The Road to Siri and Alexa

Interview with Computer Scientist Jason Eisner, Ph.D.

by Carol Blackburn, Ph.D.

Before Siri and Alexa were possible, engineers had to solve daunting technical challenges. They had to develop computer systems that could take speech audio data and identify words within it. They also had to develop systems that could "understand" questions and requests asked in conversational language (as opposed to computer language) and respond appropriately. Johns Hopkins computer science professor Jason Eisner has been a part of the research effort that made this technology possible. He spoke to Imagine *about how some of those challenges were overcome and the challenges that remain.*

WILL KIRK

The development of speech recognition systems was a monumental undertaking. What breakthroughs made it possible?

Methods for transcribing spoken language into written text took many decades to develop. Today, everything we do in this field relies on statistical machine learning, an approach that has now taken over all of artificial intelligence (AI). But speech recognition did not start out that way.

In the 1970s and 1980s, AI researchers trying to develop "expert systems" thought that by writing down their understanding of a phenomenon, they could enable a computer to reason about the phenomenon more or less the way an expert would. Linguists in the 1970s wrote rules for detecting individual phonemes, much as you teach a child the sounds associated with each letter of the alphabet. The rules written for computers used technical specifications; for example, "If the sound spectrum has energy peaks at these frequencies, then it's probably the vowel sound *ah*; if it has peaks at these other frequencies, it's probably the vowel sound *ee*." The rules got more complicated when they tried to combine different kinds of evidence. Writing them was very labor-intensive—and not very effective at actually performing speech recognition.

Starting in 1972, however, a research group at IBM led by Fred Jelinek pioneered an entirely different approach. Fred's notion—which transformed speech recognition, then natural language processing, and then the rest of AI—was that we simply can't write down all the rules we use because we don't know them all consciously. Instead, Fred reasoned, rules about how language behaves could be found by analyzing language statistically. And the way to systematize that kind of information was to write simple statistical formulas that a computer could use to identify patterns in data—then let it analyze large data sets.

A Carnegie Mellon doctoral student named Jim Baker took ideas he learned in an internship at IBM back to CMU. For his Ph.D. dissertation in 1975, he built a statistical speech-recognition system using an elegant technique called hidden Markov models. It was a game changer. That approach beat all others. In 1982, Jim and his wife Janet founded the first speech-recognition software company, Dragon Systems.

Speech recognition systems are now amazingly good.

You know the saying about restaurants: You can have fast, cheap, or good—pick two? There used to be a similar maxim about speech recognition: It could be fluent, speaker-independent, or have a large vocabulary—pick any two. If you wanted it to be speaker-independent with a large vocabulary, you would have to speak...with...pauses...between...words. But today, speech recognition is so good

"THIS IS A FIELD WHERE YOU CAN COMBINE YOUR INTERESTS IN LANGUAGE, COGNITION, MATH, AND COMPUTER SCIENCE."

that you don't have to pause, and the systems don't need to be trained on your specific voice. If you hand your phone to someone else—someone who has a cold, or is a different gender, or has a different accent—the system will not be confused by the fact that it was listening to you a moment before.

What are the major tasks at the heart of a system such as Siri or Alexa?

At its core, a speech-recognition system asks two questions of a speech utterance: Would a human ever say xyz? And if so, would xyz sound like what I heard? For any single xyz, these two questions are answered using mathematical models that are basically formulas for estimating probabilities from data. With these models in hand, speech recognition becomes in some sense a simple problem: What's an xyz that makes both probabilities high? But actually finding such an xyz requires clever search algorithms that don't have to look at all the gazillion xyz possibilities one by one.

Next, the text xyz must be transformed from ordinary human language into something an AI system can "understand" and use. This discipline is called *natural language understanding*. This challenge is somewhat simplified in the case of an AI system like Alexa because the vast majority of requests to Alexa fall into fewer than 100 categories. Alexa is actually a collection of capabilities referred to as skills, each designed to handle a particular kind of request. Skills are like apps; anyone can write one using the Alexa Skills Kit. To create a new skill, you just have to anticipate all the ways that people might make such a request. Each skill also includes procedures for responding to those requests.

Siri receives a broader range of questions than Alexa does, but its approach to handling common queries is similar. There are people at Apple whose job is to look through the query logs, cluster the queries using simple machine-learning techniques, and see if there are types of queries that they don't yet have a good answer for. What do those queries look like? And what sorts of responses should be generated for them?

Sometimes what they find is sort of fun. For example, when Apple engineers analyzed the first version of Siri, they discovered that a lot of people were flirting with her. So they had to program in some responses to respond gently and humorously, but at the same time discourage the flirting.

Do computer scientists and linguists differ in their approach to problems like these?

Linguists and engineers tend to have different concerns. Engineers are interested in what happens frequently; they'd like to get 99% of the cases correct. Linguists are just as interested in atypical cases because they are scientifically informative. When do rare events occur? How do humans understand them even though they're rare? What does *not* occur? Those outliers are of academic interest, but have not had a major impact on the development of digital servants like Siri or Alexa.

What problems are researchers still struggling with?

We would like to go beyond servicing common requests, and really understand language. IBM's Watson system was able to beat humans at *Jeopardy!* questions because it could skim Wikipedia and understand it superficially. But no current computer can reliably answer fourth-grade reading comprehension questions, whose multiple-choice answers are constructed to fool skimmers.

There have been essentially two approaches to deeper comprehension. One approach asks, if humans use reading to expand vocabulary and knowledge, why couldn't a computer get smarter by reading, too? A computer scientist named Doug Lenat has been working on this for decades. He wanted to program a computer with enough knowledge that it could read an encyclopedia and learn by itself. An immediate obstacle is that you need a lot of linguistic competence and background knowledge to start reading an encyclopedia—the authors assume you've already lived in the world for a while. So Lenat's team set out to write down all this common knowledge using formal logic.

This project ran into the same difficulties as other rule-based expert systems. Much common knowledge is not universally true. For example, I might tell the computer, "If I drop a glass on a hard floor, it will break." But what exactly do I mean by a glass? Drop it from what height? You probably supposed that I'm dropping the glass from a height at which it would be comfortable for me to hold it. But what if it's only a millimeter above the floor? What if I dropped the glass in zero gravity? It's hard to precisely write down all the knowledge we subconsciously use when reading.

What is the other approach?

Deep learning is a statistical approach that develops complex predictive formulas known as neural networks. A formula may weigh different kinds of evidence, so it does something like reasoning, but not necessarily with hard-and-fast rules. The learning system just tries to find a formula that usually works.

As an example, consider image understanding. An image can be described by a set of numbers that specify the color of each pixel. A deep-learning formula can take such a collection of numbers as input and say, "A puppy in a shoebox!"

But when the input is a sequence of words, what should take the place of pixel colors? There are no obvious dimensions we would use to quantify a word's meaning. So, what if we let an AI system come up with its own dimensions as it analyzes sentences? We might not know what those dimensions mean, but that wouldn't matter to the AI system; the only thing that mattered would be the effectiveness of the formula.

Only now do we have the mathematical models and computing power to effectively generate these kinds of representations. And this is what many people are doing now: using machine learning to analyze words based on how they are used. It turns out that by paying attention to all the contexts in which words show up, a neural network can place every word in a 300-dimensional Euclidean space. This turns the language problem into something that's more like an image problem: Important properties of each word have been converted into numbers.

While we don't know what those dimensions mean, they tell us a lot about how the words behave. Words that are close together in 300-dimensional space have similar meanings and similar usage. And vectors within that space correlate with semantic meaning. The vector that points from Italy to Rome is parallel to the vector that points from Japan to Tokyo. You can actually use these vectors to complete analogies.

Even though a neural network doesn't know what the dimensions mean, it can use them to decide what to do with words. It can decide how to incorporate a word into a sentence diagram because it knows which other words behave similarly. It can translate words in context by picking a word whose 300-dimensional position seems appropriate. And it can answer at least some kinds of questions about sentence meaning. That's a big step.

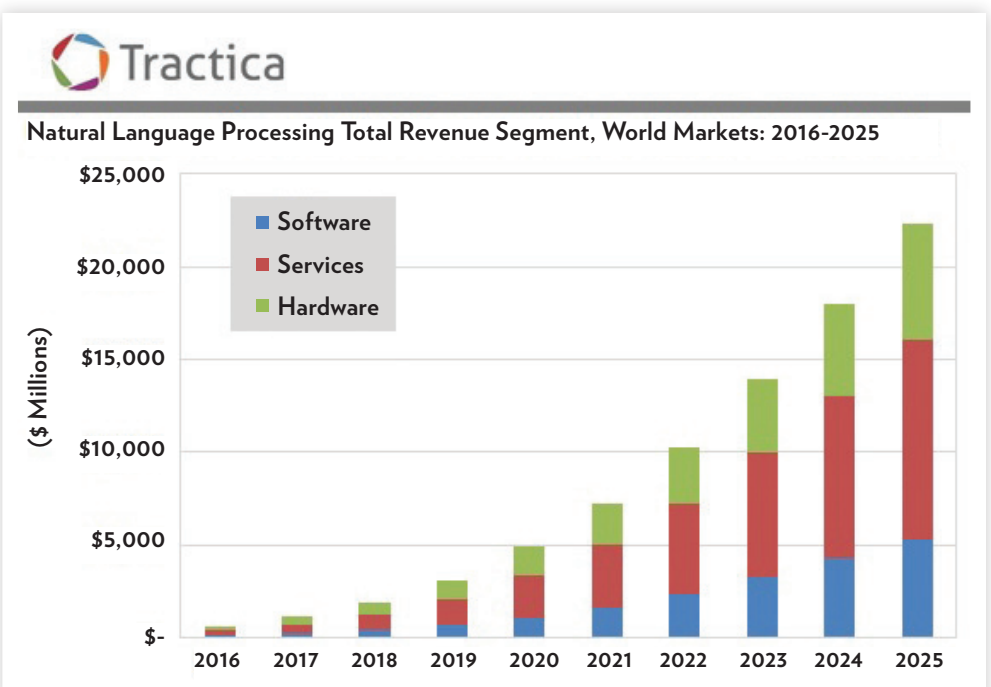
There is clearly a lot more to do. What advice do you have for students interested in these kinds of challenges?

This is a field where you can combine your interests in language, cognition, math, and computer science.

To learn about linguistics in an entertaining way, I recommend *The Art of Language Invention* by David J. Peterson. He has invented many convincing fictional languages for TV and movies, and his book will teach you a lot about the structure of real human languages.

Like the rest of AI, human language technology is based on mathematical and computational modeling. We devise math problems that are really language problems in disguise. The more math you know, the better you can be at this game. Anything you can learn about probability, statistics, or linear algebra will be a great foundation.

Being able to code your ideas is also crucial. Python is currently the most popular language for playing with data. It's a favorite with kids and also with grownup researchers, including my grad students. There are plenty of online resources for learning Python. ■



USED WITH PERMISSION TRACTICA.COM

Natural language processing technology is a rapidly growing field. This figure shows projected growth in the next decade, as estimated by Tractica, a market analysis firm that focuses on human interaction with technology. Read the report: tinyurl.com/NLP-report