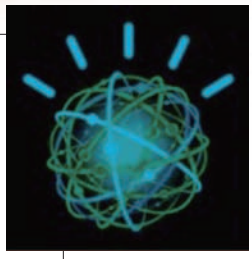
解説

# Watson：クイズ番組に挑戦する質問応答システム



金山　博　武田浩一
日本 IBM 東京基礎研究所

## Watson プロジェクトの始まり

Wikipedia には，2004 年に当時 IBM リサーチ部門のソフトウェア分野の責任者であった Charles Lickel がレストランでクイズ番組への挑戦に思い至った非常に興味深いエピソードが紹介されている☆1．米国政府の質問応答システム開発プロジェクトを IBM でリードしていた David Ferrucci が，2007 年に以下で説明する「グランド・チャレンジ」として正式にこの挑戦への取り組みを開始したことが Watson プロジェクトの始まりであった．

### ➤ グランド・チャレンジ

IBM リサーチ部門には，「グランド・チャレンジ」と呼ばれる，非常に難しい技術的な問題を設定し，その解決に向けた研究に投資するというプログラムがあり，世界中の IBM 研究員が，新たな難問のアイディアを出し合っている．その典型例が，1997 年にチェスで人間のチャンピオンに勝利したスーパーコンピュータ，Deep Blue である．

グランド・チャレンジの主な意義は，学術的進歩への貢献である．大きな課題を設定すると，その部分問題が生まれ，それらの進歩を目に見える形で評価できるようになり，研究が加速される．そのために，要素技術や結果を学術機関と共有するように努めている．また，情報科学の専門家でない一般の人々にとっても挑戦の内容が理解できるテーマが設定されている．さらに近年のグランド・チャレンジでは，ビジネスに活用できる基礎研究への投資という企業戦略を反映して，実世界への応用シナリオが同時に考案されるようになってきた．Watson の研究が重視された最大の理由は，近年の情報爆発時代を象徴するテキスト情報（非構造情報）の驚異的な増加を大きな価値に転換できる有望な技術と位置づけられたためである．


図 -1　Jeopardy! のパネル

### ➤ クイズ番組 Jeopardy! とは

Jeopardy! は，米国で 40 年以上の歴史を持つクイズ番組で，歴史・科学・文学・スポーツなどの幅広い知識を問われる．3 人の回答者が獲得金額を競う形式で，最初に 1 人が図 -1 のようなパネルの中から，カテゴリ（分野）と金額（難易度に応じて 5 段階）を指定すると，問題文が表示される．ボタンを

☆1　http://en.wikipedia.org/wiki/Watson_%28computer%29（2011 年 4 月現在）．

| | |
|---|---|
| 1 | カテゴリ：Dialing for Dialects<br>問題文：While Maltese borrows many words from Italian, it developed from a dialect of this Semitic language.<br>（マルタ語はイタリア語から多くの語彙を借りているが，それはこのセム語系言語の方言から発展した）<br>答え：Arabic（アラビア語） |
| 2 | カテゴリ：Alternate Meanings<br>問題文：4-letter word for the iron fitting on the hoof of a horse or a card-deal ing box in a casino.<br>（馬のひづめに付ける金具，またはカジノでカードを入れる箱を表す 4 文字の語）<br>答え：　Shoe |

表 -1　Watson が対戦で正答した問題の例

図 -2　Watson の筐体

最初に押した人が回答し，正解すれば該当する金額が得られ，不正解なら同額を減らされる．出題と回答は非常に速いペースで進行し，早押し問題 30 問を 2 セット行った後に，最後に 1 問だけ全員が答える筆記形式の問題（その時点での各自の持ち点を賭ける，いわば決勝問題）があり，合計 61 問が 1 つのゲームの単位である．

**表 -1** に，カテゴリと問題，それに対する答えの例を挙げる．まず，カテゴリ自体が複雑で，単に問題のジャンルを指定するというよりは，何らかのヒントや示唆を間接的に与えるようなものになっている．問題文も「米国の初代大統領は？」といった類の単純なものではなく，やや長めの英文で指定されており，問題文を予測して答えを用意しておくことは不可能である．

これらの問題は，情報の蓄積によって解けるような一般的な知識を問う問題が主である☆2．また，問題は最初に全文が一度に表示される☆3．したがって，本質的な質問応答に注力することができ，グランド・チャレンジの題材としてふさわしいといえる．

Watson は，人間の対戦相手と一緒にステージに立つ．問題文とカテゴリは司会者の声を認識するのでなく，テキスト情報として Watson に伝送される．前述の通り，問題文は画面にも表示されるため，対戦相手と同条件で問題を解いているといえる．すべての処理は背後の部屋に設置された**図 -2** のようなコンピュータ（POWER7 の 2,880 コア）で行われ，インターネットには接続されていない．Watson が回答する際はシリンダーに入ったボタンを物理的に押し，音声で答えを読み上げる．そのほか，パネルの選択や賭け金の設定など，ゲーム進行上の判断は人手を介さずに自動的に行われる．

## ➤ 克服すべき主要な技術的課題

Jeopardy! で人間と対戦するために必要なのは，「カテゴリ」と「問題文」とを入力として受け取り，「解答」を出力するシステムである．このようなタスクは従来から「質問応答（question answering）」と呼ばれており，さまざまな研究がなされてきた[2]．これは一般的に言われる「インターネット検索」とは本質的に違う．インターネット検索では，入力は問題文に含まれる数個のキーワードで，それにヒットする Web ページが出力として返される．質問応答では，特定の名詞や固有名詞の返答が要求されるが，検索結果の文書から正しい語句を探すのは人手でも容易ではない．

Jeopardy! に挑戦するうえでの主な技術的課題は，以下の 5 つに分類できる．個々の課題が従来の研究にはない難しさであるうえ，これら 5 つをすべて解決しないと人間と対戦できるレベルに至らないことは明白であった．

☆2　たとえば「今何問目？」といった，クイズの状況に特化したような質問への対処は不要である．

☆3　司会者による読み上げは，番組の視聴者向けの補助的なものであるため，早押し形式といえども，問題文の最初の部分だけを聞いて答えを先読みする必要はない．

❍ **幅広い分野への対応**

化合物の特徴や，乗り換え経路の探索など，あらかじめ限定された分野の問題であれば，エキスパートシステムのような規則の生成，オントロジー（概念体系）の構築といったアプローチによりある程度は解決できる．一方，Jeopardy! の問題のようにあらゆる分野が対象となる状況では，必要な知識の幅と量は莫大なものとなり，従来とはまったく違った情報の整理が必要となる．

❍ **問題文とカテゴリの解釈**

上記で見た通り，問題文は人間がコミュニケーションに使う言語(英語)で記述され，カテゴリは人間向けに何らかの連想をさせるような曖昧な単語列である．すなわち，システムへの入力がチェスの盤面のように明確に数値化されておらず，問われているものが何か，ヒントをどのように使うかという，入力の意味解釈が求められる．

❍ **高い正答率での回答**

Jeopardy! で過去に勝利したチャンピオン回答者の正答率は非常に高く(本番の対戦の際にも Watson を含む 3 回答者の正答率は 88 ～ 89% であった)，さまざまな分野の問題に対してこの正答率を実現するような高精度の手法が必要とされる．

❍ **確信度の推定**

誤答をすると減点となるため，どれだけの自信を持って問題に回答できるかを見積もったうえでボタンを押すという判断をする必要がある．人間は，直感的に「この問題を知っている」ということを察した上でボタンを押すが，コンピュータにとって「自分が知っているか知らないか」を知ることは難しく，インターネット検索など多くのタスクでも考慮されてきていない点である．

❍ **応答速度**

司会者が問題文を読み終えるまでの 2 ～ 3 秒の間に答えの導出と確信度の計算を終えないと，たとえ正解が求められても他の回答者に先にボタンを押されてしまう．すなわち，許された時間の範囲内で高い正答率を実現する計算処理，という難しい設計が求められる．

### ➤ 質問応答技術の歴史

Watson の背景となる質問応答の技術は，大学などの多くの研究機関で以前から研究されてきた．この分野では技術向上と情報共有を目的として，MUC（Message Understanding Conference），TREC（Text Retrieval Conference），日本では国立情報学研究所による NTCIR（NII Test Collection for IR Systems）といった評価型会議が行われ，参加者が実装したシステムの性能が，共通のタスクとベンチマークデータを利用して評価・比較されてきた．Jeopardy! のように出題分野を限定しない場合は，オープンドメイン質問応答と呼ばれる．これはアメリカ国立標準技術研究所（NIST）が主催する上記 TREC の第 8 回会議で 1999 年に初めてタスクとして設定されたもので，ファイナンシャル・タイムズやロサンゼルス・タイムズといった報道記事を主な情報源として質問応答技術が評価され，以後この分野の進展に大きな貢献をした．

質問応答では，回答として単純な対象物やイベントを要求するものをファクトイド（factoid）型質問と呼び，作曲家の作品のような集合や，語の定義，理由・原因などを要求するものを非ファクトイド型質問として区別している．したがって，Jeopardy! のクイズはオープンドメインのファクトイド型質問に分類される．Watson 開発の中心となった IBM 研究者たちは，2002 年の TREC におけるこの質問応答タスク向けに，従来の単一の回答生成プロセスを多重化した PIQUANT[3] と呼ばれるシステムを開発した．PIQUANT は TREC で上位の成績を収めることができ，Watson 開発のベースとして利用されたが，次章で述べるようにその性能は要求されるレベルと大きな隔たりがあった．

## Watson 開発の過程

Watson の研究開発を進めるうえで，高い精度で解を生成する手法の実現はもちろん，最初に性能評価の指標を定めた点が重要であった．以下ではこの 2 点を中心に，実装したシステムの動作，情報源の

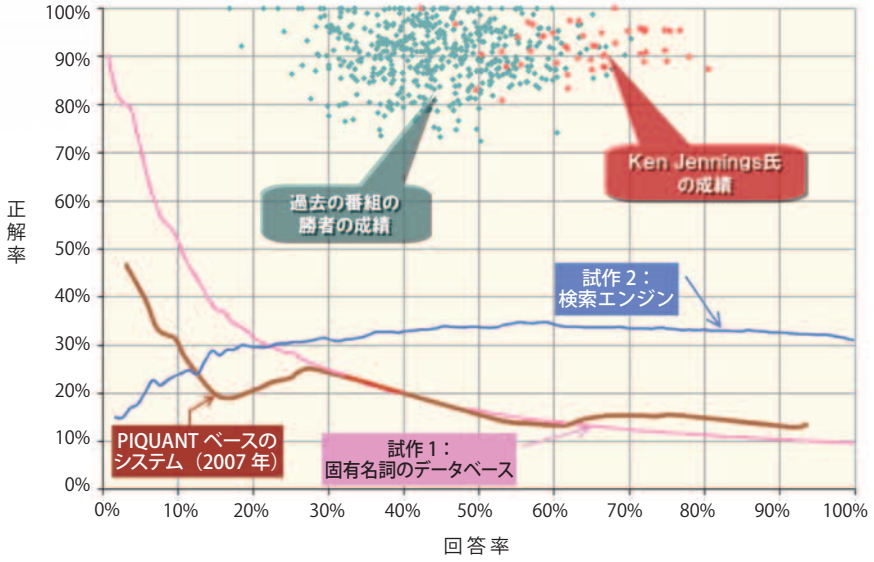



図-3　回答率と正解率のグラフ．上部の点は過去の番組における勝者のゲームごとの成績を表す．曲線は，2007年時点のPIQUANTシステムと，最初に試した2つの手法の性能（文献1）より引用）．

作り方，さらに，大規模システムを実装するために採用したプラットフォームについても説明する．

## ➤ 評価の指標

数年にわたる研究開発において，その時々のシステムの性能を客観的に測ることは重要である．そのために考案されたのが，「回答率（どれだけの質問に答えようとしてボタンを押すか）」「正解率（押したときにどれだけの割合で正解するか）」という2軸の指標である．これにより，開発中のシステムの性能を目標値と比較し，システムの着実な性能向上を確認することができた．**図-3**は，過去のJeopardy!の番組で勝利した回答者の成績と，出発点となるいくつかの手法を比較したものである．グラフ上部の点はJeopardy!の勝者のゲームごとの成績を示す．その中でも，74連勝の記録を持つKen Jennings氏は，特に高い回答率（赤の点）を示している．

茶色の線は，2007年にPIQUANTをJeopardy!向けに適用したシステムの性能を示す．すべての質問に答えようとしたときの正解率はわずか1割強，回答に自信を持てる1割の質問だけに限定して回答したときも，正解したのはそのうちわずか3割であり，番組で勝利した人たちの成績には遠く及ばなかった．この差こそが，グランド・チャレンジとしての課題の難しさを物語っている．

## ➤ 手法の検討

Jeopardy!において人間と互角の性能を得るためには，どのようなアプローチをとればよいかを知るために，基本的な手法の試作から始めた．図-3には2つの手法のグラフを加えてある．

1つ目は，固有名詞のデータベースを作って，問題文中のキーワードと符合するものを探す方法である．しかし，事前にデータベース化できる語は限られており，限られた問題には高い正解率が出せるものの，4割の問題に答えようとすると正解率は20％まで落ち込む．

2つ目は，インターネットの検索を用いる方法である．問題文中のキーワードを使って検索して，そのスコアを確信度として用いる方法を試みた．この場合，多くの文書を対象にすれば30％の問題には答えられることが分かったが，それ以上の正解率向上は望めないことが分かった．

したがって，これらのアプローチでは人間と対戦できるレベルに達することは困難である．この限界を破るために，蓄えられた大量の情報を整理して，問題と答えとの関係を多角的に調べる方法が考え出された．それが次に述べるDeepQAフレームワークである．

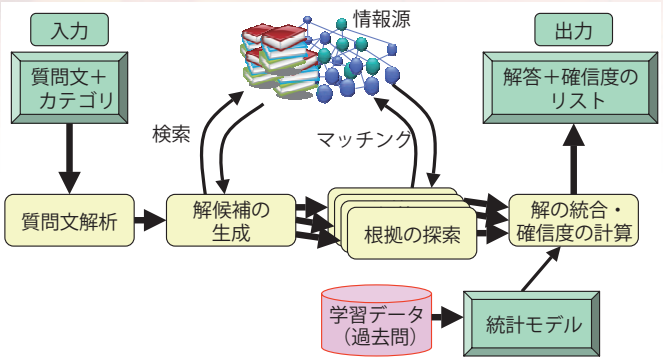



図 -4　DeepQA フレームワークの概略

## ➤ DeepQA フレームワーク

DeepQA とは，Watson を実現するために設計された質問応答の仕組みであり，「情報源と統計情報をもとに，仮説の生成と根拠の探索を行う」という点が特徴である．質問文とカテゴリを入力して，解答と確信度を出力するまでの DeepQA の処理の流れを図 -4 に示し，以下で各部分を順に解説する．

### (1) 質問の解析

英語で書かれた質問文から，何が問われているかを判断する．質問文の構文のバリエーションはほぼ無限にあるため，特定の型にあてはめることはできず，正確な構文の解析が必要となる．

表 -1 の 1 番目の例の場合は，問題文を構文構造に変換したうえで，回答すべきものが“this Semitic language”であること，“borrows many words from Italian”という従属節の部分は解答に直接関係がないことなどを認識させる．さらに，文中の代名詞“it”が“Italian”ではなく“Maltese”を指しているという解釈（照応解析）も問題を解くために重要であり，自然言語処理で培われてきた要素技術が試される場である．

### (2) 解候補の生成

次に，質問文に対する解答の候補（仮説）を，大量の情報源の中から探して列挙する．情報源には，ニュース記事，百科事典やその他のテキスト文書や，語彙体系などの辞書が含まれる．なお，Watson はクイズ番組の出場時にはインターネットに接続しないので，情報源はあらかじめ蓄えておく．

この段階で正しい答えを見落としてしまうと，後段の処理で取り返すことができない．そこで，質問文に含まれている語句と同時に現れやすい語句を検索したり，質問で問われている事物に該当する語を辞書から列挙したり，複数の手段で候補を探す．その結果，候補の数は数百に及ぶ．

### (3) 解答の根拠探し

候補の中から正しい答えを選択するために，各候補が質問文に対する解答であると言える「根拠」を情報源の中から探す．この操作はいわば，解候補の語句を問題文の該当個所に埋め込んだもの[☆4]と同じ内容が情報源のどこかに書かれているかを探すものである．もちろん，情報源の中に問題文と一語一句変わらない記述が見つかることは稀なので，問題文を分解して重要な部分を抜き出したり，候補の語が持つべき意味的性質を列挙したりして，それらと合致する情報源の記述を探索する．その合致（マッチング）のさせ方を「観点」，マッチしたものを「根拠」と呼ぶ．正しい解答に対して多くの根拠を見つけられるよう，新たな観点を増やしたり，マッチングのアルゴリズムを洗練させたり，情報源を充実させたりすることが，性能向上に向けて取り組んだ研究開発の核である．最終的に観点の数は百以上になった．

(2) で列挙した全候補に対して根拠を探す必要があり，処理時間が無視できない．高速化のために，各候補に対する処理を並列化した．本番の環境では Power7 アーキテクチャの 2,880 コアを用いた超並列の環境で動作させ，高速化を実現した．

### (4) 確信度の計算

候補に対して，(3) で見つかった根拠に応じて得点付けをする．正答に繋がりやすい強い根拠を持つ候補に大きな値が割り当てられるよう，それぞれの観点に「重み」を付与する．重みの計算のために，過去の Jeopardy! の問題と解答のデータ数万件を用いた機械学習を行う．すなわち，過去問をその時点のアルゴリズムと情報源に基づいて解こうとしたとき

☆4　たとえば，表 -1 の例 1 で，問題文中の“this Semitic Language”を解候補（“Arabic”，“Hebrew”など）で置き換えたもの．

に，正解率が最大となるように，観点への重みをロジスティック回帰により計算する．

各候補について，有効な根拠に重み付けをしたものの総和を計算し，これを確信度とする．最大の確信度を持った答えが閾値を超えたときに，Watsonはボタンを押して回答をする．

## ➤ 解答導出の例

上記のプロセスでどのように問題を解いているのかを直感的に理解するために，例を挙げて解説する．Watsonが対応しているのは英語の質問文と情報源の処理であるが，ここでは理解を容易にするために，日本語の例を導入する．

質問文：「本州のなかで最も西に位置するこの県は，
1871年に発足した．」
正　答：「山口(県)」

まず，質問文の中のキーワード，この場合「本州」「最も」「西」「県」「1871」などを検索条件として，情報源の中を検索し，それと一緒に出現しやすいキーワードを列挙する．すると，「広島」「山口」「鳥取県」「中国地方」「奥多摩」など，解候補が得られる．問われているもの(これを「質問の型」と呼ぶ)が「県」だということが分かっても，最初から日本の43の「県」だけを考えればよいわけではない．解答は日本の県に限るという知識が質問文には明示されておらず，Watsonには本州に位置する県というものが実質的に日本の県を意味するということは容易には結論づけられないからである．このほかにも，質問の型が「作曲家」だったらどの集合を調べればよいか，「液体」なら，または「形式」なら一体何を調べるか…と考えていくと，質問の型とその解答になる語句の組合せには際限がなく，関連しそうな語句を大量に調べてみるほかはない．

次に，これらの候補が答えとして適切かどうかを調べるため，情報源の中から根拠を探す．根拠を調べる観点には，「候補が，質問の型である『県』であるか？」「候補が，質問文中の制約『最も西にある』と記述されているか？」「問題文中と同じ時間表現とともに現れるか？」「該当する語句への参照(リンク)がいくつあるか？」などがある．それぞれの候補について，これらの観点から根拠を見いだせるかを**表-2**に示す．

| 観点＼解候補 | 広島 | 山口 | 鳥取県 | 中国地方 | 奥多摩 |
|---|---|---|---|---|---|
| 候補と質問で型が一致する？(「県」である) | ○ | ○ | ○ | × | × |
| 条件の一部が一致？(最も西にある) | × | ○ | × | ○ | ○ |
| 時間表現が共通？(1871年の記述を含む) | × | ○ | × | ○ | × |
| 該当する語句へのリンクの数(多いほうがよい) | 1300 | 500 | 200 | 150 | 10 |
| 総合点(確信度) | 2% | 92% | 20% | 6% | 0% |

**表-2　解候補ごとの根拠の探索**

すべての観点で根拠を見いだせる解答は存在しないことが多いので，過去の問題から学習した重み付けに基づいた確信度を計算する．この例の場合，正解である「山口」に最も高い確信度が与えられた．なお，このほかに「山口県」という候補もあった場合は，それらを統合した上で確信度を計算する．最終的な確信度が十分に大きければ，ボタンの押下を試みて，それが対戦相手よりも速ければ回答ができる．

## ➤ 情報源の整備

問題を解くために必要な情報には，大きく分けて2種類ある．1つは辞書，語彙体系，意味的関係(「坊ちゃん」の著者＝「夏目漱石」といった事物の関係)のように構造化されたデータで，もう1つはニュース記事，百科事典の本文，Weblogの記事など，通常の英語で書かれた非構造情報のテキストである．

知識を整理するという意味では，前者の構造化情報が重要かつ扱いやすいが，Jeopardy!が扱うような広い分野の知識を網羅するのは困難である．また，言語の多義性，意味的な曖昧性があるときに，矛盾がないような概念体系を人手で構築するのは困難である．たとえば，Schwarzenegger は俳優か政治家か，「イヌ」は「ネコ」であるか否か☆5，のように構造化が難しい現象は枚挙にいとまがない．

☆5　イヌ科はネコ目であるという点においては正しい．

そこでWatsonでは，従来の質問応答システムで試みたように1つの知識体系を整備するという方針ではなく，利用可能な複数の知識を用いることにした．語と語の関係などを整理した語彙体系として，WordNet[☆6]，DBPedia[☆7]，YAGO[☆8]などを参照している．これにより，「AはBである」というis-aの関係を検証する場合（質問の型がBで，解候補Aが答えとして適切かどうかを調べる），それぞれの語彙体系に関係が見つかるかを別々の観点で見る（表-2に独立の行として○・×を付けるイメージ）ことにより，各体系が網羅性や一貫性に欠ける場合のリスクを低減することができる．

もう一方の情報源の形が，非構造情報，すなわち生のテキストの情報である．重要な事実が多く書かれている百科事典や新聞記事が有用なのは想像に難くないだろう．そのほか，数々の実験を通して，正しい解答を得るために必要な情報は何かを議論していった結果，シェイクスピアの戯曲，聖書，歌の歌詞など，その引用やパロディが問題文中に使われやすいものを加えていった．最終的に情報源は約70GBとなった．これはインターネット全体のデータ量よりは遥かに小さいが，無料であるなど入手が容易であり，後述の前処理が妥当な時間で実行できて，実メモリに載せられる量であり，かつ出題される問題の多くをカバーするという点で絞り込まれたものである．

人間の頭脳では決して憶えきれない量の生のテキストを一語一句違わずに暗記していることになるが，文字列そのものよりも意味的内容が重要である場合が多い．そこで，テキストに対して事前に構文解析や関係抽出をして，その結果を生のテキストに付与しておくという「前処理」を施す．これにより，豊富な情報を高速に検査することが可能となった．

テキストからの情報抽出は，非構造情報を構造情報に転換し，扱いやすい観点を増やすことに寄与する．たとえば，Wikipediaの Backgammonの項目の一文目には，“Backgammon is one of the oldest board games for two players.”という記述があるが，「A is B」や「A is one of B」などの構文パターンが「AはBの一種である」ことを示すという知識を用いれば，「Backgammon」は「game」であるという知識が得られる．この手法により全文書を解析しておけば，新たなis-aタイプの概念体系を生成することができる．解析誤りによってノイズが生じることもあるが，人手で作った既存の概念体系とは異なる客観的で網羅性のあるデータを作ることができ，答えらしさの判定に寄与した．筆者らのIBM東京基礎研究所チームは，2007年12月にWatsonプロジェクトへの参画を依頼され，以後このような情報源からの情報抽出に主な貢献をした．

## ➤ UIMAを用いた開発

UIMA（Unstructured Information Management Architecture）は，自然言語のような解釈に曖昧性のあるデータに対して，その構造や意味を，順次メタデータとして加えていく仕組みで，2006年からオープンソース・ソフトウェアとして公開され[☆9]，2009年からOASIS（Organization for the Advancement of Structured Information Standards）[☆10]の標準となっている．Watsonでは，質問文の分析・根拠の探索と，情報源の前処理を含むすべてのプロセスがUIMA上のプラグインとして実装されている．

例として，**図-5**に質問の解析の概略を示す．すべて大文字で入力された質問文を，小文字に変換し，構文解析等の処理を行い，問題のフォーカスとなる部分を求めるまでの処理が，解析器の組合せにより実現されている．個々の解析器はWatson専用のものとは限らず，既存の自然言語処理技術も含む．各解析器，および全体の入出力の形式は，元の文に注釈のオブジェクトを加えていくというUIMA標準のデータ構造で統一されている．質問の解析，候補の生成，テキストの前処理などを合わせると，全体

☆6 http://wordnet.princeton.edu/
☆7 http://dbpedia.org/
☆8 http://www.mpi-inf.mpg.de/yago-naga/yago/
☆9 http://uima.apache.org/
☆10 http://www.oasis-open.org/

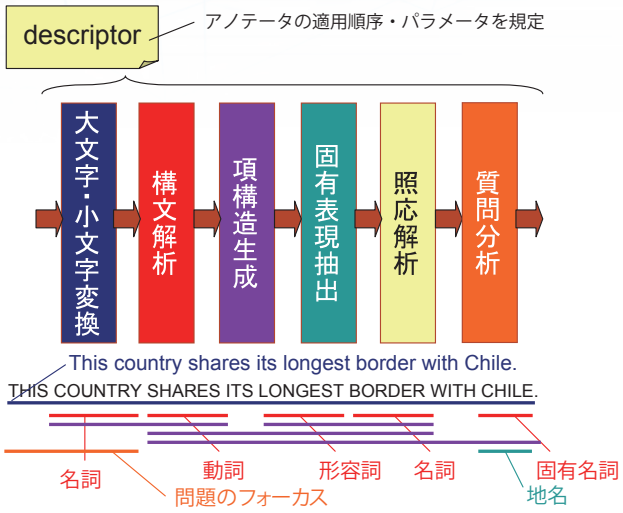



図 -5　UIMA による質問文の解析の処理の流れ．複数の解析器が接続されている．

で数百のコンポーネントが使われているものの，整合性を保ちつつ設計することができた．

さらに，UIMA は，プロセス全体のパラメータ調整や並列化の制御をする機構を持つため，個々のコンポーネントを追加・改良したり，パラメータを変更したりするときにも，UIMA の設定ファイル（descriptor）を用いてさまざまな条件下でのシステム構成を簡単に記述することができる．これは試行錯誤しつつ実験を繰り返すための強力なツールとなる．短期間でこれだけの性能を持つシステムを構築できたのは，非構造情報処理専用に開発された UIMA の最大の効果であった．

また，Watson プロジェクトの推進には大学との共同研究も重要な役割を果たした．同プロジェクトにはマサチューセッツ工科大学，テキサス大学オースチン校・カーネギーメロン大学など 8 校が参画し [4]，UIMA と同様の考え方で，質問応答で用いられる各要素の相互運用性を高めて研究開発を行える仕組みとそのオープン化について，2009 年 2 月に提言を行った [5]．

## Watson の結果

以上のように開発を進めていった Watson が，どのように性能を上げていったか，そして実際の対戦の結果や，そこから得られたものについて紹介し，今後の応用の可能性を探る．

### ➤ 性能向上の軌跡

開発当初は人間の能力には遠く及ばなかったものの，DeepQA の仕組みを設計・実装して，幾度もの実験を繰り返し，新しいアルゴリズム，必要なデータの検討を重ねることによって，**図 -6** に示すように性能はぐんぐん向上していった．2008 年の末には，過去のトップ回答者の一部の性能を上回り，チ

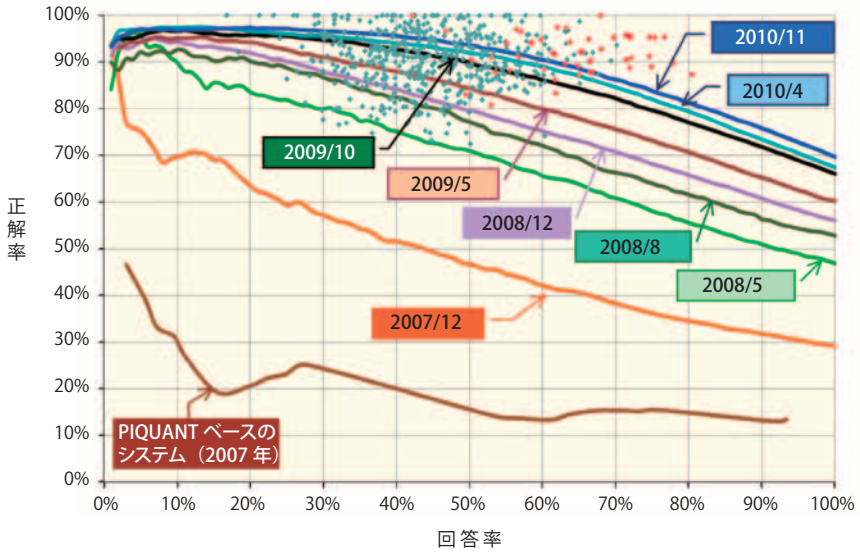



図 -6　4 年間の性能向上の履歴（文献 1）より引用）

図 -7　対戦の模様．左から，Jennings 氏，Watson，Rutter 氏．

| | |
|---|---|
| 3 | カテゴリ：US Cities<br>問題文：Its largest airport was named for a World War II hero; its second largest, for a World War II battle.<br>（この都市の最大の空港は第 2 次大戦の英雄の名が，2 番目の空港は戦いの名前が付けられている）<br>正しい答え：Chicago（シカゴ） |
| 4 | カテゴリ：Name the Decade<br>問題文：The first modern crossword puzzle is published & Oreo cookies are introduced.<br>（最初のクロスワードパズルが出版され，オレオクッキーが発売された年代）<br>正しい答え：1910s（1910 年代） |

表 -3　Watson が正答できなかった問題の例

ャンピオンとの対戦の目処がついたため，2009 年 4 月に IBM は Jeopardy! の対戦を発表した☆11．

当然ながら徐々に性能向上のスピードは緩むものの，2010 年 11 月には Ken Jennings 氏の成績を半分弱は上回る域に達した．これは，実際に対戦をしたときに互角に近くなることを意味する．そして遂に 2011 年 2 月に対戦，そしてテレビ放映が実現した．

## ➤ 対戦の実現

2011 年 2 月 14 ～ 16 日，3 日にわたって放映された Jeopardy! では，先述の連勝記録を持つ Ken Jennings 氏，累積獲得賞金額が最大の Brad Rutter 氏，そして Watson の三者（**図 -7**）で 2 ゲーム（合計 122 問）が戦われた．結果として，Watson は圧倒的な知識量により両氏を突き放した．実際には Jennings，Rutter 両氏も，解答を分かっていて，ボタンの押下を試みている場面も多かったが，Watson の応答速度が勝っていた．また，過去のゲームのシミュレーションと最適化技術に基づく絶妙なパネルの選択（ボーナスパネルを高確率で引き当てる），掛け金の設定（人間には決してできない半端な額を瞬時に計算）などのゲーム戦略も勝利に結びついていた．

## ➤ 人間と Watson との違い

一方，対戦の中で，Watson の弱点も露呈された．以下の例のように，人間は犯さないタイプのミスや，答えられないカテゴリがあった．

☆11 http://www-06.ibm.com/jp/press/2009/04/2801.html

### ❍ カテゴリの解釈

1 ゲーム目の最後の筆記問題は，**表 -3** の例 3 にある，“US Cities”というカテゴリで米国の都市を問うものであった．Jennings 氏，Rutter 氏は“Chicago”と正答したが，Watson は“Toronto”と，カナダの都市を書いて誤答をしてしまった．人間にとってはあり得ない間違え方だ．

しかし，この“US Cities”というカテゴリを見て，米国の都市が答えとなることは実は自明ではない．たとえば“Biology”というカテゴリは，問題や答えが生物学に関することを示唆するだけで，答えが生物学の一種（分子生物学など）となるわけではない．すなわち，“US Cities”というカテゴリは，答えが米国の都市だという絶対的制約としては働かず，米国の都市が答えとなる数値を相対的に上げる要素として使われるだけである．他の根拠と総合した結果，“Toronto”の確信度が“Chicago”のそれを上回ったために誤答となった．実際にはこのような柔軟なアプローチをとることによって，全体の正解率を上げることに成功している．

### ❍ 誤答の繰り返し

表 -3 の例 4 は，あるイベントが行われた年代が問われる問題である．Jennings 氏が“(19) 20s”と答えて点数を引かれてしまった直後に，Watson が“1920s”と同じ答えを言って同様に点数を引かれるという場面があった．これは，他人の誤答がシステムには伝わっていないという設定上の問題であり，音声認識は今回の挑戦の本質的問題ではない（発声の仕方や会場の反響で精度が大きく変化し，質問応答の達成度が測れなくなる）ために Watson が対象

外としていることによる．

**❍ 短い質問への反応**

映画のタイトルが質問文となって，その監督兼俳優を答えさせるカテゴリがあった．これは5問ともWatsonは答えられなかった．解答を導くことはできたものの，速度が人間に追いつかなかったのである．

人間は映画のタイトルを聞いただけで「それについて知っている」ことが瞬時に分かり，ボタンを押すことができる．一方で，Watsonはそのような直感を持っていないため，解候補を列挙→それぞれの根拠を探索→スコア付けというプロセスを経てからでしかボタンを押すか否かの判断ができない．これは短い問題でも複雑な問題でも同様である．

以上のように，最終的な金額ではWatsonが上回ったものの，人間の能力とは方向性が異なる．見方を変えれば，人間とコンピュータは互いに補完的であるといえる．

## ➤ 実用化に向けて

Watsonによって培われた技術は，さまざまな分野での応用が期待されている．しかし，Watsonがあれば世の中の質問に何でも答えられるというわけではない点には注意が必要である．クイズ番組への勝利で立証したことは，答えが1つに定まるような問題文が与えられたときに，一般的な知識が書かれたテキストを参照して解答を導くというWatsonの能力である．Watsonが，明日の天気や，独自性を持った政治についての意見について答えられるわけではない．逆に，決定的に解ける問題，たとえば掛け算や辞書引きに対してはDeepQAのような仕組みは不要である．また，Watsonが人間ではあり得ないような過ちを犯すことも分かった．

しかし，本質を理解すれば，Watsonの技術を活用してこそ解決できる，真に役に立つ課題が見つかる．その例の1つが医療分野の例である．

患者のカルテの情報，本人や親の病歴，血圧等の数値が入力として与えられたときに，その患者がどの病気であるかを推測するという課題である．情報源として，過去のカルテ，医学に関する文献などが利用できる．このとき，1つの病気を言い当てる必要はなく，複数の病気とその確信度を出力すればよい．

実際の医療の現場で，医師は自分の知識をもとに診断を行っているが，その際に本来の病気を見落としているという状況が存在するという．このように，質問応答システムが持つ情報アクセスにより人間の活動を補助できる場面は，今後も多数考え出されるであろう．

そのほか，Watsonに使われた自然言語処理の要素技術は応用が利く．たとえば，英語の構文解析器は，Watsonの開発を経て大幅に性能が改良された．全体の質問応答システムのごく一部として働くものであっても，その達成度が客観的な数値で測れたことにより，漸次的な改良を進めることができた．その構文解析を用いたテキストマイニングにより，人間では読み切れない量の文書から知識を抽出すること，事物の関係を知ることができるようになり，応用の幅はさらに広がるものと期待される．

**参考文献**


1) Ferrucci, D., Brown, E., Chu-Carroll, J., Fan, J., Gondek, D., Kalyanpur, A. A., Lally, A., Murdock, W., Nyberg, E., Prager, J., Schlaefer, N. and Welty, C. : Building Watson : An Overview of the DeepQA Project, AI Magazine, Vol.31, No.3, pp.59-79 (2010).
2) 磯崎秀樹，東中竜一郎，永田昌明，加藤恒昭，(監修：奥村学)：質問応答システム，コロナ社 (2009).
3) Chu-Carroll, J., Prager, J., Welty, C., Czuba, K. and Ferrucci, D. : A Multi-Strategy and Multi-Source Approach to Question Answering, Proceedings of TREC2002, pp.281-288 (2002).
4) IBM：IBM，Watsonコンピューティング・システム開発に貢献した8つの大学を発表，http://www.ibm.com/jp/press/2011/02/1501.html
5) Ferrucci, D., et al. : Towards the Open Advancement of Question Answering Systems, IBM Research Report, RC24789 (Apr. 2009).


(2011年4月19日受付)


**金山　博**　hkana@jp.ibm.com
2000年東京大学大学院理学系研究科情報科学専攻修士課程修了．同年より日本アイ・ビー・エム（株）東京基礎研究所に勤務，現在に至る．構文解析・意味解析など自然言語処理の研究に従事．Watsonプロジェクトに参加．

**武田浩一**（正会員）takedasu@jp.ibm.com
1983年京都大学大学院工学研究科情報工学専攻修士課程修了．同年日本アイ・ビー・エム入社．機械翻訳，テキストマイニングなどの研究に従事．Watsonプロジェクトに参加．1987～89年カーネギー・メロン大学客員研究員．博士（情報学）．