Chapter 2. In R.S. Michalski, J.G. Carbonell and T.M. Mitchell (Eds.), *Machine learning, an artificial approach*. Palo Alto, CA: Tioga Publishing Co., 1983.
Bib. #552.

2

WHY SHOULD MACHINES LEARN?

Herbert A. Simon
Carnegie-Mellon University

2.1 INTRODUCTION

When I agreed to write this chapter, I thought I could simply expand a paper that I wrote for the Carnegie Symposium on Cognition, since the topic of that symposium was also learning. The difficulty with plagiarizing that paper is that it was really about psychology, whereas this book is concerned with machine learning. Now although we all believe machines can simulate human thought—unless we're vitalists, and there aren't any of those around any more—still, I didn't think that was what was intended by the title of the book. I didn't think it was appropriate to write about psychology.

When my chapter finally was outlined and written, it surprised me a bit; whether it will surprise you or not, we can leave to the event. My chapter turned out to propose a thesis to which perhaps the other chapters in this volume will serve as antitheses. That will allow us to arrive at the great Hegelian synthesis that we all wish for.

2.2 HUMAN LEARNING AND MACHINE LEARNING

I must begin, after all, by saying something about human learning, because I want to compare and contrast what is involved in human learning with what is involved in machine learning. Out of the synthesis of that contrast—in itself a thesis and antithesis—will come my thesis.

2.2.1 Tediousness of Human Learning

The first obvious fact about human learning is that it's horribly slow. It takes decades for human beings to learn anything. It took all of us six years just to get up to starting speed for school, and then twenty more years to become cognitive scientists or computer scientists. That is the minimum—some of us took even longer than that. So, we're terribly slow learners. We maintain big expensive educational systems that are supposed to make the process effective, but with all we've been able to do with them—to say nothing of computer aided instruction—it remains a terribly slow process.

I can still remember, although it was 45 years ago, trying to learn how to do multiple regressions by the Gauss-Doolittle method with the aid of a desk calculator. There's nothing complicated about the method except when you're learning it. And then it seems terribly mysterious. You wonder why this gets multiplied by that, and after a long while it gradually dawns on you. As a matter of fact you can carry out the calculations long before you understand the rationale for the procedure.

Learning the linear programming simplex method also illustrates this point and another one as well. Even after you've learned it, even after you've understood it, even after (in principle) you can do it, you still can't *really* do it because you can't compute fast enough. I don't know of any humans who calculate solutions of LP problems by the simplex method; as far as I know it's all done by computers. The human doesn't even have to know the simplex method; he just has to know the program library—cookbook statistics, or cookbook computing, which we all do most of the time.

So human learning is a long, slow process. It should give us some pause, when we build machine learning systems, to imagine what can possibly be going on during all the time a human being is mastering a "simple" skill. We should ask whether we really want to make the computer go through that tedious process, or whether machines should be programmed directly to perform tasks, avoiding humanoid learning entirely.

Of course we might discover a trick: a method of machine learning that was orders of magnitude faster than human learning. Whether such tricks exist depend on whether the inefficiencies of human learning derive from peculiar properties of the human information processing system or whether they will be present in any system that tries to extract patterns or other kinds of information from complex, noisy situations and to retain those patterns in a manner that makes them available for later use. The search for such tricks that manage to escape the tediousness of human learning, however, provides a strong motivation for research in machine learning.

2.2.2 Learning and Copying

The second distinctive feature about human learning is that there's no copy process. In contrast, once you get a debugged program in the computer you can have as many copies as you want (given equivalence in operating systems and hardware). You can have these copies free, or almost free. When one computer has learned it, they've all learned it—in principle. An algorithm only has to be invented once—not a billion times.

I've been involved a little bit in tutoring someone during the last few weeks in beginning calculus. I think I know the calculus pretty well—I haven't used it much for years, but it comes back. Yet I find it terribly frustrating trying to transfer my knowledge and skill to another human head. I'd like to open the lid and stuff the program in. But for one thing, I don't know where it is in *my* head, I don't even know what language it's encoded in. For another thing, I have no way of transferring it to the other head. That, of course, is why we humans go through the slow learning process—because we can't copy and transfer programs.

2.2.3 Why Machine Learning?

Contrast this with the machine learning task. In machine learning, the minute you have the debugged program you read it into the computer and it runs. The computer does what the psychologists call "one-trial learning". And, as I've already indicated, what is learned can be copied *ad nauseam*. So, if one thinks about that a little, one says, "What's all this about machine learning? Why are we interested in it—if by machine learning we mean anything that's at all like human learning? Who—what madman—would put a computer through twenty years of hard labor to make a cognitive scientist or a computer scientist out of it? Let's forget this nonsense—just program it." It would appear that, now that we have computers, the whole topic of learning has become just one grand irrelevancy—for computer science.

I have already qualified that conclusion in one respect: we do have reason to search for machine learning programs that will avoid the inefficiencies of human learning, although we must be alert to the possibility that such programs cannot, in principle, be constructed. The difficulty may be intrinsic in the task; human learning, though slow, may be close to optimally efficient.

I must also enter another caveat because you'll ask me, "What were you saying in that talk you gave two months ago? Why were you talking about learning?" The caveat is: Even in a world in which there are lots of computers it still may be important for us to understand human learning. Artificial intelligence has two goals. First, AI is directed toward getting computers to be smart and do smart things so that human beings don't have to do them. And second, AI (sometimes called cognitive simulation, or information processing psychology) is also directed at using computers to simulate human beings, so that we can find out how humans work and perhaps can help them to be a little better in their work.

None of the doubts I have just raised about computer learning apply to this second application of AI. Anybody who is interested in machine learning because he wants to simulate human learning—because he wants to understand human learning and thinking, and perhaps improve it—can pursue his interest in good conscience. But what about those who have other goals?

2.3 WHAT IS LEARNING?

When I had arrived at this point and surprised myself by writing down these notes, I asked myself, "What can we talk about legitimately for the next three days, other than cognitive psychology?" But I looked at the names of the people who were going to be here and at some of the titles of papers in the program, and I decided that a good deal of what we were talking about wasn't really learning anyway, so it was all right.

Let me elaborate on that remark. The term "learning", like a lot of other everyday terms, is used broadly and vaguely in the English language, and we carry those broad and vague usages over to technical fields, where they often cause confusion. I just saw a notice of a proposed special issue of SIGART, with a list of kinds of learning. It's a long list, and I'd be astonished if all of the items on it denote the same thing. Maybe it is just a list of the different species of learning, but I suspect that it also reflects the great ambiguity of the term "learning".

2.3.1 A Definition of Learning

The only partially satisfactory definition I've been able to find is that learning is any change in a system that allows it to perform better the second time on repetition of the same task or on another task drawn from the same population. The change should be more or less irreversible—not irreversible in the sense that you can't unlearn (although that sometimes is hard, especially unlearning bad habits) but irreversible in that the learning doesn't go away rapidly and autonomously. *Learning denotes changes in the system that are adaptive in the sense that they enable the system to do the same task or tasks drawn from the same population more efficiently and more effectively the next time.*

Since we may want the same task done over and over and over again, tuning a system so that it runs very fast is a great thing. Human beings seem to have some tuning capabilities, often called automating task performance. But more often, particularly in the university, we're interested in learning, not so that the same task can be done over and over again, but so that we acquire the ability to perform a wide range of tasks (for example, solving problems that appear on examinations, or performing similar tasks that may occur afterwards in real life).

2.3.2 Learning and Discovery

There are relations between learning and some other activities. For one thing, learning is related to discovery. By discovery I mean finding new things. Very little human learning is discovery. Most of what we know somebody told us about or we found in a textbook. At the very best we acquired it by working out some very carefully selected exercises, which guided us nicely in the right direction and provided most of the selective heuristics for our search. There can be all kinds of learning without discovery, and there usually are. Most of the things we know were discovered by other people before we knew them, and only a few were even reinvented by us.

Nevertheless, there is a relation between learning and discovery, because if you do discover something and it's good, you'd like to retain it. So, if you have a discovery system, you would like (somehow or other) to associate a learning system with it, even a simple memorization and indexing scheme. That doesn't quite get us off the hook. If you have a computer that discovers the proof for Goldbach's Theorem or the Four Color Theorem, you don't have to have a separate learning program, for you can simply get the proof out of the computer and transport it around on paper in the usual way. But, it would be very convenient if the computer could store the proof so that it could be used in subsequent work.

One of the first learning programs for computers was the little learning routine in the Logic Theorist (LT) [Newell & Simon, 1956]. When the Logic Theorist had the good fortune to prove a theorem in *Principia Mathematica* it had the good sense to keep the theorem around. On the next problems, it didn't start from the axioms alone but could use the new theorem along with the axioms. It wasn't any great feat to program this learning program. It did what we teachers call (pejoratively) "rote learning"—just memorizing. LT memorized only the theorem, not the proof; but giving it the latter capability also would have been a trivial matter.

In the Artificial Intelligence literature, the distinction I have been maintaining here between discovery and learning is not usually observed. That is to say, a great many machine "learning" systems are also discovery systems; they discover new knowledge that they subsequently retain. Most of the skeptical arguments I have raised about machine learning do not apply to the discovery process. Hence, I think it quite appropriate that a large part of the research effort in the domain of "machine learning" is really directed at "machine discovery". As long as we are not ourselves confused by the terminology, I do not even see a strong reason to object to this substitution of terms.

2.3.3 Learning and Understanding Natural Language

So, there's a connection here between learning and discovery. There is also a connection between learning and understanding. Understanding includes the whole natural language problem. In human life (and I'll try later to connect

this up with computers) most of what we learn we get from other people, communicated to us in natural language. A good many of the tasks that people have undertaken for machine learning have involved a natural language front end as an important part of the task. It is also a very annoying part of the task, eating up all of your time and energy when you wish you were doing something else.

2.3.4 Learning and Problem-Solving

Additionally, some things we might call "learning" could also be called "problem-solving". I've heard "automatic programming" called "learning". The aim of automatic programming is to be able to say the same brief vague things to a computer you'd say to a good human programmer in defining a task for him and to come out with a program on the other end. What the automatic programming program does is not really learning; it is solving the problem of getting from the sloppy ill-structured input statement of the programming problem to a well-structured program in the programming language. This kind of "learning" could readily come under the usual heading of "problem-solving".

Nevertheless, traditionally at least, the tasks of discovery, of natural language understanding, and of self-programming have often been intermingled with, or even identified as, learning tasks. If you want to call it learning you won't get an argument from me. It really isn't learning but ...

2.4 SOME LEARNING PROGRAMS

I'm going to back off one step further from my unkind words about machine learning and look at some "classical" examples ("classical" in the field of computer science is anything twenty years old) of learning programs, to see whether they really justify my harsh judgment.

2.4.1 Learning to Play Checkers

The first that I ought to mention is surely Arthur Samuel's checker program [Samuel, 1959]. Here was a program that, in the morning, wasn't very much of a checker player. But after you switched on its learning process and gave it games to play and other training exercises, by evening it was a State-champion-level checker player. That is a lot better than any of us could do. So there's a very impressive example of a learning program going back twenty-five years.

Let me submit that however fine this program was from an AI standpoint, it only made sense if we really didn't understand checkers. If Samuel had understood checkers well, he could have put the final evaluation function in right at the beginning. (You may recall that he used two kinds of learning, but the only one I want to mention at the moment is tuning the evaluation function for positions on the basis of outcomes. When good things happened, items that were heavily weighted in the evaluation function got additional weight, and when bad things happened they lost some of their weight.) If Samuel had known the right evaluation function at the outset, he would have put it in the program; he would not have gone through all the learning rigamarole. It cost only one day of computing time, to be sure, but computers were expensive then, even for one day.

It does make sense to provide for such learning in a task where you don't know enough to do the fine tuning. We might think of this as an area of machine learning (or, more accurately, machine discovery) where we can get the system to behave better than it would behave if we just sat down and programmed it. Nobody writing chess programs has had this feeling yet. They all think they know more chess than a computer could acquire just by tuning itself. As far as I know, none of the successful chess-playing programs have had any learning ability.

So there are cases where the computer can learn some things that we didn't know when we programmed it. But if you survey the fields of AI and knowledge engineering today, you will find very few cases where people have had the feeling this could or should be done, or have had any ideas of how to do it. Nevertheless, this potential application of learning procedures is certainly one further qualification on my general stricture against such programs.

I've already mentioned learning by the Logic Theorist, but that was just convenience, unless LT had reached the point where it was discovering genuinely new things. If Doug Lenat had let AM [Lenat, 1977] run for another two hours—as I kept telling him he should—and it had discovered something completely new, then the learning would make sense, for you would want to save what had been discovered.

2.4.2 Automatic Indexing

There's something to be said (again, largely on convenience grounds) for systems that are capable at least of learning discrimination nets—EPAM nets, if you like [Feigenbaum, 1963]. If you're building up a big data base and adding information to it all the time, you want easy access to that information, and so you want an index. It's a lot more convenient to have the system index itself as it goes along, than to index it by hand.[1] Or if you're building a large production system and don't want to search it linearly, you're going to incorporate an index in the production system to select the order in which it performs the tests. There is no difficulty in automating that; we have known for twenty-five years how to do it. So why not?

So there's some room for learning there. I don't know whether there's much room for learning *research*, since the technology of growing discrimination

[1]By "indexing" I mean building up a tree or network of tests so that you can access a data store in ways other than by linear search.

nets, alias indexes, is already pretty well developed, but someone may find a great new way of doing it.

2.4.3 Perceptrons

A final "classical" example (this is a negative example to prove my point) is the whole line of Perceptron research and nerve net learning [Rosenblatt, 1958]. A Perceptron is a system for classifying objects (that is, a discovery and learning system) that computes features of the stimulus display, then attempts to discriminate among different classes of displays by computing linear additive functions of these features. Functions producing correct choices are reinforced (receive increased weight), those producing incorrect choices have their weights reduced. I have to conclude (and here I don't think I am in the minority) that this line of research didn't get anywhere. The discovery task was just so horrendous for those systems that they never learned anything that people didn't already know. So they should again strengthen our skepticism that the problems of AI are to be solved solely by building learning systems.

2.5 GROWTH OF KNOWLEDGE IN LARGE SYSTEMS

In the remainder of my remarks, I would like to focus attention on large knowledge-based AI systems, particularly systems that can be expected to continue to grow and accumulate over a period of years of use. We may find in such systems some reasons to qualify a general skepticism about the role of learning in applied AI. Medical diagnosis systems like INTERNIST [Pople, 1977] and MYCIN [Shortliffe, 1976], and the venerable DENDRAL program [Feigenbaum *et al.*, 1971] are examples of the kinds of systems I have in mind.

There has been attention (as in TEIRESIAS [Davis, 1981] and other such efforts) to designing an effective programming interface between these knowledge-based systems and the humans who are supposed to improve them, and you can call that learning (or instruction). Most of the work has been aimed at making the job of the human easier. (Perhaps that's unfair, for it's a mutual job for the two of them.) So one might think of the man-machine interface as a good locus for learning research.

2.5.1 The ISAAC Program

To make my remarks more concrete, I would like to discuss for a bit Gordon Novak's well-known ISAAC system, which solves English-language college physics problems of the sorts found in textbooks [Novak, 1977]. Although ISAAC is primarily a performance or problem-solving program, one can think of some interesting ways of complementing it with a learning program.

ISAAC has a data bank containing schemas that describe various kinds of simple objects that physicists talk about—levers, masses, pivots, surfaces, and the like. A schema is just what you'd expect—a description list of an object with slots that you can fill in with information about its characteristics. In addition, ISAAC has, of course, some productions and a control structure.

When you give ISAAC a physics problem in natural language out of the physics textbook, it uses its schemas and productions to produce an internal representation of the problem. The representation is another node-link structure, which you can think of as a super-schema made up by assembling and instantiating some of the basic schemas. ISAAC will assemble some levers, and some masses, and a pivot or two, and a surface in the way the problem tells it, and make a problem schema out of them. At the outset it parses the sentences stating the problem, using its schemas to extract structure and meaning from them, and builds its internal representation, a problem schema.

This internal representation contains so much information about the problem that ISAAC uses a little subsidiary program to depict the problem scene on a CRT. Of course the real reason ISAAC wants this internal representation is not to draw a picture on a scope, but to use it to set up an appropriate set of equations and solve them.

Notice that ISAAC doesn't try to translate the natural language problem directly into equations, as Bobrow's STUDENT program did for algebra [Bobrow, 1968]. It first builds up an internal representation—what I think a physicist would call a physical representation (a "mental picture") of the situation. It then uses that intermediate representation to build the equations, which it ultimately solves. The internal representation does a lot of work for ISAAC because it identifies the points where forces have to be equilibrated and therefore identifies which equations have to be set up.

2.5.2 A Learning Extension of ISAAC

We can enlarge ISAAC by adding to it an UNDERSTAND program [Hayes & Simon, 1974]. Now you're going to say, "Ah ha! ISAAC already has an understanding program, because ISAAC can understand the problems it is given." That is true. But to do this, ISAAC must already have in memory a rich set of schemas describing physical devices, and it must already have the set of productions that allow it to organize these schemas into an internal representation. So ISAAC already knows all the physics it's going to know. While it understands problems, how about understanding *physics*? This would require the ability to use natural language information to construct new schemas and new productions. This is what the UNDERSTAND program does—not for physics, but for slightly simpler domains. UNDERSTAND creates, from the natural language, schemas for the kinds of objects being talked about and their relations. (In fact, Novak is presently exploring similar lines of investigation.)

What I want to ask about this whole amalgam of ISAAC and UNDERSTAND is, what is the place here for learning research in AI? (I know what the place is here for learning research in psychology. I think this is a very important area.

But let's continue to talk about the AI side of it.) If we understand the domain ourselves, if we understand physics, why don't we just choose an internal representation and provide the problems to the system in that internal representation? What's all this learning and natural language understanding about? Or, if we still want to give the system a capability of doing the problems in the back of the textbook, which are in natural language, then lets build Novak's ISAAC system. Why go through all the rigamarole of an UNDERSTAND program to learn the schemas and the productions painstakingly instead of just programming them? Before you launch into such a project as an AI effort (as distinct from a psychological research project), you have to answer that question.

2.6 A ROLE FOR LEARNING

Since you have listened very patiently to my skeptical challenge to learning as the road to the future in AI, I think I should own up to one more important qualification that needs to be attached to my thesis—a little fragment of the more complete antithesis that the other papers of this volume develop.

I began by running down the human species—emphasizing how stupid we all are as revealed by our agonizingly slow rates of learning. It is just possible that the complexity of the learning process is not an accident but is, instead, an adaptive product of evolution. The human brain is a very large collection of programs that cumulates over a lifetime or a large part of a lifetime. Suppose that we were allowed to open up the lid and program ourselves directly. In order to write debugged programs, modifications of our present programs, we would have to learn a lot about the internal code, the internal representations of the knowledge and skills we already possess.

Perhaps you know how knowledge is organized in your brain; I don't know how it is organized in mine. As a consequence, I think it would be exceedingly difficult for me to create a new, debugged code that would be compatible with what is already stored. This is, of course, a problem that we already encounter with our time-shared computers today. As we add new utility programs, or modify the monitors or operating systems, we encounter all sorts of interactions that make these modifications cumulatively harder to effect. At best, we encapsulate knowledge in hosts of separate programs that can operate independently of each other, but by the same token, cannot cooperate and share their knowledge effectively. Old programs do not learn, they simply fade away. So do human beings, their undebuggable programs replaced by younger, possibly less tangled, ones in other human heads. But at least until the state of undebuggability is reached, human programs are modified adaptively and repeatedly by learning processes that don't require a knowledge of the internal representation.

It may be that for this kind of system (a human brain or the memory of a very large time-shared computing system) the *only* way to bring about continual modification and improvement of the program is by means of learning procedures that don't involve knowing the detail of the internal languages and programs. It is a salient characteristic of human learning procedures that neither teacher nor learner has a detailed knowledge of the internal representation of data or process. It may turn out that there aren't procedures more efficient than these very slow ones that human beings use. That's just a speculation, but we ought to face the grim possibilities as well as the cheery possibilities in the world.

Even if we had to accomplish our complex programming in this indirect way, through learning, computers still would have a compensation—the costless copying mechanism that is not shared by human beings. Only one computer would have to learn; not every one would have to go to school.

2.7 CONCLUDING REMARKS

By now you are aware that my case against AI research in learning is a very qualified case with several important exceptions—exceptions you may be able to stretch until they become the rule. Let me put the matter in a positive way, and rephrase these exceptions as priorities for learning research. They are five in number.

1. I would give a very high priority to research aimed at simulating, and thereby understanding, human learning. It may be objected that such research is not AI but cognitive psychology or cognitive science or something else. I don't really care what it is called; it is of the greatest importance that we deepen our understanding of human learning, and the AI community possesses a large share of the talent that can advance us toward this goal.
2. I would give a high priority, also, to basic research aimed at understanding why human learning is so slow and inefficient, and correspondingly, at examining the possibility that machine learning schemes can be devised that will avoid, for machines as well as people, some of the tediousness of learning.
3. I would give a high priority to research on the natural language interface between computer systems and human users. Again, it does not matter whether you call it research on learning or research on understanding. We do want systems, particularly in the knowledge engineering area, in which we don't have to know the internal language or representation in order to interact with them. This is especially true, as I have just argued, if the systems are to be cumulative over many years.
4. I think there is an important place for research on programming from incomplete instructions (automatic programming), which is not unrelated to the preceding item. Giving instructions to a skilled programmer is different from writing the program yourself—else why hire the programmer? It is a very important research question to ask whether we can get the computer to be the skilled programmer.

5. My final priority is research on discovery programs—programs that discover new things. We may regard discovery itself as a form of learning, but in addition we will want to give a discovery system learning capabilities because we will want it to preserve and to be able to use all the new things it finds.

So now, I guess, I have come full circle, and have made a strong case for machine learning. But I do not think the effort in addressing my initial skepticism has been wasted. Research done in the right area for the wrong reasons seldom achieves its goals. To do good research on machine learning, we must have clear targets to aim at. In my view, the usual reasons given for AI learning research are too vague to provide good targets, and do not discriminate with sufficient care the learning requirements for people and computers, respectively.

Perhaps the deepest legitimate reason for doing machine learning research is that, in the long run for big knowledge-based systems, learning will turn out to be more efficient than programming, however inefficient such learning is. Gaining a deeper understanding of human learning will continue to provide important clues about what to imitate and what to avoid in machine learning programs. If this is true, then it follows that among the most important kinds of learning research to carry out in AI are those that are oriented toward understanding human learning. Here as elsewhere, Man seems to be the measure of all things.

ACKNOWLEDGMENTS

This research was supported by Research Grant MH-07722 from the National Institute of Mental Health, and a grant from the Alfred P. Sloan Foundation.

REFERENCES

Bobrow, D. G., "Natural language input for a computer problem-solving system," *Semantic Information Processing*, Minsky, M. (Ed.), MIT Press, Cambridge, MA, 1968.

Davis, R., "Applications of meta level knowledge to the construction and use of large knowledge bases," *Knowledge-Based Systems in Artificial Intelligence*, Davis, R. and Lenat, D. (Eds.), McGraw-Hill Book Company, New York, NY, 1981.

Feigenbaum E. A., "The simulation of verbal learning behavior," *Computers and Thought*, Feigenbaum, E. A. and Feldman, J. (Eds.), McGraw-Hill Book Company, New York, NY, 1963.

Feigenbaum, E. A., Buchanan, B. G. and Lederberg, J., "On generality and problem solving: A case study using the DENDRAL program," *Machine Intelligence*, Meltzer, B. and Michie, D. (Eds.), Edinburgh University Press, Edinburgh, Scotland, 1971.

Hayes, J. R. and Simon, H. A., "Understanding written problem instructions," *Knowledge and Cognition*, Gregg, L. W. (Ed.), Lawrence Erlbaum Associates, Potomac, MD, 1974.

Lenat, D. B., "Automated theory formation in mathematics," *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, IJCAI, Cambridge, MA, pp. 833-842, August 1977.

Newell, A. and Simon, H. A., "The logic theory machine," *IRE Transactions on Information Theory*, Vol. IT-2, No. 3, pp. 61-79, September 1956.

Novak, G. S., "Representations of knowledge in a program for solving physics problems," *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, IJCAI, Cambridge, MA, pp. 286-291, August 1977.

Pople, H., "The formation of composite hypotheses in diagnostic problem solving," *Proceedings of the Fifth International Joint Conference on Artificial Intelligence*, IJCAI, Cambridge, MA, pp. 1030-1037, August 1977.

Rosenblatt, F., "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review*, Vol. 65, pp. 386-407, 1958.

Samuel, A. L., "Some studies in machine learning using the game of checkers," *IBM Journal of Research and Development*, No. 3, pp. 210-220, 1959.

Shortliffe, E., *Computer-Based Medical Consultations: MYCIN*, American Elsevier Publishing Company, New York, NY, 1976.