McKinsey Quarterly

The programmer's dilemma: Building a *Jeopardy!* champion

IBM computer scientist David Ferrucci and his team set out to build a machine that could beat the quiz show's greatest players. The result revealed both the potential—and the limitations—of computer intelligence.

February 2011 • Stephen Baker

In 2007, IBM computer scientist David Ferrucci and his team embarked on the challenge of building a computer that could take on—and beat—the two best players of the popular US TV quiz show *Jeopardy!*, a trivia game in which contestants are given clues in categories ranging from academic subjects to pop culture and must ring in with responses that are in the form of questions. The show, a ratings stalwart, was created in 1964 and has aired for more than 25 years. But this would be the first time the program would pit man against machine.

In some sense, the project was a follow-up to Deep Blue, the IBM computer that defeated chess champion Garry Kasparov in 1997. Although a TV quiz show may seem to lack the gravitas of the classic game of chess, the task was in many ways much harder. It wasn't just that the computer had to master straightforward language, it had to master humor, nuance, puns, allusions, and slang—a verbal complexity well beyond the reach of most computer processors. Meeting that challenge was about much more than just a *Jeopardy!* championship. The work of Ferrucci and his team illuminates both the great potential and the severe limitations of current computer intelligence—as well as the capacities of the human mind. Although the machine they created was ultimately dubbed "Watson" (in honor of IBM's founder, Thomas J. Watson), to the team that painstakingly constructed it, the game-playing computer was known as Blue J.

The following article is adapted from *Final Jeopardy: Man vs. Machine and the Quest to Know Everything* (Houghton Mifflin Harcourt, February 2011), by Stephen Baker, an account of Blue J's creation.

It was possible, Ferrucci thought, that someday a machine would replicate the complexity and nuance of the human mind. In fact, in IBM's Almaden Research Center, on a hilltop high above Silicon Valley, a scientist named Dharmendra Modha was building a simulated brain equipped with 700 million electronic neurons. Within years, he hoped to map the brain of a cat, and then a monkey, and, eventually, a human. But mapping the human brain, with its 100 billion neurons and trillions or quadrillions of connections among them, was a long-term project. With time, it might result in a bold new architecture for computing, one that could lead to a new level of computer intelligence. Perhaps then, machines would come up with their own ideas, wrestle with concepts, appreciate irony, and think more like humans.

But such machines, if they ever came, would not be ready on Ferrucci's schedule. As he saw it, his team had to produce a functional *Jeopardy!*-playing machine in just two years. If *Jeopardy!*'s executive producer, Harry Friedman, didn't see a viable machine by 2009, he would never green-light the man–machine match for late 2010 or early 2011. This deadline compelled Ferrucci and his team to build their machine with existing technology—the familiar semiconductors etched in silicon, servers whirring through billions of calculations and following instructions from many software programs that already existed. In its guts, Blue J would not be so different from the battered ThinkPad Ferrucci lugged from one meeting to the next. No, if Blue J was going to compete with the speed and versatility of the human mind, the magic would have to come from its massive scale, inspired design, and carefully-tuned algorithms. In other words, if Blue J became a great *Jeopardy!* player, it would be less a triumph of science than of engineering.

Blue J's literal-mindedness posed the greatest challenge. Finding suitable data for this gullible machine was only the first job. Once Blue J was equipped with its source material—from James Joyce to the Boing Boing blog—the IBM team would have to teach the machine to make sense of those texts: to place names and facts into context, and to come to grips with how they were related to each other. Hamlet, just to pick one example, was related not only to his mother, Gertrude, but also to Shakespeare, Denmark, Elizabethan literature, a famous soliloquy, and themes ranging from mortality to self-doubt, just for starters. Preparing Blue J to navigate all of these connections for virtually every entity on earth, factual or fictional, would be the machine's true education. The process would involve creating, testing, and fine-tuning thousands of algorithms. The final challenge would be to prepare the machine to play the game itself. Eventually, Blue J would have to come up with answers it could bet on within three to five seconds. For this, the *Jeopardy!* team

would need to configure the hardware of a champion.

Every computing technology Ferrucci had ever touched had a clueless side to it. The machines he knew could follow orders and carry out surprisingly sophisticated jobs. But they were nowhere close to humans. The same was true of expert systems and neural networks. Smart in one area, clueless elsewhere. Such was the case with the *Jeopardy!* algorithms that his team was piecing together in IBM's Hawthorne, New York, labs. These sets of finely honed computer commands each had a specialty, whether it was hunting down synonyms, parsing the syntax of a *Jeopardy!* clue, or counting the most common words in a document. Outside of these meticulously programmed tasks, though, each was fairly dumb.

So how would Blue J concoct broader intelligence—or at least enough of it to win at *Jeopardy!*? Ferrucci considered the human brain. "If I ask you what 36 plus 43 is, a part of you goes, 'Oh, I'll send that question over to the part of my brain that deals with math,'" he said. "And if I ask you a question about literature, you don't stay in the math part of your brain. You work on that stuff somewhere else." Ferrucci didn't delve into how things work in a real brain; for his purposes, it didn't matter. He just knew that the brain has different specialties, that people know instinctively how to skip from one to another, and that Blue J would have to do the same thing.

The machine would, however, follow a different model. Unlike a human, Blue J wouldn't know where to start answering a question. So with its vast computing resources, it would start everywhere. Instead of reading a clue and assigning the sleuthing work to specialist algorithms, Blue J would unleash scores of them on a hunt, and then see which one came up with the best answer. The algorithms inside of Blue J, each following a different set of marching orders, would bring in competing results. This process, a lot less efficient than the human brain, would require an enormous complex of computers. More than 2,000 processors would each handle a different piece of the job. But the team would concern itself later with these electronic issues—Blue J's body—after they got its thinking straight.

To see how these algorithms carried out their hunt, consider one of the thousands of clues the fledgling system grappled with. Under the category Diplomatic Relations, one clue read: "Of the four countries the United States does not have diplomatic relations with, the one that's farthest north."

In the first wave of algorithms to handle the clue was a group that specialized in grammar. They diagrammed the sentence, much the way a grade-school teacher would, identifying the nouns, verbs, direct objects,

and prepositional phrases. This analysis helped to clear up doubts about specific words. In this clue, “the United States” referred to the country, not the Army, the economy, or the Olympic basketball team. Then the algorithms pieced together interpretations of the clue. Complicated clues, like this one, might lead to different readings—one more complex, the other simpler, perhaps based solely on words in the text. This duplication was wasteful, but waste was at the heart of Blue J’s strategy. Duplicating or quadrupling its effort, or multiplying it by 100, was one way the computer could compensate for its cognitive shortcomings, and also play to its advantage: speed. Unlike humans, who can instantly understand a question and pursue a single answer, the computer might hedge, launching searches for a handful of different possibilities at the same time. In this way and many others, Blue J would battle the efficient human mind with spectacular, flamboyant inefficiency. “Massive redundancy” was how Ferrucci’s described it. Transistors were cheap and plentiful. Blue J would put them to use.

While the machine’s grammar-savvy algorithms were dissecting the clue, one of them searched for its focus, or answer type. In this clue about diplomacy, “the one” evidently referred to a country. If this was the case, the universe of Blue J’s possible answers was reduced to a mere 194, the number of countries in the world. (This, of course, was assuming that “country” didn’t refer to “Marlboro Country” or “wine country” or “country music.” Blue J had to remain flexible, because these types of exceptions often popped up.)

Once the clue was parsed into a question the machine could understand, the hunt commenced. Each expert algorithm went burrowing through Blue J’s trove of data in search of the answer. One algorithm, following instructions developed for decoding the genome, looked to match strings of words in the clue with similar strings elsewhere, maybe in some stored Wikipedia entry or in articles about diplomacy, the United States, or northern climes. One of the linguists focused on rhymes with key words in the clue. Another algorithm used a Google-like approach and focused on documents that matched the greatest number of keywords in the clue, paying special attention to the ones that popped up most often.

While they the algorithms worked, software within Blue J would be comparing the clue to thousands of others it had encountered. What kind was it—a puzzle? A limerick? A historical factoid? Blue J was learning to recognize more than 50 types of questions, and it was constructing the statistical record of each algorithm for each type of question. This would guide it in evaluating the results when they came back. If the clue turned out to be an anagram, for example, the algorithm that rearranged the

letters of words or phrases would be the most trusted source. But that same algorithm would produce gibberish for most other clues.

What kind of clue was this one on diplomatic relations? It appeared to require two independent analyses. First, the computer had to come up with the four countries with which the United States had no diplomatic ties. Then it had to figure out which of those four was the farthest north. A group of Blue J's programmers had recently developed an algorithm that focused on these so-called nested clues, in which one answer lay inside another. This may sound obscure, but humans ask these types of questions all the time. If someone wonders about "cheap pizza joints close to campus," the person answering has to carry out two mental searches, one for cheap pizza joints and another for those nearby. Blue J's "nested decomposition" led the computer through a similar process. It broke the clues into two questions, pursued two hunts for answers, and then pieced them together. The new algorithm was proving useful in *Jeopardy!*. One or two of these combination questions came up in nearly every game. They are especially common in the all-important Final Jeopardy, which usually features more complex clues.

It would take Blue J almost an hour for its algorithms to churn through the data and return with their candidate answers. Most were garbage. There were failed anagrams of country names and laughable attempts to rhyme "north" with "diplomatic." Some suggested the names of documents or titles of articles that had strings of the same words. But the nested algorithm followed the right approach. It found the four countries on the outs with the United States (Bhutan, Cuba, Iran, and North Korea), checked their geographical coordinates, and came up with the answer: "What is North Korea?"

At this point, Blue J had the right answer. But the machine did not yet know that North Korea was correct, or that it even merited enough confidence for a bet. For this, it needed loads of additional analysis. Since the candidate answer came from an algorithm with a strong record on nested clues, it started out with higher-than-average confidence in that answer. The machine would proceed to check how many of the answers matched the question type: "country." After ascertaining from various lists that North Korea appeared to be a country, confidence in "What is North Korea?" rose further up the list. For an additional test, it would place the words "North Korea" into a simple sentence generated from the clue: "North Korea has no diplomatic relations with the United States." Then it would see if similar sentences showed up in its data trove. If so, confidence climbed higher.

In the end, it chose North Korea as the answer to bet on. In a real game,

Blue J would have hit the buzzer. But being a machine, it simply moved on to the next clue.

About the Author

Stephen Baker is the author of *The Numerati* (Houghton Mifflin Harcourt, 2008) and was previously a writer at *BusinessWeek*.