Category Overview

8/15/07

Automation

Delegating tasks to computational systems

A pervasive question in computing is, What can be automated? The "what" in this question usually refers to any human task, most especially cognitive tasks, that might be performed computationally. Can we build computers to play chess? Solve math problems? Search the Internet for what is on our minds given clues in our keywords? Translate between Arabic and English in real time? Drive a car through backcountry terrain? Label images accurately? See what our eyes see?

Automation questions are traditionally concerned with the effect of machines such as assembly lines or printing presses on productivity and jobs. Computing has significantly extended the set of tasks that might be considered for automation.

Computational Automation

The relatively easy, tractable computational tasks present few automation challenges. We just do them.

The intractable computational tasks, on the other hand, present many automation challenges. If the known computational methods are all exponential or worse, then we can solve these problems only if they are not very large. The large ones are infeasible.

The class of very hard computational tasks includes most of the problems people want to solve for science, engineering, and commerce; for example, figuring route capacities in a transportation network or numerically simulating an aircraft in flight. (These are all NP-complete problems.) There is tremendous motivation to find approximate or heuristic ways to solve them -- not perfectly, but acceptably well. A heuristic method means that what was previously beyond reach can now be done by machine. A particularly good heuristic may well be labeled a "technology breakthrough" because so many people are pleased with it.

Thus we can define computational automation as the discovery of a heuristic that maps a very hard computational task to a system that performs the task acceptably well. Not perfectly, but good enough.

Cognitive Automation

In much the same way as computational automation, the field of artificial intelligence (AI) is concerned with mappings from tasks to good-enough systems. In this case the tasks are cognitive tasks of humans and possibly animals. The big difference is that many cognitive tasks are not known to be computational, and therefore solutions cannot start with the assumption that they are trying to simplify hard computational problems.

This leaves the AI problem-solver with two basic choices for assumptions about human tasks:

> **The strong AI hypothesis** holds that human tasks are ultimately computational; that is, they are a subset of the hard computational tasks noted earlier. Strong AI therefore seeks to discover and automate the computational methods behind cognition.
>
> **The weak AI hypothesis** makes no assumption about whether the human tasks are computational or not; instead it searches for good-enough computational equivalents. Weak AI seeks to build computational systems that mimic human tasks acceptably well. Weak AI is inherently experimental because experiments are the only way to confirm that a computation can mimic a human behavior.

There has been considerable debate over the validity of the strong AI hypothesis. A famous "Chinese Room Argument" by John Searle showed that a standard computer that speaks fluent Chinese would internally be doing nothing more than manipulating symbols without regard to their meaning. Its mechanism would be completely unintelligent. Therefore a perception by a Chinese speaker that it is intelligent would be ungrounded. In practice, the strong AI hypothesis is mostly theoretical. Most people won't accept that a computational system can perform human tasks unless the system passes a battery of acceptance tests. In other words, they employ the weak AI stratagem to guide their acceptance even if they worked from the strong AI hypothesis to propose a system.

Behavioral acceptance tests help decide whether a computational system successfully performs human tasks. They compare the observed behavior of the computational system with the observed behavior of the humans doing the task. The Turing test, proposed by Alan Turing in 1950, was an early example; it compares conversational behavior of humans and machines.

Still, the notion that the brain may be fundamentally computational has inspired many people. The plausible scientific argument for this belief is that the brain is made up of billions of neurons, each with a definite structure of connections to other neurons, and each having a definite electro-chemical function that determines when it “fires”. With sufficiently precise instruments to map the brain and then with sufficiently powerful computers, it may one day be possible to simulate a human brain in real time. In other words, the mind is software that can be ported from the brain to any computing system. Ray Kurzweil is a best-selling author and ardent proponent of this view.

Others find this view to be irrelevant to cognition. They hold that consciousness is a statistical phenomenon caused by the biological structure of the brain, not by the mechanics of neuronal firings. They believe that an artificial brain may one day be built, but advances in software and hardware technology will not lead to the breakthrough. Philosopher John Searle is an articulate spokesman for this view. Searle likes to point out that a simulated brain is not the same thing as a real brain, just as simulated digestion is not real digestion. In other words, there is a world of difference between a simulation and the actual process; simulation does not duplicate the process.

Still others believe that mechanical computation has nothing to do with intelligence and that computers shed no light on intelligence; we would be better off focusing on meaningful questions of design.

All these parties engage the fray with great passion. When you think about it, it is not surprising: people care about what it means to be human.

We in the computing field have a quirky relationship with AI. When there is no working computational system for a task, we are likely to characterize the task as “intelligent” and its operation as a “mystery”. But once an acceptable computational system has been found, we are likely to characterize the task as “unintelligent” and “machine like”. In other words, tasks stop looking intelligent when we find computational systems to do them.

Where Philosophy and Science Meet

Many of the questions about computational feasibility of cognitive tasks turn on philosophical interpretations of human cognition as well as on scientific interpretations of the machinery. Philosophical truth is not the same as scientific truth. Consequently, many controversies regarding the feasibility and significance of computational cognition are not likely to be settled soon. To be an effective designer in this domain, one has to learn to make philosophical distinctions as well as scientific distinctions.

The most obvious example of this is the question of whether a computer can be intelligent. This question was on the minds of many people when the first electronic computers were being designed in the 1940s. In a famous book, *God and Golem* (around 1948), the great engineer Norbert Wiener meant to cast doubt on the premise of intelligent machines. One of the earliest treatises on computer circuitry, by Warren McCulloch and Walter Pitts (1943), showed that a neuron-inspired model of threshold circuits was capable of the same operations as Boolean logic, subtly implying that the brain is a universal computer. The press releases about the installations of the first commercial computers referred to them as electronic brains. In his famous paper on the possibility of intelligence (1950), Alan Turing, who believed that machines would one day be intelligent, deftly skirted disagreements on the definition of intelligence by proposing a behavioral simulation game now known as the Turing test. He forecast that by the year 2000, computers would, at least 50% of the time, be able to fool humans into thinking they are humans for at least 10 minutes. Around 1964, Joseph Weizenbaum of MIT built a program called Eliza to imitate sessions with a psychiatrist; although he intended it to mock the Turing test, many people took it seriously and spilled their deepest secrets to the machine. Around 1970 Hubert Dreyfus, a philosopher working in the AI lab at MIT, challenged the feasibility of “expert systems” on the grounds that the hallmarks of expert behavior are not the applications of rules, which is all computers can do. This issue is still debated today. When one reviews all these debates, one can cynically conclude that all we have to show for many years of trying to build computers that are like brains, is that our brains think they are computers.

In the 1950s, a group of scientists who were studying these issues called their field Artificial Intelligence (AI). They were interested in machines that could simulate tasks that were ordinarily seen as intelligent tasks that only humans could perform. They started with computer programs that would play mental games such as tic-tac-toe, nim, bridge, checkers, and chess. They built programs that solved word problems of the kind commonly seen in high school algebra. They built programs that controlled robot arms in stacking blocks or grouping objects on a table. They sought programs to translate among natural languages, for example, English to French and back again. They connected speech synthesizers to the output of computers and designed speech recognizers for the input. They sought to capture the knowledge of human experts into programs called expert systems. They sought to simulate human processes of creativity and invention, such as music composition, writing novels, or making connections among newswire stories. These efforts produced numerous practical results. But they made little headway toward the Holy Grail of AI, machine intelligence. By 1985, federal research sponsors became disillusioned with the lack of progress toward machine intelligence and became much more demanding for deliverables. The leaders of AI have described the decade following as “AI winter”. Many AI leaders called for the reformulation of the AI project as the search for systems that enhance and amplify human intelligence -- rather than replace it.

Computing with Patterns

Many early speculators noted that the brain functions as an associative store -- memories are stored not with names but with links to other memories. They believed that the brain works by storing patterns that are retrieved by other patterns. They believed that consciousness is a never-ending series of moments where new patterns are retrieved based on the current pattern and sensory input. They studied the simulation of cognition in devices with associative memories and in so doing they seemed to avoid the criticisms leveled at cognitive capabilities of rule-based computation.

A popular associative structure was the neural net, a circuit made of threshold gates that fired with the weighted sum of their inputs exceeded an internal threshold. (Threshold circuits were proposed by McCulloch and Pitts.) In the 1950s, Frank Rosenblatt proposed a class of neural nets called perceptrons that would respond with a 0 or 1 output to a binary input representing a visual field. A perceptron could recognize a letter engraved as pixels in a raster field. In 1969, Marvin Minsky and Seymour Papert of MIT attacked perceptrons as too limited and incapable of recognizing important patterns. While this was a setback for neural networks, others found extensions that overcame the limitations of perceptrons. In 1982 John Hopfield of Caltech proposed a new kind of neural network that contained internal feedback and thus could enter numerous stable internal states triggered by external inputs. Hopfield networks could be trained to perform complex functions, like controlling the glide path of an airplane during a landing.

In the 1980s, Pentti Kanerva proposed an architecture called sparse distributed memory that would store bit-patterns associated with given cue patterns. A single pattern was stored in many cells of the memory simultaneously and was retrieved by statistical reconstruction of the states of many cells. Kanerva said this resembled the way the human brain handles long term memory.

Also in the 1980s, several researchers used a statistical method called Bayesian Inference to compute the most probable hypothesis that explained given data. They claimed that the brain is a Bayesian inference machine and even that the sparse distributed memory model implicitly employed Bayesian inference to generate its outputs.

The importance of these studies is that they showed successful statistical models for many aspects of human cognition, aspects that appear to be unreachable by rule-based, deductive computations. While these methods have produced many useful results, they too have not shed much light on whether human intelligence can be exhibited by a machine.

The Four Methods of Artificial Intelligence

Even though its agenda is ambitious, the business of finding good-enough computational systems for cognitive tasks rests on four basic methods. They are models, search, deduction, and induction. Many systems use more than one of these methods.

In all four, knowledge representation is a major issue. All computations within these systems are over representations. When humans perform the same tasks, they almost always incorporate tacit, contextual information that they are not aware of. Therefore, the mapping from the original setting into the computational representation necessarily loses information. Even if the computational steps exactly mimic the real thing, the representation error can still lead the computation to produce results not satisfactory to human observers. This is called the *frame* or *context problem*.

Modeling means to employ or posit an information-process model of the cognitive behavior, then simulate the model with a computer. A model is a representation of a physical system by equations or algorithms that can be used to calculate the system's response to stimuli. A model of cognitive behavior may be computational even if the actual brain processes are not. A model is validated by comparing its responses to those of the real system and confirming that they are close most of the time. A few examples of models include:

(1) Logic systems model cognitive tasks as the sifting of many facts and previous deductions until a conclusion is reached. Fuzzy systems generalize by allowing truth values to be anything from 0 to 1, and not just 0 or 1.

(2) Expert systems assume that human experts apply internal rules that map current sensory input and experiential memories to action.

(3) Neural networks seek to realize cognitive functions by partially imitating the neuronal connection structures of nervous systems.

(4) Sparse distributed memory models consciousness as a stream of states circulating through a large memory system that stores copies of binary patterns in many cells distributed throughout memory.

Search uses various heuristics to reduce the number of states that must be examined for a solution in very large spaces to a manageable number. Search is important because many cognitive tasks could be accomplished perfectly by searching for the best among many alternatives. Unfortunately, the spaces of alternatives can be prohibitively large, forcing us to use abbreviated searches that don't always yield the best answer. Examples of search include:

(1) Chess. Theoretically, one can win a chess game by enumerating all possible future board configurations and identifying move sequences that lead to wins. Unfortunately, the size of the tree of possible future boards is so large that only a limited number of branches can be examined. Various rules of thumb, such as "exchange for pieces of equal or greater value", are used to limit the search. Interestingly, it is not known whether human chess masters use any of these heuristics as they play.

(2) Genetic Algorithms. These algorithms mimic the way cells divide and recombine genes. A solution to the task is represented as a binary string and each such string is assigned a "fitness" number. The higher the fitness, the better the solution represented by the string. At each generation, the most fit strings are randomly cut and recombined, and the least fit strings are discarded. After a few generations, a string of dominant fitness emerges and is offered as the solution. It may not be the best solution, but it is often very close in fitness to the best.

(3) Internet Search. Google visits web pages and ranks them according to the number of links pointing to them and the

reputations of the pages hosting those incoming links. When a user searches on keywords, Google returns the highest ranked pages. Most of the time, users find a useful answer among the top ten web pages proposed by Google.

(4) Neural Networks. A network is trained by presenting it with a set of (X,Y) pairs: Y represents the desired response to input X. The network's internal parameters are adjusted so that the network produces Y whenever X is presented. Once this has been done, Y values can be found quickly by presenting their corresponding X values. New Y values can be estimated by presenting new X values.

Deduction uses rules of logic to derive conclusions allowed by given inputs and assumptions. During the early years of AI, many people including Alan Turing assumed that the rational brain works by logical deduction. They explored whether a "deduction machine (or language)" would behave intelligently. A deduction machine generates statements that follow logically from previously deduced statements or from given statements stored in a database. Prolog is an example of a logic deduction language. It was the basis of the Japanese Fifth Generation Computing Project in the 1980s, which sought to build supercomputers that would process billions of deductions per second. Expert systems were envisioned as a major application of deduction systems. Unfortunately only a handful of expert systems worked well. Some philosophers (notably Hubert Dreyfus) have argued that the reason for the failure is that human experts don't always act logically and that many expert actions cannot be explained by rules or deductions at all. Still, the hope that an intelligent deductive system can be built lingers on. A 20-year-old project called Cyc.com has been attempting to amass a database of billions of common-sense facts to find out whether the failure of expert system can be attributed to insufficient common sense. Luis von Ahn of CMU has proposed an Internet game called Verbosity that would populate such a database as a side effect of tens of thousands of people playing the game.

Induction is the main process for constructing models by finding patterns in data. Induction is a very creative process and there is no one way to explain how humans do it. However, there are some common categories of induction that are useful. Examples:

(1) Curve Fitting. Given a set of data in the form of (X,Y) pairs, find a curve of given type that best fits the data. If we hypothesize that the data conform to a curve $y = f(x)$, we could measure the error between the actual Y values and the hypothesized curve by adding up the errors $(Y-f(X))^2$; the curve fitter would find values of

parameters of function “f” that minimize the error. Someone who wants to predict the future response to a new input “Z” simply proposes “f(Z)”. The one function “f” becomes the model of the large data set of (X,Y) pairs.

(2) Bayesian Inference. This method aims to find the most likely hypothesis H given observed data D. A theorem due to Thomas Bayes (circa 1750) says $p(H|D) = p(D|H)p(H)/p(D)$. It is usually easy to determine the probability the hypothesis is true in general $p(H)$, the probability the data will be observed $p(D)$, and the probability that that data will be observed given the hypothesis $p(D|H)$. Some very sophisticated models have been built around this simple law. Wikipedia tells the story of the search that found the lost submarine Scorpion in 1968 by using Bayesian Information to calculate which squares on an ocean grid would most likely yield the missing submarine if searched. They found the submarine. A Bayesian search engine can be used in several ways; it could evaluate several hypotheses to find the most probable or it could iteratively improve a hypothesis until it had the highest probability of being true.

(3) Reinforcement. A system with adjustable parameters is presented with inputs and a feedback mechanism tells the error in the resulting response. The system adjusts some of its parameters to lessen the error. After a while, the system has “learned” the data and gives low errors all the time. The neural net example given earlier is also an example of reinforcement.

Collective Intelligence

The recent books “Wisdom of Crowds” (James Surowiecki) and “Wikinomics: How Mass Collaboration Changes Everything” (Don Tapscott and Anthony Williams) popularized the notion that social networks exhibit intelligence.

The Wisdom book illustrates the idea that many people can pool little bits of knowledge to construct useful new knowledge. Sometimes this is as simple as averaging together numbers they give you. For example, Google and other companies run internal stock markets that buy and sell imaginary stocks representing future products. The stock prices serve as remarkably accurate predictions of a product’s success in the real market. Sometimes the pooling of knowledge relies on sophisticated signal processing of weak signals gathered from many people in a network. For example, John Petersen of the Arlington Institute proposes using the Internet to let people register premonition dreams to see whether disasters can be predicted.

Google has developed sophisticated inference methods to find the context in which a searcher is searching; useful clues include the history of the searcher's past searches, other people's use of the same keywords, and the geographic location. These methods enable much higher quality answers to searches.

The Wikinomics book illustrates the idea that people can design new ways of collaborating; organizations that practice them are smarter than more traditional organizations. An example is "crowdsourcing" where challenge problems are posted on a Web site, then anyone can volunteer a solution, and the challenger selects the best solution. Even the Rockefeller Foundation is using such an approach to stimulate innovation (innocentive.com). Another example is the genre of games invented by Luis van Ahn for "human computation"; thousands of people play his games (e.g., esp.com) on the Internet and as a byproduct do seemingly impossible tasks such as labeling images with accurate keywords.

Collective intelligence is a relatively new field of study. The examples above show that this field is enriching the four basic AI methods. It may also be opening a new method, the design of games and collaborative systems that harness collective intelligence.

