Watson and healthcare

How natural language processing and semantic search could revolutionize clinical decision support

Michael J. Yuan

IBM Watson's stellar performance in the Jeopardy! show captured the world's imagination. The first real world application for Watson involves healthcare. How does Watson address issues that previous generations of tools have not been able to address? What are the technical approaches Watson takes to set it apart from other systems? This article answers how Watson takes on those questions and gives you a sneak peek of the technology behind Watson based on scientific papers, specifications, articles published by the IBM team, and interviews with university collaborators.

“I, for one, welcome our new computer overlords” – Ken Jennings, 74-time Jeopardy! Champion.

On Feb 16 2011, the IBM supercomputer, Watson, defeated two human all-time champions in the game Jeopardy! on national TV. Jeopardy! is a quiz game that is uniquely human. The questions are often nuances with puns, irony, and humor. It is a remarkable feat that a computer can even play this game, let alone beating human champions! Is the age of artificial intelligence finally upon us after over 50 years of disappointments? More specifically, can Watson's intelligence be used to advance science and business outside of a game show? As with Deep Blue before it, Watson started as a public demonstration of state-of-the-art technology, but it may well make significant impact to the society at large in the coming years. So, what are those real world applications for Watson?

It seems that Watson's very first real world application is going to be in healthcare. Potentially, by answering questions for physicians at the point of care, it could help improve healthcare quality and reduce costly errors. In this article, I will discuss how the DeepQA technology behind Watson can be used to solve specific problems in healthcare. All information presented in this article is based on scientific papers published by the Watson research team and public interviews given by IBM executives. IBM is still formulating exactly how it will apply DeepQA to the medical domain.

I would like to specifically thank Dr. Herbert Chase, a professor of Clinical Medicine at Columbia University College of Physicians and Surgeons, who is a key collaborator with IBM on Watson's application in clinical decision support.
Does Watson have true “intelligence”?  
One of the best ways to judge artificial intelligence is the Turing test proposed by Alan Turing in 1950. The test is to have human judges engage in natural language conversations with another party that may be a human or a machine. If by the end of the conversations, the judges cannot reliably agree on which party is human and which party is machine, the machine is said to have passed the Turing test and demonstrated artificial intelligence. While Watson can answer human questions very well, it cannot engage in long conversations with a human. So, no, Watson has not met Turing’s standard for true artificial intelligence. One can argue that Watson merely searches for an answer. It does not have the intelligence to actually understand the question or the answers.

However, from the practical point of view, Watson helps humans solve problems. If it is not strictly artificial intelligence (AI), Watson is certainly intelligence argument (IA) that extends human brains.

A brief history of clinical decision support systems  
For 40 years, clinical decision support systems (CDSS) have promised to revolutionize healthcare. In fact, when the government recently mandated electronic health record (EHR) systems in all healthcare facilities, one of the key objectives is to promote better and cheaper healthcare using CDSS based on the patient data collected from the EHRs. With the large amount of new data collected by the newly installed EHR systems, computers like the Watson will be able to find optimal answers to clinical questions much more efficiently than the human mind.

Two major categories of CDSS are diagnostic support tools and treatment support tools. Diagnostic support helps physicians make a better diagnosis based on the patient symptoms, medications, and medical records. Diagnostic error is the number one cause of malpractice lawsuits against healthcare providers (see Related topics). Therefore, helping physicians avoid common cognitive errors and make better diagnoses is a priority. Treatment support, on the other hand, helps clinicians stay compliant with known treatment guidelines such as avoiding known drug interactions, dispensing the right medication to the right patients, and staying on schedule with catheter changes.

The first generation of CDSS focuses on diagnostic support. Differential diagnostic tools, like the DXPlain, use the Bayesian inference decision process to take into account one piece of clinical finding (such as a symptom or lab result) at a time, and then calculate the statistical probabilities of various potential diagnoses. The knowledge bases of such system are large arrays of prior probabilities linking clinical findings to diagnoses. The issue with these first generation tools is that physicians rarely have time to sit in front of a computer, sift through medical records, and enter one piece of finding at a time into a computer. Then, after several likely diagnoses emerge, the physician has to research potential treatment options. That is especially a problem now since primary care physicians only spend fifteen minutes per patient.

The second generation of clinical decision support tools aims to improve the workflow and ease-of-use. A representative product in this category is Isabel. Isabel has two major innovations. First, it takes a natural language patient summary from the physician notes in the EHR, identifies keywords and findings contained in the summary, and then generates a list of related diagnoses from its probabilities database. Second, Isabel indexes published medical literature for treatment
options for each diagnosis, and it makes treatment suggestions to the physician together with diagnoses. The natural language process in both parsing the physician notes and medical literature is a significant innovation that makes Isabel appealing.

However, even with Isabel, it is still often too slow to extract physician notes electronically, and then search for answers. A study indicated that Isabel's diagnoses are accurate when large paragraphs of text are used, but the accuracy drops significantly when the input contains less text (see Related topics).

For a trained medical professional, a better way to resolve a puzzling finding or to find a new treatment is often simply to ask a more experienced clinician.

The power of simple Q&A

According to an observational study published in 1999 by BMJ (British Medical Journal) a team of researchers observed 103 physicians over one work day. Those physicians asked 1,101 clinical questions during the day. The majority of those questions (64 percent) were never answered. And, among questions that did get answered, the physicians spent less than two minutes looking for answers. Only two questions out of the 1,101 triggered a literature search by the physicians attempting to answer them. Hence, providing quick answers to clinical questions could have major impact in improving the quality of healthcare (see Related topics). Enter Watson.

People often ask, doesn't Google already do that? Sure, you can enter a clinical finding or a diagnosis into Google, and search for answers. In fact, a doctor used Google as a diagnostic aid in high profile medical case published in the New England Journal of Medicine (see Related topics). However, Google is fundamentally a keyword search engine. It returns documents not answers.

The “Paris Hilton” problem

A good illustration of Google's problem with natural language queries is known as the “Paris Hilton problem”. If someone types "Paris Hilton" into the search engine, Google does not know if the person is looking for a hotel in France or looking for celebrity gossip. The only way to answer such questions is for the computer to "understand" the context through analysis of the natural language.

- Google does not understand the question. The physician is responsible for parsing the question into keyword combinations that would yield the right Google results. Aside from the simplest factoid questions, this proves to be a difficult task. In fact, there are whole books on how to optimize search queries to make the most out of Google.
- Google finds millions of documents for each query, and orders the results by keyword relevance. The user needs to read the document and parse the meaning based on context, and then extract a list of potential answers.

Hence, while Google is tremendously useful, especially in answering factoid questions, it suffers the same problem as CDSS tools that come before it: it simply requires too much time and mental energy from the physicians to make it useful as an everyday decision support tool.

In a 2006 study published by BMJ, two investigators went through a whole year of diagnostic cases published in the New England Journal of Medicine, and evaluated whether a trained
professional can derive a diagnosis by simply evaluating Google search results. Note that the human investigator must look at the cases to construct search queries and then go through the Google results to identify potential diagnoses – a labor intensive and time consuming process. The answer is that they can come across the correct diagnoses in 58 percent of the cases (see Related topics). The hope for Watson is that it would be able to improve those percentages while saving the human clinician time and effort.

For a good analysis of Google versus Watson in answering the type of questions posed on Jeopardy!, see Danny Sullivan's writing in the Related topics section.

Watson in action

To see the kinds of questions Watson can answer, check out the two example questions Dr. David Ferrucci showed to German Chancellor Merkel and Turkish PM Erdogan at the CeBIT 2011 Opening Ceremony. Try typing those questions into Google to see how hard it is to parse the answer from a list of documents, and how easy it is to miss less frequent diagnoses.

1. Question: Streptococci cause this childhood “fever” characterized by a bright red rash and high temperature.
   Answer: 98% Scarlet fever, 15% Rheumatic fever, 8% Strep throat

2. Question: This disease can cause uveitis in a patient with family history of arthritis presenting circular rash, fever, and headache.
   Answer: 76% Lyme Disease, 1% Behcet's Disease, 1% Sarcoidosis

From the CDSS workflow point of view, we need to add a natural language and semantic layer on top of a search engine like Google, so that the computer can actually answer questions. That is exactly what Watson does.

Furthermore, Watson evaluates each potential answer based on evidence it can gather through refined secondary search. That allows Watson to give a confidence level for each answer. That is crucial for a medical Q&A system since it guards against a very common type of cognitive error physicians make—premature closure. Premature closure happens when a physician forms and accepts a diagnosis and fails to consider plausible alternatives in the face of new evidence once the diagnosis. For example, when a patient walked into the office complaining of chest discomfort after a big meal, the physician diagnosed heartburn and prescribed simple medication for heartburn. But, when the patient later deteriorated and demonstrated clear signs of heart attack, the physician was unable to consider the possibility of a heart attack because he was puzzled why the heartburn medication did not work and ended up prescribing more heart burn medications to the patient. This type of diagnostic error happens when the physician is "anchored" to the wrong conclusion. In this case, a question to the Q&A system asking why the heartburn medication did not work could save lives. Watson could do a great job in reminding physicians to consider low probability but potentially severe cases.

The language of Watson

From the technical point of view, Figure 1 shows the steps Watson goes through to answer a question. In summary, the steps are:

1. Watson parses the natural language question to generate a search query.
2. Watson's embedded search engine searches a large document knowledge base to find related documents.

3. Watson parses the natural language based search results and generates potential answers (the hypotheses).
   a. For each hypothesis, Watson constructs and initiates another search to collect evidence that supports this hypothesis.
   b. Watson's embedded search engine searches supporting evidence for each hypothesis.
   c. The search results are again parsed and each piece of evidence is scored for its strength.
   d. Each hypothesis is now assigned a score based on the strength of all of its supporting evidence.

4. The hypotheses are turned into a list of answers returned to the user.

The key tasks performed by Watson include natural language processing (steps 1, 3, and 3c), searching (steps 2, 3a, and 3b), and evidence scoring (steps 3d and 4). (View a larger version of Figure 1.)

**Figure 1. The workflow Watson goes through to answer a question**

Fundamentally, both the natural language parsing and evidence scoring are processing and evaluating unstructured text documents. Inside Watson, software components based on the Apache UIMA (Unstructured Information Management Architecture) project perform those tasks. Watson generates multiple search queries for each question, and uses a variety of different search techniques to find hypotheses or evidence from the knowledge base. Search technologies used in Watson include Apache Lucene (term frequency to rank results), Indri (Bayesian network to rank results), and SPARQL (search relationships between terms and documents). See Related topics.
Watson and open source software

For all the sophistication and advanced algorithms that go into Watson, the Watson system is largely based on freely available open source software. Apache UIMA, Apache Lucene, Indri, and SPARQL and the Jena Toolkit are all open source software. Specifically, Watson uses open source software as basic frameworks but builds on proprietary algorithms as components that plug into the framework. Open source software has come a long way to be a key enabler of software innovation.

Watson's way of reasoning is to generate hypotheses (that is, candidate answers) from a large body of documents, as opposed to from pre-conceived theories as humans typically do. In fact, a major trend in scientific research is to “mine” discoveries from data. While Watson is trying to emulate human intelligence, humans seem to think more like Watson too! See the Related topics section for an excellent article in Wired magazine for more on this issue.

Apache UIMA

The Apache UIMA project is an open source implementation of the OASIS UIMA specification. UIMA provides a scalable architecture framework to run text processing applications like Watson.

A key feature of the UIMA framework is that it allows applications (called components in UIMA terms) to be chained together. This way, each application component can focus on one text processing task, and pass the results to the next component in the chain to do more work. That is ideally suited for the Watson workflow outlined earlier in the article.

Furthermore, UIMA provides a parallel processing framework, called UIMA-AS, that allows multiple components to be executed concurrently. For Java™ developers, the UIMA-AS framework is based on another Apache open source framework, ActiveMQ, which uses Java Messaging Services to facilitate communication between tasks asynchronously. In the case of Watson, once a set of hypotheses is generated, the computer should be able to independently collect and score evidence for each hypothesis independently. For example, all the parallel lines in Figure 1 represent tasks that can be processed concurrently by multiple CPUs. UIMA-AS is the reason Watson can leverage 2880 CPUs to come up with Jeopardy! answers within 3 seconds. This architecture also allows UIMA deployments to scale back when needed. For example, a “physician assistant” application might not need to give answers within seconds, and hence the hardware requirement for such an application could be substantially less than Watson.

From an application developer's point of view, writing a UIMA application mostly consists of annotators. An annotator is a Java or C++ class that can take a piece of input text and extract structured information from it. The UIMA documentation has an excellent tutorial on how to write an annotator to extract addresses and room numbers of free text using regular expressions.

Obviously, natural language processing requires more than just extracting structured text via regular expressions. A large body of algorithms has been developed to detect, tokenize, tag, categorize, and parse words, phases, and sentences in a natural language paragraph to extract its meaning based on context. A key strength of the open source framework approach of UIMA is that it encourages other developers to contribute annotators that implement those known algorithms and hence make life easier for new developers. Several commonly used annotators
are already included in the standard UIMA distribution to help developers get started quickly. The UIMA website hosts several large repositories of annotators available for download, including the OpenNLP (Open Natural Language Processing) annotators and IBM Semantic Search annotators. The IBM Semantic Search annotators allow you to search a document repository for all documents that are authored by a person with a specific name.

Through the development of Watson, the IBM team developed many annotators on UIMA. In fact, a large amount of work went into developing scoring functions to evaluate the validity of each piece of evidence extracted from the search result documents. According to the Watson team, more than 100 scoring functions were developed for the Jeopardy! project, and those UIMA annotators are the reason Watson can correctly identify a single correct answer from a large volume of documents returned by the search. Like Google, Watson’s secret is not in how it finds the answers, but in how it scores and ranks the answers so that the most likely ones come out at the top.

### Software frameworks

Most software developers are familiar with the concept of frameworks. In the world of Java development, there are hundreds of widely used frameworks from web to data modeling. In the world of C++, the Standard Template Library (STL) is a well known framework. The framework provides an API (application programming interface) for applications. Applications that are built on a framework follow the architectural patterns provided by the framework, and delegate many of the mundane tasks to the framework libraries.

### Apache Lucene

As we described earlier, Watson searches large document databases to generate hypotheses and then finds supporting evidence for each hypothesis. One of the key search engines used to index and search those unstructured text documents is Apache Lucene.

Apache Lucene is a fully featured free text indexer and search engine written in Java. It provides a set of simple APIs that allow developers to easily embed the search engine into their own applications. The developer can customize how the documents are indexed and scored for relevance. Lucene also supports a rich query language that resembles the Google query language (the operators that you can type into the Google search box).

Lucene is embedded inside Watson to index the large documentation repositories: a trivia database and Wikipedia articles for Jeopardy! and medical publications for the healthcare application. The UIMA annotators invoke Lucene as needed to search the document base at various stages of the question answering process. Examples of such medical databases include peer reviewed medical publication repositories such as Medline, as well as official treatment guidelines such as the Agency for Healthcare Research and Quality (AHRQ) National Guideline Clearinghouse.

Lucene not only indexes free text in documents based on word frequency, it can also store structured information extracted from the document by UIMA. By storing structured information, the documents are searchable through metadata (such as the author name or whether it contains an address in New York City). As UIMA processes natural language text in the document database, it
creates common analysis structure (CAS) objects, which contain the structured results extracted from the document, type system, and indices. The Lucene CAS indexer (Lucas) is a standard annotator shipped with UIMA. Lucas saves CAS information into Lucene index files.

UIMA and Lucene work together to form the analytics and knowledge engine for Watson. Even if Watson's annotators and algorithms are not open source, the successful demonstration at the Jeopardy! show validates the overall approach, and provides a path for developers to create similar applications in specific domains.

**Future of natural language processing in healthcare**

Natural language processing in healthcare goes beyond questions and answers. In 2008, the prestigious scientific journal *Nature* published a special issue on how medical research is entering an area of big data. The scientific discovery process is shifting from theory, hypothesis, experiment, and proof, to mining the data directly for conclusions. While the *Nature* issue focuses on genomic data as the target of big data mining, we can argue that natural language text data is also an invaluable source for research.

For instance, as EHR systems are adopted by government mandate, physician notes are digitized in a computer readable format. It is a large repository of information to mine for symptom indicators, treatment efficacy, and potential medical errors. In fact, the Mayo Clinic and IBM have already announced a partnership to open source much of the UIMA annotators Mayo developed to mine its own medical records.

Mining patient reported data is another interesting area. Patient communities such as PatientsLikeMe and Association of Cancer Online Resources, Inc. (ACOR) have collected vast amounts of email, forum posts, blog posts, and self monitoring data from patients. Important research has already been done based on that data to identify drug adverse effects that are not uncovered in FDA trials, as well as comparative studies on treatment efficacies. Using natural language tools, we can take these kinds of research to a new level.

Using open source software and off-the-shelf hardware, Watson has shown us what can be done. The R&D effort around Watson has already started to pay dividends to the developer community, in the form of IBM contributions to UIMA, UIMA-AS, and related modules. It is now up to developers to write innovative applications to take advantage of these capabilities!
Related topics

- Clinical Decision Support Capabilities of Commercially-available Clinical Information Systems is a nice review article on how the state-of-art of CDSS in EHR systems by Wright and Sittig et al.
- Analysis of questions asked by family doctors regarding patient care is a 1999 observational study published by BMJ.
- A paper published in 2005 analyzed records from the VA hospitals and determined that diagnostic errors is the number one cause of litigation against hospitals.
- DXplain is one the first and most widely used diagnostic decision support system based on Bayesian inference.
- Bayesian inference is a fundamental approach to all differential decision support systems.
- Indri is a new search engine from the Lemur project; a cooperative effort between the University of Massachusetts and Carnegie Mellon University to build language modeling information retrieval tools.
- Learn more about how SPARQL and the Jena Toolkit open up the semantic web from this developerWorks article by Philip McCarthy.
- Isabel represents a second generation of diagnostic and treatment decision support tools with emphasis on natural language processing, searching, and workflow efficiency.
- A simulated study published in 2008 has indicated that Isabel is accurate if a large amount of text is copied and pasted into it. But its accuracy deteriorates quickly if only limited information is available.
- The End of Theory: The Data Deluge Makes the Scientific Method Obsolete, published by Wired Magazine, is a good article on how the scientific discovery process is becoming increasingly data driven, much like Watson's reasoning process.
- Final Jeopardy!: Man vs. Machine and the Quest to Know Everything is Stephen Baker's fascinating book on the story and people behind Watson.
- IBM research has a three part blog series on the inner works of Watson. Part one is How Watson “sees,” “hears,” and “speaks” to play Jeopardy!. Part two is Knowing what it knows: selected nuances of Watson's strategy. Part three is Watson's wagering strategies.
- Two physicians investigated how effective Google is as a diagnostic support tool. The result is that, with human help in constructing the search query and interpreting the search results, a Google search can reveal the correct diagnosis for 58 percent of cases they evaluated. See Googling for a diagnosis—use of Google as a diagnostic aid: internet based study.
- Dr. David Ferrucci together with other members of the Watson team wrote an article for the AI Magazine to discuss the technical approaches behind Watson. See Building Watson: An Overview of the DeepQA Project.
- The Apache UIMA project and OASIS UIMA specification are the core of Watson's technical infrastructure.
- UIMA component repositories provide ready-to-use UIMA annotators for various data analysis tasks.
• The IBM Semantic Search UIMA annotator is available from IBM alphaworks.
• Apache Lucene project is a highly efficient and fully featured search engine that can be embedded into your own applications. Watson uses Lucene to index and search documents in its knowledge base.
• Nature’s special issue on big data points way to the future of biomedical research.
• The Open Health Natural Language Processing (OHNLP) Consortium is a joint effort by IBM and Mayo Clinic to foster natural language processing of EHR records. It releases open source UIMA components contributed by IBM and the Mayo Clinic.
• Patient communities such as PatientsLikeMe and ACOR have a lot of patient-generated natural language materials that can be mined for research and treatment.