can have several choices for the counterexample.

Computationally the interaction is modelled by providing the learning algorithm with an extra oracle tape for printing a representation of its hypothesis and receiving the response. It is assumed that the representations of instances from X_n, concepts from $\mathcal{C}_n$ and hypotheses from $\mathcal{H}_n$ have size polynomial in n.

A concept class $\mathcal{C}$ is said to *polynomially learnable with equivalence queries from* $\mathcal{H}$, if there is a polynomial algorithm A such that for every $n \geq 1$, every target concept $C \in \mathcal{C}_n$ and every choice of the counterexamples, the following holds: if A is given n in unary then it terminates by identifying C.

In analogy with the previous section, $\mathcal{C}$ is *properly polynomially learnable with equivalence queries* if it is polynomially learnable with equivalence queries from $\mathcal{C}$.

In the next section we shall discuss a simplified version of the PAC model considering only the sample size needed for probably approximately correct learning, disregarding the amount of computation required. Similarly, in an on-line model it is interesting to ask, how much interaction is needed between the learning algorithm and the environment

to solve a learning problem, again disregarding the amount of computation required to produce the next query. This approach is analogous to the study of decision trees in theoretical computer science. For example, the decision tree model of comparison based sorting algorithms does not take into consideration the difficulty of determining the next comparison.

Thus, given a finite set X, a concept class $\mathcal{C} \subseteq 2^X$ and a hypothesis space $\mathcal{H} \subseteq 2^X$, an algorithm learning $\mathcal{C}$ with equivalence queries from $\mathcal{H}$ is just a function assigning the next query to the previous queries and the counterexamples received for these queries. In fact, as the algorithms are assumed to be deterministic, the next query depends on the previous counterexamples only. The learning complexity of a learning algorithm can be defined as the largest number of counterexamples required to identify a target concept, considered over all target concepts and all possible choices of counterexamples. The *complexity of learning $\mathcal{C}$ using equivalence queries from $\mathcal{H}$* is the smallest learning complexity of a learning algorithm solving this problem.

We note that this definition applies to a single finite concept class $\mathcal{C}$. The relationship between the previous definition and this one is that if the concept class $\mathcal{C} = \bigcup_{n\geq 1} \mathcal{C}_n$ is polynomially learnable with equivalence queries from $\mathcal{H} = \bigcup_{n\geq 1} \mathcal{H}_n$ then for some polynomial p, for every $n \geq 1$, the complexity of learning $\mathcal{C}_n$ using equivalence queries from $\mathcal{H}_n$ is at most $p(n)$.

Two important special cases are to be mentioned here. The *learning complexity of $\mathcal{C}$* is the complexity of learning a concept from $\mathcal{C}$ with equivalence queries from $\mathcal{C}$. The *learning complexity of $\mathcal{C}$ with arbitrary equivalence queries* is the complexity of learning $\mathcal{C}$ using equivalence queries from 2^X. This is the same as the *mistake bound* of $\mathcal{C}$ considered by Littlestone (1988).

Finally we consider another query type. A *membership query* is specified by an element x of the domain. The response to such a query is the classification of x with respect to the target concept. It turned out that there are several interesting learning problems which can be solved efficiently using equivalence queries combined with membership queries. Some of these will be mentioned in Section 5.

A detailed discussion of the relationship between the different models is given in Maass and Turán (1990 b). Here we only mention that, as observed by Angluin (1987), a learning algorithm using equivalence queries can be simulated by a PAC learning algorithm. An equivalence query H is simulated by taking a sufficiently large sample of the target concept. If this sample contains a counterexample then the simulation continues. Otherwise H is output as the final hypothesis.

4. Characterizations of learnability

In this section we mention results which characterize learnability in the different models considered, in terms of other notions such as the Vapnik-Chervonenkis dimension, Occam algorithms, weak learnability and adversary trees. These characterizations can be used to obtain generic learning algorithms and to prove negative results for learnability.

4.1 Vapnik-Chervonenkis dimension

Given a concept class $\mathcal{C}$ over a domain X, its *Vapnik-Chervonenkis dimension* $VC(\mathcal{C})$ is defined as follows. A subset $Y \subseteq X$ is called *shattered* by $\mathcal{C}$ if for every subset $Z \subseteq Y$ there is a concept $C \in \mathcal{C}$ such that $Z = C \cap Y$. Then $VC(\mathcal{C})$ is the size of a largest shattered subset of X. It turns out that this combinational parameter is very closely related to the difficulty of learning a concept from $\mathcal{C}$.

First let us consider still another modified version of proper PAC learning (referred to in the previous section), where A is not required to be a polynomial time algorithm. Thus A is only assumed to be a function which assigns a hypothesis $H \in \mathcal{H}$ to a sample of some given size m. For a given ϵ, δ $(0 < \epsilon, \delta < 1)$ A is called an (ϵ, δ)-learning function if for every concept $C \in \mathcal{C}$ and every distribution D over X it holds that if H is the hypothesis assigned to a random sample of size m selected from X according to D then with probability at least $1-\delta$ the error of H is at most ϵ. What is the smallest m (depending on ϵ and δ) for which there is an (ϵ, δ)-learning function? Improving a result of Blumer, Ehrenfeucht, Haussler and Warmuth (1989) (which in turn built on the work of Vapnik and Chervonenkis (1971)), Anthony, Biggs and Shawe-Taylor (1990) showed that if

$$m \geq \frac{1}{\epsilon(1-\sqrt{\epsilon})}\left(\log\left(\frac{VC(\mathcal{C})/(VC(\mathcal{C})-1)}{\delta}\right) + 2VC(\mathcal{C})\log\left(\frac{6}{\epsilon}\right)\right)$$

then every A assigning a consistent hypothesis to a sample satisfies the requirement. (A hypothesis H is consistent with a sample if it contains all its positive examples and none of its negative examples.) Here one has to assume some weak measurability conditions in the case when $X = \mathbf{R}^n$ (see Blumer, Ehrenfeucht, Haussler and Warmuth (1989)). On the other hand, improving a bound of Blumer, Ehrenfeucht, Haussler and Warmuth (1989), it was shown by Ehrenfeucht, Haussler, Kearns and Valiant (1988) that if $\mathcal{C}$ is nontrivial then no learning function exists (for any $\mathcal{H}$) if $m < \frac{1-\epsilon}{2\epsilon}\log\frac{2}{\delta} + \frac{VC(\mathcal{C})-1}{64\epsilon}$. ($\mathcal{C}$ is trivial if it contains either one concept, or two concepts partitioning the domain.) Thus apart from the gap of a factor of $O\left(\log\frac{1}{\epsilon}\right)$, these results give a precise characterization of the required sample size in terms of the Vapnik-Chervonenkis dimension. It is remarkable that there is a sharp dichotomy here - either all consistent functions work, or no function works at all.

Now returning to the PAC model, it follows that the polynomial growth of $VC(\mathcal{C}_n)$ and the existence of an efficient algorithm finding a consistent hypothesis for a given sample is a sufficient condition for proper PAC-learnability. The notion needed here is that of *randomized polynomial hypothesis finder*. This is a randomized polynomial time algorithm, which given a sample, outputs a consistent hypothesis with probability at least α, for some fixed $\alpha > 0$. It also follows from the characterization above that the polynomial growth of $VC(\mathcal{C}_n)$ is a necessary condition for proper PAC-learnability. In addition, it can be shown that a PAC-learning algorithm can be used to construct a randomized polynomial hypothesis finder (Pitt and Valiant (1988); see Section 6. for further applications of this fact). This implies the following characterization of proper PAC-learnability (Blumer, Ehrenfeucht, Haussler and Warmuth (1989)): a concept class $\mathcal{C}$ is properly PAC learnable iff the growth of $VC(\mathcal{C}_n)$ is polynomial and there is a randomized polynomial hypothesis finder for $\mathcal{C}$. In the Boolean case $X_n = \{0,1\}^n$ this characterization is further simplified

by the fact that the polynomial growth of $VC(\mathcal{C}_n)$ is equivalent to the polynomial growth of $\log |\mathcal{C}_n|$ (Natarajan (1987), Haussler, Kearns, Littlestone and Warmuth (1988)).

4.2 Occam algorithms

An Occam algorithm is an algorithm which, given a sample of the target concept, outputs a consistent and relatively simple hypothesis. With other words, an Occam algorithm is a hypothesis finder which is capable of some *data compression*. This notion, introduced by Blumer, Ehrenfeucht, Haussler and Warmuth (1987), corresponds to the principle called Occam's Razor stating that one should prefer simple hypotheses.[1]

First let us assume that $X_n = \{0,1\}^n$. In this case an *Occam algorithm* is a randomized polynomial algorithm A for which there is a polynomial p and a constant $\alpha (0 \leq \alpha < 1)$ such that for every $n \geq 1$, every target concept $C \in \mathcal{C}_n$ of size at most s and every ϵ $(0 < \epsilon < 1)$ the following holds: if A is given a sample of size m of C as input, then with probability at least $1 - \epsilon$ it outputs the representation of a consistent hypothesis from $\mathcal{C}_n$ having size at most $p\left(n, s, \frac{1}{\epsilon}\right) \cdot m^{\alpha}$. The assumption $\alpha < 1$ represents the amount of compression required.

Blumer, Ehrenfeucht, Haussler and Warmuth (1987) showed that if there is an Occam-algorithm for $\mathcal{C}$ then $\mathcal{C}$ is properly s-PAC learnable. The proof is based on the observations that a hypothesis with large error is unlikely to be consistent with a large sample, and that there are only few short hypotheses. (We note that a similar argument, combined with Sauer's lemma (1972) about the Vapnik-Chervonenkis dimension, forms the basis of the result of the previous section.)

In the case of real inputs a similar result can be obtained if the definition of an Occam algorithm is modified by replacing the existence of short hypotheses with the existence of a hypothesis space of small Vapnik-Chervonenkis dimension (Blumer, Ehrenfeucht, Haussler and Warmuth (1989)). This implies, for example, that unions of boxes in a Euclidean space are properly s-PAC learnable.

Recently Board and Pitt (1990) and Schapire (1990) obtained partial converses of these results, showing that under quite general conditions s-PAC learnability implies the existence of an Occam algorithm. This emphasizes the canonical role of Occam algorithms and data compression for learnability.

In the case of $X_n = \{0,1\}^n$ only simple encodings appear to have Occam algorithms. If this is indeed the case, the above results could be used to prove negative results for learnability by showing the nonexistence of Occam algorithms, as suggested by Board and Pitt (1990). Some further comments on this relationship between computational learning theory and combinatorial optimization are given in Section 6.

---

[1]It is interesting that the same principle was formulated by Maimonides around 1190 in The Guide of the Perplexed (Maimonides (1963)): "... if we assume, for instance, that we suppose as a hypothesis an arrangement by means of which the observations regarding the motions of one particular star can be accounted for through the assumption of three spheres, and another arrangement by means of which the same observations are accounted for through the assumption of four spheres, it is preferable for us to rely on the arrangement postulating the lesser number of motions." (II.11.)

4.3 Weak predictability

An interesting characterization of predictability is obtained by considering the notion of weak predictability, introduced by Kearns and Valiant (1989). Referring to the reformulation of predictability given at the end of Section 3.1, a concept class $\mathcal{C}$ is called *weakly predictable* if there exist a hypothesis space $\mathcal{H}$, a polynomial algorithm A and polynomials p_1, p_2 such that for every $n \geq 1$, every target concept $C \in \mathcal{C}_n$, every probability distribution D_n on X_n and every δ $(0 < \delta < 1)$ the following holds: if A is given a sample of size $p_1\left(n, \frac{1}{\delta}\right)$ of C selected from X_n according to D_n, then it outputs a hypothesis $H \in \mathcal{H}_n$ such that with probability at least $1 - \delta$, the error of H is at most $\frac{1}{2} - \frac{1}{p_2(n)}$.

Thus instead of outputting a hypothesis with error at most ϵ, the learning algorithm is only required to output a hypothesis with error slightly smaller than $\frac{1}{2}$.

Does weak predictability imply predictability? It was noted that the approach of running a weak learning algorithm several times and taking majority vote does not work as the hypotheses need not be independent.

Nevertheless, Schapire (1990) showed that the notions of predictability and weak predictability are equivalent. The result is actually formulated as referring to the more general notions of s-predictability and weak s-predictablity, where s is again a parameter for the size of the target concept. It is interesting, that the existence of an Occam algorithm in this case (mentioned in the previous section) is obtained as a corollary of this theorem. Another proof was found by Freund (1990). Both proofs are built on simulations of the weak learning algorithm on different distributions, making essential use of the distribution-free property of the PAC model.

4.4 On-line learning with arbitrary hypotheses

We close this section by giving a characterization, due to Littlestone (1988) of the complexity of learning a concept class $\mathcal{C}$ over a finite domain X, with arbitrary equivalence queries.

A learning algorithm using membership queries only can be viewed as a *decision tree,* where each node is labelled by an element x of the domain, the edges leaving a node are labelled by 0 or 1, giving the classification of x, and the leaves are labelled by concepts C from $\mathcal{C}$. For a given target concept, the learning algorithm starts at the root and follows a path down the tree, arriving to a leaf where it identifies the label of the leaf as the target concept. The complexity of learning $\mathcal{C}$ with membership queries is the smallest depth of any decision tree for $\mathcal{C}$.

On the other hand, for each decision tree T one can consider the minimal depth of a leaf in T, and maximize this quantity over all decision trees for $\mathcal{C}$. This complexity measure, called the *adversary tree* complexity of $\mathcal{C}$, turns out to be equal to the complexity of learning $\mathcal{C}$ using arbitrary hypotheses. An application of this characterization for proving lower bounds will be mentioned in Section 6.2.

5. Some learning algorithms

As noted above, the characterizations of PAC learnability suggest general approaches to the construction of PAC learning algorithms. On the other hand the characterization of PAC learnability in terms of the Vapnik-Chervonenkis dimension implies that the model does not distinguish any two learning algorithms producing consistent hypotheses from a large enough sample. It is desirable, also from the practical point of view, to find further criteria for evaluating the performance of learning algorithms and the complexity of learning problems. In this section we consider the complexity of some concrete learning problems in the on-line models, which provide a framework for such an evaluation.

The problem of learning a halfspace over $\{0,1\}^n$ (or a Boolean threshold function) defined in Section 2. is one of the first ones considered in learning theory. The perceptron algorithm of Rosenblatt (1962) can be viewed as a learning algorithm using equivalence queries. The current weight vector is updated after each counterexample by essentially adding the counterexample to the weight vector. The Winnow algorithms of Littlestone (1988) are multiplicative versions of this algorithm to learn monotone Boolean threshold functions, with a good performance guarantee in several cases. Both algorithms need an exponential number of counterexamples in the worst case (Minsky and Papert (1988), resp. Maass and Turán (1990 a), (1990 d)).

A learning algorithm demonstrating the proper polynomial learnability of halfspaces over $\{0,1\}^n$ with equivalence queries is given in Maass and Turán (1989), (1990 d). The algorithm maintains a *version space*, i.e. a set of representations of those concepts which are still candidates for being the target concept. The next query is the center of this set, with a suitable notion of center. It turns out that every counterexample reduces the volume of the version space by a constant factor. This approach is an adaptation of the ellipsoid method in combinatorial optimization (Khachian (1979), Grötschel, Lovász and Schrijver (1988)). More generally, every algorithm for finding a point in a convex polytope given by a separation oracle, and having a guaranteed lower bound for its volume (see Grötschel, Lovász and Schrijver (1988)) can be used to construct an algorithm for learning halfspaces over $\{0,1\}^n$. In particular, adapting Vaidya's algorithm (Vaidya (1989)) one gets an algorithm of learning complexity $O(n^2 \log n)$ (see Section 6.2 for an almost matching lower bound). In comparison, the sample size required for PAC learning a halfspace over $\{0,1\}^n$ for fixed ϵ and δ is $\Theta(n)$.

Now we turn to the problem of learning boxes in a discrete d-dimensional space (also defined in Section 2.). Note that the standard representations have length $O(d \log n)$. For fixed d, an algorithm showing proper polynomial learnability with equivalence queries is given in Maass and Turán (1989), (1990 c). The learning complexity of the algorithm is $O(\log n)$, which is shown to be optimal.

The algorithm is again based on the use of a version space and the existence of a hypothesis for which every counterexample reduces the version space significantly. Thus both algorithms mentioned may be considered as adaptations of the paradigm of binary search to concept learning.

Using a similar approach, Beals (1990) found an algorithm of complexity $O(\log^3 n)$ for learning a square over the domain $\{0,\ldots,n-1\}^2$.

The range of applicability of this approach is not clear. For example, it is shown in Maass and Turán (1990 a), (1990 c) that the complexity of learning boxes which are not required to be axis-parallel is $\Omega(n)$ even in the two-dimensional case.

The complexity of the box learning algorithm is exponential in d, while the complexity of the algorithm which always outputs a minimal consistent hypothesis is $O(dn)$ (thus polynomial in d, but exponential in $\log n$). Recently Zhixiang Chen (1991) announced a box learning algorithm of complexity $O(d^2 \log^2 n)$.

By now there is a large number of interesting learning algorithms which make essential use of both equivalence and membership queries. The first example is Angluin's algorithm for learning DFA (Angluin (1987)). In this algorithm the membership queries are used to interpret the counterexample received, in order to be able to form a new hypothesis. We note that the complexity measure used here is different from the ones considered so far, as an efficient learning algorithm is required to be polynomial in the number of states of the target DFA and in the length of the longest counterexample received. This is necessary as the counterexamples can be of arbitrary length and the algorithm has to be able to read them.

Among the other problems which are efficiently learnable using equivalence and membership queries we mention read-once formulas (Angluin, Hellerstein and Karpinski (1989)) and conjunctions of Horn clauses (Angluin, Frazier and Pitt (1990)).

6. Negative results

Corresponding to the two types of learning models considered so far, a negative result for learnability can be either complexity theoretic or information theoretic. A complexity theoretic negative result shows that there is no computationally efficient learning algorithm for a given learning problem, using some complexity theoretic assumption such as $P \neq NP$. An information theoretic negative result provides a lower bound to the amount of information, e.g. the number of counterexamples, that must be obtained by a learning algorithm. These lower bounds apply to all learning algorithms, without considering their computational complexity.

6.1 Complexity theoretic negative results

As noted in Section 4.1 the proper PAC learnability of a concept class C implies the existence of a randomized polynomial hypothesis finder for C. Using this argument, Pitt and Valiant (1988) showed that if $R \neq NP$ then k-term DNF are not properly PAC learnable for $k \geq 2$. (R is class of languages accepted by randomized polynomial machines with one-sided error.) They also noted that if the hypothesis space is enlarged to k-CNF then k-term DNF become PAC-learnable (Valiant (1984)).

Similarly, it follows from the results described in Section 4.2. that for certain classes such as DFA and Boolean formulas proper s-PAC learnability implies the existence of an Occam algorithm for C. Now an Occam algorithm may be viewed as an approximation algorithm for the problem of finding a shortest representation of a sample. Thus if one proves, e.g. assuming $R \neq NP$, that there is no such approximation algorithm, then this implies a negative result for proper s-PAC learnability. In this direction we mention the result of Pitt and Warmuth (1989): if $P \neq NP$ then the smallest DFA consistent with a

given sample cannot be approximated within any polynomial. A somewhat stronger result is needed to prove the nonlearnability of DFA in the proper s-PAC model.

Another approach provides stronger negative results, showing unpredictability of certain learning problems, using potentially stronger cryptographic assumptions.

As already observed by Valiant in his fundamental paper (Valiant (1984)), the task of breaking a cryptosystem may also be viewed as a learning problem (more precisely, as a prediction problem), and thus the result of Goldreich, Goldwasser and Micali (1986) may be interpreted as showing the unpredictability of Boolean circuits, assuming the existence of a one-way function. Here the learner is even allowed to ask membership queries. Kearns and Valiant (1989) considered "easier" learning problems such as Boolean formulas and DFA, and using similar cryptographic assumptions showed that these are also unpredictable. Thus their result implies the strong nonapproximability of a minimal consistent DFA, using the stronger assumption. It is interesting to note that in view of Angluin's algorithm mentioned in Section 5. the result for DFA cannot be extended to include membership queries. On the other hand, recently Angluin and Kharitonov (1991) proved, using the same cryptographic assumptions and a construction of Naor and Yung (1990) that the class of NFA is unpredictable even if membership queries are allowed.

6.2 Information theoretic negative results

As the first example of a negative result of this kind, we recall the lower bound to the sample size of any learning function in terms of the Vapnik-Chervonenkis dimension of the concept class (Ehrenfencht, Haussler, Kearns, Valiant (1989)), described in Section 4.1.

In view of the important role played by the Vapnik-Chervonenkis dimension in PAC learnability, one may consider its relationship to learning complexity in the different on-line models. Littlestone (1988) observed that for every finite concept class $\mathcal{C}$, $VC(\mathcal{C})$ is a lower bound to the complexity of learning a concept from $\mathcal{C}$ with arbitrary equivalence queries. Similarly $VC(\mathcal{C})$ is a lower bound to learning complexity if only membership queries are allowed.

On the other hand there are concept classes which can be learned with at most 0.42 $VC(\mathcal{C})$ queries if *both* equivalence and membership queries are allowed. Nevertheless, $\frac{1}{7}\,VC(\mathcal{C})$ does provide a lower bound to the complexity of learning $\mathcal{C}$ with equivalence and membership queries, for every finite concept class $\mathcal{C}$ (Maass and Turán (1990 a), (1990 b)). This general information theoretic lower bound is sharp in some cases, e.g. it gives a sharp $\Omega\big(k(1+\log\frac{n}{k})\big)$ bound for the complexity of learning a conjuction of k out of n variables.

Finally we mention lower bounds for learning halfspaces over $\{0,1\}^n$ and DFA.

The characterization of the complexity of learning with arbitrary equivalence queries can be used to show that the complexity of learning a halfspace over $\{0,1\}^n$ with arbitrary equivalence queries is $\Omega(n^2)$ (Maass and Turán (1989), (1990 d)). This matches the upper bound (which is achieved by equivalence queries from the concept class itself) given in Section 5. up to a factor of $\log n$.

Complementing her polynomial algorithm for learning DFA with equivalence and membership queries (1987), and her remark that the number of membership queries required to learn DFA's is exponential (1981), Angluin showed that the number of equivalence

queries required to learn n-state DFA accepting a subset of $\{0,1\}^{O(n)}$ is also superpolynomial (Angluin (1990)).

7. Summary

In the introduction of his paper starting computational learning theory, Valiant observed that the intuitive notion of learning merits similar attention from the point of view of formal theoretical study as that of the notion of computing. In this comparison, learning appears to be more elusive, more difficult to capture by a unified mathematical theory (as noted by Haussler (1990), it is not clear whether such a theory is even possible or desirable). Research was focused on concept learning, which is in fact closely related to computing in that several approaches developed in theoretical computer science can be adapted to its study. Interesting connections were found with other fields such as combinatorial optimization, cryptography and statistical pattern recognition. In this survey we gave a short account of some aspects of the results obtained in computational learning theory, by describing several learning models, characterizations of learnability, some learning algorithms and negative results.

Acknowledgement I would like to thank Wolfgang Maass for several valuable discussions.

REFERENCES

ANGLUIN, D. (1981). A note on the number of queries needed to identify regular languages. *Information and Control,* **51**, 76-87.

ANGLUIN, D. (1987). Learning regular sets from queries and counterexamples. *Information and Computation,* **75**, 87-106.

ANGLUIN, D. (1988). Queries and concept learning, *Machine Learning,* **2**, 319-342.

ANGLUIN, D. (1990). Negative results for equivalence queries. *Machine Learning,* **5**, 121-150.

ANGLUIN, D., FRAZIER, M. AND PITT, L. (1990). Learning conjunctions of Horn clauses. *31. FOCS,* 186-192.

ANGLUIN, D., HELLERSTEIN, L. AND KARPINSKI, M. (1989). *Learning read-once formulas with queries.* (Technical Report UCB/CSD 89/527). University of California at Berkeley, Computer Science Division. (also, Technical Report TR-89-050, International Computer Science Institute, Berkeley, California.) To appear, *JACM.*

ANGLUIN, D. AND KHARITONOV, M. (1991). When won't membership queries help. *23. STOC,* 444-454.

ANTHONY, M., BIGGS, N. AND SHAWE-TAYLOR, J. (1990). The learnability of formal concepts. *3. COLT,* 232-246.

BEALS, R. (1990). Unpublished manuscript.

BLUMER, A., EHRENFEUCHT, A., HAUSSLER, D. AND WARMUTH, M.K. (1987). Occam's razor. *Information Processing Letters,* 24, 377-380.

BLUMER, A., EHRENFEUCHT, A., HAUSSLER, D., AND WARMUTH, M.K. (1989). Learnability and the Vapnik-Chervonenkis dimension. *JACM,* 36, 929-965.

BOARD, R. AND PITT, L. (1990). On the necessity of Occam algorithms. *22. STOC,* 54-63.

CHEN, Z.X. (1991). Unpublished manuscript.

EHRENFEUCHT, A., HAUSSLER, D., KEARNS, M. AND VALIANT, L.G. (1988). A general lower bound on the number of examples needed for learning. *COLT 1988,* 139-154.

FAIGLE, U. AND KERN, W. (1990). On learnability of monotone DNF functions under uniform distribution. University of Twente, Faculty of Applied Mathematics, Memo. No. 863.

FLOYD, S. (1989). Space-bounded learning and the Vapnik-Chervonenkis dimension. *2. COLT,* 349-364.

FREUND, Y. (1990). Boosting a weak learning algorithm by majority. *3. COLT,* 202-216.

FULK, M.A. AND CHASE, J., EDITORS. (1990). *Proceedings of the Third Annual Conference on Computational Learning Theory.* Morgan Kauffman.

GOLD, E.M. (1967). Language identification in the limit. *Information and Control,* 10, 447-474.

GOLDREICH, O., GOLDWASSER, S. AND MICALI, S. (1986). How to construct random functions. *JACM,* **33**, 792-807.

GRÖTSCHEL, M., LOVÁSZ, L. AND SCHRIJVER, A. (1988). *Geometric Algorithms and Combinatorial Optimization.* Springer, Algorithms and Combinatorics Vol. 2.

HAUSSLER, D. (1989). Generalizing the PAC model for neural net and other learning applications. (Technical Report UCSC-CRL-89-30). University of California at Santa Cruz, Computer Research Laboratory. To appear, *Information and Computation.*

HAUSSLER, D. (1990). Probably approximately correct learning. *Eight National AI Conference, AAAI'90,* 1101-1108.

HAUSSLER, D., KEARNS, M., LITTLESTONE, N. AND WARMUTH, M.K. (1988). Equivalence of models for polynomial learnability. *COLT 1988,* 42-55. To appear, *Information and Computation.*

HAUSSLER, D., LITTLESTONE, N. AND WARMUTH, M.K. (1988). Predicting $\{0,1\}$-functions on randomly drawn points. *29. FOCS,* 100-109.

HAUSSLER, D. AND PITT, L., EDITORS. (1988). *Proceedings of the 1988 Workshop on Computational Learning Theory.* Morgan Kauffman.

KEARNS, M.J. AND SCHAPIRE, R.E. (1990). Efficient distribution-free learning of probabilistic concepts. *31. FOCS,* 382-391.

KEARNS, M. AND VALIANT, L.G. (1989). Cryptographic limitations on learning Boolean formulae and finite automata. *21. STOC,* 433-444.

KHACHIAN, L.G. (1979). A polynomial algorithm in linear programming. *Doklady Akademii Nauk SSSR,* 244, 1093-1096. English translation: *Soviet Mathematics Doklady,* 20, 191-194.

LINIAL, N., MANSOUR, Y. AND NISAN, N. (1989). Constant depth circuits, Fourier transform and learnability. *30. FOCS,* 574-579.

LITTLESTONE, N. (1988). Learning quickly when irrelevant attributes abound: a new linear-threshold algorithm. *Machine Learning,* 2, 285-318.

MAASS, W. AND TURÁN, GY. (1989). On the complexity of learning from counterexamples. *30. FOCS,* 262-267.

MAASS, W. AND TURÁN, GY. (1990 a). On the complexity of learning from counterexamples and membership queries. *31. FOCS,* 203-210.

MAASS, W. AND TURÁN, GY. (1990 b). Lower bounds and separation results for on-line learning models. To appear, *Machine Learning.*

MAASS, W. AND TURÁN, GY. (1990 c). Algorithms and lower bounds for on-line learning of geometrical concepts. Unpublished manuscript.

MAASS, W. AND TURÁN, GY. (1990 d). How fast can a threshold gate learn? To appear in: Hanson, S., Drastal, G. and Rivest, R., Editors, *Computational Learning Theory and Natural Learning Systems: Constraints and Prospects,* MIT Press.

MAIMONIDES, M. (1963). The Guide of the Perplexed. Translated by Shlomo Pines. University of Chicago Press.

MINSKY, M. AND PAPERT, S. (1988). *Perceptrons: an introduction to computational geometry, Expanded edition.* MIT Press.

NATARAJAN, B.K. (1987). Learning functions from examples. Technical Report CMU-R1-TR-87-19. Carnegie-Mellon University.

NILSSON, N.J. (1965). *Learning machines.* McGraw-Hill.

NAOR, M. AND YUNG, M. (1990). Public-key cryptosystems provably secure against chosen ciphertext attacks. *22. STOC,* 427-437.

PITT, L. AND VALIANT, L.G. (1988). Computational limitations on learning from examples. *JACM,* **35**, 965-984.

PITT, L. AND WARMUTH, M.K. (1989). The minimum consistent DFA problem cannot be approximated within any polynomial. *21. STOC,* 421-432. To apear, *JACM.*

RIVEST, R., HAUSSLER, D. AND WARMUTH, M.K., EDITORS. (1989). *Proceedings of the Second Annual Conference on Computational Learning Theory.* Morgan Kauffman.

ROSENBLATT, F. (1962). *Principles of neurodynamics.* Spartan Books.

RUMELHART, D.E. AND MCCLELLAND, J.L. (1986). *Parallel distributed processing: explorations in the microstructure of cognition.* MIT Press.

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

SCHAPIRE, R.E. (1990). The strength of weak learnability. *Machine Learning*, **5**, 197-227.

SLOAN, R.H. (1988). Types of noise in data for concept learning. *COLT 1988*, 91-96.

VAIDYA, P.M. (1989). A new algorithm for minimizing convex functions over convex sets. *30. FOCS*, 338-343.

VALIANT, L.G. (1984). A theory of the learnable. *CACM*, **27**, 1134-1142.

VAPNIK, V.N. AND CHERVONENKIS, A.YA. (1971). On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and Its Applications*, **16**, 264-280.