Question analysis: How Watson reads a clue

A. Lally
J. M. Prager
M. C. McCord
B. K. Boguraev
S. Patwardhan
J. Fan
P. Fodor
J. Chu-Carroll

The first stage of processing in the IBM Watson™ system is to perform a detailed analysis of the question in order to determine what it is asking for and how best to approach answering it. Question analysis uses Watson's parsing and semantic analysis capabilities: a deep Slot Grammar parser, a named entity recognizer, a co-reference resolution component, and a relation extraction component. We apply numerous detection rules and classifiers using features from this analysis to detect critical elements of the question, including: 1) the part of the question that is a reference to the answer (the focus); 2) terms in the question that indicate what type of entity is being asked for (lexical answer types); 3) a classification of the question into one or more of several broad types; and 4) elements of the question that play particular roles that may require special handling, for example, nested subquestions that must be separately answered. We describe how these elements are detected and evaluate the impact of accurate detection on our end-to-end question-answering system accuracy.

Introduction

The question-answering process in IBM Watson*, like that of most other question-answering systems, begins with a question analysis phase that attempts to determine what the question is asking for and how best to approach answering it. Broadly speaking, question analysis receives as input the unstructured text question and identifies syntactic and semantic elements of the question, which are encoded as structured information that is later used by the other components of Watson. Nearly all of Watson's components depend in some way on the information produced by question analysis.

Question analysis is built on a foundation of general-purpose parsing and semantic analysis components. Although these components are largely domain-independent, some tuning to the special locutions of Jeopardy!** questions has been done, which we describe in this paper.

Based on this foundation, we apply numerous detection rules and classifiers to identify several critical elements of the question. There are a variety of such elements, each of which is vital to different parts of Watson's processing.

Digital Object Identifier: 10.1147/JRD.2012.2184637

The most important elements are the *focus, lexical answer types (LATs), Question Classification*, and *Question Sections (QSections)*. The definitions of these terms refer to the following example Jeopardy! question:

POETS & POETRY: He was a bank clerk in the Yukon before he published "Songs of a Sourdough" in 1907.

The *focus* is the part of the question that is a reference to the answer. In the example above, the focus is "he". (In the case where multiple words refer to the answer, it is generally sufficient to detect one focus and then apply general-purpose co-reference resolution to find the other references.) The focus is used, for example, by algorithms that attempt to align the question with a potential supporting passage [1]; for proper alignment, the answer in the passage should align with the focus in the question.

LATs are terms in the question that indicate what type of entity is being asked for. The headword of the focus is generally a LAT, but questions often contain additional LATs, and in the Jeopardy! domain, categories are an additional source of LATs. In the example, LATs are "he", "clerk", and "poet". LATs are used by Watson's type coercion components [2] to determine whether a candidate answer is an instance of the answer types.

0018-8646/12/$5.00 © 2012 IBM

Question Classification identifies the question as belonging to one or more of several broad types. The example question is of the most common class, Factoid, but many Jeopardy! questions are of other classes such as Definition, Multiple-Choice, Puzzle, Common Bonds, Fill-in-the-Blanks, and Abbreviation. These Question Classes (QClasses) are used to tune the question-answering process by invoking different answering techniques [3], different machine learning models [4], or both.

QSections are question fragments whose interpretation requires special handling. Some of the most important uses of QSections are to identify lexical constraints on the answer (e.g., "4-letter" or "3-word") and to decompose a question into multiple subquestions [5].

Most of our rule-based question analysis components are implemented in Prolog [6, 7], a well-established standard for representing pattern-matching rules. Our implementation can analyze a question in a fraction of a second, which is necessary to be competitive at the Jeopardy! task.

Nearly all other question-answering systems include some form of question analysis. A large number of these systems have been developed under the influence of organized evaluation efforts such as TREC/TAC (Text REtrieval Conference/Text Analytics Conference) [8] and CLEF (Cross-Language Evaluation Forum) [9]. They have a strong emphasis on factoid questions such as "Which was the first movie that James Dean was in?" and "What is the population of Japan?" However, Jeopardy! questions are often more complex than these typically studied factoid questions; thus, accurate question analysis is more difficult [10]. In addition, Jeopardy! questions are organized into categories, and determining how to make use of the name of the category is an important and challenging task that does not exist in prior work.

The remainder of this paper is structured as follows. In the next section, we summarize the parsing and semantic analysis capabilities that provide the foundation for question analysis, and we discuss how they were adapted to the unique challenges presented by Jeopardy! questions. Next, we explain how we implemented rule-based portions of question analysis using Prolog. Then, we discuss focus and LAT detection in more detail, evaluate the accuracy of our implementation, and evaluate its impact on our end-to-end question-answering performance. This is followed by a similar discussion and evaluation of Question Classification and QSection detection. Finally, we compare Watson's question analysis with that of other question-answering systems and conclude with a summary of this paper.

Foundation of question analysis

Parsing and semantic analysis

Watson's parsing and semantic analysis suite is composed of the Slot Grammar parser ESG (English Slot Grammar) with an associated predicate-argument structure (PAS) builder [11], a named entity recognizer (NER), a co-reference resolution component, and a relation extraction component [12].

ESG is used to parse each sentence in a question into a tree that shows both surface structure and deep logical structure. Each tree node has attached to it: 1) a word or a multiword term with an associated predicate and its logical arguments; 2) a list of features, some morphosyntactic, others semantic; and 3) the left and right modifiers of the node, each with the slot it fills. The deep structure lies in the predicates and their arguments. The predicate arguments are other tree nodes, but they may come from remote positions of the tree or may be the active-form (logical) arguments in passives. Such predications are useful for building logical forms and working with standardized relations associated with the predicates. The PAS builder produces a simplification of the ESG parse. In the PAS, parses with small variations in syntax are mapped to a common form, which assists in subsequent pattern matching.

In the example question from the introduction, parsing and semantic analysis identify, among many other things, predications `publish(e1, he, ''Songs of a Sourdough'')` and `in(e2, e1, 1907)`, the latter saying that publishing event `e1` occurred in 1907. It also resolves the co-reference of the two occurrences of "he" and "clerk", identifies "Yukon" as a geopolitical entity and "Songs of a Sourdough" as a composition, and extracts relations such as `authorOf(focus, ''Songs of a Sourdough'')` and `temporalLink(publish(...), 1907)`.

Question analysis adaptations

On the Jeopardy! show, questions are displayed in all uppercase, and this is exactly how Watson received them. Although this does not present much trouble to the human contestants, it does present challenges for a computer. Mixed case provides information, for example, to indicate proper names or titles. Our existing parsing and semantic analysis capabilities were developed for mixed case and heavily relied on these cues. To address this problem, we apply a statistical true-caser component that has been trained on many thousands of correctly cased example phrases [13].

ESG has been adapted in several ways to the special locutions of Jeopardy! questions. In place of "wh" pronouns, Jeopardy! questions typically use "this/these" and "he/she/it", which sometimes leads to constructions that are very rare in English, as in "A stock that is a low risk investment is a blue this". The parser was adapted to handle these. In addition, a fair number of Jeopardy! questions consist only of noun phrases. Although ESG had been developed to handle text segments of any phrase type (not just complete sentences), noun phrases occur much more frequently in Jeopardy! questions than in average English; hence, modifications were made to the parser to prefer noun phrases

Table 1 Relations in Jeopardy! questions.

Question	*Relations*
A.k.a., the Flavian Amphitheatre, this ancient structure was begun by the Roman Emperor Vespasian around 72 A.D.	`altName(focus,Flavian Amphitheatre)`
A myocardial infarction, better known as this, is a common reason for ICU admission.	`altName(focus, myocardial infarction)`
In May 1898 Portugal celebrated the 400th anniversary of this explorer's arrival in India.	`anniversaryOf(this explorer's arrival in India, 400, May 1898)`
Chile shares its longest land border with this country.	`borderOf(focus,Chile)`
In 1867 the U.S. bought this island group named for a Russian captain and leased it to seal hunting companies.	`rdfTriple(buy,U.S.,focus)`

under certain conditions. For example, in the noun-phrase question, "Number of poems Emily Dickinson gave permission to publish during her lifetime," early versions of ESG preferred an awkward verb-phrase analysis with "publish" as the main verb; this has been corrected since. In spite of these adaptations, care was taken not to degrade parsing of normal English. This is done in part by use of switches for the parser that are turned on only when parsing Jeopardy! questions.

Our co-reference component also required adaptation because Jeopardy! questions often include an unbound pronoun as an indicator of the focus, which is not expected in normal text. For example, in the question, "Astronaut Dave Bowman is brought back to life in his recent novel 3001: The Final Odyssey," the "his" refers to the answer (Arthur C. Clarke), not Dave Bowman. We addressed this by running focus finding before co-reference resolution and adapting our co-reference processing to be focus-aware.

Special-purpose relations

In addition to being applied to general text [12], our relation extraction component has also been adapted in two ways to meet the specific characteristics and idiosyncrasies of Jeopardy! questions. On the one hand, special-purpose relations of particular relevance to Jeopardy! are identified, and relation detectors for them are developed. On the other hand, some of the same underlying pattern-matching mechanisms that are used for relation detection can also be used to identify certain (non-relational) aspects of Jeopardy! questions, which are subsequently used in Question Classification and special question processing.

Special-purpose relations address Jeopardy!-specific locutions, which express relationships common enough in the genre of Jeopardy! questions, but not necessarily frequent enough to warrant the development of relation extractors situated within the parsing and semantic analysis suite of relations. Their frequency in Jeopardy!, however, is sufficiently high that detecting and exposing them can make a difference to subsequent components. Such relations include (see example questions in **Table 1**) the following: alternate names for the focus element, temporal arithmetic in questions, and geospatial relations. We also detect relations called `rdfTriple` that identify individual facts within a complex question and enable a general approach for checking whether these facts are true.

For certain classes of questions, we deploy a separate line of processing using semantic frames. Broadly speaking, a frame gathers typical and characteristic properties of an entity or an event: examples of frames we use for Jeopardy! question answering include books, presidents, countries/states, and awards (e.g., "who was awarded/won what, for what, and when"). The notion is that, by knowing the values and the semantics of some of a frame's slots, domain-specific reasoning could lead to the value of the slot element that is the focus of the question [14]. The deep semantic relations framework is well suited for detecting and instantiating slot values, and an adaptation for that purpose is a part of the semantic frame-based answering pipeline. As an example, the question "His 2 acting Oscars have been awarded for playing a tough cop in 1971 and a brutal sheriff in 1992" leads to the identification of the following relations: `awardType(Oscar)`, `awardWinner(focus)`, `awardRole(tough_cop)`, and `awardCategory(acting)`. These are then used by a general frame instantiation mechanism (this example would lead to two frame instances, identical in all but the `awardRole` slot).

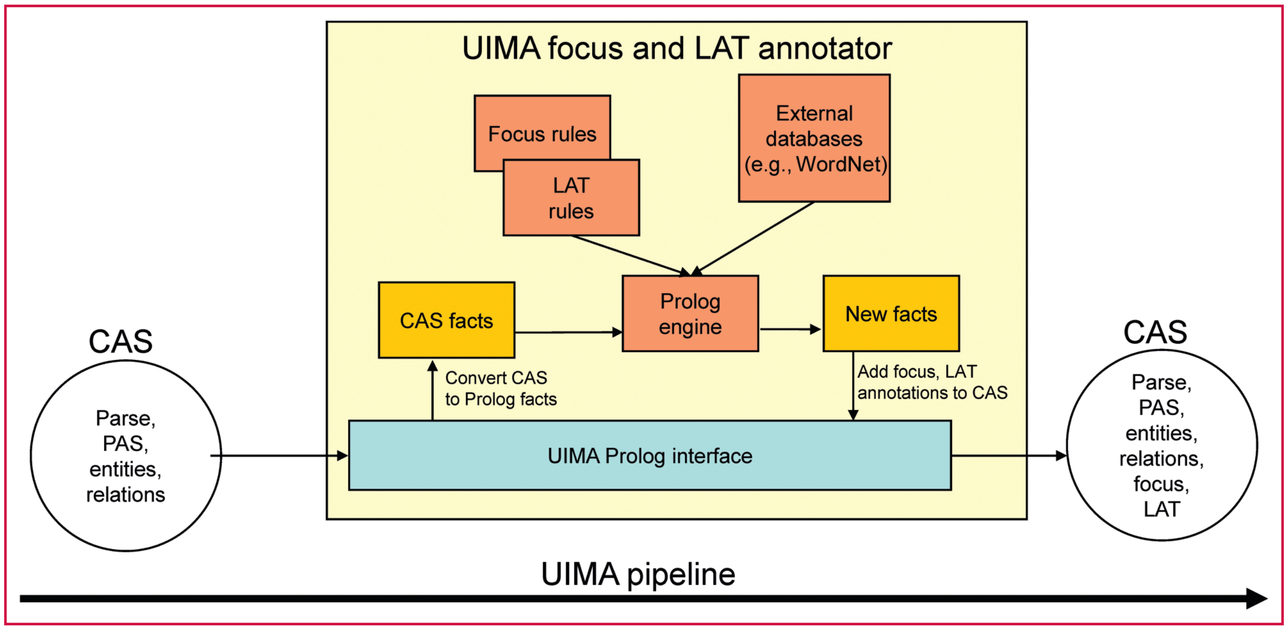
UIMA focus and LAT annotator
Focus rules
LAT rules
External databases (e.g., WordNet)
CAS facts
Prolog engine
New facts
Convert CAS to Prolog facts
Add focus, LAT annotations to CAS
UIMA Prolog interface
CAS
Parse, PAS, entities, relations
CAS
Parse, PAS, entities, relations, focus, LAT
UIMA pipeline

Figure 1

UIMA Prolog interface specialized for focus and LAT detection. (Figure used with permission from P. Fodor, A. Lally, and D. Ferrucci, "The Prolog Interface to the Unstructured Information Management Architecture," Computing Research Repository, 2008; available: http://arxiv.org/abs/0809.0680v1.)

Given that the machinery and methods for pattern-driven relation detection are, fundamentally, developed for pattern matching over PAS configurations, they can be directly applied to a task different from relational analysis: identification of QSections (i.e., text segments in questions associated with certain question characteristics), which can also be used to influence Question Classification. For example, see the discussion of abbreviation question detection and question decomposition in the section "Question Classification and QSection detection", below.

Prolog implementation of rules

As in most of Watson, question analysis is implemented as a pipeline of components assembled using the Unstructured Information Management Architecture (UIMA) [15]. Most of the question analysis tasks in the Watson project are implemented as rules over the PAS and various external databases such as WordNet** [16]. We required a language in which we could conveniently express a large number of rules over a dependency-based parse, including matching over long-distance relationships. We found that Prolog was the ideal choice for the language because of its simplicity and expressiveness.

Prolog was a natural fit for integration with UIMA, as shown in **Figure 1**. In UIMA, the common analysis structure (CAS) is a dynamic data structure that contains unstructured data (i.e., data whose intended meaning is still to be inferred) and structured information inferred from this data, encoded as *feature structures*. The translation of the CAS to Prolog facts is straightforward. Each CAS feature structure is assigned a unique integer ID. Each feature (property) of that feature structure becomes a fact of the form `feature_name(id, value)`. If the value of a feature is another feature structure, then the ID of the target feature structure is used in the value slot of the Prolog fact. Array values are represented as Prolog lists. Any new facts that result from querying Prolog rules are asserted back into show the CAS as new feature structures and will be subsequently passed to annotators after the current one in the UIMA pipeline.

For example, the PAS nodes produced by the parsing and semantic analysis annotators are represented as Prolog facts such as the following (the numbers representing unique identifiers for PAS nodes):

```
lemma(1, ''he'').
partOfSpeech(1,pronoun).
lemma(2, ''publish'').
partOfSpeech(2,verb).
lemma(3,''Songs of a Sourdough'').
partOfSpeech(3,noun).
subject(2,1).
object(2,3).
```

Such facts were consulted into a Prolog system, and several rule sets were executed to detect the focus of the question, the LAT, and several relations between the elements of the parse. For example, a simplified rule for detecting the `authorOf` relation can be written in Prolog as follows:

```
authorOf(Author, Composition) :-
  createVerb(Verb),
  subject(Verb, Author),
  author(Author),
  object(Verb, Composition),
  composition(Composition).-

  createVerb(Verb) :-
  partOfSpeech(Verb, verb),
  lemma(Verb, VerbLemma),
  [ ''write'', ''publish'', ...].
```

The `author` and `composition` predicates, not shown, apply constraints on the nodes (i.e., "he" and "Songs of a Sourdough", respectively) to rule out nodes that are not valid fillers for the author and composition roles in the relation.

This rule, applied to the example, results in the new fact `authorOf(1, 3)`, which is recorded and passed to downstream components in the Watson pipeline.

Prior to our decision to use Prolog for this task, we had implemented custom pattern-matching frameworks over parses. These frameworks ended up replicating some of the features of Prolog but lacked the full feature set of Prolog or the efficiency of a good Prolog implementation. Using Prolog for this task has significantly improved our productivity in developing new pattern-matching rules and has delivered the execution efficiency necessary to be competitive in a Jeopardy! game. We have implemented different Prolog rule sets for shallow and deep relation extraction [12], focus detection, LAT detection, Question Classification, and QSection detection. In all, these rule sets consist of more than 6,000 Prolog clauses.

Focus and LAT detection

We begin our discussion of focus and LAT detection by presenting a set of baseline patterns that are readily observable from Jeopardy! questions and that are simple to implement. We then show where this simple baseline is inadequate and demonstrate improvements that we have made.

Baseline focus and LAT detection

The baseline focus detection implementation consists of the following patterns, in order of their priority, with ties broken by taking the lexically first. Each is illustrated by an example, where the italicized phrase is the focus and its headword is in bold.

- A noun phrase with determiner "this" or "these":
  THEATRE: A new play based on *this Sir Arthur Conan Doyle canine* ***classic*** opened on the London stage in 2007.
- "This" or "these" as a pronoun:
  '88: In April 1988, Northwest became the first U.S. air carrier to ban ***this*** on all domestic flights.
- When the question is a noun phrase, we conventionally label the entire question as the focus:
  AMERICAN LIT: ***Number*** *of poems Emily Dickinson gave permission to publish during her lifetime.*
- One of the pronouns "he/she/his/her/him/hers":
  OUT WEST: ***She*** joined Buffalo Bill Cody's Wild West Show after meeting him at the Cotton Expo in New Orleans.
- One of the pronouns "it/they/them/its/their":
  ME "FIRST"!: ***It*** forbids Congress from interfering with a citizen's freedom of religion, speech, assembly, or petition.
- The pronoun "one":
  12-LETTER WORDS: Leavenworth, established in 1895, is a federal ***one***.

When none of the above applies, the question may have no focus,

MOVIE TITLE PAIRS: 1999: Jodie Foster & Chow Yun-Fat.

Although these rules are straightforward, the requirements they impose on the parser may be challenging. It is critical for the parser to correctly attach the determiner "this" to its headword; note that as in "This Sir Arthur Conan Doyle canine classic," the headword is often not the word immediately following "this." It is also very important for the parser to correctly distinguish a noun-phrase question from a verb phrase. As previously noted, we have made some improvements to ESG in this regard.

The baseline LAT detection approach generally chooses the focus headword as the only LAT, with the only exceptions as follows (focus in italics and LAT in bold):

- If the focus is a conjunction, extract the conjuncts:
  HENRY VIII: Henry destroyed the Canterbury Cathedral Tomb of *this* ***saint*** *and* ***chancellor*** *of Henry II.*
- "⟨Focus⟩ of X". extract LAT X when ⟨Focus⟩ is any of one/name/type/kind:
  HERE, PIGGY, PIGGY, PIGGY: Many a mom has compared her kid's messy room to *this kind of hog* ***enclosure***.
- "⟨Focus⟩ for X". extract LAT X when ⟨Focus⟩ is any of name/word/term:
  COMPANY NAME ORIGINS: James Church chose

this name for his ***product*** because the symbols of the god Vulcan represented power.

- If no focus was detected and the category is a noun phase, take headword of the category as LAT: HEAVY METAL **BANDS**: "Seek & Destroy", "Nothing Else Matters", "Enter Sandman".

Improving detection within a question

The simple baseline rules are reasonably precise, but they do make some mistakes, and there are many useful LATs that are not detected. In this section, we present some of the common failure cases that we have addressed. At this point, we consider only how to use the portion of the question excluding the category; the category is addressed in the next section. The following examples all represent cases where the baseline patterns fail:

1) PAIRS: An April 1997 auction of Clyde Barrow's belongings raised money to fund moving his grave next to ***hers***.
2) FATHER TIME (400): On Dec. 13, 1961, Father Time caught up with *this 101-year-old artist* with a relative in **her** nickname.
3) CRIME: Cutpurse is an old-time word for *this type of crowd-working* ***criminal***.
4) THE FUNNIES: "Marmaduke'' is *this breed of* ***dog***.
5) ISLAND HOPPING: Although most Indonesians are Muslims, *this* is the predominant **religion** on Bali.
6) I FORGET: Mythical **rivers** of Hades include the Styx and *this one* from which the dead drank to forget their lives.
7) FAMOUS AMERICANS: Although he made no campaign speeches, *he* was elected **president** in 1868 by a wide electoral margin.

Example 1 illustrates a common focus detection problem of selecting the correct pronoun. The simple baseline rule selects the lexically first but incorrect focus "his". To select the correct focus, we try to determine which pronouns are co-referential with named entities in the question (e.g., "his" to "Clyde Barrow"), and then choose as the focus the pronoun that is not bound to any entity. Once the focus is detected, we can apply our general-purpose anaphora resolution component. This can help us find pronoun LATs that indicate the gender of the answer, for instance, in example 2.

When the focus expresses a subclass relationship (examples 3 and 4), we mark the superclass from the embedded prepositional phrase as the LAT.

LATs are frequently indicated by different types of relationships in which the focus participates, such as co-reference (example 5), expressions of set membership (example 6), or roles played by the focus (example 7).

A further difficulty in LAT detection is determining whether the LAT should be a single word or a multiword term. In most cases, we consider the LAT to be a single word, excluding any modifiers. For example, if a question's focus were "this U.S. president," we would say that the LAT is "president." Modifiers of the LAT may be taken into account by some of the type coercion components, but they are not considered part of the LAT. However, this can be wrong in some cases (consider, if the question were "this vice president," we would not want to extract the LAT "president"). Therefore, we treat a multiword term as a LAT when it does not represent a subtype of its headword (as in "vice president") or when the modifier changes the sense of the headword to an infrequent sense (e.g., "prime minister" can be a LAT because, although it is a subtype of a sense of "minister", it is not a frequent sense, at least in the United States).

Extracting LATs from the category

One of the more difficult challenges is to detect LATs that occur in the category. Jeopardy! categories sometimes but not always express LATs, and it is not always obvious which word in the category might represent a LAT. There are a few category patterns that precisely determine the LAT (e.g., "NAME THE X" or "WHO'S THE X?"), but these cover a very small number of questions.

In the general case, we observe that category words are likely to be LATs if they meet three conditions: they refer to types of entities (e.g., "countries" rather than "geography"), the entity type is consistent with the question LATs (if any), and there are not any mentions or instances of that type in the question. Consider the following examples:

8) BRITISH **MONARCHS**: *She* had extensive hair loss by the age of 31.
9) ATTORNEYS GENERAL: Edmund Randolph helped draft and ratify the Constitution before becoming *this man*'s Attorney General.
10) **ACTRESSES**' FIRST FILMS: Oklahoma!
11) U.S. **CITIES**: *It*'s home to the University of Kentucky and to horseracing's Toyota Blue Grass Stakes.
12) U.S. CITIES: St. Petersburg is home to Florida's annual tournament in *this game popular on shipdeck*.

In example 8, the category contains the LAT "monarch," which expresses an entity type that is compatible with the LAT "she". Example 9 has a very similar form, but "Attorney General" is not a LAT. The fact that the category has a mention in the question makes it much less likely to be a LAT.

In example 10, looking just at the category, both "actress" and "film" seem like potential LATs. If we can identify that the question contains a film (or that it does not contain an actress), we can determine that "actress" is the LAT.

Examples 11 and 12 have the same category, but only in example 11 does it express a LAT. In example 12, "city" is not compatible with the LAT "game," and the presence of an instance of a city in the question lowers the likelihood that "city" would be a LAT rather than a topic.

Estimating LAT confidence

The LAT detection rules, particularly those that operate on the category, can produce false positives. Since some LAT detection rules are more reliable than others, we would like to have a confidence value in each LAT that can be used to weight each LAT appropriately during answer scoring. To accomplish this, we trained a logistic regression classifier using a manually annotated LAT gold standard. The classifier uses the focus and LAT rules that have fired as features, along with other features from the parse, NER, and the prior probability of a particular word being a LAT. Low-confidence LATs are filtered to improve precision.

Learning LATs from previous questions in a category

Finally, during the play of a game, Watson can adjust its category LAT detection by learning from prior (question, answer) pairs that have been revealed within the category. For each word in the category, Watson hypothesizes that that word could have been a LAT and checks whether correct answers to prior questions have been instances of that type. We have precomputed statistics over large numbers of historical Jeopardy! questions that tell us the probability that a category word that has been used as a LAT on a given number of prior questions will continue to be used as a LAT in subsequent questions. These probabilities were computed separately for the case where a LAT detection rule had selected the category word and the case where no rule applied. We set the LAT confidence value of the category word to be equal to this probability estimate, overriding the value that was produced by the logistic regression classifier. As mentioned, low-confidence LATs are filtered. This procedure can infer entirely new LATs that would not have been detected by the rules.

For example, one of the sparring matches that Watson played contained the category "CELEBRATIONS OF THE MONTH," consisting of the following questions:

"D-Day Anniversary and Magna Carta Day"
"National Philanthropy Day and All Souls' Day"
"National Teacher Day and Kentucky Derby Day"
"Administrative Professionals Day and National CPAs Goof-Off Day"
"National Magic Day and Nevada Admission Day."

Upon seeing the first question in the category, Watson incorrectly detected "day" as the LAT and did not interpret the category as expressing a LAT. After that clue was played, Watson was informed of the correct answer ("June") and determined that this matched the category word "month". This increased the probability that "month" would be a LAT for subsequent clues in the category, although it was still far from certain. As Watson saw more correct answers within the category, this probability increased and Watson was able to produce months as its top answers to the later clues in the category.

Table 2 LAT detection evaluation.

	Baseline	*Watson*
Precision	0.817	0.829
Recall	0.613	0.766
F_1	0.700	0.796
Per Question Recall	0.840	0.905

Evaluation

We compared the accuracy of LAT detection (and, indirectly, focus detection, which it depends on) between the baseline rules (from the "Baseline focus and LAT detection" section above) and the complete Watson system (with all of the improvements we have just presented). For the evaluation, we used a manually annotated set of 9,128 questions. We used ten-fold cross validation to train and evaluate our statistical LAT classifier. The results are shown in **Table 2**. The metrics are defined as follows:

$$Precision = \frac{\#Correctly\ Detected\ LATs}{\#Detected\ LATs}$$

$$Recall = \frac{\#Correctly\ Detected\ LATs}{\#LATs\ in\ Manually\ Annotated\ Set}$$

$$F_1 = \frac{2\ (Precision)\ (Recall)}{Precision + Recall}$$

$$Per\ Question\ Recall = \frac{\#Questions\ with\ at\ least\ one\ correctly\ detected\ LAT}{\#Questions\ with\ at\ least\ one\ manually\ annotated\ LAT}.$$

The results show that the baseline patterns are reasonably precise but are severely lacking in recall. Much of the work on Watson's focus and LAT detection has been directed toward improving the recall without sacrificing precision. The improvement in the *Per Question Recall* indicates that, in 6.5% of Jeopardy! questions, Watson detects a correct LAT that was not found by the baseline patterns. We expect that this will greatly improve Watson's chances of coming up with the correct answer to those questions.

One common class of error occurred when the human annotator chose a multiword term as a LAT, but Watson chose just the headword. This decision can be somewhat subjective according to our definition of LAT. Examples of

Table 3 Impact on question-answering system.

Focus/LAT	*QClass/QSection*	*Question-answering accuracy (%)*
Baseline	No	65.1
Baseline	Yes	67.5
Full	No	68.1
Full	Yes	71.0

disagreements were "singer" versus "lead singer" and "body" versus "legislative body". If the evaluation is relaxed so that a LAT is considered correct if it matches one of the words within a multiword gold-standard LAT, Watson's F_1 score increases by 0.02 and the baseline's increases by 0.01.

To directly measure the impact of accurate LAT detection (and, indirectly, that of focus detection) on the end-to-end question-answering system, we created a configuration of the Watson system in which the focus and LAT detection is replaced by the baseline patterns. These different system configurations are evaluated on an unseen test set of roughly 3,500 questions. The results are shown in **Table 3**. Comparing the second row (baseline focus and LAT detection) with the fourth row (Watson's full focus and LAT detection) shows that with its full focus and LAT detection, Watson answers an additional 3.5% of questions correctly. This improvement is statistically significant by McNemar's test [17] with $p < 0.01$.

Question Classification and QSection detection

The Jeopardy! domain includes a wide variety of kinds of questions, and we have found that a one-size-fits-all approach to answering them is not ideal. In addition, some parts of a question may play special roles and can benefit from specialized handling. One of the important jobs of question analysis is to identify these QClasses and QSections.

Question Classification

We have defined a set of QClasses to guide processing after question analysis. In some cases, a question may not be answerable at all by our default factoid answering approaches, and classification of the question into a specialized type is essential, for example:

13) BEFORE & AFTER: 13th Century Venetian traveler who's a Ralph Lauren short sleeve top with a collar.
14) THE SOUTHERNMOST CAPITAL CITY: Helsinki, Moscow, Bucharest.

In example 13, the correct answer ("Marco Polo shirt") will never even appear in our sources; therefore, our basic factoid question-answering approaches will never succeed. In example 14, the default factoid system would always fail because the correct answer ("Bucharest") appears in the question, which, for most questions, we do not allow.

Other classes of questions may not be impossible for the factoid approaches to answer but may still benefit from some processing that is not appropriate for general question types. **Table 4** lists Watson's QClasses along with their frequency, manually measured over a set of approximately 3,500 Jeopardy! questions.

We detect QClasses using a variety of techniques. The recognizers are independent of one another; hence, more than one QClass can be potentially associated with any particular question. There are some pairwise incompatibilities, which are enforced by a QClassConsolidator process that runs after recognition and removes the less preferred of any mutually inconsistent QClass pair.

The QClasses PUZZLE, BOND, FITB (Fill-in-the blank), and BOND, and MULTIPLE-CHOICE have fairly standard representations in Jeopardy! and are detected primarily by regular expressions. Presumably, because humans themselves require a signal in order to use different answering techniques, PUZZLE and BOND questions are almost always indicated by a familiar phrase (e.g., "BEFORE & AFTER", "ANAGRAMS", and "COMMON BONDS"), possibly with a domain-indicating modifier, in the category alone. Simple regular expression patterns are used to detect these and observed and anticipated synonyms (such as "JUMBLED" or "SCRAMBLED" for anagrams). FITB is detected by regular expressions over the question text, which match either underscores appearing in the question or (more typically) quoted phrases adjacent to the focus. MULTIPLE-CHOICE is detected either when the category provides a sequence of (usually three) items and the question provides a relation used to select the correct answer, or vice versa. More details on these "special" question types are provided in [3].

Other QClasses do not have such standard representations and are identified by syntactic rules that match over the PAS. The ETYMOLOGY QClass is triggered by the pattern "From ⟨language⟩ for ⟨phrase⟩, ⟨focus⟩ . . .," as well as many other similar patterns. The VERB QClass is detected by cases where the focus is the object of "do" (as shown in the example in Table 4) or in definition-style questions involving an infinitive (e.g., "This 7-letter word means to presage or forebode"). The TRANSLATION QClass can be triggered either by a pattern over the question PAS ("⟨focus⟩ is ⟨language⟩ for ⟨phrase⟩") or cases where the question is just a simple phrase with no focus and the category specifies a language to translate to or from.

The QClasses NUMBER and DATE are dependent only on the LAT and are detected when the LAT is a member of a manually created list of LATs that have numbers or dates as instances.

Table 4 QClasses.

QClass	*Description*	*Example questions (correct answer)*	*Frequency (%)*
DEFINITION	A question that contains a definition of the answer	CONSTRUCTION: It can be the slope of a roof, or the gunk used to waterproof it. (Answer: “pitch”) CONSTRUCTION: The name of this large beam that supports the joists literally means "something that encircles". (Answer: “a girder”)	14.2
CATEGORY-RELATION	The answer has a semantic relation to the question, where the relation is specified in the category	FORMER STATE GOVERNORS: Nelson A. Rockefeller. (Answer: “New York”) COUNTRIES BY NEWSPAPER: Haaretz, Yedioth Ahronoth. (Answer: “Israel”)	7.2
FITB	A fill-in-the-blank question asks for completion of a phrase	COMPLETE IT: Attributed to Lincoln: "The ___ is stronger than the bullet." (Answer: “ballot”) SHAKESPEARE IN LOVE: "Not that I loved Caesar less," says Brutus, "but that I loved" this city "more." (Answer: “Rome”)	3.8
ABBREVIATION	The answer is an expansion of an abbreviation in the question	MILITARY MATTERS: Abbreviated SAS, this elite British military unit is similar to the USA's Delta Force. (Answer: “the Special Air Service”)	2.9
PUZZLE	A puzzle question: the answer requires derivation, synthesis, inference, etc.	BEFORE & AFTER: 13th Century Venetian traveler who's a Ralph Lauren short sleeve top with a collar. (Answer: “Marco Polo shirt”) THE HIGHEST-SCORING SCRABBLE WORD: Zoom, quiz or heaven. (Answer: “quiz”)	2.3
ETYMOLOGY	A question asking for an English word derived from a foreign word having a given meaning	ARE YOU A FOOD"E"?: From the Spanish for "to bake in pastry", it's South America's equivalent of a calzone. (Answer: “an empanada”)	1.9
VERB	Question asks for a verb	THE NOT-SO-DEADLY SINS: To capitalize all text in an email is an abomination that signifies the person is doing this. (Answer: “shouting”)	1.5
TRANSLATION	A question asking for translation of a word or phrase from one language to another	FRUITS IN FRENCH: Pomme. (Answer: “apple”)	1.1
NUMBER	The answer is a number	YOU NEED TO CONVERT: One eighth of a circle equals this many degrees. (Answer: “45”)	1.0
BOND	The question asks for what is in common between a set of entities	EDIBLE COMMON BONDS: Mung, snap, string. (Answer: “bean”)	0.7
MULTIPLE-CHOICE	The question contains multiple possible answers from which to choose the correct answer	THE SOUTHERNMOST CAPITAL CITY: Helsinki, Moscow, Bucharest. (Answer: “Bucharest”) OSCAR, GRAMMY OR BOTH: Mickey Rooney. (Answer: “Oscar”)	0.5
DATE	A question asking for a date or year	THE TEENS: World War I ended in November of this year. (Answer: “1918”)	0.3

The DEFINITION QClass is difficult to identify with great accuracy, but this is compensated for by the fact that our standard question-answering approach is reasonably good at handling these questions; hence, detection accuracy is not as critical as it is for the other QClasses. Definition questions are identified by key phrases such as "is the word for" and "means", as well as a small number of recurring category types such as "CROSSWORD CLUES".

The CATEGORY-RELATION QClass is detected when the question text is just a single entity or a short list of entities (there is no focus detected). This QClass is suppressed if a more specific QClass applies.

Of particular interest is the detection of the ABBREVIATION QClass, which applies to questions in which the expansion of an abbreviation is sought. Consider the following examples:

15) CHEWING THE "FAT": Abbreviated CFS, this medical condition is called "the thief of vitality."
16) ABBREV.: On a tombstone: RIP.
17) YANKEE MAGAZINE: An article called "A Tip of the Hat to Danbury," in this state, tells how JFK helped kill an industry with his bareheaded ways.

Watson annotates the examples in 15 and 16 with the ABBREVIATION QClass and identifies "CFS" and "RIP" as the abbreviations to be expanded, respectively. In example 17, JFK is also recognized as an abbreviated term; however, this question is not annotated with the ABBREVIATION QClass. This distinction is crucial as the expansion of any abbreviated term can be either the correct answer (examples 15 and 16) or a very stupid answer (example 17).

We recognize the ABBREVIATION QClass in a way that is similar to our LAT detection approach: rule-based recognition combined with a statistical classifier that makes the final decision primarily on the basis of the learned reliability of each rule. The rule-based recognizer includes regular expression patterns that capture canonical ways that abbreviation questions may be expressed in Jeopardy! (example 16), as well as rules that match against the PAS of the question, to capture more complex syntactic variations for questions that seek abbreviation expansions (example 15). We train a logistic regression model, in which each of these rules is used as a binary feature, along with features that indicate the number of previously seen abbreviation questions in the same category. Questions that are classified in the ABBREVIATION QClass employ a special solving technique to produce expansions as candidate answers, as described in [3].

QSections

A QSection is an annotation made over a contiguous span of text in the question (occasionally, in the category) to represent a function that the text plays in the interpretation of the question. Like QClasses, QSections are identified either by Prolog rules over the PAS or by regular expressions over the text. In some cases, QClasses and QSections are recognized and output by the same component because their existence and function are codependent.

Altogether, many types of QSections are recognized. Some of the more important are as follows.

- LexicalConstraint—A phrase such as "this 4-letter word" that should not be used in a query but is critical for selecting the correct answer.
- Abbreviation—A term in a question that is identified as an abbreviation, which is associated with its possible expansions. For questions of QClass ABBREVIATION, one of these Abbreviation QSections is further identified as the abbreviation whose expansion is sought.
- SubQuestionSpan—When a question can be decomposed into two or more disjoint sections that individually indicate or contribute to the answer, these sections are marked as SubQuestionSpans. Some of the more complex Factoids, as well as all "BEFORE & AFTER"s, most "RHYME TIME"s, and double definitions such as the first DEFINITION example in Table 4, get SubQuestionSpan annotations [3, 5].
- McAnswer—The (usually three) strings that represent the answer choices in a multiple-choice question are marked with a McAnswer QSection.
- FITB—This annotates the string that adjoins the focus term (i.e., the text that forms the term or expression that the focus completes).

Evaluation

We evaluated the accuracy of our Question Classification against a manually annotated gold standard of approximately 3,500 questions. The results are shown in **Table 5**. Over all the QClasses, we achieved an *F* measure of 0.637. Individually, many of the QClasses have a much higher *F* measure. The class with the lowest detection *F* measure is DEFINITION, perhaps because it is one of the more subjective classifications. Errors in DEFINITION detection often do not hurt, however, because definition questions and factoid questions are similar enough that they can be often correctly answered even with an incorrect QClass identification. Also of note is that the PUZZLE class has low recall because there are many infrequently occurring types of puzzles for which it was not worthwhile to develop specialized detection mechanisms.

To measure the impact on the end-to-end question-answering system, we compared the full Watson system to a configuration that had the QClass and QSection detection entirely removed (baseline system). The results are shown in Table 3. Comparing the third and fourth rows shows that with its full QClass and QSection detection,

Table 5 Question Classification evaluation.

QClass	*Precision*	*Recall*	F_1
DEFINITION	0.508	0.487	0.497
CATEGORY-RELATION	0.644	0.806	0.716
FITB	0.676	0.711	0.693
ABBREVIATION	0.728	0.806	0.765
PUZZLE	0.977	0.525	0.683
ETYMOLOGY	0.886	0.600	0.716
VERB	0.796	0.811	0.804
TRANSLATION	0.885	0.590	0.708
NUMBER	0.842	0.471	0.604
BOND	1.000	0.652	0.789
MULTIPLE-CHOICE	0.650	0.684	0.667
DATE	0.692	0.818	0.750
All	0.646	0.629	0.637

Watson answers an additional 2.9% of questions correctly. This difference is statistically significant by McNemar's test with $p < 0.01$.

Related work

It is common in question-answering systems to represent a question as a graph either of syntactic relations in a parse or PAS [18–20] or of deep semantic relations in a handcrafted ontology [21–23]. Watson uses both approaches.

The concept of a question focus has been used in prior work [24] but often does not have a clear definition. The most comprehensive discussion of focus detection can be found in Bunescu and Huang [25], who define a focus as "the set of all maximal noun phrases in the question that co-refer with the answer." They also state that these focus words can be used to influence the selection of the answer type. They do not have a separate concept of a LAT and, thus, do not mention a way for a word to not be a focus but still influence the answer type. We have found it essential to make the focus/LAT distinction because of Watson's heterogeneous set of answer-scoring algorithms, some of which attempt to align an answer in a passage with the focus in the question, and others that use the LAT to determine whether the candidate answer is of the correct type. For example, in the question "He was elected president in 1868," the word "president" is clearly a LAT, but we do not want to consider it a focus since it would be wrong to align it with an answer in a passage (we do not expect the answer to this question to appear as the complement of "elect").

Most other question-answering systems use question analysis to identify a semantic answer type from a fixed ontology of known types [19, 26–29]. Because of the very broad domain that Jeopardy! questions cover, this is not practical. Instead, we employ a separation of concerns where question analysis identifies the LAT, but the type coercion components determine semantics. The work that is perhaps most similar to our concept of LAT is that of Pinchak and Lin [30], who directly use the words in the question for answer typing, although they have one specific algorithm for evaluating candidate answers on the basis of how often they appear in the same context in the corpus, whereas Watson identifies LATs independently of how they are used and then applies a suite of type coercion algorithms [2].

In prior work, there is often not a clear distinction between Question Classification and answer type detection. For example, some systems [26, 27] have QClasses for abbreviation expansion (which we also consider a QClass), as well as various types of Human and Entity (which we cover under LAT detection). We treat these concepts separately since a question such as "This television network is abbreviated ABC" is an abbreviation expansion question that also specifies the semantic type of the entity. Question Classification and QSection detection to the extent done in Watson has not been the subject of previous work, mainly because the TREC question sets [8] used by most of the prior question-answering work do not exhibit the wide variety of QClasses that Jeopardy! does [10].

Conclusion

In this paper, we have presented the processes by which Watson analyzes a question. Question analysis is built on a foundation of general-purpose parsing and semantic analysis, although the style of Jeopardy! questions and the requirements of question analysis have necessitated some adaptation of those capabilities. Question analysis is primarily concerned with the identification of four critical elements, namely, focus, LAT, Question Classification, and

QSection. This is a more comprehensive view of question analysis than that covered by prior work, which includes only some of these elements and often conflates them.

Many of the requirements of question analysis are handled by pattern matching over data structures from parsing and semantic analysis. We chose to use Prolog for this task since it is a well-established standard that is very flexible and well suited for representing pattern-matching rules. Our implementation is efficient enough to analyze a question in a fraction of a second, which is critical for competing at Jeopardy!.

We have evaluated our implementation in terms of its performance on the LAT detection and Question Classification tasks, as well as its impact on Watson's end-to-end question-answering accuracy. For the evaluation, we proposed a baseline question analysis implementation based on easily observable patterns from Jeopardy! questions and have shown that our question analysis provides a significant improvement in the question-answering accuracy over this baseline.

In all, the Watson system with its full question analysis capabilities correctly answers an additional 5.9% of questions versus the system that includes baseline focus and LAT detection and no QClass and QSection detection. On the basis of our analysis of human Jeopardy! performance [10], an accuracy reduction of this magnitude would substantially degrade Watson's chances to win a Jeopardy! game against good human competition.

*Trademark, service mark, or registered trademark of International Business Machines Corporation in the United States, other countries, or both.

**Trademark, service mark, or registered trademark of Jeopardy Productions, Inc., or Trustees of Princeton University in the United States, other countries, or both.

References

1. J. W. Murdock, J. Fan, A. Lally, H. Shima, and B. K. Boguraev, "Textual evidence gathering and analysis," *IBM J. Res. Develop.*, vol. 56, no. 3/4, Paper 8, pp. 8:1–8:14, May/Jul. 2012.
2. J. W. Murdock, A. Kalyanpur, C. Welty, J. Fan, D. Ferrucci, D. C. Gondek, L. Zhang, and H. Kanayama, "Typing candidate answers using type coercion," *IBM J. Res. Develop.*, vol. 56, no. 3/4, Paper 7, pp. 7:1–7:13, May/Jul. 2012.
3. J. M. Prager, E. W. Brown, and J. Chu-Carroll, "Special questions and techniques," *IBM J. Res. Develop.*, vol. 56, no. 3/4, Paper 11, pp. 11:1–11:13, May/Jul. 2012.
4. D. C. Gondek, A. Lally, A. Kalyanpur, J. W. Murdock, P. Duboue, L. Zhang, Y. Pan, Z. M. Qiu, and C. Welty, "A framework for merging and ranking of answers in DeepQA," *IBM J. Res. Develop.*, vol. 56, no. 3/4, Paper 14, pp. 14:1–14:12, May/Jul. 2012.
5. A. Kalyanpur, S. Patwardhan, B. K. Boguraev, A. Lally, and J. Chu-Carroll, "Fact-based question decomposition in DeepQA," *IBM J. Res. Develop.*, vol. 56, no. 3/4, Paper 13, pp. 13:1–13:11, May/Jul. 2012.
6. L. Sterling and E. Shapiro, *The Art of Prolog—Advanced Programming Techniques,* 2nd ed. Cambridge, MA: MIT Press, 1994.
7. M. Covington, *Natural Language Processing for Prolog Programmers*. Englewood Cliffs, NJ: Prentice-Hall, 1994.
8. E. Voorhees, "Overview of the TREC 2002 question answering track," in *Proc. 11th Text REtrieval Conf.*, 2002, pp. 115–123.
9. D. Giampiccolo, P. Froner, A. Peñas, C. Ayache, D. Cristea, V. Jijkoun, P. Osenova, P. Rocha, B. Sacaleanu, and R. Suteliffe, "Overview of the CLEF 2007 multilingual question answering track," in *Proc. Cross Language Eval. Forum—Advances in Multilingual and Multimodal Information Retrieval*, vol. 5152, *Lecture Notes In Computer Science*, C. Peters, V. Jijkoun, T. Mandl, M. Henning, D. W. Oard, P. Anselmo, V. Petras, and D. Santos, Eds. Berlin, Germany: Sringer-Verlag, 2007, pp. 200–236.
10. D. Ferrucci, E. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. Kalyanpur, A. Lally, J. W. Murdock, E. Nyberg, J. Prager, N. Schlaefer, and C. Welty, "Building Watson: An overview of the DeepQA project," *AI Mag.*, vol. 31, no. 3, pp. 59–79, 2010.
11. M. C. McCord, J. W. Murdock, and B. K. Boguraev, "Deep parsing in Watson," *IBM J. Res. Develop.*, vol. 56, no. 3/4, Paper 3, pp. 3:1–3:15, May/Jul. 2012.
12. C. Wang, A. Kalyanpur, J. Fan, B. K. Boguraev, and D. C. Gondek, "Relation extraction and scoring in DeepQA," *IBM J. Res. Develop.*, vol. 56, no. 3/4, Paper 9, pp. 9:1–9:12, May/Jul. 2012.
13. L. Vita, A. Ittycheriah, S. Roukos, and N. Kambhatla, "tRuEcasing," in *Proc. ACL*, Sapporo, Japan, 2003, pp. 152–159.
14. A. Kalyanpur, B. K. Boguraev, S. Patwardhan, J. W. Murdock, A. Lally, C. Welty, J. M. Prager, B. Coppola, A. Fokoue-Nkoutche, L. Zhang, Y. Pan, and Z. M. Qiu, "Structured data and inference in DeepQA," *IBM J. Res. Develop.*, vol. 56, no. 3/4, Paper 10, pp. 10:1–10:14, May/Jul. 2012.
15. D. Ferrucci and A. Lally, "Building an example application with the unstructured information management architecture," *IBM Syst. J.*, vol. 43, no. 3, pp. 455–475, 2004.
16. C. Fellbaum, *WordNet: An Electronic Lexical Database*. Cambridge, MA: MIT Press, 1998.
17. Q. McNemar, "Note on the sampling error of the difference between correlated proportions or percentages," *Psychometrika*, vol. 12, no. 2, pp. 153–157, Jun. 1947.
18. U. Hermjakob, E. H. Hovy, and C. Lin, "Knowledge-based question answering," in *Proc. 6th World Multiconf. Syst., Cybern. Inform.*, 2002. [Online]. Available: http://research.microsoft.com/en-us/people/cyl/sci2002.pdf
19. J. Chu-Carroll, J. Prager, C. Welty, K. Czuba, and D. Ferrucci, "A multi-strategy and multi-source approach to question answering," in *Proc. Text REtreival Conf.*, 2003. [Online]. Available: http://trec.nist.gov/pubs/trec11/papers/ibm.prager.pdf
20. D. Moldovan, C. Clark, S. Harabagiu, and S. Maioran, "COGEX: A logic prover for question answering," in *Proc. HLT-NAACL*, 2003, pp. 87–93.
21. W. Lehnert, "Human and computational question answering," *Cognit. Sci.*, vol. 1, no. 1, pp. 47–73, Jan. 1977.
22. D. B. Lenat, "CyC: A large-scale investment in knowledge infrastructure," *Commun. ACM*, vol. 38, no. 11, pp. 33–38, Nov. 1995.
23. K. Forbus and P. Paritosh, "Analysis of strategic knowledge in back of the envelope reasoning," in *Proc. 20th Nat. Conf. AAAI*, vol. 2, A. Cohn, Ed., 2005, vol. 2, pp. 651–656.
24. J. M. Prager, "Open-domain question answering," *Found. Trends Inform. Retrieval*, vol. 1, no. 2, pp. 91–231, 2006.
25. R. Bunescu and Y. Huang, "Towards a general model of answer typing: Question focus identification," in *Proc. 11th Int. Conf. Intell. Text Process. Comput. Linguist.*, Iasi, Romania, 2010. [Online]. Available: http://ace.cs.ohiou.edu/~razvan/papers/cicling10.pdf
26. X. Li and D. Roth, "Learning question classifiers," in *Proc. 19th Int. Conf. Comput. Linguist.*, 2002, vol. 1, pp. 1–7.
27. D. Zhang and W. S. Lee, "Question classification using support vector machines," in *Proc. 26th Annu. Int. ACM SIGIR Conf. Res. Dev. Inform. Retrieval*, 2003, pp. 26–32.

28. N. Schlaefer, J. Ko, J. Betteridge, G. Sautter, M. Pathak, and E. Nyberg, "Semantic extensions of the Ephyra QA system for TREC," in *Proc. TREC*, 2007. [Online]. Available: http://www.cs.cmu.edu/~nico/pubs/trec2007_schlaefer.pdf
29. A. Mikhailian, T. Dalmas, and R. Pinchuk, "Learning foci for question answering over topic maps," in *Proc. ACL-IJCNLP Conf.*, Suntec, Singapore, 2008, pp. 325–328.
30. C. Pinchak and D. Lin, "A probabilistic answer type model," in *Proc. 11th Conf. Eur. Chapter Assoc. Comput. Linguist.*, Trento, Italy, 2006, pp. 393–400.

Received August 2, 2011; accepted for publication January 10, 2012

Adam Lally *IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY 10598 USA (alally@us.ibm.com).* Mr. Lally is a Senior Technical Staff Member at the IBM T. J. Watson Research Center. He received the B.S. degree in computer science from Rensselaer Polytechnic Institute, Troy, NY, in 1998 and the M.S. degree in computer science from Columbia University, New York, NY, in 2006. As a member of the IBM DeepQA Algorithms Team, he helped develop the Watson system architecture that gave the machine its speed. He also worked on the natural-language processing algorithms that enable Watson to understand questions and categories and gather and assess evidence in natural language. Before working on Watson, he was the lead software engineer for the Unstructured Information Management Architecture project, an open-source platform for creating, integrating, and deploying unstructured information management solutions.

John M. Prager *IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY 10598 USA (jprager@us.ibm.com).* Dr. Prager has been working in technical fields related directly or indirectly to question answering for most of his professional career. Most recently, while at the T. J. Watson Research Center, he has been part of the Watson project, building a system that plays the Jeopardy! quiz-show game. He has been involved in both the algorithms area, in particular working on puns and other wordplay, and the strategy area. Previously, he led IBM's successful entries in Text REtrieval Conference Question-Answering (TREC-QA) tasks, an annual evaluation at the National Institute of Standards and Technology (NIST). Prior to that, he worked in various areas of search, including language identification, web search, and categorization. He has contributed components to the IBM Intelligent Miner for Text product. For a while in the early 1990s, he ran the search service on www.ibm.com. While at the IBM Cambridge Scientific Center, Cambridge, MA, he was the Project Leader of the Real-time Explanation and Suggestion project, which would provide users with help by taking natural-language questions and processing them with an inference engine tied to a large repository of facts and rules about network-wide resources. He has degrees in mathematics and computer science from the University of Cambridge, Cambridge, U.K. and in artificial intelligence from the University of Massachusetts, Lowell; his publications include conference and journal papers, nine patents, and a book on Alan Turing.

Michael C. McCord *IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY 10598 USA (mcmccord@us.ibm.com).* Dr. McCord is a Research Staff Member in the Semantic Analysis and Integration Department at the T. J. Watson Research Center. He received the Ph.D. degree in mathematics from Yale University, New Haven, CT, and spent a year at the Institute for Advanced Study. His initial research was in mathematics (topology and foundations), and he then moved into computer science and natural-language processing, with emphasis on syntax and semantics. He has been at IBM Research since 1983. He originated the Slot Grammar parsing system, which has been applied to machine translation and grammar checking, and which is used in the Watson question-answering system. He is author or coauthor of 47 refereed articles and one book.

Branimir K. Boguraev *IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY 10598 USA (bran@us.ibm.com).* Dr. Boguraev is a Research Staff Member in the Semantic Analysis and Integration Department at the Thomas J. Watson Research Center. He received the Engineering degree in electronics from the Higher Institute for Mechanical and Electrical Engineering, Sofia, Bulgaria, in 1974, and the diploma and Ph.D. degrees in computer science (1976) and computational linguistics (1980), respectively, from the University of Cambridge, Cambridge, U.K. He worked on a number of U.K./E.U. research projects on infrastructural support for natural-language processing applications, before joining IBM Research in 1988 to work on resource-rich text analysis. From 1993 to 1997, he managed the natural-language program at Apple's Advanced Technologies Group, returning to IBM in 1998 to work on language engineering for large-scale, business content analysis. Most recently, he has worked, together with the Jeopardy! Challenge Algorithms Team, on developing technologies for advanced question answering. He is author or coauthor of more than 120 technical papers and 15 patents. Until recently, he was the Executive Editor of the Cambridge University Press book series *Studies in Natural Language Processing*. He has also been a member of the editorial boards of *Computational Linguistics* and the *Journal of Semantics*, and he continues to serve as one of the founding editors of *Journal of Natural Language Engineering*. He is a member of the Association for Computational Linguistics.

Siddharth Patwardhan *IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY 10598 USA (siddharth@us.ibm.com).* Dr. Patwardhan is a Post-Doctoral Researcher in the Knowledge Structures Group at the T. J. Watson Research Center. He received the B.E. degree in computer engineering from the University of Pune, Pune, India, in 2001, the M.S. degree in computer science from the University of Minnesota, Duluth, in 2003, and the Ph.D. degree in computer science from the University of Utah, Salt Lake City, in 2010. He has been working at the IBM T. J. Watson Research Center since 2009, exploring research projects in natural-language processing and artificial intelligence. He is a member of the Algorithms Team working on the IBM Jeopardy! challenge and is an author or coauthor of more than 25 technical publications covering his work on information extraction, opinion/sentiment analysis, computational lexical semantics, and question answering. Dr. Patwardhan is a member of the Association for Computational Linguistics and a member of the Association for the Advancement of Artificial Intelligence.

James Fan *IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY 10598 USA (fanj@us.ibm.com).* Dr. Fan is a Research Staff Member in the Semantic Analysis and Integration Department at the T. J. Watson Research Center, Yorktown Heights, NY. He joined IBM after receiving the Ph.D. degree at the University of Texas at Austin, Austin, in 2006. He is a member of the DeepQA Team that developed the Watson question-answering system, which defeated the two best human players on the quiz show Jeopardy!. He is author or coauthor of dozens of technical papers on subjects of knowledge representation, reasoning, natural-language processing, and machine learning. He is a member of Association for Computational Linguistics.

Paul Fodor *Computer Science Department, Stony Brook University, Stony Brook, NY 11794 USA (pfodor@cs.stonybrook.edu).* Dr. Fodor is a Research Assistant Professor in the Computer Science Department at Stony Brook University (SUNY of New York), Stony Brook. He received the B.S. degree in computer science from the Technical University of Cluj-Napoca, Cluj-Napoca, Romania, in 2002, and the M.S. and Ph.D. degrees in computer Science from the Stony Brook University, in 2006 and 2011, respectively. His work on declarative rule languages and logic used as a specification language and implementation framework for knowledge bases was applied in areas ranging from natural language processing to complex event processing and the Semantic Web. Through his research, he has contributed to several large software projects: IBM Watson, OpenRuleBench suite of benchmarks for analyzing the performance and scalability of rule systems for the Semantic Web, ETALIS declarative complex event processing, and SILK Semantic Inferencing on Large Knowledge. He is an author or coauthor of 12 technical papers.

Jennifer Chu-Carroll *IBM Research Division, Thomas J. Watson Research Center, Yorktown Heights, NY 10598 USA (jencc@us.ibm.com).* Dr. Chu-Carroll is a Research Staff Member in the Semantic Analysis and Integration Department at the T. J. Watson Research Center. She received the Ph.D. degree in computer science from the University of Delaware in 1996. Prior to joining IBM in 2001, she spent 5 years as a Member of Technical Staff at Lucent Technologies Bell Laboratories. Her research interests are in the area of natural-language processing, more specifically in question-answering and dialogue systems. Dr. Chu-Carroll serves on numerous technical committees, including as program committee co-chair of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT) 2006 and as general chair of NAACL HLT 2012.