AI Magazine Volume 14 Number 2 (1993) (© AAAI)

The Gardens of Learning

A Vision for AI

Oliver G. Selfridge

"Can we actually know the universe? My God, it's hard enough finding your way around Chinatown."

– Woody Allen, 1966

"Know then thyself, presume not God to scan; The proper study of mankind is man."

– Alexander Pope, An Essay on Man, 1733

■ The field of AI is directed at the fundamental problem of how the mind works; its approach, among other things, is to try to simulate its working—in bits and pieces. History shows us that mankind has been trying to do this for certainly hundreds of years, but the blooming of current computer technology has sparked an explosion in the research we can now do.

The center of AI is the wonderful capacity we call learning, which the field is paying increasing attention to. Learning is difficult and easy, complicated and simple, and most research doesn't look at many aspects of its complexity. However, we in the AI field are starting. Let us now celebrate the efforts of our forebears and rejoice in our own efforts, so that our successors can thrive in their research.

This article is the substance, edited and adapted, of the keynote address given at the 1992 annual meeting of the American Association for Artificial Intelligence on 14 July in San Jose, California.

Before I start, I should first make an acknowledgment—I am grateful to many people. I have had enormous help from friends and colleagues and from the GTE Laboratories where I worked. There are too many people to try to name them all.

I want to start this morning by sharing what I feel—my joy and my exultation at being here with you, at what we are all trying together to do to begin to solve the mysteries of the human mind.

I want to set it out straight. This is our ambition. Let us celebrate AI; let us celebrate, let us rejoice in it, all of us who have worked in AI; let us celebrate, all of us who now work in AI. Let us rejoice for our children in what we are striving for.

We are in a different position from the physicists who claim to know the universe with quasars and bosons. I was talking about the physicists who have placed us in the universe.

Now we know where we are, we have stars, we know that atoms are rather like stars. We know how the universe grows and shrinks. We know the big bang and everything about the universe. However, the question, of course, is, Who are we? What is inside us? What makes us work? I don't mean the physiology, the bones and the muscles. What I mean is who we are, how we think, how we feel, how we talk to each other. What makes a civilization of us?

Here we are; now that we know the atoms and the stars, we have to know ourselves. All of us here in this hall are really joining in that great search, and to it many of us have dedicated our careers and lives.

Here is what I am going to talk about. It's not even an overview. It's really how I feel about where we started, how we've been going, and where we are going to—all of us now.

> The beginnings of AI,
> The founders of AI,
> The nature of AI: one view—
> Learning is the essence.
> The nature of learning—
> What is AI for? What are the visions of AI?

This morning I should like to talk about how AI arose; I should also like to honor the founders of our discipline AI. Who are they that you select as giants? I have my own ideas, of course, of the giants in our field.

Then I want to talk about the discourse on what I think is the real center of AI, that is, learning as it occurs in animals, peoples, and machines. Machine learning, I think, is playing an increasingly central role in AI, and this role will grow. Of course, one cannot talk about learning without talking about the meaning of AI in what we do now. Is AI some simple goal to be achieved? This view is the popular one. Are the machines intelligent? Can a machine be intelligent? Do we think there is some simple Turing or meta-Turing test that will persuade the ungodly of our godliness? The answer is—of course not!

What great ideas we have! What new expressions! What new representations! We have ideas that, as Pat Hayes remarked, would have been totally unthinkable 50 years ago. These ideas and hopes we have are profound and disturbing and challenging. Many people find some sense of comfort in the notion that the nature of a person is somehow magic and unknowable. The human soul, the very nature of people! Can these be just clockwork, merely algorithms?

A popular response is rejection of the very idea. This rejection comes in many sizes and styles, like fonts, but there are three underlying ideas:

> It's a threat—it belittles people and makes them less worthy.
>
> It's no use—it's inherently not feasible.
>
> It's wrong—if God had meant us to fly, he would have given us wings.

All of us, I am sure, have had arguments about these aspects of AI and often with an ideological fervor. I think I know where most of you stand. We have all argued about Joe Weizenbaum and his condemnation of much of what AI is trying to do:

> I would argue that, however intelligent machines may be made to be, there are some aspects of thought that *ought* to be attempted only by humans. (Weizenbaum 1976, p. 13)

Weizenbaum is in fact a thoughtful and wise person, even if most of us disagree with him. We all know many less coherent objections to the ideas behind our field.

Does this question have to be answered? Well, not by us! What is our Holy Grail? It really is to understand the mindness of mind, to explain what makes a person behave in a human way, to interrelate the emotions and the hungers and the logic of us with our powers and our planning and our enormous joint enterprises that constitute civilization. Our approach, well supported by millennia of science, is, among other things, to build a model of the mind and its processes.

We have all seen, and no doubt will continue to see, articles questioning the faith of AI. Just the other day, in the British journal *New Scientist*, I read "Will Machines Ever Think?" by Harry Collins, a sociologist from the University of Bath in England.[1] Collins discusses many of the ideas of Bert Dreyfus, who is no doubt a favorite philosopher for nearly everyone in this hall.

Enough already! Mimicking behaviors is really not the issue. Turing tests will not tell us the answer because there is no answer! *AI Magazine* discusses the Turing test in an article by Bob Epstein (1992), "Can Machines Think?"

> Of course machines can think.
> They just don't—yet!
> And perhaps when they can they will choose not to; just like us.

It reminds me of when we used to believe that life was something that had to be defined. How could one tell whether something was alive or not? I remember that as a child I read that viruses, those often vicious infective agents, could be crystallized. How could life be crystallized and still be life? This question seems to me to be parallel to the popular question, But how can a machine think? It seems to me that although there might be popular questions, like the Turing test, most of us are beyond them now.

As the science of biology grew, the researchers tended not to worry about whether something was alive or not. They were too busy finding out things about it. That is why we are here. Instead of worrying about whether a particular machine can be intelligent, it is far more important to make a piece of software that is intelligent. Many of us are beginning to do so now.

I am speaking here as a scientist for that is how I regard myself. To some extent, you in this hall are all scientists, too, for I claim that to some extent you are all driven by the dreams that drive me. No doubt we all have other aims and goals: some to make money, some to build wonderful devices, and others for other things. It is the striving that counts.

But the way is difficult:

> The stumbling way in which even the ablest of scientists have had to fight through thickets of erroneous observations, misleading generalizations, inadequate formulations, and unconscious prejudice is rarely appreciated by those who obtain their scientific knowledge from textbooks. (Conant 1951)

It is also rarely appreciated by those who get their scientific knowledge from newspapers or family friends. You have to be in it, as we all

It is more important to make intelligent software than to worry about proving that a machine can be intelligent.

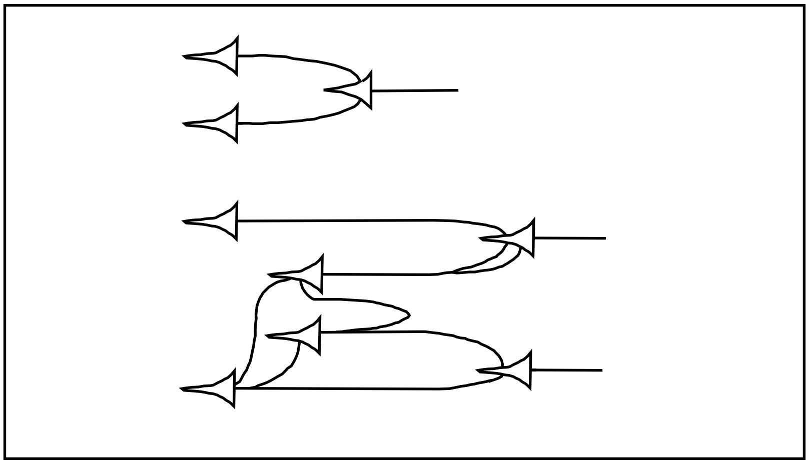


Figure 1. Two Neural Nets from McCulloch and Pitts (1943).

are. As the poet W. H. Auden said 40 years ago, “The way is both difficult and steep, Look if you like, but you will have to leap.”

I have watched AI since its beginnings, and I cannot properly express to you the thrill of it all, the heartfelt exultation. It is not the thrill of a roller coaster, nor the satisfaction of cooking a really good roast beef; it is not the excitement of writing a really good sonnet, nor discovering a new mathematical truth. It is, rather, the deep and enduring satisfaction of watching and helping a child or a family grow up to be responsible and creative. That’s us. That’s you.

In 1943, I was an undergraduate in mathematics at the Massachusetts Institute of Technology (MIT) and met a man whom I was soon to be a roommate with. He was but three years older than I, and he was writing what I deem to be the first directed and solid piece of work in AI (McCulloch and Pitts 1943). His name was Walter Pitts, and he had teamed up with a neurophysiologist named Warren McCulloch, who was busy finding out how neurons worked (McCulloch and Pitts 1943).

This paper followed the startling work by mathematicians in the 1930s about the computability of certain numbers and the undecidability of certain questions. What Walter and Warren showed was that a network of certain neurons working in a rigidly defined way could compute any number that any other machine could compute. They knew that their model of the neuron was highly simplified, but even so, it is one that neural net researchers of today instantly recognize.

Figure 1 shows a couple of examples of neural nets taken from this paper—the first AI paper ever. Notice that there are weights because some of the connections are doubled. Notice that it’s a many-to-many mapping—the lower one, that is. Notice that there are hidden units—all the pieces that work today. It seems to me that connectionists or neural net researchers will find them even familiar. Another paper came from Pitts and McCulloch three years later called “How We Know Universals: The Perception of Auditory and Visual Forms” (Pitts and McCulloch 1947). This paper was the beginning of pattern recognition, concept analysis, and concept learning for neural nets.

One of the astonishing things about both papers is the foresight and relevance to the work of today, nearly 50 years later. I urge those of you who have not read them to do so.

The authors were astonishing themselves. Walter was an extraordinary and aberrant genius who led a not very long life, and he provided the meat of the first paper when he was but 19. We were roommates for some years, and retrospectively I’ve always been astonished at how fast he understood things and how gentle and thorough were his judgments.

Warren McCulloch was born early in the century and listed himself as a neurophysiologist. Warren was also a philosopher and a man of great wisdom and supported a large number of students both professionally and personally. He discovered Walter in 1942 in Chicago. I want to pay Warren homage. There was within him the soul of a great man with a handsome admixture of the curiosity and the energy of a child. Warren and Walter were the most exciting people I have ever known, I think.

In the late 1940s, other developments were rife. Remember that there were no computers yet. My adviser Norbert Wiener (1948) was writing *Cybernetics*; Schockley was busy at Bell Laboratories inventing transistors, Claude Shannon was discovering information theory there as well, and Donald Hebb was looking at Walter’s and Warren’s ideas and thinking about cell assemblies.

Warren McCulloch, Walter Pitts, John von Neumann, Heinz von Foerster, Donald Hebb, Gray Walter, Norbert Wiener, J. C. R. Licklider: I call these people now the heralds of AI. Von Neumann, although he was chiefly known for computers, was enormously interested in AI and wrote about self-reproducing automata. Heinz von Foerster was working in perception and ethics in machines in the 1940s. Donald Hebb was working on cell assemblies. Gray Walter was building feedback machines. Norbert Wiener was writing about cybernetics, feedback, behavior, and thinking. J. C. R. Licklider played a special role. He was a young psychologist then, but he was also a catalyst for supporting AI as a means of understanding what AI was about. Through the Defense Advanced Research Projects Agency and Bolt Beranek and Newman (BBN), he funded nearly all the organized beginning AI work. He funded the AI Lab work, and he supported John McCarthy when he put together the first time-sharing system on the PDP1 at BBN.

As important as all these heralds are, I should like to remind you that in those days, before many of you were born, Marvin Minsky was entering Princeton University. Like Warren McCulloch and Walter Pitts, like von Neumann and Heinz von Foerster, like Donald Hebb and Gray Walter, like Norbert Wiener and J. C. R. Licklider, Marvin Minsky wanted to find out how brains work and what a mind is. For his bachelor thesis, he built and analyzed a neural net. I spoke to him earlier this year:

> *O*: You were around when AI started, and by AI, I mean not the forties work, which was all paper, but the beginning of trying to make actual machines or actual programs. Your own thesis was on neural nets, and, I think, it can be reasonably said to be the first doctoral thesis in AI at Princeton.
>
> *M*: That's a nice idea and certainly the first construction of a neural analog machine of any scale, but I suppose Gray Walter's machine was … but mine actually had Hebb synapses, so that that was a real reinforcement machine.
>
> *O*: That was in 1950, 1951?
>
> *M*: I built the machine in the summer of '51. I designed it in about '50.
>
> *O*: And you built it at?
>
> *M*: I built it at the Harvard Psycho-Acoustics Laboratory in the middle of that wonderful environment with Skinner at one end doing behaviorism and….

We all then wanted to find out how brains work and what a mind is. We've always wanted to build a man out of clockwork—not to denigrate man but, rather, to honor clockwork and show the perfections of the universe mirrored in the perfection and eternity of mankind.

Let me pay tribute to the other early founders of our field; I call them the "great ones." You know them all. It's hard to talk about John McCarthy without going on for too long. I treasure him for his solid drive to formalize his ideas: the notions of implementing commonsense in a formal system and Lisp. God knows, the love and hate we have for that language!

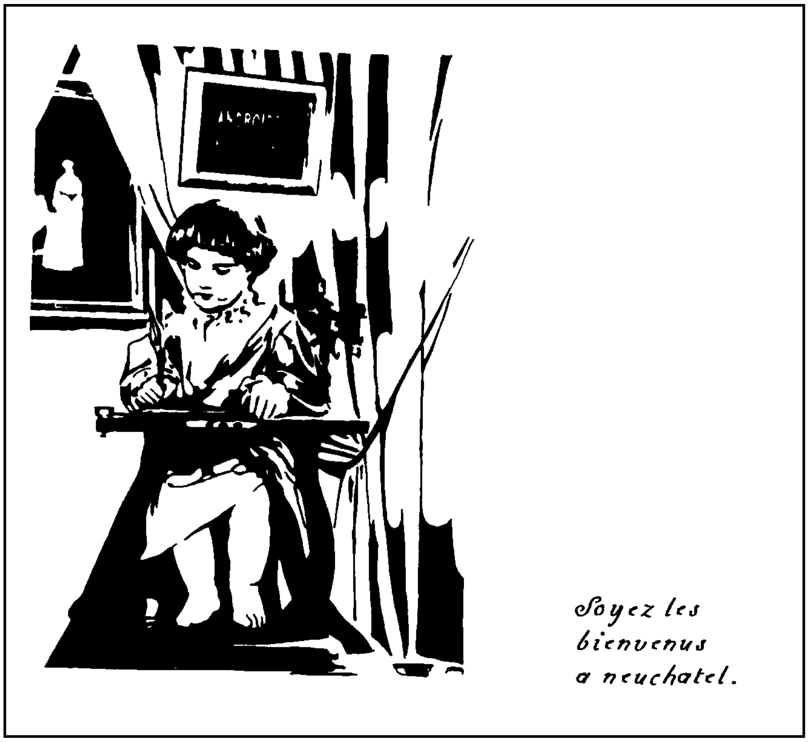


Figure 2. The Jaquet Droz Writer, 1744, and a sample of its writing (adapted from Chapuis and Droz 1958).

I helped introduce Allen Newell to AI. I remember him for his early interest in cognitive science, and his expressions of enthusiasm for it. I think of him as having brought cognitive science naturally into AI; or, rather, not just brought it into AI, but synergized and catalyzed cognitive science to play its proper role in AI. Specifically, of course, he and Herb Simon put together CPS. Alas, Allen died in Pittsburgh, Pennsylvania, on July 19, 1992; requiescat in pace.

My own specialty is learning, and I should like to pay homage to Arthur Samuel who did his really astonishing work with checkers and learning in the middle and late 1950s.

A strange interlude here: I don't call him a founder or a great one, but Frank Rosenblatt is worth a great deal of attention. Rosenblatt, who died in the early 1970s in a boating accident, built the PERCEPTRON in the late 1950s. Now, the PERCEPTRON was a single-layer neural net with no feedback and no hidden units. Frank analyzed these and built them. Most of the ones he built never worked. That is, they never did what they were intended to do.

Ed Feigenbaum is the last great one I have, and he is here today. His work in idealizing knowledge, putting it together into systems that can work and control, is the stuff of greatness. One of the things about these people that I've been listing is that they're all young. When I look around this hall, you guys are all young. That is as it should be.

Before I go into learning, I want to talk a little bit about where these ideas came from. Why are we so interested in what the mind is? Well, it turns out, if you look at history, that we have always been interested in building a man, to show that a person is in some sense a piece of clockwork.

In the seventeenth and eighteenth centuries, the clock makers built automata of the most ingenious kind, and there is little doubt of what they were really trying to do. Figure 2 shows an early automatic writer and its writing; it was built about 150 years ago with a pen and a face (Chapuis and Droz 1958).

Then there was a famous duck (figure 3). In 1738, Vaucanson in Vienna built a duck that quacked and flapped its wings. Ninety years later, a researcher in Vienna called Rechsteiner built another one; it was apparently modeled on the Vaucanson duck, or even conceivably used it. In 1847, the *Allgemeine Bayrische Kronik* wrote:

> All the movements and attitudes of this automaton faithfully reproduce nature, copying it to the life even down to the tiniest detail.... Here is clearly something more than mere mechanical ability. The artist has penetrated into the deepest secrets of [life]. This grasp of the secret of natural processes and the practical application of knowledge [is] an immense step forward in the world of natural science. (Chapuis and Droz 1958, p. 239)

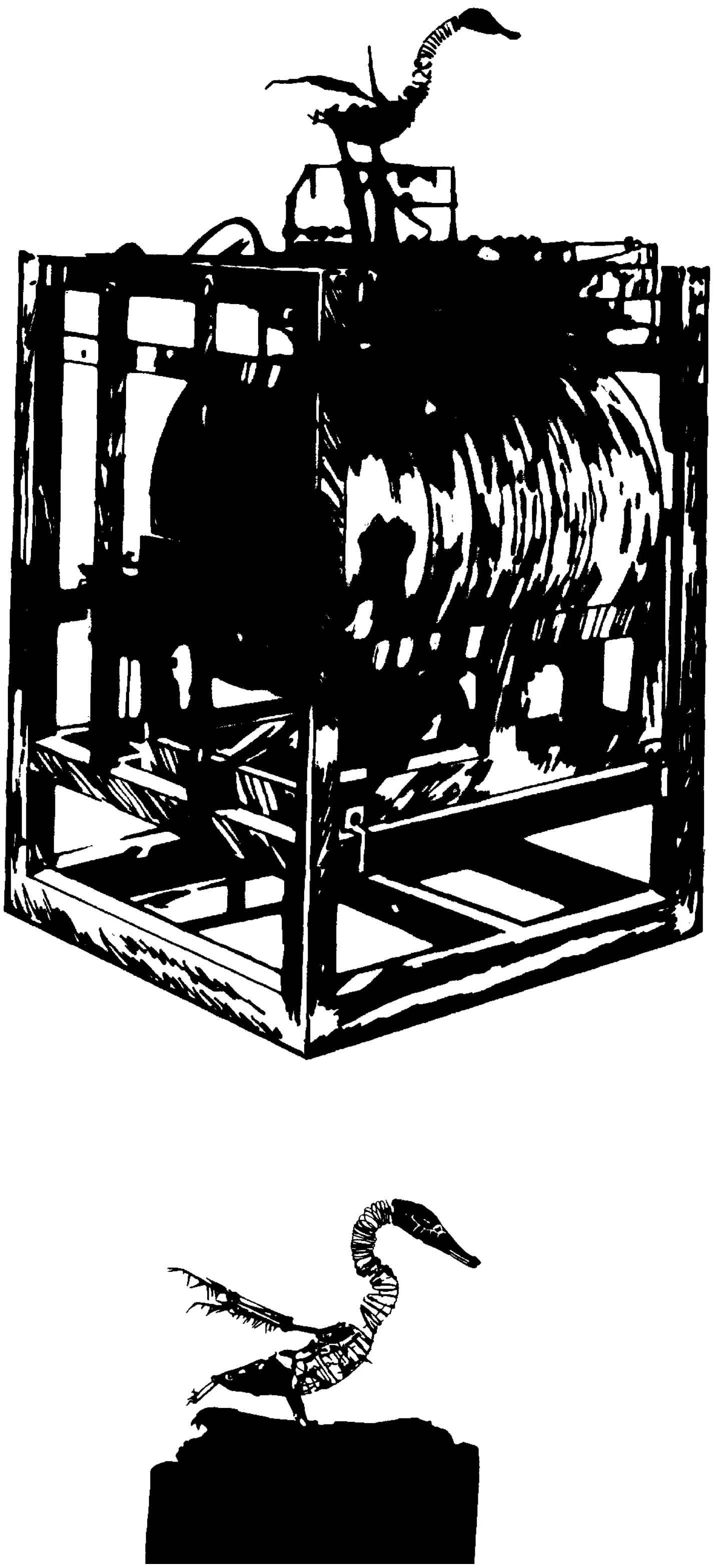


Figure 3. Rechsteiner's Duck (1846), Rebuilt from Vaucanson's Duck (1738) (adapted from Chapuis and Droz 1958).

Notice the drum memory underneath (figure 3, left). This is, I think, still extant today, but these pictures were taken 60 years ago. The big disks control the individual motions. The artist has penetrated, they say, "the deepest secrets of life" (Chapuis and Droz 1958).

Here is a description from another newspaper in 1847; it gives the game away (Chapuis and Droz 1958, p. 238).

> ...the duck in the most natural way begins to look around him.... [on seeing a bowl of porridge, the duck] plunges his beak deep into it, showing his satisfaction by wagging his tail. [It is] extraordinarily true to life. In next to no time the bowl has been half emptied, although on several occasions the bird, as if alarmed by some unfamiliar noises, has

> raised his head and glanced around.... [satisfied, he] begins to flap his wings and [gives] several contented quacks.
>
> But most astonishing are the contractions of his stomach, showing that [it] is upset by this rapid meal ... and after a few moments we are convinced in the most concrete manner that he has overcome his internal difficulties. The truth is that the smell of the fart that now spreads through the room becomes almost unbearable. We wish to express to the artist inventor the pleasure that his demonstration gave to us.

Now, the question for us is when we make our demos—and Norbert Wiener sees this too—we see that the logic of the machine resembles human logic, but is the machine just making a mock fart? Has the machine a more eminently human characteristic? Well, can it? I remember Norbert asking this and my noting it down: Has it the ability to learn?

> We thus see that the logic of the machine resembles human logic.... Has the machine a more eminently human characteristic as well—the ability to learn?... Thus the brain, under normal circumstances, is not the complete analogue of the computing machine but rather the analogue of a single run on such a machine. (Wiener 1948)

Norbert's comments were made in 1948, and it is now nearly half a century later. The great ones have started our field. As I said at the beginning of this talk, it thrills me to see all of us here trying to find out how the brain works. In this endeavor, of course, we are the scientists and the seekers.

Now we get to the meat of what I have to say: To me, the most important part of intelligence is not the knowing of something, whether that something is a fact or a skill or anything else. Rather, it is the changing of what you know, usually so that you know more and for the better. I mean knowing not only in the sense of knowing an answer but also in the senses of knowing how to do something, knowing how to assess someone's feelings, knowing what's funny and what's beautiful, and knowing how to catch a ball and thread a needle. All these things have to be learned.

My own view is that an expert system, a knowledge-based system is not, of itself, intelligent. If an expert system—brilliantly designed, engineered, and implemented—cannot learn not to repeat its mistakes, it is not as intelligent as a worm or a sea anemone or a kitten.

I want to pay tribute now to the researchers in machine learning. You are the core, the essence of AI. I salute you, machine learners!

What I am going to do now is to describe to you what I think learning is. I think it is much broader than what most of the researchers in machine learning believe. In some ways it is simpler, and in some it is much harder indeed.

This model is mine, and there are many other ways of talking about it. I believe that learning is a complicated thing in people, nearly always. We all know that there are several kinds of learning, and many of them do not seem to me to be covered by current research in machine learning.

Here are some examples of what I mean by learning:

> A child learns to talk.
>
> A dog learns to salivate on hearing a bell: the classical conditioned reflex.
>
> A rat learns to run a maze.
>
> A child learns that two plus three is five.
>
> A mosquito learns to drill for blood vertically.
>
> Habituation and sensitization are learned in many lower animals.
>
> I have lots more. These are interesting:
>
> A graduate student learns to integrate by parts.
>
> A worm learns to take the right path in a maze.
>
> As England greens, its moths become whiter.
>
> Bacteria become resistant to bactericides.

A graduate student learns to integrate by parts. In fact, it is this kind of problem that many people are looking at in machine learning, not so much a worm learning to take the right path in a maze.

The last two examples are interesting. "As England greens, its moths become whiter": It was noticed a hundred years ago that as the dark satanic mills in the middle of England blackened the countryside, the moths grew darker by evolution to be less detectable. Recently, the dark satanic mills began switching to less ecologically harmful activity, and the moths have become noticeably whiter. Now this example and the last one about bacteria are also some sort of tribute, as many of you are no doubt thinking, to genetic learning and John Holland, whom I hereby acknowledge and honor.

A child of 9 … not only knows what 29 x 29 is but also can work out what 17 x 17 is.

This kind of learning distinction is one that we need to put more of into our machines.

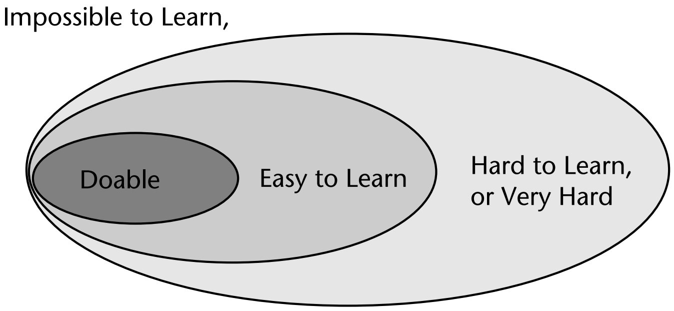


Figure 4. What You Can Do and What You Can Learn.

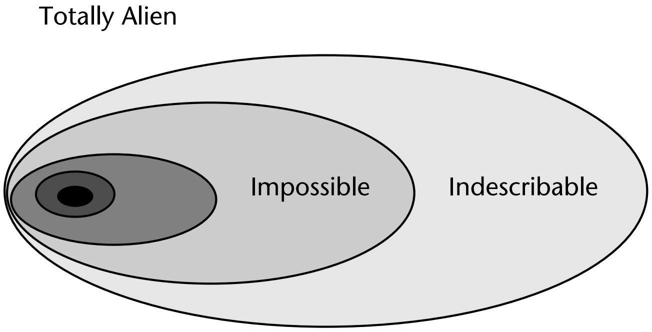


Figure 5. What You Can Learn and the Ocean of Ignorance.

Some of those other points are also worth a little more discussion. For example, I make a distinction between learning to do something and learning how to do something. It is clearly a tricky point, and it is a distinction that does not always hold. Let me try again with a conditioned reflex:

Mother Hubbard shows a dog a bone; the dog salivates.

Mother Hubbard rings a bell and shows a dog a bone; the dog salivates.

Mother Hubbard rings a bell; the dog salivates.

Mother Hubbard’s dog has learned to salivate.

Now how about learning how?

Mother Hubbard’s dog is gently wired to detect salivation.

Mother Hubbard’s dog is having a cat nap [sic] and dreams of a bone; so, he salivates. The detection of saliva causes a bone to drop, waking up the dog. After three or four episodes, the dog salivates to get the bone.

The dog has learned how to salivate.

I can go further than these examples; for me, any kind of purposive adaptation has within it the essential elements of learning (which is not to say that it is useful to describe a servomechanism, for example, as a learning device.)

The parallel of that example in people can be made easily:

Learning to multiply 29 and 29:

“What’s 29 x 29?” “841!”

“What’s 17 x 17?” “Er ... what?”

Learning how to multiply 29 and 29:

“What’s 29 x 29? “ “841!”

“What’s 17 x 17? “ “Er ... 289!”

That is to say, a bright 3 year old, such as we have all had or are going to have, can easily be taught to say, “29 x 29 = 841.” The child then knows that 29 x 29 = 841. You can prove it by asking the child. Unless he is being difficult, which is by no means beyond the realm of possibility, he will reply 841, or he might say, “Give me a quarter and then I’ll tell you.” However, if you then ask, What is 17 x 17? he will either reply, “841” or say “Er.”

A child of 9, however, not only knows what 29 x 29 is but also can work out what 17 x 17 is. This kind of learning distinction is one that we need to put more of into our machines.

Sometimes, this distinction does not apply, of course. Learning to drive a car is not noticeably different from learning how to drive a car. However, it is a point that has real relevance to machine learning. Remember that we do not know how we do most of the things we have learned. Which of us can provide a model of how we can tell a person from the way she moves? Which of us can work out how we know that person’s voice even when she whispers? I telephoned one of my sons the other day and coughed before I spoke, and he knew who it was. How?

There are some simple rules about learning:

Don’t make the same mistake twice.

Try what has worked before in similar situations.

Keep trying.

Don’t stop when you succeed.

Don’t make the same mistake twice: Each one of these rules is worth a full conference. What does “same” mean? What does “twice” mean? What’s a “mistake”? How do you tell? Sometimes, you should make the same mistake twice but mostly not.

Try what has worked before in similar situations: Many of us take this rule seriously. How do you measure similarity? The formalist would like to build metrics. People don't build metrics, at least not in the same sense and not formally. We have some kind of a sense. We don't know what these senses are. Do we have metrics of any kind? I am not sure. All these points need to be understood.

Keep trying: One of the pieces of knowledge that we acquire—and it's not an assertion—about a field or domain is when does it pay to keep trying? It clearly pays sometimes. It clearly does not at other times.

Don't stop when you succeed: This last rule is important—deep as well. When you've finally grasped a skill, you still have to tune it; you still have to improve it. You still have to learn when to use it and when not to use it.

I'm talking here about the broad range of changes we call learning. I'm not going to try to make a definition of learning. Many of you know that I think that making definitions at this point in our science is not only a waste of time but positively harmful.

Teaching is a special kind of encouragement to learning. Many people think that teaching is a kind of magic and that if we could just learn to teach in the right way, then any child would learn anything the teacher chose to teach. I'm reminded of *Henry IV* and this discussion between Glendower and Hotspur:

> *Glendower:* I can call spirits from the vasty deep!
>
> *Hotspur:* Why so can I, or so can any man;
> But will they come when you do call for them?
>
> *Glendower:* Why, I can teach you, cousin, to command
> The devil.
>
> *Hotspur:* [rather] … tell truth, and shame the devil.
>
> —*W. Shakespeare,* Henry IV, *part I, III, i 53*

Yet we do make ambitious claims about learning systems:

> "I can teach my machine to write a poem."
> "Ah, but will it learn to do what you are teaching?"
>
> "I will show your program how to have common sense."
> "Just show it how not to make the same mistake twice."

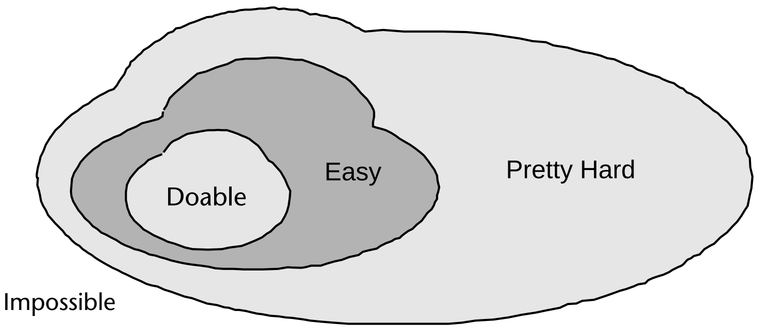


Figure 6. The Magic of Learning Pushes Back the Limits.

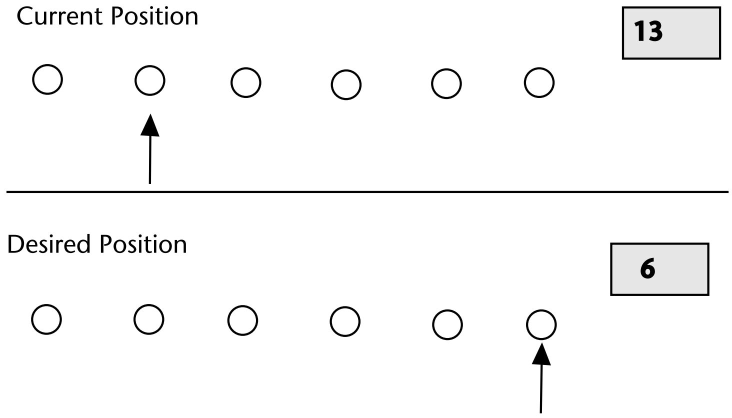


Figure 7. The Board for Counting.

The popular view is that most learning occurs through teaching. That is, the child or whatever is some kind of container into which you pour knowledge. It is believed that teaching is a kind of action done to an object that causes the object to have learned something. I think that this view is a backward and foolish way to think about learning. What is important is the student who is learning. The following is a way to think about teaching:

> Each person is in a garden full of growing knowledge and powers. Education is opening a door and showing another garden and its further delights.
>
> Learning is going through the door.
>
> You can't push a student through a door; you must entice him/her.
>
> Remember to ask why the student should care.

Here is a way to think about learning:

> Think of the things to do, the tasks to be undertaken, a vast sea of them. Most of them are impossible and unimaginable, but some are merely hard, a few are easy,

> and a very few you can actually do.
>
> Then you use the things you can actually do and learn how to do an easy task.
>
> The magic is that then some of the hard ones become easy, and some of the impossible ones become merely hard.

It looks like figure 4: the doable, the easy ones, the hard ones, and the impossible. Actually, the portrait in figure 4 is inaccurate. It is much more like figure 5, just a little spot of the doable ones.

The magic is that when the doable ones spread a little bit after something has been learned, then all the other ones spread too (figure 6).

What follows from this discussion is that you don't necessarily learn a hard task by facing that hard task. Let me put it another way to make the point clearer. I think of what a newborn baby can do; it's not much. However, as the child learns one thing, other things become easier. As the baby learns to reach for something, it learns not to overshoot its hand. Each learning adds to its repertoire. We do not yet have any good model of how the varied skills interact, how much is needed for a next step, or what are the multifarious ways of taking a next step.

You have a newborn daughter whom you want to be chess champion of the world. How do you start? Do you start by teaching the moves of the king, or do you teach about the value of the center, and so on? The answer is obvious. First, you teach the infant—help her—to reach something, to babble, to crawl, to walk, to talk, to interact. All those things of being a child contribute in unknown ways to ready her for the intellectual tasks of playing chess, although six or seven years later or, maybe for some of us, only three or four.

Of course, the progress made by a child is not a simple step, and it is not even a simple sequence. Think of it more as an entwined braid of tasks and aims and capabilities. Capabilities arise and grow; some can be replaced. Each new capability can till the soil, grow new flowers, and interact. Of course, all the flowers interbreed, giving new powers and freedoms.

In geology, one discusses rivers that flow out of the termini of melting glaciers; they are often said to be *braided* because of the substreams that form and reform, melt into each other, and separate. The early branches form the later branches.

> All learning should be easy.
>
> Learn the easy things first, and some hard things become easy.
>
> It is nearly impossible to learn anything that is hard to learn.

One example that illustrates these points is a program that my son Mallory and I wrote a decade ago; it is called COUNTING, and it is a kind of model for a machine learning to count without knowing how. Here is a board (figure 7); the arrow represents a finger pointing at different objects to be counted, so the top position is the board. I can give it a task to change the top position to the bottom position by means of the operations that follow:

Starting Capabilities (operations)

1. Move left	L(eft)
2. Move right	R(ight)
3. Increment counter	I(ncrement)
4. Decrement counter	D(ecrement)
5. Repeat operation	A(gain)

The program can move the pointer or finger one object to the left. This movement is supposed to be analogous to a child's moving his finger from one object to another. It can move the finger left or move it right. It can also move the counter up or down by 1, that is, adding 1 to the counter or subtracting 1 from it.

The other thing it can do is a kind of meta-operation. It can repeat any operation arbitrarily, often until it cannot do it anymore. If you move from one object to the left, you have to stop at the end of the row.

Now, all we are allowed to do with this program is to ask it to try and reach some position, the *desired position* that we specify for it. One works this program, therefore, by giving it a new board position and then presenting it with some desired position; the program has to search among sequences of operations to try and reach the desired position. What we want as a whole is that the program learn to count the number of circular blocks that we put before it on the board.

> Ask the program to try and reach the desired position.
>
> Show the program the desired position for several starting points.
>
> On success, the program can accept the procedure as another primitive operation.

Here, for example, the desired position was counting—I wanted it to say six in the counter and have the box to the right and the finger all the way to the right, meaning it has gone to the left and counted one by one, incrementing the counter each time. If I present lots of different board positions and desired positions in this way, the program never finds out how to count. I have to give it easier tasks first.

Well, what easier task is that? Some of you who have played with this program know that one easy thing derives from what you would do with a child. If you saw a child start counting from the middle finger of a hand, you would say something like, "Move the pointer all the way to the left; that is, start at the thumb."

This task can be done easily enough by merely saying, "Repeat left." The move is just two operations. This sequence can be found by an easy search, merely dozens of trials, and then it always works for this task. After that task is learned, here is the important point: Change the initialize pointer into a primitive operation of its own, so that it enters the list of operations and is usable as a unit.

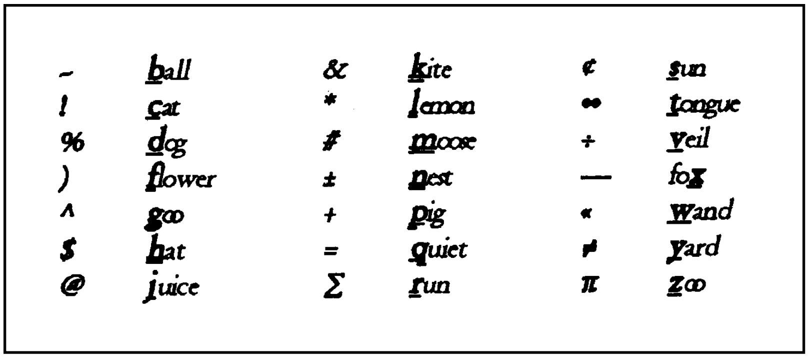
Another precursor task is to initialize the counter, that is, to set it to zero. Again, this task can be simply "repeat decrement." We can call this operation *zero counter.*

Another task to learn is usually the last one a child learns: You go from one finger to the next, go from one number to the next, counting. We now have three more primitive operations, making a total of 8:

Move right
Move left
Increment
Decrement
Initialize pointer
Set counter to zero
Move right and increment
Repeat

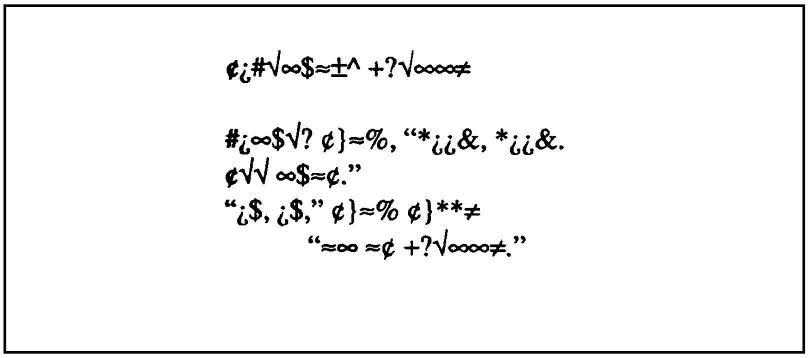
Suddenly, counting is easy. All we have to do is initialize pointer, set zero counter, and then repeat move right and increment, and the program counts! This program is not terribly deep, but it does learn without being instructed, without being programmed. It doesn't even look at examples or cases, and it is primitive. Learning, of course, is much more than what this program shows. Most of the people in machine learning are doing much higher, more complicated things.

There are always several ways of learning something. Marvin (Minsky 1986), in his wonderful and seminal book *Society of Mind,* stresses the importance of learning to learn. We learn for lots of reasons, and I touch on this point soon. However, even straightforward learning by people, the kind they do without teachers, can show us many things that are not echoed in our machine-learning programs.



~	ball	&	kite	¢	sun
!	cat	*	lemon	••	tongue
%	dog	#	moose	÷	veil
)	flower	±	nest	—	fox
^	goo	+	pig	«	wand
$	hat	=	quiet	≠	yard
@	juice	Σ	run	π	zoo

Figure 8. Alphabet and Text with Symbol Substitutions. From Adams (1990). Copyright 1990, The MIT Press. Reprinted with permission.



¢¿#√∞$≈±^ +?√∞∞∞≠

#¿∞$√? ¢}≈%, "*¿¿&, *¿¿&.
¢√√ ∞$≈¢."
"¿$, ¿$," ¢}≈% ¢}**≠
"≈∞ ≈¢ +?√∞∞∞≠."

Figure 9. Simple Text Using Substitute Characters. From Adams (1990). Copyright 1990, The MIT Press. Reprinted with permission.

Marilyn Adams (1990) is a colleague of mine at BBN, a cognitive scientist, and a couple of years ago she published a book called *Beginning to Read.*

She stresses the enormous importance of not just knowing how to identify a character but also being able to do so instantaneously—well, nearly instantaneously. Let me show you. Figure 8 shows an alphabet that uses familiar characters, except that the characters are not letters. Figure 9 shows a simple text based on these characters. The text in figure 9 translates to

Something pretty

Mother said "Look, Look.
See this."
"Oh, oh," said Sally.
"It is pretty."

Imagine trying to read if you have to think about each character. Now, I put it to you that many kids have not been read to, as yours have been, have not discussed the letters, have not said the noises of A, B, C, D for years and years so that they are familiar with and know the letters inherently; this is true of some of the kids in our society. When these kids get to school in first grade and are asked to read, this alphabet is what the ABCs look like to them. Now, in fact, this example is a primitive text of the most awful kind

No, learning to read involves knowing the characters instantaneously—more than that too! Many of you know my eldest son Mallory (who is here today); he was a smart kid, and before he was four, he could read nearly a hundred different words. However, it took him another eight months before he could read a three-word sentence.

More simply put, merely searching the state space for a solution is not enough when dealing with a capability to be learned. The learning has to go deep enough so that the search doesn't have to be done at all or, certainly, not consciously. The point is that after we sort of learn something, we usually have to learn it well, we have to tune the pieces of it, we have to think about when and where it works.

Here are some other considerations: What is the context? How efficiently is the task done? Keep track of circumstances—they change! When does it work? Where does it work? What are we trying to do? Each one of these points is extraordinarily important.

Context: Different things we learn have different effects in different contexts. When I ask a friend, a physicist, why the sky is blue, he talks to me about the behavior of the molecules and the different wavelengths from the sun. When a small child asks why the sky is blue, she doesn't want a lecture on physics. She wants to be told about the nature of the sky and the clouds, rain and where it comes from, light coming from the sun, and rainbows that sometimes appear in the sky. Context really makes all these differences. We have to learn how to make them too. Context in local domains, like reading, has to be learned.

Efficiency: Whenever we learn a skill, we have to perfect it. For example, a child learning to ride a bicycle—once the child has learned to ride the bicycle, she has to learn to ride it well without thinking. When one teaches an older child to drive a car, the child is spending all her energy, all his/her attention, on the driving of the car. None of us do it now. We all drive cars and carry on conversations at the same time without thinking about it. That's much the point of learning.

We have few models in machine learning of this kind of continuing to learn. When does it work; where does it work; and to me perhaps most important, what are we trying to do while we learn? We are never trying to do just one thing. We are always trying to do lots and lots of things, both in teaching and in learning. We have all our purposes to satisfy, and our purposes grow and develop as our skills do.

Well, what good is machine learning? Where can we apply it? I want to give a couple of examples here before I wind up.

> **Where can we apply machine learning?**
>
> Experts know more than facts—they know when and how to change the rules; let us find out these things too!
>
> Expert systems should be built for change and, perhaps, for changes in the rules for change.

Expert systems have been around for 20 years, as I remarked. Brilliantly useful they are, but they miss some of the aspects of some of the kinds of learning that we do. Learning is not just a set of assertions. Experts know more than facts. They also know when to change the rules and how to change them or what to change them to or where to get a new rule or when to expunge an old one. Let us find out these things too. Let us ask the experts how they do things and why they do things as well as what they do. Expert systems should be built for change and, perhaps, for changes in the rules for change as well.

> **Where can we apply machine learning?**
>
> We all know that software is more updating, revising, and modifying than rigid design.
>
> Software systems must be built for change; our dream of a perfect, consistent, provably correct set of specifications will always be a nightmare—and impossible too.
>
> We must therefore begin to describe change, to write our software so that (1) changes are easy to make, (2) their effects are easy to measure and compare, and (3) the local changes contribute to overall improvements in the software.

A similar case in software: In all our software systems, software technology is presented as a rigid discipline in which you have to follow the certain ways of doing it. The dream is that somehow if our software systems could be produced with a perfect set of specifications, if only the specifications were truly perfect, wow!, could we show them how to get a rigid, verifiable, provably correct program! Well, of course, that is nonsense. We all know that software is more updating, revising, modifying, rewriting than it is perfecting any kind of rigid design. Software systems must be built for change. It must be easy to change software systems. Our dream of a perfectly consistent, provably correct set of specifications will always be a nightmare—an impossible one too. We must therefore begin to describe change in software; write our software so that changes are easy to make; write our software so that the effects of changes are easy to measure and compare; and write our software so that the small local changes we make in a module, if we're building modularly, can be measured and evaluated. Then perhaps, local changes in one module measurably contribute to an overall improvement in the software.

> **Where can we apply machine learning?**
>
> "For systems of the future, we need to think in terms of shifting the burden of evolution from programmers to the systems themselves.… [we need to] explore what it might mean to build systems that can take some responsibility for their own evolution." (Huff and Selfridge 1990)

Of course, this shift is what we all want in the long run.

A few summary conclusions here:

> The essence of AI is learning and adapting.
>
> Learning is complicated; it deals with change and changing.
>
> Learning has to make a difference.
>
> Learning is longitudinal; it is not a single act.
>
> Learning is multitudinal; there are many ways to go.
>
> Learning is parallel; there are many things to learn at once.
>
> Learning is manifold; you always learn many things at once.
>
> Learning one thing makes a hundred others easier.
>
> To learn many easy things is better and easier than learning one thing that is hard to learn.

I think of all of us here as practicing all those precepts and beginning the science of what the mind is in its mindness, not psychology, not psychiatry, but in working by simulating the processes of reasoning and feeling. I think of us in our work, and I look out at the sea of urgent, eager, ambitious people, and I'm reminded of Henry the V's speech just before the battle of Agincourt:

> This day is called the feast of Crispian:
> He that outlives this day, and comes safe home,
> Will stand a tip-toe when this day is named,
> And rouse him at the name of Crispian.
> He that shall live this day, and see old age,
> Will yearly on the vigil feast his neighbours,
> And say, 'To-morrow is Saint Crispian:'
> …
> But he'll remember with advantages
> What feats he did that day: …
> This story shall the good man teach his son;
> And Crispin Crispian shall ne'er go by
> From this day to the ending of the world,
> But we in it shall be remembered;
> We few, we happy few.…
>
> *—W. Shakespeare,* King Henry V*, IV, iii*

The essence of AI is learning and adapting…

In the days ahead, so will our children and grandchildren, the graduate students of all the tomorrows there will be, celebrate our research and our publishing and our conferences. They might laugh at our hyperbole, but they will respect the questions we asked; so, they might say:

> "My grandmother was one of them; she went to AAAI in San Jose."
>
> "My uncle presented a paper at IJCAI in Australia or Kiev or France."
>
> "I have a cousin who told me that she wrote a paper with Nils Nilsson."
>
> "My father said he knew Jim Slagle and Danny Bobrow and Wendy Lehnert."
>
> "He worked with Bob Lawler."
>
> "She did her doctorate under Ryszard at Illinois."

So bold you are—you thinkers, you hackers, you wonderers with formalisms and heuristics with experiments, with axioms, trying all the wonderful things. I want to tell you this, that we have a tiger by the tail in this, our field.

> Tyger! Tyger! burning bright
> In the forests of the night,
> What immortal hand or eye
> Could frame thy fearful symmetry?
>
> —*William Blake,* Songs of Experience*,* *1794*

Who better than all of us?

Notes

[1] He also wrote *Artificial Experts: Social Knowledge and Intelligent Machines* (MIT Press, 1991).

References

Adams, M. 1990. *Beginning to Read*. Cambridge, Mass.: MIT Press.

Chapuis, A., and Droz, E. 1958. *Automata: A Historical and Technological Study.* London: Batsford.

Conant, J. 1951. *Science and Common Sense*. New Haven, Conn.: Yale University Press.

Epstein, R. 1992. Can Machines Think? The Quest for the Thinking Computer. *AI Magazine* 13(2): 80–95.

Huff, K., and Selfridge, O. G. 1990. Evolution in Future Intelligent Information Systems. In Proceedings of the International Workshop on the Development of Intelligent Information Systems.

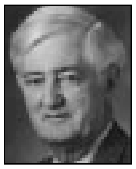
McCulloch, W. S., and Pitts, W. 1943. A Logical Calculus of the Ideas Immanent in Nervous Activity. *Bulletin of Mathematics and Biophysics* 5:115.

Minsky, M. 1986. *The Society of Mind*. New York: Simon and Schuster.

Pitts, W. H., and McCulloch, W. S. 1947. How We Know Universals. The Perception of Auditory and Visual Forms. *Bulletin of Mathematics and Biophysics* 9:127.

Weizenbaum, J. 1976. *Computer Power and Human Reason*. San Francisco: Freeman.

Wiener, N. 1948. *Cybernetics*. Cambridge, Mass.: MIT Press.



Oliver Selfridge studied at MIT under Norbert Wiener an eon ago. He has worked at Lincoln Laboratory, MIT Cambridge Project, and Bolt Beranek and Newman. He moved to GTE Laboratories in Waltham, Massachusetts, in the early 1980s, where Bud Frawley and he helped to start a considerable effort in machine learning, probably the largest in the industrial world. He was a Senior Staff Scientist in the Computer and Intelligent Systems Laboratory until early 1993.

Selfridge has consulted extensively for the U.S. government and has served on panels for the Department of Defense, the Intelligence Community, and the National Institutes of Health. He belongs to many professional societies in his fields of interest. He is a Fellow of the American Association for the Advancement of Science, and an AAAI Fellow.