VOICES

ONE MORE PARADIGM SHIFT, AND YOU'LL BE ABLE TO TELL

ACCORDING TO MANY COMPUTER-industry observers, we're on the brink of the voice-recognition revolution. In mid-1997, Dragon Software released NaturallySpeaking, the first desktop program that can recognize continuous speech rather than just isolated words. IBM, Philips, and emerging language-software giant Lernout & Hauspie have since shipped rival packages. And in March, a coalition of internet companies released Voice XML, an open standard for making websites voice-accessible. "Speech [recognition] is not just the future of Windows," says Bill Gates, "but the future of computing itself." Five years from now, a computer without ears will seem as antiquated as one without a mouse.

But although today's speech-recognition systems perform at levels that would have seemed dazzling twenty years ago, they are still inferior to humans in almost every way. They have trouble deciphering mumbles, stutters, and accents. They get flummoxed in noisy environments or when several people are speaking at once. Humans solve these problems more or less effortlessly, but if the last half century of research is any guide, knowing how it works in the natural world won't help the machines. As Reinhard Karger, a project manager at Germany's Verbmobil speech-translation project puts it, "An airplane does not flap its wings."

To overcome these performance barriers—to justify the gleam in Bill Gates's eye—computer speech recognition needs a paradigm shift. In fact, it was just such a paradigm shift, thirty years ago, that made today's systems possible.

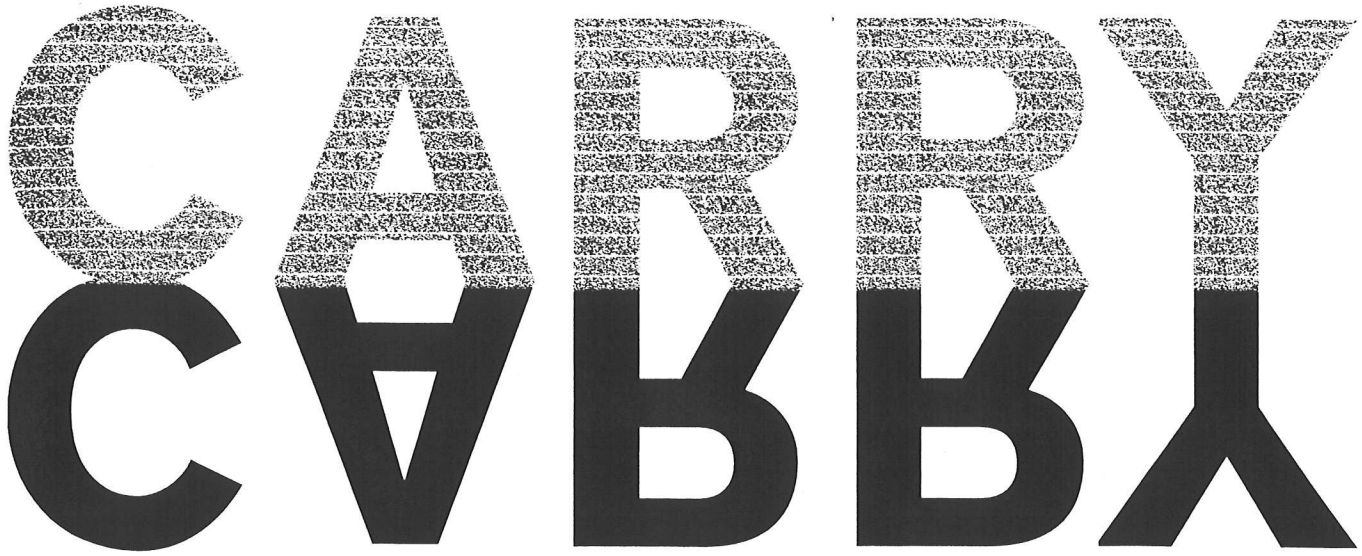
THE SUCCESS of modern speech recognition isn't due to any one person, but it just might be due to two: Jim and Janet Baker. The Bakers met as graduate students at New York's Rockefeller University in 1970. Janet was a biophysicist studying the nervous system. Jim was getting his Ph.D. in mathematics. He became interested in speech because Janet was working on it.

The dominant model of speech interpretation at that time was template matching. To get a template, researchers would record test subjects pronouncing a word and then do a frequency analysis. The frequencies might be higher when pronounced by a child and lower when pronounced by a linebacker, but the overall shape of both frequency progressions would be similar. When several such progressions were combined, the result was a sausage-shaped zone of the possible frequencies for a given word.

An automatic speech recognizer might have a vocabulary of several hundred such word templates. When a user spoke a word into the computer's microphone, the computer would do a frequency analysis and compare it with all the templates in its memory. After picking the closest-matching template, the computer would spell out the corresponding word on the screen.

There were a number of problems with the template-matching model. First, it was inflexible with regard to speed of pronunciation. Humans have no trouble recognizing that "goal" is the same word as "goooooaaaal!" But to the template-matching computer, they look completely different; one is ten times longer than the other. To resolve this problem, scientists came up with something called dynamic time warping. This is a set of functions that the computer can use to shrink or stretch the utterance until it conforms to one of the templates it knows. Unfortunately, dynamic time warping is tricky. For example, in "goooooaaaal!" the letter *g* doesn't stretch at all. The computer has to know to warp different sounds in different ways.

Second, while template matching is well suited to recognizing discrete words, it's clumsy with the smaller building blocks of speech, known as phonemes. Phonemes are the basic sounds in a language—*sh, ee, mm,* and so on. English has about fifty of them. In principle, recognizing phonemes should be more efficient than recognizing entire words because the computer only has to store fifty-odd templates, rather than thousands. But phonemes are constantly getting pressed up against each other in speech. The beginnings and ends of a phoneme get twisted by the phonemes on either side of it, so that the *o* in "goal" looks very different from the *o* in "yoga." It's also hard for a computer to tell where one phoneme ends and the next begins. When you give a computer continuous speech rather than single words, the problem is compounded. "Did you"



YOUR COMPUTER EVERYTHING. BY MATTHEW STEINGLASS

becomes "didja"; "get Ted" becomes "ge' Ted." All of this leads to the speech-recognition scientist's nemesis: ambiguity.

This is where mathematician Jim Baker came in. As an undergraduate, Baker had worked on stochastic modeling, a way of using probability and statistics to handle ambiguous information. Baker had a hunch that stochastic modeling, which emerged in the late 1960s, would prove useful in speech recognition, and the 1975 paper in which he set forth the idea has become a classic. Every speech-recognition application on the market today uses stochastic modeling—more specifically, a set of techniques known as hidden Markov models, or HMMs.

The equations behind HMMs are extremely difficult, but the overall concept is at least comprehensible to the layman. Let's say you have a random, or stochastic, process, such as the weather. Suppose you know the probability that any given state in this system will be followed by any other state. For example, there might be a 40 percent chance that a sunny day will be followed by another sunny day, a 20 percent chance that it will be followed by an overcast day, and so forth. (This is called a Markov process—each state is determined probabilistically by the state immediately before it.) Let's say you also know the probability that each state will be associated with rainfall. There might be a 20 percent chance that an overcast day will yield an inch of rain but only a 1 percent chance that a sunny day will.

Now, let's suppose you are given a series of precipitation reports. You don't know whether it was sunny or overcast on these days; you only know how much rainfall there was. What sequence of sunny or overcast days is most likely to have produced this sequence of precipitation? This is a hidden Markov model. It's "hidden" because you have no direct knowledge of the underlying Markov process—the sequence of sunny or overcast days.

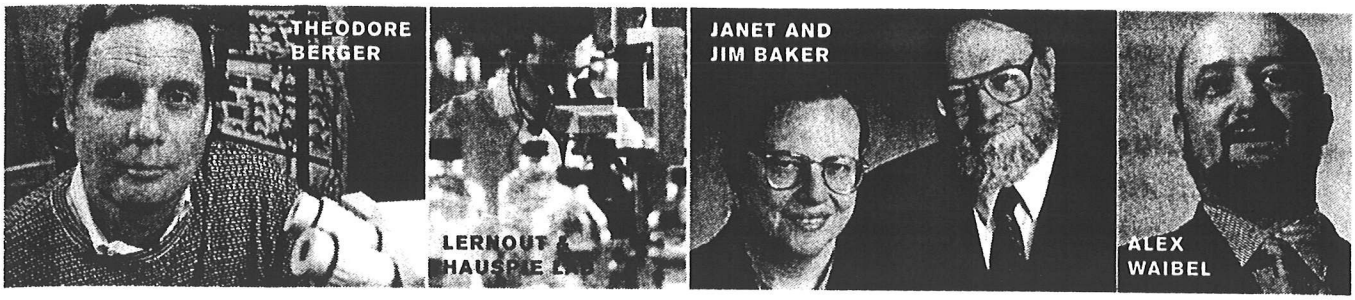
By the early 1970s, mathematicians had developed a number of techniques for handling this sort of problem. Baker's insight was to apply them to speech recognition. He was looking for a hidden sequence of phonemes instead of a hidden sequence of sunny or cloudy days, and he was working with sound frequencies rather than precipitation reports. But the underlying mathematical problem was the same. A computer could use HMM equations to match the speaker's frequencies to the phonemes she was most likely to have been pronouncing.

The Bakers did more than hypothesize that HMMs might work; they set out to prove it. In 1972, the newly married couple transferred to Carnegie Mellon, where the government's Defense Advanced Research Projects Agency sponsored their work. The goal was to design a system with a thousand-word vocabulary and 90 percent accuracy, and they achieved it with an HMM-based system they named DRAGON. By 1976, a version of DRAGON was outperforming the best of the template-matching models. In 1982, after a number of frustrating years at IBM and at Verbex, an Exxon subsidiary, the Bakers went into business on their own, founding Dragon Systems, Inc. Today, Dragon's NaturallySpeaking is the No. 1 voice-recognition program on the U.S. market.

TWENTY-FIVE years after DRAGON, HMM-based speech recognition is a billion-dollar industry. Continuous-speech voice-recognition systems, such as dictation programs, can achieve accuracy rates of up to 98 percent when trained to the voice of a single user. But "speaker-independent" systems rarely go much higher than 80 percent. HMMs are statistical tools, and the more variables that enter the system—differences in accent, sex, and age of speakers—the more ambiguous the results.

The most obvious way of minimizing this uncertainty is to feed the programs more and better data. But compiling vast acoustical databases is incredibly resource-intensive, and it favors big companies. In fact, these economies of scale recently spelled the end of Dragon Systems as an independent company. In March, it was acquired by Lernout & Hauspie.

Another way of minimizing ambiguity is by enlisting the aid of a "language model," which describes what sort of words are likely to follow other words. Language models tend to be partly grammatical (eliminating impossible combinations like "the



FROM LEFT: USC/IRENE FINK; LERNOUT & HAUSPIE; DRAGON SYSTEMS; CARNEGIE MELLON

BERGER'S NEURAL NET CAN RECOGNIZE SPOKEN WORDS THROUGH WALLS OF WHITE NOISE NO HUMAN EAR CAN PENETRATE.

the") and partly statistical (preferring common phrases over unlikely ones).

Alex Waibel is famous for having attacked the problem of voice recognition from yet another angle. In the late 1980s, the Carnegie Mellon researcher helped pioneer what then seemed to be the paradigm shift everyone was looking for: the artificial neural network. In an artificial neural network, or neural net, the "neurons" are pieces of software able to perform simple calculations. They are connected to each other, and each connection is assigned a certain weight. In a speech-recognition neural net, sound frequencies go in at one end, and numbers denoting phonemes come out the other side.

Neural nets are programmed by a trial-and-error training process. When a net gets an answer wrong, it changes the weights of the connections between its neurons and tries again. When it gets an answer right, it reinforces the weights it has. In 1986, Waibel trained a neural net to recognize the phonemes *b*, *d*, and *g*. It achieved 98.5 percent accuracy—better than contemporary HMM systems.

But neural nets as conceived in the 1980s have a problem: They aren't very good at handling time. They get confused if the phoneme comes in a little bit earlier or later than usual, and they need to have their data nicely segmented for them in advance. As a result, they aren't practical for continuous-speech recognition. Many systems today use them, but only as part of a hybrid approach, where neural nets function as front ends or back ends for specific tasks.

Last October, however, neural nets were back in the speech-recognition news. University of Southern California neurobiologist Theodore Berger announced that he and a colleague had developed a neural net that broke one of the field's most significant barriers. Under noisy conditions, where most systems perform badly, Berger's net could actually outperform people. It could recognize spoken words through walls of white noise so thick that no human could understand what had been said.

Berger didn't set out to build a speech-recognition system. He set out to build a neural net that more closely modeled the behavior of neurons in the brain. In real neurons, Berger explains, what matters is how fast and how often neurons fire, not simply whether they fire or not. Thus Berger built a neural net that transferred information by pulsing at varying frequencies. Then he looked for problems to turn it loose on. "We finally decided to try speech recognition because it's such a hard problem," says Berger. "We figured, if it could do speech, it could do anything."

Although Berger's achievement sounds remarkable, the response in the speech-recognition community has been muted to skeptical. Part of the reason is its limited scale; Berger's system has a vocabulary of just a dozen or so words. It operates at the level of words rather than phonemes, so it is unable to make anything of a word it hasn't seen before. Nor can it parse continuous speech.

And then there's the problem of flapping airplane wings. Speech researchers have learned to distrust "humanlike" approaches. As Hans Uszkoreit of Germany's DFKI research institute puts it, "It would be very surprising if this were a real breakthrough. These things don't happen that often in engineering fields."

NEVERTHELESS, Uszkoreit is among those who think the field is due for another breakthrough. "Current technology, with hidden-Markov-model-based speech recognition, is just not improving that fast," he says. "So we're all expecting new progress to come from some other kind of development."

Uszkoreit himself holds out hope for a technique called feature-based recognition. If today's HMM-based systems search for the sequence of phonemes "hidden" beneath sound waves, then tomorrow's feature-based systems will try to penetrate to an even deeper layer—to the features from which phonemes are composed. "For example, *p* and *b* have one feature in common—they're bilabials, you use both lips to produce them," Uszkoreit explains. "But they differ in another feature—one is voiced, the other unvoiced." Uszkoreit hopes that computers might be able to zero in on the differences in acoustic profile that separate, say, voiced phonemes from unvoiced ones. This, he hypothesizes, must be how humans do it. "It may even be that we have a special part of our own neural net which is trained to certain features," he says.

But wait a minute. It sounds suspiciously as if Uszkoreit wants his machines to imitate humans....

"Did you know that babies can distinguish phonemes which occur only in foreign languages?" Uszkoreit asks. "In the beginning, you're open to all feature differences. Then when you adapt your speech perception to your own language's sound system, you lose this ability."

There is indeed a great temptation to find approaches that seem closer to how humans do it. And sometimes even the most hardened speech-recognition scientist is tempted to think like a human being rather than a machine.

Matthew Steinglass is a freelance writer living in Amsterdam. His article "International Man of Mystery" appeared in the April 1998 issue of *Lingua Franca*.