Machine Learning, 13, 145–149 (1993)

Book Review
Machine Learning: A Theoretical Approach
by Balas K. Natarajan. Morgan Kaufmann Publishers, Inc., 1991.

LISA HELLERSTEIN
Department of Electrical Engineering and Computer Science, Northwestern University, Evanston, IL 60208-3118

Editor: Alberto Segre

1. Overview

The PAC (probably approximately correct) model of learning was introduced in 1984 by L. Valiant as a formal framework for studying the computational complexity of machine learning. In *Machine Learning, a Theoretical Approach*, Balas K. Natarajan provides an introduction to research in the PAC model, and more generally, to the field of computational learning theory. The book was written for use as a text in an introductory graduate course. It is also advertised as a reference for AI researchers. Readers are assumed to be familiar with topics such as polynomial-time algorithms, NP-completeness, and deterministic finite automata, although more advanced topics such as randomized complexity classes and cryptographic schemes are covered briefly at an introductory level.

Natarajan contrasts the goals and methods of research in computational learning theory to the goals and methods of two other approaches to learning—AI and inductive inference. AI research is different, he says, because it tends to be less formal and more experimental. He might also have mentioned that the problems in AI are often more general and less well defined than problems in computational learning theory: for example, learning to understand English stories, rather than learning classes of Boolean monomials. This difference is certainly evident in his book. Natarajan also points out that although there are many similarities between inductive inference and computational learning theory, computational learning theory is distinguished by its emphasis on computational complexity. A major focus of the book is the existence or non-existence of polynomial-time learning algorithms for different classes of concepts and functions.

Natarajan presents the PAC learning paradigm as a candidate for a good model of the natural learning process. His criteria for a good model are that it is usable and robust. In particular, it should mesh well with existing theories of learning, provide a basis for the design of practical algorithms, and agree with relevant experimental data.

Clearly the PAC paradigm is usable. As Natarajan points out, the research covered in the book shows it is possible to prove results in the model that are both rigorous and deep. Natarajan also claims that the model is robust. He cites a number of results that demonstrate that fundamental properties of learnability in the standard PAC model remain true in some

variants of the model. However, in other variants (such as the noisy PAC model, or the PAC model augmented with queries treated in chapter 6), these properties may not hold. In fact, it may be impossible to design a model that is as robust as we might like.

Natarajan also claims, controversially, that the PAC paradigm meshes well with existing theories of learning. As evidence, he presents some of the same arguments that are in Valiant's original paper. He also points to a particular result presented in the book, the proof that Occam algorithms are an effective way to learn. An Occam algorithm is (essentially) one that uses Occam's Razor; it picks the simplest explanation of the observed data. This result is certainly suggestive, and is significant in giving a proof of Occam's Razor in a technical setting. However, such results show only that the PAC paradigm is useful in modeling "vanilla" concept-learning. There has been relatively little research exploring the usefulness of the PAC paradigm in modeling other aspects of learning, such as learning with background information.

Natarajan calls for further research in the PAC paradigm to test whether it is useful in developing practical algorithms. One problem in developing such algorithms is how to model noise. Natarajan cites results on learning with noise, but does not cover the proofs in his book.

It is odd that Natarajan proposes the PAC paradigm as a potential model for *natural* learning. The title of the book is *Machine Learning*, not Natural Learning. At the start of the book Natarajan states "Suppose that we agree that the human brain is nothing more than a very large computer," and then carefully adds the comment, "(This is a controversial supposition. We do not debate the issue here ...)." Natarajan calls for research to test whether the PAC paradigm agrees with experimental data in the behavioral sciences. Such research might be interesting, but is largely irrelevant to the focus of the book.

Computational learning theory is the study of machine learning from the perspective of theoretical computer science—a rigorous approach to algorithm design with an emphasis on computational complexity. Natarajan's discussion of the philosophical issues behind the PAC paradigm is relatively short. Readers will have to look elsewhere to find a careful and critical analysis of the PAC paradigm's validity. Natarajan's main contribution is the presentation of theoretical results. These results form a rich, unified body of work. Readers of the book will be in a position to understand and appreciate current research in the PAC paradigm.

The only other book now available on computational learning theory is *Computational Learning Theory: An Introduction* by M. Anthony and N. Biggs (1992). That book is more readable than Natarajan's, but covers a narrower range of topics.

2. Summary of contents

In the PAC model, the goal of the learning algorithm is to output, with "high" probability, a "good" approximation to an unknown target concept drawn from a known class of concepts (thus giving a Probably Approximately Correct, or PAC, output). The learning algorithm is assumed to have access to a source of random examples, labeled according to whether or not they exemplify the target concept. The examples are generated according to a fixed but unknown distribution. A "good" approximation misclassifies only a "small" fraction of the possible examples, weighted according to this distribution. Much of the

research in PAC learning, or more generally, in computational learning theory, is devoted to the question of whether it is possible to develop learning algorithms that are probably correct and run in time polynomial in relevant parameters of the target concept.

There are two measures of interest in evaluating the complexity of a learning problem in the PAC model. The first is the number of random examples an algorithm would have to draw in order to have enough information to learn (the *sample complexity*); the second is the amount of time the algorithm would have to spend in computation.

Chapters 2 and 4 treat the problem of sample complexity for concept classes on countable and uncountable domains. The problem of sample complexity is directly related to a measure called the *VC-dimension*, named for two statisticians, Vapnik and Chervonenkis. The VC-dimension of a concept class is a combinatorial measure of the inherent complexity of the class. It follows from the work of Vapnik and Chervonenkis that there is a direct connection between the VC-dimension of a class, and the number of random examples a learning algorithm must see before it has enough information to learn the concept class (independent of computational complexity considerations). By showing that the VC-dimension is high, one can obtain non-learnability results.

Chapter 3 discusses the computational complexity of learning. Pitt and Valiant showed that learning a class of concepts in the standard PAC model is at least as hard as the following problem: Given a set of labeled examples, find a concept in the class that agrees with that labeling (i.e., that fits the data). If this problem is NP-hard for a given class of concepts, then the class cannot be learned in the standard PAC model unless RP = NP (roughly speaking, RP consists of problems that can be solved by polynomial-time randomized algorithms). An example of such a class is the class of concepts representable by DNF formulas with at most k-terms. Chapter 3 also covers Occam algorithms.

In Chapter 5, the VC-dimension results of the previous chapters are extended to the problem of learning classes of functions, rather than classes of concepts. This is an important topic because many practical problems, such as weather prediction, can be modeled as function learning problems.

Some concept classes that cannot be learned in polynomial time in the PAC model (unless RP = NP) can be learned if the algorithm has access not only to random labeled examples, but also to an oracle that answers queries about the target concept. Chapter 6 contains Angluin's PAC (with oracle) algorithm for learning concepts representable by deterministic finite automata. The algorithm assumes the existence of an oracle that will answer questions of the form "Is string x accepted by the target DFA?" Such a question can be viewed as performing an experiment, or asking a question of a teacher.

Chapter 7 discusses Blum and Rivest's result that three-unit feed-forward neural nets cannot be learned in polynomial time in the standard PAC model unless RP = NP. The result is based on the techniques of chapter 3, and Natarajan includes the proof that the consistency problem for three-unit feed-forward neural nets is NP-complete. Because solving the consistency problem for such nets is equivalent to training them to properly classify a set of training examples, this result explains the difficulty of training this particular class of nets. The chapter gives the technical definitions necessary for understanding the result. However, it provides almost no background information about why neural nets are interesting, and it does not explain the (somewhat limited) implications of the result for neural net research.

Chapter 8 covers an important variant of the standard PAC model, called the *prediction-model*, in which the algorithm for learning a class is allowed to output an approximation from outside that class. In this model, learning is not equivalent to solving a consistency problem, so the hardness results of chapter 3 do not apply. It is possible to prove hardness results in this model using techniques from cryptography. Natarajan gives a broad overview of these techniques, but does not provide complete proofs. Chapter 8 also covers Schapire's boosting techniques, which show that the accuracy of a learning algorithm in this model can be improved to be arbitrarily high.

In several chapters of the book, Natarajan covers his own results. Some of the results in chapter 5 on learning functions are due to Natarajan. Other results of Natarajan concern the problem of learning with one-sided error. A learning algorithm is said to have one-sided error if the approximation output by the algorithm is guaranteed to classify an example as exemplifying the target concept only if it does in fact exemplify that concept. In situations where a misclassification may have particularly severe consequences (such as in medical diagnosis), it might be vital for a learning algorithm to make only one-sided error.

The book contains many pointers to the literature, for readers who want to explore topics that Natarajan chooses not to cover in detail. It also contains an extensive bibliography. A minor problem is that the index is not as complete as it should be.

Since the book was published, there have been new results related to problems mentioned in the book. For example, Angluin and Kharitonov proved that for many important concept classes such as DNF formulas, learning in the PAC model with membership queries is as hard as learning without membership queries (under cryptographic assumptions). Readers wanting a more recent summary of results in the field should consult Dana Angluin's survey article in STOC '92 (Angluin, 1992), which also contains an extensive bibliography.

3. Suitability as a textbook

In his foreword, Natarajan suggests that instructors using his book supplement it with additional material of their own choosing. This is a good suggestion. The book covers many important topics, but is not comprehensive. Instructors might want to add additional material on topics such as learning with noise, learning with queries, and on-line learning, which are not covered in any depth. In chapter 9, Natarajan provides citations to relevant papers on these topics. If the students have had no previous exposure to neural nets, instructors might want to provide background information on the history and implications of neural net research before covering the relevant chapter of the book.

Probability plays a central role in Natarajan's book, and his treatment of probabilistic arguments assumes the students will be comfortable with probability. Instructors using the book in a course in which the students are not strong in math and probability should provide extra help in this area.

Natarajan introduces the VC-dimension in chapter 2, in the context of countable concept classes. It is possible to prove good results on countable concept classes without using the VC-dimension by simply using the size of the class (see, e.g., Rivest, 1987). Since many students need time to become comfortable with the PAC model, it might be better to defer the presentation of VC-dimension until covering the material in chapter 4 on learning uncountable concept classes.

Natarajan includes a list of problems at the end of every chapter. They vary greatly in difficulty.

The biggest drawback of using the book as a text is that it is difficult to read. It is heavy on notation. In some cases, Natarajan proves general theorems instead of simpler, slightly weaker theorems. For instance, in presenting the central theorem on Occam algorithms, he does not assume that the size of the target concept is known in advance. The proofs of these general theorems are fairly long and complex. Pedagogically, it would have been better for Natarajan to present the weaker theorems first, and then (either in a comment or in a separate proof) to handle the general case.

Natarajan is to be congratulated, however, on producing a book that presents results from different sources in one place, and does so in a unified and technically correct way. His notation and definitions are consistent, and he has made rigorous some sketchy proofs. If the book is used as the basis for a course, as Natarajan suggests, instructors should find it a valuable resource.

References

Angluin, A. (1992). Computational learning theory: Survey and selected bibliography. *Proceedings of the Twenty-Fourth Annual ACM Symposium on Theory of Computing* (pp. 351–369). Victoria, BC: ACM Press.

Anthony, M., & Biggs, N. (1992). *Computational learning theory: An introduction*. Cambridge, UK: Cambridge University Press.

Rivest, R.L. (1987). Learning decision lists. *Machine Learning*, *2*, 229–246.