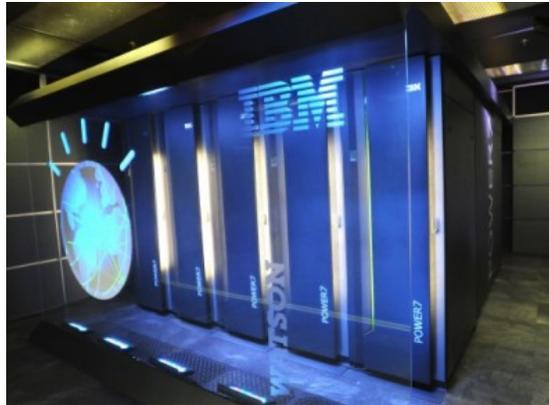


How IBM's Watson Computer Excels at *Jeopardy!*

By [John Rennie](#)

Posted: February 14, 2011



Watson, the *Jeopardy!*-playing computer system, runs on a cluster of IBM Power 750 servers. (Credit: IBM)

Over the next three evenings (Feb. 14th-16th), much of TV-watching America and any number of others will be watching the latest man-vs.-machine challenge as an IBM computer codenamed [Watson takes on two human champions in a highly publicized match of the game show *Jeopardy!*](#). The contest is in effect the long-awaited successor to the 1997 chess match in which another IBM computer, [Deep Blue, defeated between then-reigning grandmaster Garry Kasparov](#). Given how strongly Watson reportedly performs—it bested the human players in a preview match last month—I became curious about how exactly it does so well, since open-ended natural-language problems of the sort that *Jeopardy!* embodies have usually been seen as phenomenally difficult for computers to solve.

Finding detailed explanations has been a bit difficult because the IBM researchers behind the project have apparently not yet published a paper on their work. Nonetheless, I have pieced together a few helpful comments from those familiar with the project. The answers, I think, can help to put Watson's skills and the current state of artificial intelligence (AI) into perspective, especially with respect to benchmarks for AI such as the Turing test. More generally, they may also help to explain why, when it comes to AI, strong believers and skeptics may routinely talk right past each other.

To start, a sense of why *Jeopardy!* represents such a challenge to computers is useful. Here's [how IBM views the problem](#):

What does it take to win at Jeopardy!?

Jeopardy! is a game covering a broad range of topics, such as history, literature, politics, arts and entertainment, and science. *Jeopardy!* poses a grand challenge for a computing system due to its broad range of subject matter, the speed at which contestants must provide accurate responses, and because the clues given to contestants involve analyzing subtle meaning, irony, riddles, and other complexities in which humans excel and computers traditionally do not.

To win, it takes the deep analysis of large volumes of content to deliver high accuracy, confidence and speed. The best Jeopardy! players, according to our analysis, provide correct, precise responses more than 85% of the time. They also “know what they don’t know” and choose not to answer questions when they are unsure, since there is a penalty for being wrong.

So to play the game well, Watson needs to be able to parse human language with considerable sophistication as well as the relationships among real-world concepts—a tall order for a machine without sentience and intuitions about reality. It needs command over a vast wealth of knowledge that hasn't been explicitly defined ahead of time, but as all of us who have played along at home know, hints can be found in the definition of the categories and the phrasing of the clues in the game that can help to narrow down the range of answers.

Nevertheless, the answers are not simply sitting in Watson's data banks, waiting to be found: it will typically need to synthesize an answer based on pulling together various lines of thought suggested by the clue (while ignoring others that the game's producers may have inserted to make the problem more complex). As a matter of strategy, Watson also needs to be able to reflect on the state of its own certainty about an answer before it gives one. And it needs to be able to do all of the preceding very quickly: in virtually no more time than host Alex Trebek takes to ask the question. (On average, that means less than three seconds.)

In the words of the very nearly omniscient supercomputer Deep Thought from *A Hitchhiker's Guide to the Galaxy*, “Tricky.”

Watson does get one small break: perhaps contrary to appearances on TV, it does not need to use speech recognition to decipher what Alex Trebek says when he reads the clues. Rather, [according to Eric Brown](#), a research manager at IBM, as soon as the clue is revealed on the board, Watson begins interpreting a transmitted ASCII file of the same text. This allowance is not so much a break for Watson as it is a slight leveling of the playing field—after all, the human players do get to read the clues on the board as they're being read, too. After the computer has arrived at an answer and buzzed in, it does use speech synthesis hardware to give the answer that the players, Trebek and the audience will hear.

IBM designed a system of hardware and software optimized for the purpose of crushing the spirit of returning *Jeopardy!* champions Ken Jennings and Brad Rutter, or at least giving it a good shot. Racked into a space that is roughly equivalent to 10 refrigerators in volume, the system is a cluster of 90 IBM Power 750 servers, each of which contains 32 90 POWER7 processor cores running at 3.55 GHz, according to the company. This architecture allows massively parallel processing on an embarrassingly large scale: as David Ferrucci, the principal investigator on the Watson project, [told Daily Finance](#), Watson can process 500 gigabytes per second, the equivalent of the content in about a million books.

Moreover, each of those 32 servers is equipped with 256 GB of RAM so that it can retain about 200 million pages worth of data about the world. (During a game, Watson doesn't rely on data stored on hard drives because they are too slow to access.) As David Ferrucci, the principal investigator on the Watson project, [told Clive Thompson for the New York Times](#), they have filled that gigantic base of memory with uploads of “books, reference material, any sort of dictionary, thesauri, folksonomies, taxonomies, encyclopedias, any kind of reference material you can imagine getting your hands on or licensing. Novels, bibles, plays.”

Eric Brown explains how Watson tries to make sense of all the information at its digital fingertips in preparation for a game [this way](#):

Watson does not take an approach of trying to curate the underlying data or build databases or structured resources, especially in a manual fashion, but rather, it relies on unstructured data —

documents, things like encyclopedias, web pages, dictionaries, unstructured content.

Then we apply a wide variety of text analysis processes, a whole processing pipeline, to analyze that content and automatically derive the structure and meaning from it. Similarly, when we get a question as input, we don't try and map it into some known ontology or taxonomy, or into a structured query language.

Rather, we use natural language processing techniques to try and understand what the question is looking for, and then come up with candidate answers and score those. So in general, Watson is, I'll say more robust for a broader range of questions and has a much broader way of expressing those questions.

The key development is that Watson is organizing the information in its memory on its own, without human assistance.

The secret of Watson's performance resides within its programming at least as much as in its hardware and memory, however. How, indeed, does a machine solve the kinds of quirky puzzles posed by the *Jeopardy!* clues?

The nontechnical, hand-waving explanation, according to various sources, is that Watson doesn't have a single golden method for gleaning an answer. Quite the opposite, in fact: it uses its massive parallel processing power to simultaneously develop and test thousands of hypotheses about possible answers. It also evaluates each of these possible answers to reflect a state of "confidence" in its correctness—in effect, how well the answer seems to fit all the clues and how unambiguously associated inferences lead to that answer.

This evaluation process isn't rigid, however. Based on its experience with the game, Watson constantly learns more about how to interpret clues correctly, which alters how much confidence it assigns to all the answers under consideration. Watson will eventually "hit the buzzer" to give the answer for which it has the highest confidence, but only if that confidence level exceeds a pre-set 70 or 80 percent threshold. [**Correction (2/15):** Having watched the game, it's clear that the required confidence threshold can slide up or down and is not rigidly set. Watson seems to have some basis for determining what sufficient confidence is at least somewhat independently of the answers in hand. Often the confidence level does seem to be in the 80s, but it goes up and down. The principle described here still generally applies, though.]

Technology Review offered an example from Ferrucci about [how Watson would extrapolate](#) from this sample *Jeopardy!* clue: "It's the opera mentioned in the lyrics of a 1970 number-one hit by Smokey Robinson and the Miracles."

...Watson can identify "Pagliacci" as being "an opera," although this on its own would not be much help, since many other passages also identify opera names. The second result identifies a hit record, "The Tears of a Clown," by "Smokey Robinson," which the system judges to be probably the same thing as "Smokey Robinson and the Miracles." However, many other song titles would be generated in a similar manner. The probability that the result is accurate would also be judged low, because the song is associated with "the '60s" and not "1970." The third passage, however, reinforces the idea that "The Tears of a Clown" was a hit in 1970, provided the system determines that "The Miracles" refers to the same thing as "Smokey Robinson and the Miracles."

From the first of these three passages, the Watson engine would know that Pagliacci is an opera about a clown who hides his feelings. To make the connection to Smokey Robinson, the system has to recognize that "tears" are strongly related to "feelings," and since it knows that Pagliacci is about a clown that tries to keep its feelings hid, it guesses—correctly—that Pagliacci is the answer. Of course, the system might still make the wrong choice "depending on how the wrong answers

may be supported by the available evidence,” says Ferrucci.

It's easy, Ferrucci says, for less sophisticated natural-language systems to conclude that “The Tears of a Clown” is the answer by missing the fact that the request was for an opera referenced by that song. Such a conclusion could be triggered by passages that have lots of keywords that match the question.

This approach to understanding and solving natural-language problems is an outgrowth of IBM's work on what it calls its DeepQA effort (where QA stands for “question answering”). Here's an excerpt from IBM's FAQ on the subject?

How does DeepQA's approach compare to purely knowledge-based approaches?

Classic knowledge-based AI approaches to Question Answering (QA) try to logically prove an answer is correct from a logical encoding of the question and all the domain knowledge required to answer it. Such approaches are stymied by two problems: the prohibitive time and manual effort required to acquire massive volumes of knowledge and formally encode it as logical formulas accessible to computer algorithms, and the difficulty of understanding natural language questions well enough to exploit such formal encodings if available. Consequently they tend to falter in terms of breadth, but when they succeed they are very precise.

<...>

The DeepQA hypothesis is that by complementing classic knowledge-based approaches with recent advances in NLP, Information Retrieval, and Machine Learning to interpret and reason over huge volumes of widely accessible naturally encoded knowledge (or “unstructured knowledge”) we can build effective and adaptable open-domain QA systems. While they may not be able to formally prove an answer is correct in purely logical terms, they can build confidence based on a combination of reasoning methods that operate directly on a combination of the raw natural language, automatically extracted entities, relations and available structured and semi-structured knowledge available from for example the Semantic Web.

As brilliantly as this approach seems to work in general, it leaves Watson with a curious mix of strengths and weaknesses that will no doubt surface in the *Jeopardy!* tournament. For example, Watson is surprisingly worse than human players when the clues are extremely short—say, only one or two words. As [CNN learned from Stephen Baker](#), the author of the forthcoming newly published book [Final Jeopardy: Man vs. Machine and the Quest to Know Everything](#) about Watson's game-show challenge, if the category were “First Ladies” and the clue were “Ronald Reagan,” most Americans could jump to the response “Who is Nancy Reagan?” before Watson could finish developing hypotheses about what the clue meant. Conversely, Watson is extremely good at solving from clues that involve some level of brute force searching through its memory for answers that meet multiple criteria.

In the interest of brevity (too late!), I'll close this post here. My next one will take up the question of [what Watson's success as a Jeopardy! expert means for the current and future development of AI](#).

[Some corrections and updates to the further readings added since this was first posted.]

For more information:

- [FAQs](#) on “Watson and Jeopardy” and IBM's DeepQA project, prepared by the company.
- [“How Watson works: a conversation with Eric Brown, IBM Research Manager,”](#) by Amara D. Angelica (KurzweilAI.net, Jan. 31, 2011).