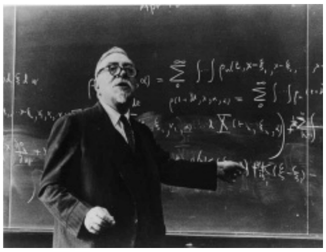
13 Artificial intelligence and neural networks

> It is not my aim to shock you – if indeed that were possible in an age of nuclear fission and prospective interplanetary travel. But the simplest way I can summarize the situation is to say that there are now in the world machines that think, that learn and that create. Moreover, their ability to do these things is going to increase rapidly until – in a visible future – the range of problems they can handle will be coextensive with the range to which the human mind has been applied.
>
> Herbert Simon and Allen Newell[1]

Cybernetics and the Turing Test

One of the major figures at MIT before World War II was the mathematician Norbert Wiener (B.13.1). In 1918, Wiener had worked at the U.S. Army's Aberdeen Proving Ground, where the army tested weapons. Wiener calculated artillery trajectories by hand, the same problem that led to the construction of the ENIAC nearly thirty years later. After World War II, Wiener used to hold a series of "supper seminars" at MIT, where scientists and engineers from a variety of fields would gather to eat dinner and discuss scientific questions. J. C. R. Licklider usually attended. At some of these seminars, Wiener put forward his vision of the future, arguing that the technologies of the twentieth century could respond to their environment and modify their actions:

> The machines of which we are now speaking are not the dream of the sensationalist nor the hope of some future time. They already exist as thermostats, automatic gyrocompass ship-steering systems, self-propelled missiles – especially such as seek their target – anti-aircraft fire-control systems, automatically controlled oil-cracking stills, ultra-rapid computing machines, and the like.…[2]



B.13.1. Norbert Wiener's (1894–1964) name is mainly associated with the term *cybernetics*. Cybernetics is an interdisciplinary theory describing how complex systems regulate themselves using feedback mechanisms. Wiener was only eighteen when he earned his PhD degree in mathematics from Harvard University. During World War II, he worked on automatic control of antiaircraft guns.

All these applications rely on feedback for their ability to learn and adapt. To see how such environmental feedback works, consider a simple thermostat. A bimetallic thermostat has a strip made of two metals fastened together that expand and contract at different rates when the temperature rises and falls. As a result, the thermostat bends when cold and straightens out when warmed (Fig. 13.1). When the temperature drops low enough, the thermostat bends far enough to close an electrical circuit that causes the heating to come on. When

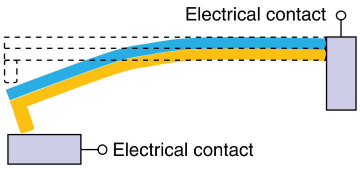


Fig. 13.1. Bimetallic thermostat made from iron (blue) and copper (orange). In cold, the copper contracts more and it bends the bimetal strip downward.

the temperature is too hot, the strip straightens, the circuit is broken, and the heating goes off. Nowadays, sensitive temperature sensors have replaced most metal thermostats, but the principle of how they control the heating system using feedback from the environment remains the same.

Wiener argued that although the physical sciences had been the dominant sciences of the past, the future would be more concerned with communication and control, and he believed that the computer would play a major role in such a future. He called his new science *cybernetics* from the Greek word *kybernetes*, meaning *steersman*. Wiener used the name to refer to the control of complex systems, but the prefix *cyber-* has acquired a variety of computer-related meanings. For example, we now talk about *cyberspace*, meaning the online world of computer networks, and *cyberwarfare*, for attacks on an enemy's information systems.

Scientists at a neurophysiology meeting in New York in 1942 took the first steps toward defining the field of artificial intelligence (AI). Wiener, with his colleagues Julian Bigelow and Arturo Rosenblueth, argued that an animal's nervous system could be thought of in engineering terms as a complicated network of *neurons*, the cells in the brain that process information, with feedback loops. They suggested we can think of computing systems in biological terms in the same way. It was through feedback, they concluded, that an engineering system could have a "definite purpose." This discussion marked the beginning of the fields of AI and *cognitive science* – the study of thinking, learning, and intelligence – although these terms were not introduced until more than a decade later. Cognitive science is now seen as bringing together computer modeling, neurophysiology, and psychology to try to understand how the human mind works.

Wiener and von Neumann were not the only ones thinking about the possibility of AI. In 1941, during World War II, Alan Turing had been exploring ideas about what he called "machine intelligence." Turing had helped design the *bombe*, a mechanical device used in the British code-breaking center at Bletchley Park to decrypt secret messages generated by the German Enigma machine. The bombe had demonstrated the value of performing "guided searches" to save time by reducing a large range of possible solutions to a manageable number. Turing and his fellow code breaker Donald Michie (B.13.2) had many discussions about how similar ideas could be used to create a computer chess program. In 1950, in his famous paper "Computing Machinery and Intelligence," he introduces the idea of what is now known as the *Turing Test*. In the Turing Test, if a human being cannot consistently tell whether questions are being answered by a computer or by another human being, then the computer has passed the test. In his paper, Turing considered the question "Can machines think?" He proposed replacing this question by another, more practical question based on what he called the *imitation game*. The essence of the game is that there are three people in different rooms – a man A, a woman B, and an interrogator C. The three people can communicate only by sending typewritten messages. The object of the game is for the interrogator C to determine whether A or B is the woman by asking questions of each of them. Turing now asks the question:


B.13.2. Donald Michie (1923–2007) worked in the British code-breaking center at Bletchley Park during World War II. He was one of the pioneers of AI in the U.K. computer science research community.

> "What will happen when a machine takes the part of A in this game?" Will the interrogator decide wrongly as often when the game is played like this

> as he does when the game is played between a man and a woman? These questions replace our original, "Can machines think?"[3]

The Turing Test is often taken as an operational definition of intelligence. In its modern form, it reads, "A computer passes the test if a human interrogator, after posing some written questions, cannot tell whether the written responses come from a person or from a computer."[4] To pass the test, computers will need to complete the following tasks: to understand natural language; reason about the information expressed by words and sentences; and learn from experience. In 1950, Turing was cautiously optimistic:

> I believe that in about fifty years' time it will be possible, to programme computers, with a storage capacity of about 10^9, to make them play the imitation game so well that an average interrogator will not have more than 70 per cent chance of making the right identification after five minutes of questioning. The original question, "Can machines think?" I believe to be too meaningless to deserve discussion. Nevertheless I believe that at the end of the century the use of words and general educated opinion will have altered so much that one will be able to speak of machines thinking without expecting to be contradicted.[5]

Turing also gave a famous example of the type of conversation that he imagined it would be possible to have with a "sonnet-writing" machine in the future. It would be difficult to learn whether the machine has really understood something or whether, as he says, it has just "learnt it parrot fashion":[6]

> **Interrogator:** In the first line of your sonnet which reads "Shall I compare thee to a summer's day", would not "a spring day" do as well or better?
> **Witness:** It wouldn't scan.
> **Interrogator:** How about "a winter's day"? That would scan all right.
> **Witness:** Yes, but nobody wants to be compared to a winter's day.
> **Interrogator:** Would you say that Mr. Pickwick reminded you of Christmas?
> **Witness:** In a way.
> **Interrogator:** Yet Christmas is a winter's day, and I do not think Mr. Pickwick would mind the comparison.
> **Witness:** I don't think you are serious. By a winter's day one means a typical winter's day, rather than a special one like Christmas.[7]

If a computer were capable of such a sophisticated dialog, requiring knowledge both of literature and Mr. Pickwick as well as of the significance of Christmas, it would be hard to make a distinction between "real" and artificial thinking. At present, we still seem to be far from this goal. ELIZA, one of the earliest "chatbot" programs, simulated an interview with psychotherapist and could be superficially very convincing (see the short summary of ELIZA at the end of this chapter, for an example). Its author, Joseph Weizenbaum, chose the psychotherapy model precisely because it would not require a significant knowledge base. ELIZA imitated client-centered therapy, a form of psychotherapy that tries to increase the patient's insight and self-understanding by restating the patient's feelings and thoughts.

Fig. 13.2. The Loebner Prize for $100,000 was established in 1990 for the AI system that first passes the Turing Test.

Since 1991, a New York businessman, Hugh Loebner (Fig. 13.2), has sponsored an annual Turing Test competition; and in 2012, the centenary of Turing's birth, the contest was held at Bletchley Park. In more than twenty years of competition, no chatbot program has come close to deceiving a sophisticated judge.

An everyday demonstration of a computer's inability to pass something like a Turing Test is a reverse version based on recognizing distorted letter shapes. To pass this reverse Turing Test, a computer would need highly developed perceptual abilities that are currently beyond the capability of the most advanced computer vision algorithms. These puzzles were called CAPTCHAs (Fig. 13.3) by Luis von Ahn (B.13.3), an acronym standing for "Completely Automated Public Turing test to tell Computers and Humans Apart." Humans can easily recognize the distorted letters, so CAPTCHAs enable websites to distinguish between human users and automated "robot" programs trying to access the site. It is estimated that more than two hundred million CAPTCHAs are solved every day.

Fig. 13.3. CAPTCHAs can be easily read by a human, but not by a computer. This is one commonly used mechanism to distinguish between human visitors to websites and robotic crawlers.

From logic theorist to DENDRAL

The term *artificial intelligence* was coined by John McCarthy (B.13.4) in a workshop at Dartmouth College in New Hampshire in 1956. McCarthy and fellow AI pioneers Marvin Minsky, Claude Shannon, and Nathaniel Rochester wrote a proposal for the workshop stating:

> The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it. An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves. We think that significant advance can be made in one or more of these problems if a carefully selected group of scientists work on it together for a summer.[8]

B.13.3. Luis von Ahn is an associate professor at Carnegie Mellon University in Pittsburgh. He is perhaps best known for his invention of CAPTCHAs, those annoying distorted characters that only humans, not computers, are supposed to be able to read.

The highlight of the Dartmouth workshop was a reasoning program developed by Allen Newell and Herbert Simon (B.13.5) from Carnegie Tech, now Carnegie Mellon University. Their "Logic Theorist" program was able to prove theorems using simple symbolic logic. It represented each problem as a tree structure with the root being the initial *hypothesis*, a tentative explanation that could be tested by further investigation. Each branch of the tree was a deduction based on the rules of logic. To prevent the tree from growing uncontrollably, Newell and Simon needed a way to remove unwanted branches. To do so, they introduced *heuristics*, rules of thumb that enabled the program to select only those branches of the overall search tree that seemed most promising. They said, "Logic Theorist's success does not depend on the 'brute force' use of the computer's speed, but on the use of heuristic processes like those employed by humans."[9]

In their monumental work *Principia Mathematica*, Alfred Whitehead and Bertrand Russell had attempted to systematize the principles of mathematical

B.13.4. A famous photograph of four of the founding fathers of AI. From left to right they are Claude Shannon, John McCarthy, Ed Fredkin, and Joseph Weizenbaum.

logic. Newell and Simon attempted to use Logic Theorist to reproduce the proofs of fifty-two theorems in Whitehead and Russell's book:

> Let us consider more specifically whether we should regard the Logic Theorist as creative. When the Logic Theorist is presented with a purported theorem in elementary symbolic logic, it attempts to find a proof. In the problems we have actually posed it, which were theorems drawn from Chapter 2 of Whitehead and Russell's *Principia Mathematica*, it has found the proof about three times out of four.[10]

On being told that the program had found a shorter proof for one of their theorems, Bertrand Russell was reportedly delighted. Newell, Shaw, and Simon attempted – unsuccessfully – to publish their result in the *Journal of Symbolic Logic* with the Logic Theorist program listed as a co-author.

McCarthy moved from Dartmouth to MIT in 1958 and in the same year made three major contributions to computer science. One was the suggestion for time-sharing systems, as we have seen in Chapter 3. A second was his invention of the Lisp programming language, an acronym derived from LISt Processing. Lisp was the dominant language for AI applications for the next thirty years. McCarthy's third major contribution was to lay out a research agenda for the AI community in a paper called "Programs with Common Sense." In the paper, McCarthy described a hypothetical AI program he called Advice Taker. Like Newell and Simon's Logic Theorist and their ambitious follow-up, the General Problem Solver, Advice Taker would not only use logic and *symbol manipulation*, the manipulation of characters rather than numbers, but also incorporate general knowledge about the world to assist in its deductive process:

> The main advantages we expect the advice taker to have is that its behavior will be improvable merely by making statements to it, telling it about its symbolic

B.13.5. Herbert Simon (1916–2001) and Allen Newell (1927–92) were pioneers in the field of AI. They were awarded the Turing Award in 1975 for their work in AI, and Simon also won the Nobel Prize in economics in 1978 for his theory of decision making.

environment and what is wanted from it. To make these statements will require little if any knowledge of the program or the previous knowledge of the advice taker. One will be able to assume that the advice taker will have available to it a fairly wide class of immediate logical consequences of anything it is told and its previous knowledge. This property is expected to have much in common with what makes us describe certain humans as having common sense. We shall therefore say that a program has common sense if it automatically deduces for itself a sufficiently wide class of immediate consequences of anything it is told and what it already knows.[11]

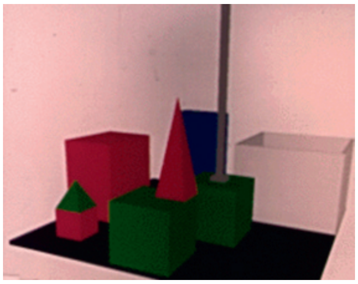


Fig. 13.4. The SHRDLU "blocks world" program was written by Terry Winograd at MIT. The program could understand and execute instructions given in natural language to move different types of blocks around in a virtual box.

Advice Taker embodied the idea that an AI system needed an explicit representation of the world and the ability to manipulate this knowledge with logical deductive processes. This vision set the agenda for AI research for the next few decades.

Marvin Minsky (B.13.6) arrived at MIT at the same time as McCarthy, and together they set up the MIT Artificial Intelligence Laboratory. Their research collaboration lasted only a few years before their approaches to AI diverged. Minsky concentrated on just getting systems to do interesting things – "scruffy AI." Minsky's students focused on solving problems in very limited *domains*, application areas not requiring a broad general knowledge. Successful examples included the domains of integral calculus, geometry, and algebra as well as a famous series of problems in the "blocks world" (Fig. 13.4), a simplified world consisting of some toy blocks sitting on a table. SHRDLU, developed by Terry Winograd, was a computer program that could understand instructions and carry on conversations about the blocks world. Unlike Minsky, McCarthy emphasized knowledge representations and reasoning using formal logic. In 1963 McCarthy left MIT to start the Stanford Artificial Intelligence Laboratory.

B.13.6. Marvin Minsky is one of the original pioneers of AI. He is also credited with the invention of the first head-mounted graphical display in 1963. He also acted as a scientific adviser for Stanley Kubrick's film *2001: A Space Odyssey*. Science fiction writer Isaac Asimov described Minsky as one of only two people he would admit were smarter than he was. The other was cosmologist and astronomer Carl Sagan.

As computers became more powerful and as computer memories became larger, there was a movement for researchers to explicitly build "knowledge" into AI applications and to develop *expert systems*, computer programs that imitate the decision making of a human expert. One of the pioneers of the expert-systems approach to AI was Ed Feigenbaum (B.13.7) at Stanford University. In 1969, with Bruce Buchanan and Joshua Lederberg, a geneticist and recipient of the Nobel Prize, Feigenbaum developed the DENDRAL program that attempted to capture the expert knowledge of chemists and to apply that knowledge by employing a set of rules. The name DENDRAL was an abbreviation of *dendritic algorithm*, *dendritic* referring to the branching fibers of neurons that pick up nerve impulses. The problem that DENDRAL attempted to solve was that of determining the molecular structure of a substance using data provided by a mass spectrometer, an instrument that separates particles of different masses in a similar way to light spread out into different colors by a prism. To identify the precise structure of a compound, a chemist must deduce its chemical elements from the set of masses of fragments of the compound. For large molecules, this generates a huge number of possible structures. To make the problem manageable, expert chemists use their own rules of thumb – in other words, *heuristics* – to recognize well-known substructures and thereby reduce the number of possibilities for the overall structure of the compound. DENDRAL combined a knowledge base, written in the form of rules, with a reasoning engine written in Lisp. DENDRAL was

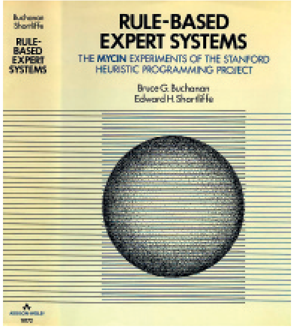


Fig. 13.5. The MYCIN project was an expert system designed to diagnose and treat blood infections. It was developed at Stanford by Edward Shortliffe with Bruce Buchanan and Ed Feigenbaum.

thus the first successful "knowledge-intensive" AI system because it automated the decision-making and problem-solving processes of experts in a field.

Feigenbaum looked at other domains where this approach could be applied. With Bruce Buchanan and Edward Shortliffe, he developed the MYCIN expert system to diagnose and treat blood infections (Fig. 13.5). Using about 450 rules developed from interviews with experts, MYCIN performed better than many doctors. The success of DENDRAL, MYCIN, and other expert systems led to an overenthusiastic rush to produce commercial systems in the 1970s and 1980s. Although the high hopes of the pioneers were not fully realized, knowledge-based expert systems are still used for applications ranging from straightforward help-desk and technical support to manufacturing and robotics. For narrow, well-defined problems, expert systems can be successful. However, a major limitation of this rule-based approach to knowledge is that these systems do not generalize well to larger, broader problems. In addition, the development and capture of the knowledge rules are very labor intensive and usually very specific to the case at hand. Because almost nothing in real life is simply true or false in the way that abstract logic requires, for any commonsense rule about the world there must also be a large number of exceptions.

The creation of taxonomies and classifications dates back to the 300s B.C., when Aristotle wrote his *Organon*, a collection of his works on logic. This included a section on categories that we would now call a type of *ontology*, the study of what kinds of things exist. It was the Swedish biologist Carolus Linnaeus who invented our present system of biological classification in the 1700s (Fig. 13.6). Computer scientists have borrowed the word *ontology* from the philosophers to describe a structural framework for organizing knowledge. An ontology specifies a set of concepts within a domain that a computer can use to reason about objects in the domain and about the relationships between them. AI researchers have long believed that useful ontologies are essential for effective AI systems. One response to this need is therefore to expand the knowledge base of the computer by producing a comprehensive vocabulary of all of the important concepts in a given domain, including the objects in the domain and the properties, relations, and functions needed to define the objects and specify their actions.

B.13.7. Ed Feigenbaum received the Turing Award for his work in expert systems in 1994. He is often known as the "father of expert systems." He also served as chief scientific adviser for the U.S. Air Force and received their Exceptional Civilian Service Award in 1997.

One of the most ambitious ontology projects is the Cyc project, started by Douglas Lenat in 1984. The name Cyc is a shortened form of *encyclopedia*. The project is an attempt to build a knowledge base containing much of the everyday, commonsense knowledge of a human being. Typical pieces of knowledge represented in the database are statements such as "Every tree is a plant" and "Plants die eventually." After more than twenty-five years, Cyc's knowledge base contains more than one million assertions, rules, or commonsense ideas. However, its creators estimate that it will need more than one hundred times that many entries before Cyc can begin to learn for itself from written material.

The DBPedia (Fig. 13.7) project has taken a different approach and uses a method called *crowdsourcing*, soliciting content from a large group of people, to extract structured data from Wikipedia. DBPedia's 2012 release contained an ontology with more than two million concepts together with about one hundred facts per concept. The researchers hope that the Cyc and DBPedia projects will help realize Tim Berners-Lee's vision of the Semantic Web, in which machines

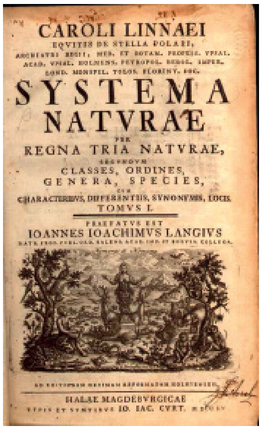


Fig. 13.6. Swedish botanist, physician, and zoologist, Carolus Linnaeus, published his classification of living things in 1735. This was an early attempt at constructing a knowledge representation for species of animals and plants.

can process and understand the actual data on the web. This vision will become reality when web search engines have access to machine-readable knowledge that enables them to reason and make "intelligent" decisions.

The early optimism of the Dartmouth workshop attendees – Allen Newell, Herbert Simon, John MacCarthy, and Marvin Minsky – was typified by the quotation that introduces this chapter. A more realistic perspective has now replaced this optimism. As the computer scientist David McAllester said in a 1998 paper on machine learning:

> In the early period of AI it seemed plausible that new forms of symbolic computation … made much of classical theory obsolete. This led to a form of isolationism in which AI became largely separated from the rest of computer science. This isolationism is currently being abandoned. There is a recognition that machine learning should not be isolated from information theory, that uncertain reasoning should not be isolated from stochastic modeling, that search should not be isolated from classical optimization and control, and that automated reasoning should not be isolated from formal methods and static analysis.[12]

Computer chess and Deep Blue

In the early days of computing, most people thought that computers were just machines that were capable of carrying out complex arithmetic calculations very rapidly. A few of the early pioneers, like Turing and Shannon, speculated that computers one day would be able to play chess, a task that had always up until then been considered to require human intelligence. Donald Michie summarized the interest of chess for AI as follows:

> Computer chess has been described as the "Drosophila melanogaster" of machine intelligence. Just as Thomas Hunt Morgan and his colleagues were able to exploit the special limitations and conveniences of the "Drosophila" fruit fly to develop a methodology of genetic mapping, so the game of chess holds special interest for the study of the representation of human knowledge in machines. Its chief advantages are: (1) chess constitutes a fully defined and well-formalized domain; (2) the game challenges the highest levels of human intellectual capacity; (3) the challenge extends over the full range of cognitive functions such as logical calculation, rote learning, concept-formation, analogical thinking, imagination, deductive and inductive reasoning; (4) a massive and detailed corpus of chess knowledge has accumulated over the centuries in the form of chess instructional works and commentaries; (5) a generally accepted numerical scale of performance is available in the form of the U.S. Chess Federation and International ELO rating system.[13]

Fig. 13.7. The DBpedia project is trying to structure the content of Wikipedia by using an army of volunteers – 'crowdsourcing' – to perform the work required.

Claude Shannon's 1950 article "Programming a Computer for Playing Chess" that spelled out a complete set of ideas for computer chess, including how to represent board positions, searching the "game tree" of possible moves, and using procedures called *evaluation functions*, by which players use knowledge of the game to judge each possible move and choose the best ones. In game theory, a *game tree* is a graphical representation of a sequential game consisting of

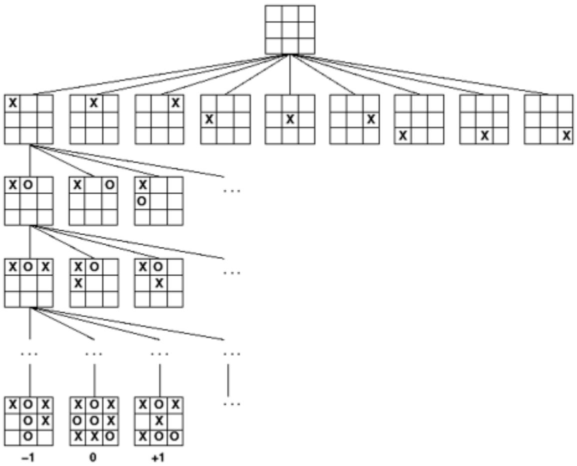


Fig. 13.8. A (partial) game tree for tic-tac-toe or noughts and crosses.

nodes, the points at which players can take actions, and *branches*, which represent the possible moves at each node. In 1951, Dietrich Prinz wrote the first chess program able to solve simple endgame problems. Prinz worked for Ferranti, a British computer company marketing the Manchester Mark I machine, the first commercially available general-purpose computer. Five years later, Stan Ulam and a group at Los Alamos National Laboratory wrote a program that could play a full game of chess, but only on a reduced board of 6 × 6 squares and no bishops. It was not until 1957 that IBM programmer Alex Bernstein wrote the first complete chess program for the IBM 704 computer, one of the last vacuum-tube computers. The chess program took about eight minutes to make a move after completing a search that could look about two moves ahead. Before we examine how a chess program works, let us look at a simpler problem, a computer program for the game called tic-tac-toe or noughts and crosses.

We shall label the two players MAX, who makes the X moves, and MIN, who makes the O moves. The total game tree consists of all the legal moves from all the possible configurations of Xs and Os. MAX moves first, and from the top node of the tree, MAX can make nine possible moves – a branching factor of nine (see Fig. 13.8). Then it is MIN's turn to make one of the eight remaining moves. This alternation continues until either a line of Xs or Os is achieved or all spaces on the board are filled. There are $9 \times 8 \times 7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1$ nodes, or 362,880 nodes, in the tree. The game has a simple evaluation function by which a player chooses the best move: +1 for a win for MAX, ½ for a draw, and 0 for a win for MIN. A computer program can easily evaluate all possible paths and positions leading to the final move of the game.

For the first move in chess, there are twenty possible moves, sixteen with the eight pawns and four with the two knights. A typical game is around forty moves, and for each position there is an average of thirty to thirty-five possible moves to explore. Because the entire chess game tree would contain more than 10^{40} nodes, an exhaustive search strategy looking at all the final positions is not possible. Because we cannot get to the final positions, the evaluation function for chess is also much more complicated. For example, a chess evaluation function usually has a weighted sum of the various factors that are thought to

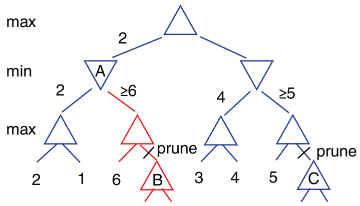


Fig. 13.9. In this example of alpha-beta pruning, the moves of the game are represented by a binary min-max tree. The algorithm traverses the tree starting from the bottom left and chooses the maximum of the first two values. The algorithm then moves to the red branch and finds that the first value is six. Thus what must be passed on to the min node must be more or equal to six. Because we already know that there is a lower value of two at the min node we do not need to evaluate branch B of the red node because the algorithm will never use this part of the tree. We proceed in the same way to eliminate branch C.

influence the value of a position. These include factors such as the power of each piece and its possible mobility, control of the center of the chess board, and the safety of the king. A program therefore needs to find strategies that optimize moves for the player MAX, at the same time assuming that the opponent, MIN, will make an optimum move in response. Such a strategy is provided by the *minimax algorithm*, a procedure that minimizes the risk for a player. Because the number of moves that need to be examined by the minimax algorithm grows at an increasingly rapid rate with the depth of the tree, computer chess programs can only afford to evaluate several moves ahead, not all the way to the final result node. In their 1958 chess program dubbed NSS from their initials, Allen Newell and Herbert Simon from Carnegie Mellon University and Cliff Shaw from the RAND Corporation introduced an optimization technique called *alpha-beta pruning* (Fig. 13.9) for the minimax search algorithm. Alpha-beta pruning decreases the search time by stopping evaluation of a move when at least one possibility has been found that proves the move to be worse than a previously examined move. In this way, several branches of the search tree can be "pruned" and the search time devoted to deeper exploration of more promising branches. In addition to such pruning techniques, modern chess programs also include tables of the standard openings and endgames.

The first computer versus computer chess match featured the Kotok-McCarthy program written by Alan Kotok, John McCarthy, and their colleagues from MIT pitted against the Russian ITEP program written by scientists at the Institute of Theoretical and Experimental Physics in Moscow (Fig. 13.10). Playing by telegraph in 1967, the match ended in a 3 to 1 victory for ITEP. In the same year, MIT's Mac Hack, written by Richard Greenblatt and colleagues, became the first chess program to play in a tournament with humans. Its Elo rating was 1400, well above the novice level of 1000 on the chess rating system developed by the Hungarian-born physicist Árpád Élő. In 1968, the international chess master David Levy made a famous bet with John McCarthy that no computer would beat him at chess in the next ten years, saying:

> Clearly, I shall win my … bet in 1978, and I would still win if the period were to be extended for another ten years. Prompted by the lack of conceptual progress over more than two decades, I am tempted to speculate that a computer program will not gain the title of International Master before the turn of the century and that the idea of an electronic world champion belongs only in the pages of a science fiction book.[14]

Levy played his 1978 match against the Chess 4.7 program, the strongest computer chess program of the time, written by Larry Atkin and David Slate from Northwestern University. Levy won by 4.5 to 1.5 but he said later, "I had proved that my 1968 assessment had been correct, but on the other hand my opponent in this match was very, very much stronger than I had thought possible when I started the bet."[15]

Fig. 13.10. A photograph of the Institute of Theoretical and Experimental Physics in Moscow.

In 1980, the celebrated MIT computer scientist Ed Fredkin offered prizes for successive milestones in computer chess. The smallest prize of $5,000 went to Ken Thompson, inventor of the Unix operating system, and Joe Condon, when their Belle chess program earned a U.S. Master rating in 1983. Belle was the first computer chess system to use custom-designed chips, and it won the

Fig. 13.11. IBM's Deep Blue chess computer first played Kasparov in 1996. On that occasion the world champion managed to beat the machine. Kasparov famously lost the rematch a year later.

world computer chess championship in 1980. The U.S. Department of State temporarily confiscated Belle in 1982 as it was heading to the Soviet Union to participate in a computer chess tournament. The State Department claimed it was a violation of U.S. technology transfer law to ship a high-technology computer to a foreign country. The next prize of $10,000 for the first program to achieve an Elo rating of 2500 was awarded to a computer called Deep Thought in August 1989. Deep Thought was a computer specifically designed to play chess by Feng-hsiung Hsu and his fellow graduate student Murray Campbell at Carnegie Mellon University. IBM then recruited Hsu and Campbell to develop a successor to Deep Thought. The result was Deep Blue, a parallel computer with thirty processors, enhanced by 480 special-purpose chess chips (Fig. 13.11). Deep Blue was capable of evaluating two hundred million positions a second and could typically search six to eight moves ahead, and sometimes more. A team of three chess grand masters provided its opening library, and its endgame database included many six-piece endgames as well as those with five pieces and fewer. In May 1997, world champion Garry Kasparov took on Deep Blue in a six-game match held in New York (Fig. 13.12). The computer won a close match with two wins for Deep Blue, one for Kasparov, and three draws. The $100,000 Fredkin Prize went to Feng-hsiung Hsu, Murray Campbell, and Joseph Hone from IBM. After the match, Kasparov wrote:

> The decisive game of the match was Game 2, which left a scar in my memory … we saw something that went well beyond our wildest expectations of how well a computer would be able to foresee the long-term positional consequences of its decisions. The machine refused to move to a position that had a decisive short-term advantage – showing a very human sense of danger.[16]

Neural networks

Fig. 13.12. The newspapers and other media portrayed the 1997 match between World Chess Champion Garry Kasparov and IBM's Deep Blue computer as a battle between human and machine. The cover of *Newsweek* proclaimed it "The Brain's Last Stand."

In the audience for Norbert Weiner's neurophysiology talk in 1942 was Warren McCulloch (B.13.8), a professor of psychiatry in Chicago. With a precocious eighteen-year-old mathematician called Walter Pitts, McCulloch developed the first model of the brain as an electrical network of interconnected neurons. They argued that their idealized "neural network" model captured the key features of the brain's physiology. Von Neumann was so impressed by this work that, together with Wiener and Howard Aiken from Harvard, he organized a small workshop at Princeton in January 1945 at which McCulloch and Pitts were invited to present their neural network model. Ideas from neural networks were fresh in von Neumann's mind when he wrote his "Draft Report on the EDVAC" later that year – in which he referred to the basic functional units of the computer as "organs" and made comparisons of the functions of these units with the biological functions of neurons.

The importance of the brain in determining human emotions was recognized by Hippocrates, the "Father of Medicine," as long ago as 400 B.C. He said, "Men ought to know that from nothing else but the brain come joys, delights, laughter and sports, and sorrows, griefs, despondency, and lamentations."[17] The human brain has a similar structure to brains of other mammals but is significantly larger in relation to body size compared to most animals. The relative increase in size of the human brain is mainly due to the greater size of

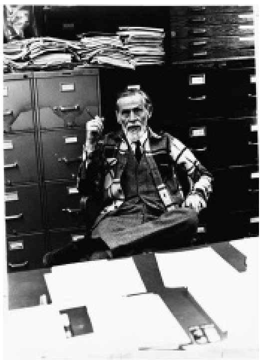
Fig. 13.13. A diagram of the cerebral lobes of a human brain: frontal lobes in pink, parietal lobe in green, and the occipital lobe in blue.

the *cerebral cortex*, a thick layer of neural tissue that covers most of the brain (see Fig. 13.13). The name *cortex* comes from the Latin word for the bark of a tree, but in this case it means the outer layer of an organ. The cortex is deeply folded and ribbed because such folding maximizes the amount of brain surface that can fit into the limited space of the skull. More than two-thirds of the surface area of a human brain is buried in these folds, called *sulci*. The cerebral cortex plays a key role in memory, perception, thought, language, and consciousness.

The nineteenth century brought rapid progress in biological science thanks to the wide use of microscopes. Theodor Schwann and Matthias Schleiden had suggested cell theory, according to which all living organisms are made up of cells, in 1838. But not all scientists were convinced that cell theory applied to brain tissue. As a result, many scientists experimented with different chemical substances for coloring the brain tissue so that individual cells would be made visible. A Spanish physician, Santiago Ramón y Cajal improved on a cell-staining method originally developed by the Italian doctor Camillo Golgi, and used this new technique to investigate the central nervous system of many living creatures. It was Ramón y Cajal's work that first revealed the complexity of biological neural networks. He wrote:

> What beauty is shown in the preparations obtained by the precipitation of silver dichromate deposited exclusively onto the nervous elements! But, on the other hand, what dense forests are revealed, in which it is difficult to discover the terminal endings of its intricate branching.… Given that the adult jungle is impenetrable and indefinable, why not study the young forest, as we would say in its nursery stage.[18]

We now know that neurons consist of a cell body or *soma* with two types of nerve fiber growing from the cell, *dendrites* and *axons*. The cell body contains the genetic information and the molecular machinery required for the functioning of the neuron. The role of the dendrites is to receive electrical or chemical signals from other neurons and provide the input to the cell of the neuron. The axon, usually much longer than the dendrites, carries nerve impulses from the cell body to other neurons. Ramón y Cajal also suggested that these signals always flow in one direction, from the dendrites of the cell to the axon, and that the axon is connected to dendrites of other cells by structures called *synapses* (see Fig. 13.14). The word *synapse* comes from the Greek words *syn*, meaning *together*, and *haptein*, meaning *to clasp*. Golgi and Ramón y Cajal were awarded the 1906 Nobel Prize in physiology or medicine "in recognition of their work on the structure of the nervous system."[19]



B.13.8. Warren McCulloch (1898–1969) was an early pioneer of AI. With Walter Pitts he proposed the first mathematical model of a neural network. John von Neumann was very impressed by the McCulloch-Pitts model and the paper and its physics terminology influenced him as he wrote the EDVAC draft report.

The number of neurons in the brain varies widely from species to species. The human brain is believed to contain more than eighty-five billion neurons, while the brain of a cat has only one billion and a chimpanzee about seven billion neurons. In addition to these vast numbers of neurons, the brain has an even larger number of synapses. Each human neuron has, on average, seven thousand synaptic connections to other neurons. There are many different types of neurons, and we will describe only how a "typical" neuron functions. The incoming signals reaching a neuron from all of its dendrites are collected and processed inside the cell body. Any output signal resulting from this input

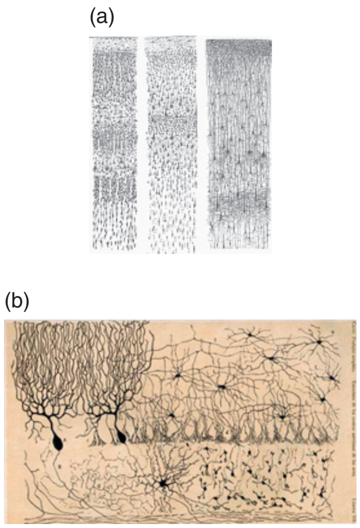


Fig. 13.14. Santiago Ramón y Cajal's drawing of: (a) a Golgi-stained cortex of a six-week-old human infant and (b) cells of the chick cerebellum.

travels down the axon and is passed on to the dendrites of neighboring neurons through the synapses (see Fig. 13.15). A typical neuron operates on a "threshold" or "all-or-none" principle meaning that the input stimulation, represented by the sum of all the incoming signals, must be above a certain threshold for the cell to produce an output signal.

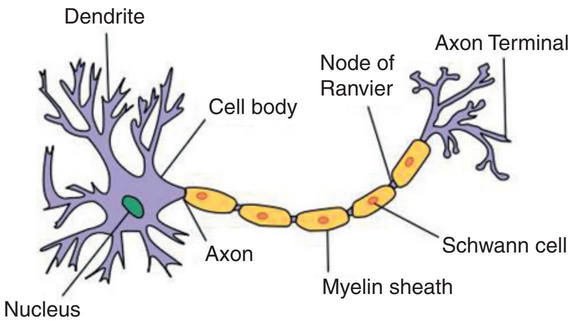
The cerebral cortex consists of up to six horizontal layers of neurons and is about 2.5 millimeters or one-tenth of an inch thick. The neurons in each of these layers connect vertically to neurons in adjacent layers. With these new discoveries about the brain in the first half of the twentieth century, Nobel Prize recipient Charles Sherrington poetically imagined how the workings of the brain would look as it woke up from sleep:

> The great topmost sheet of the mass, that where hardly a light had twinkled or moved, becomes now a sparkling field of rhythmic flashing points with trains of traveling sparks hurrying hither and thither. The brain is waking and with it the mind is returning. It is as if the Milky Way entered upon some cosmic dance. Swiftly the head mass becomes an enchanted loom where millions of flashing shuttles weave a dissolving pattern, always a meaningful pattern though never an abiding one; a shifting harmony of subpatterns.[20]

As we have seen, Wiener, von Neumann, Turing, and other early computer pioneers were fascinated with the possibility of computers performing operations that would normally be classified as requiring intelligence. Warren McCulloch and Walter Pitts had produced a simple mathematical model of a neuron that only "fired" when the combination of its input signal exceeded a certain threshold value (see Fig. 13.16). In their famous 1943 paper "A Logical Calculus of the Ideas Immanent in Nervous Activity," they showed that a network of such neurons could carry out logical functions. They also suggested that, much like a human brain, these artificial neural networks (ANNs) could learn by forming new connections and by modifying the neural thresholds. Alan Turing put forward similar ideas in an unpublished paper on "Intelligent Machinery" in 1948. Turing suggested, "The cortex of an infant is an unorganised machine, which can be organised by suitable interfering training."[21]

The basis of modern ANNs is a mathematical model of the neuron called the *perceptron* introduced by Frank Rosenblatt in 1957. In the original model

Fig. 13.15. Sketch of a biological neural network showing dendrites, axons, and synapses.



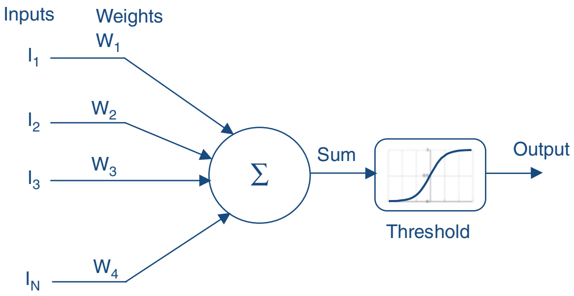


Fig. 13.16. Representation of an artificial neuron with inputs, connection weights, and the output subject to a threshold function.

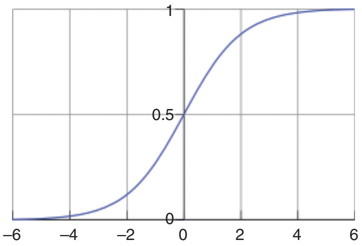


Fig. 13.17. A simple threshold function for an artificial neuron. The strength of the output signal depends on the magnitude of the sum of the input signals.

of McCulloch and Pitts, the input could only be either 0 or 1. In addition, each input "dendrite" had an associated "weight" that was either +1 or –1 to represent inputs that tended either to excite the neuron to fire or to inhibit the neuron from firing, respectively. The model calculated the weighted sum of the inputs – the sum of each input multiplied by its weight – and checked whether this sum was greater or smaller than the threshold value. If the weighted sum was greater than the threshold, the model neuron fired and emitted a 1 on its axon. Otherwise, the output remained 0. Rosenblatt's perceptron model allowed both the inputs to the neurons and the weights to take on any value. In addition, the simple activation threshold was replaced by a smoother *activation function*, a mathematical function used to transform the activation level of the neuron into an output signal, such as the function shown in Figure 13.17. ANNs are just interconnected layers of perceptrons as shown in Figure 13.18.

For numerical calculations, computers are very much faster than humans at performing arithmetic. For tasks involving *pattern recognition* – the automatic identification of figures, shapes, forms, or patterns to recognize faces, speech, handwriting, objects, and so on – even young children are still very much better than the most powerful computers. The hope for ANN research is that by mimicking how our brains learn, these artificial networks can be trained to recognize patterns. The study of ANNs is sometimes called *connectionism*.

The publication of a famous book *Perceptrons* in 1969 by Marvin Minsky and Seymour Papert from MIT dashed early hopes for progress with neural networks. Minsky and Papert showed that a simple two-layer perceptron network was incapable of learning some very simple patterns. While they did not rule out the usefulness of multilayer perceptron networks with what they called "hidden" layers, they pointed out "the lack of effective learning algorithms"[22] for such networks. This situation changed in the 1980s with the discovery of just such an effective learning algorithm. A very influential paper in the journal *Nature* gave the algorithm its name: "Learning Representations by Back-Propagating Errors" by David Rumelhart, Geoffrey Hinton (B.13.9), and Ronald Williams. Let us see how this *back-propagation* algorithm enables neural networks to learn.

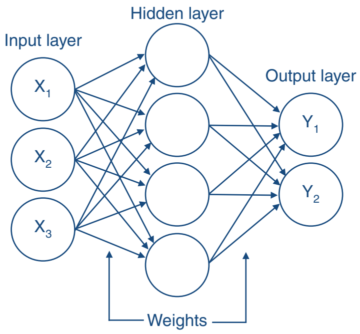


Fig. 13.18. An example of a three-layer ANN, with all connections between layers. The output of the neural network is specified by the connectivity of the neurons, the weights on the connections, the input signals, and the threshold function.

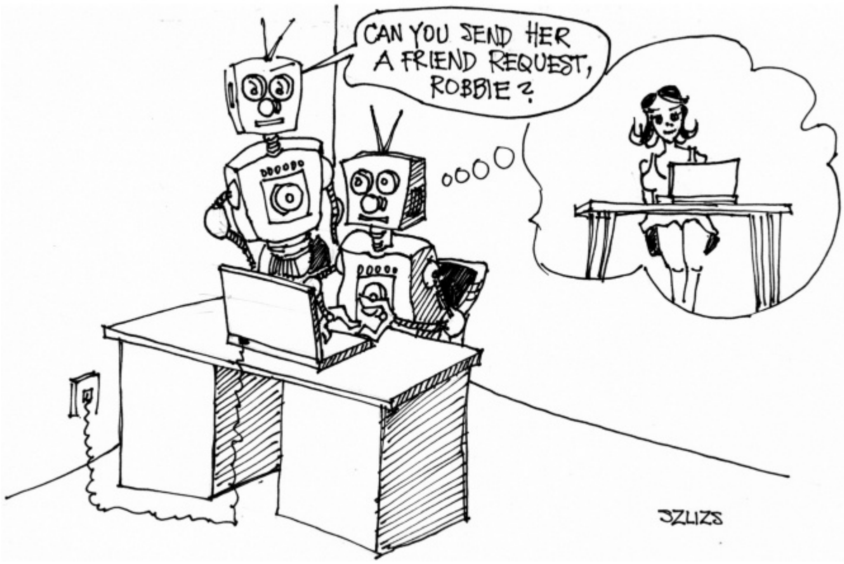
Imagine a simple three-layer neural network: a layer of input neurons, connected to a second hidden layer of neurons, which in turn is linked to a layer of output neurons (Fig. 13.18). Each neuron converts its inputs into a single output, which it transmits to neurons in the next layer. The conversion process has two stages. First, each incoming signal is multiplied by the weight of the connection, and then all these weighted inputs are added together to give a total weighted input. In the second stage, this combined input is passed through an activation function, such as the function of Figure 13.17, to generate the output signal for the neuron. To train the network to perform a particular task, we must set the weights on the connections appropriately. The amount of weight on a connection determines the strength of the influence between the two neurons. The network is trained by using patterns of activity for the input neurons together with the desired pattern of activities for the output neurons. After assigning the initial weights randomly, say to be between –1.0 to +1.0, then, by calculating the weighted input signals and outputs of the neurons in each layer of the network, we can determine the strength of the signals at the output neurons. For each input pattern, we know what pattern we want to see at the output layer, so we can see how closely our model output matches the desired output. We now have to adjust each of the weights so that the network produces a closer approximation to our desired output. We do this by first calculating the error, defined as the square of the difference between the actual and desired outputs. We want to change the weight of each connection to reduce this error by an amount that is proportional to the rate at which the error changes as the weight is altered. We first make such changes for all the neurons in the output layer. We then repeat the calculation to find the sensitivity to the weights connecting each layer, working backward layer by layer from the output to the input. The idea is that each hidden node contributes some fraction of the error at each of the output nodes to which it is connected. This type of network is known as a *feed-forward* network because the signals between the neurons travel in only one direction, from the input nodes, through the hidden nodes, to the output nodes. The learning algorithm to train the neural network is called *back propagation* because the error at the output layer is *propagated* – that is, passed along – backward through the hidden layer of the network (see Fig. 13.18).

In the 1990s, researchers found it hard to train neural networks with more than one hidden layer and two layers of weights. The problem was that the weights on any extra layers could not be adapted to produce significant improvements in learning. However, within the last few years, Geoffrey Hinton and colleagues from the University of Toronto, and researchers Li Deng and Dong Yu from Microsoft Research have shown that much deeper layered networks can not only be trained efficiently but can also deliver significantly improved learning outcomes. This *deep learning* approach is currently causing great excitement in the machine-learning community and is already leading to new commercial applications.

B.13.9. Geoffrey Hinton is a computer scientist based in Toronto who was one of the first computer scientists to show how to make computers "learn" more like a human brain. He has recently participated in exciting advances using so-called deep neural networks. His start-up company on such approaches to computer learning and recognition problems was bought by Google in 2013. Hinton is the great-great-grandson of logician George Boole. Photo by Emma Hinton

Key concepts

- Feedback loop
- Turing Test
- Expert systems
- Game tree
- Artificial neural network



ELIZA

The quest for a program that can imitate a human dates from the earliest days of computing. One of the earliest "chatbot" programs was written by Joseph Weizenbaum at MIT in 1964. The program was called ELIZA and was designed to create the impression that the user was having a conversation with a psychotherapist. Weizenbaum named the program ELIZA after Eliza Doolitle, a working-class character who learns to speak with an upper-class accent in George Bernard Shaw's play *Pygmalion*. ELIZA uses an open-ended style of questioning similar to the client-centered psychotherapy introduced by psychologist Carl Rogers in the 1940s. This style of therapy aims to increase a patient's understanding and insight by asking questions about the person's thoughts and feelings and by rephrasing the responses. ELIZA has no understanding of the meaning of the conversation, but the program does "pattern matching" and follows some simple rules for formulating questions. The style of conversation that results can be seen from the following example script from an ELIZA session:

Person: Men are all alike.
ELIZA: In what way?
Person: They're always bugging us about something or other.
ELIZA: Can you think of a specific example?
Person: Well, my boyfriend made me come here.
ELIZA: Your boyfriend made you come here?
Person: He says I'm depressed much of the time.
ELIZA: I am sorry to hear that you are depressed.
Person: It's true. I am unhappy.
ELIZA: Do you think coming here will help you not to be unhappy?
Person: I need some help, that much seems certain.
ELIZA: What would it mean to you if you got some help?
Person: Perhaps I could learn to get along with my mother.
ELIZA: Tell me more about your family.

Weizenbaum has said that ELIZA provided a parody of the responses of a nondirectional psychotherapist in an initial psychiatric interview. He also said that he chose the context of psychotherapy to "sidestep the problem of giving the program a data-base of real-world knowledge"[23] because the therapeutic situation is one of the few human situations in which a human being can reply to a statement with a question that indicates very little specific knowledge of the topic under discussion. The dialog could sometimes be so convincing that some users thought they were dealing with a human therapist instead of a machine, and there are many anecdotes about people becoming emotionally engaged with the ELIZA program.