USING SEMANTIC REPRESENTATIONS IN QUESTION ANSWERING

Sameer S. Pradhan, Valerie Krugler, Wayne Ward, Dan Jurafsky and James H. Martin

Center for Spoken Language Research
University of Colorado
Boulder, CO 80309-0594, USA

ABSTRACT

This paper describes the architecture of a Question Answering system for answering complex questions that require the integration of various techniques such as Robust Semantics, Event Detection, Information Fusion and Summarization. The focus of the paper is on a general Semantic Representation and how it can be used in the question answering process. We describe the vision for the system, report on its current state of development, and evaluate its accuracy on TREC-9 and TREC-10 questions. We also discuss a confidence annotation scheme and evaluate it using the NIST scoring metric.

1. INTRODUCTION

The Center for Spoken Language Research (CSLR) at the University of Colorado and Columbia University are collaborating to develop a new technology for question answering. This project is supported by the ARDA AQUAINT (Advanced QUestion and Answering for INTelligence) program. The project proposes to integrate robust semantics, event detection, information fusion, and summarization technologies to enable a multimedia question answering system. The goal is to develop a system capable of answering complex questions; these are questions that require interacting with the user to refine and clarify the context of the question, whose answer may be located in non-homogeneous databases of speech and text, and for which presenting the answer requires combining and summarizing information from multiple sources and over time. Generating a satisfactory answer to complex questions requires the ability to collect all relevant answers from multiple documents in different media, weigh their relative importance, and generate a coherent summary of the multiple facts and opinions reported.

We propose to integrate four core technologies:

- ▷ **Semantic annotation (CSLR)** – We use a shallow, domain-independent, semantic representation and a statistical semantic parser for building this representation. The semantic representation is a basic building block for dialog management, event detection, and information fusion.

- ▷ **Context management (CSLR)** – We are developing a dialogue interface to allow the system to carry on a focused dialogue with users to answer queries. The interface will maintain context through the interaction to allow followup questions and will conduct clarification dialogues with the user to clarify ambiguities and refine queries.

- ▷ **Event recognition and information tracking (Columbia)** – An event is an activity with a starting and ending point, involving a fixed set of participants. We propose to identify atomic events within each input document by extracting information about named entities that play a prominent role in the document and about the time period that the text covers. We will rely on the semantic representation of documents to allow us to identify participants and their functions.

This work was supported by ARDA through the AQUAINT program

▷ **Information fusion and summary generation (Columbia)** – Rather than listing a set of relevant responses to a question, we will investigate techniques to integrate summarization and language generation to produce a brief, coherent, and fluent answer. Critical to this task is the problem of selecting fragments of text from different documents that should be included in the answer and determining how to combine them, removing redundancy and integrating complementary information fluently.

This paper will focus on the domain-independent semantic representation and how we propose to use it for question answering applications.

2. CUAQ SYSTEM

2.1. Semantic Representation

The novel feature of our approach is the use of shallow semantic representations to enhance potential answer identification. Most successful systems first identify a list of potential answer candidates using pure word-based metrics. Varying granularity of syntactic and semantic information is then used to re-rank those candidates [8, 6]. However, most of these semantic units are quite specific in what they label. We identify a small set of *thematic roles* – viz., *agent*, *patient*, *manner*, *degree*, *cause*, *result*, *location*, *temporal*, *force*, *goal*, *path*, *percept*, *proposition*, *source*, *state*, and *topic*, in the candidate answer sentences, using a statistical classifier [5]. The classifier is trained on the FrameNet database [2]. This is an online lexical resource for English, based on *frame semantics*. It contains hand-tagged semantic annotations of example sentences from a large text corpora, and covers thousands of lexical items – verbs, nouns, and adjectives – representative of a wide range of semantic domains. We map the more specific *frames* and the corresponding *frame elements* to the reduced, more general set before training the classifier.

2.2. Architecture

We plan to use two generally available Information Retrieval search engines: 1) `mg` (Managing Gigabytes) [9], and 2) Google [3]
The following sequence of actions will be taken in response to an input query:

1. Question type classification – Identify the Named Entity and Thematic Role of the expected answer type. This also defines a set of answer type patterns, and includes named entity tagging and parsing the question for thematic roles.
2. Focus identification – Identify certain salient words/phrases in question that are very likely to be present in the answer string in one form or the other.
3. Extract a set of query words from the question, and apply semantic expansion to them.
4. Submit the query words to the IR engine and get back a rank-ordered set of documents.
5. Keep the top N (approximately 500) documents and prune the rest.
6. Segment documents into paragraphs and prune all but top N' paragraphs.
7. Generate scoring features for the paragraphs, including named entity tagging and parsing of paragraphs to add thematic roles.
8. Re-rank documents based on the set of features that we compute, including answer type patterns. Some of the answer type patterns are based on the semantic labels.
9. Compute confidence for each paragraph (that it contains some relevant information). This includes N-Best count as one of the features.
10. Send tagged paragraphs that exceed a confidence threshold, for summarization.

For the problem of question answering, we are more concerned with precision than recall, so we have to be careful in expanding the query words to get answers that are expressed in words quite different from the ones mentioned in the question. Semantic expansion will be performed when the system's confidence in the best candidate answer string – without expansion, is found to be below a certain threshold. Our mechanism for expansion is:

a. Submit original query words to IR engine and get back a rank-ordered set of documents.
b. Generate set of target words from top n documents based on the *idf* values of the words.
c. Generate a set of target words from WordNet [4] synsets of original keywords.
d. Take the intersection of the two sets and add to the keyword set.

2.3. Features

2.3.1. Answer Identification

We now discuss the features used for ranking the documents. The features are roughly ordered by decreasing salience.

- **Answer type** – In order to extract the answer to a question, the system needs to identify the expected answer type. Answers for short answer questions generally can be categorized as named entities, and/or propositions. Summary information is often required for descriptive and definition questions. The system classifies the answer type by two features: 1) named entity class and 2) thematic role. Named entity class specifies one (or more) of 56 classes as the named entity class of the answer. We use 54 real named entity classes, one class representing the case where the answer is expected to be a named entity but one not known to the system, and one class for cases where the answer is not expected to be a named entity. The thematic role class identifies the thematic role in which the potential answer would tend to be found in the answering sentence.

- **Answer Surface patterns** – Once the question type is known the next step is to identify candidate answers. One technique we use is based on generating a set of expected surface patterns for the answer. The patterns specify word and named entity based regular expressions that are derived from a large corpus annotated with named entities. Some examples of surface pattern types are:

  1. Some common question types, e.g., `[<PERSON_DESCRIPTION><PERSON_NAME>]`[1], for questions like “Who is `<PERSON_NAME>`?”; `[<ORGANIZATION>, <CITY>, <STATE>, <COUNTRY>]`, for questions asking about location/address of an organization.
  2. Likely re-phrasings of the question, e.g., `[<PERSON> invented <PRODUCT>]`, for questions like “Who was the inventor of `<PRODUCT>`?”
  3. Occurrence statistics of the pattern in the corpus, e.g., `[<PERSON> (<YEAR>-<YEAR>)]`, for birth dates of people.

  Sentences or snippets matching these patterns would get better scores than ones that did not.

- **Named Entities in Answer** – In the case of questions that expect a specific named entity (including the unknown named entity type) as the answer, candidates that do not contain that named entity are penalized. In the case that the answer is expected to be an unknown named entity, then candidates that contain an untagged sequence of capitalized words (a strong indicator of unknown named entities) are preferred.

- **Presence of focus word** – The presence of the focus word is an important feature for the overall score. For our purposes, a focus word is a word in the question that, or its synonym, is very likely to appear in the sentence that contains the answer. Even if all the other keywords are present in a candidate answer, but the focus word is not present, then it is not a sufficiently good candidate. The focus word can also be considered as a necessary, but not a sufficient condition for a string to qualify as a candidate answer.

  Question: How many people die from snakebite poisoning in the U.S. per year?
  Keywords: people die from snakebite poisoning U.S. per year

  Answer: About 10 people die a year from snakebites in the United States, and most were either handling or otherwise fooling around with the snakes, according to Plock.

  An answer to the above question will contain *snakebite* or its synonym, so it is the focus word. In this instance, when the focus word was not used in the ranking process, the system ranked 16 documents that did not contain the word *snakebite* before the first one that contained it, and which was the one containing the

[1] all regular expressions simplified for clarity.

right answer. However, it is not always easy to identify the focus word in a question with high probability, and there are instances when the focus word might not be present directly in the answer string, and some form of world knowledge is required to realize its presence.

▷ **Thematic Role patterns** – While surface patterns for answers can provide valuable information when a match is found, the specific nature of the patterns and the limited occurrences of the answer string within the reformulations obtainable from the question does not always guarantee a surface pattern match. We also provide a more general set of expected answer patterns based on thematic roles. We expect that these patterns will have higher coverage than the more specific surface patterns. This feature scores sentences based on the presence of expected thematic roles and named entities existing in specific thematic roles.

Thematic role patterns serve two main purposes. Firstly, they help identify false positive answer candidates that are ranked above the correct answer candidate, and secondly, they help extract the exact answer boundary from the string, especially if the answer is not a named entity. This can be illustrated with the following example[2]:

Question: Who assassinated President McKinley?
Parse: [$_{role=agent}$ Who] [$_{target}$ *assassinated*] [$_{role=patient}$ [$_{ne=person\_description}$ **President**] [$_{ne=person}$ **McKinley**]]?
Keywords: assassinated President McKinley
Answer named entity (ne) Type: Person
Answer thematic role (role) Type: Agent of target synonymous with "assassinated"
Thematic role pattern[3]: [$_{role=agent}$ [$_{ne=person}$ `ANSWER`] ∧ [$_{target}$ `synonym_of(assassinated)`] ∧ [$_{role=patient}$ [$_{ne=person}$ `reference_to(President McKinley)`]]

False Positives:
`Note: The sentence number indicates the final rank of that sentence in the returns, without using thematic role patterns.`

1. In [$_{ne=date}$ **1904**], [$_{ne=person\_description}$ **President**] [$_{ne=person}$ **Theodore Roosevelt**], who had succeeded the [$_{target}$ *assassinated*] [$_{role=patient}$ [$_{ne=person}$ **William McKinley**]], was elected to a term in his own right as he defeated [$_{ne=person\_description}$ **Democrat**] [$_{ne=person}$ **Alton B. Parker**].
4. [$_{ne=person}$ **Hanna**]'s worst fears were realized when [$_{role=patient}$ [$_{ne=person\_description}$ **President**] [$_{ne=person}$ **William McKinley**]] was [$_{target}$ *assassinated*], but the country did rather well under TR's leadership anyway.
5. [$_{ne=person}$ **Roosevelt**] became president after [$_{role=patient}$ [$_{ne=person}$ **William McKinley**]] was [$_{target}$ *assassinated*] [$_{role=temporal}$ in [$_{ne=date}$ **1901**]] and served until [$_{ne=date}$ **1909**].

Correct Answer:

8. [$_{role=temporal}$ In [$_{ne=date}$ **1901**]], [$_{role=patient}$ [$_{ne=person\_description}$ **President**] [$_{ne=person}$ **William McKinley**]] was [$_{target}$ *shot*] [$_{role=agent}$ by [$_{ne=person\_description}$ **anarchist**] [$_{ne=person}$ **Leon Czolgosz**]] [$_{role=location}$ at the [$_{ne=event}$ **Pan-American Exposition**] in [$_{ne=us\_city}$ **Buffalo**] , [$_{ne=us\_state}$ **N.Y.**]] [$_{ne=person}$ **McKinley**] died [$_{ne=date}$ **eight days later**].

▷ **Okapi scoring** – This is the score assigned by the information retrieval engine to the paragraph extracted in the very beginning. The formula for calculating the score [7] uses features like the number of query words present in the document, inverse document frequency of the words, the length of the document, etc. We update the Okapi score for the sentences after extracting them from paragraphs – that is, if the system expects it to be a short answer and decides on reducing the granularity below the paragraph level.

[2] All the named entities, but only roles pertaining to the *target* predicate are marked in the sentences.
[3] This is one of possibly more than one patterns that will be applied to the answer candidates.

- ▷ **N-gram** – Another feature that we use is based on the length of the longest n-gram (sequence of contiguous words) in the candidate answer sentences after removing the stopwords from both the question and the answer.
- ▷ **Case match** – Documents with words in the same case tend to be more relevant than those that do not have the same case words in them, so the former are given a relatively higher score.

2.3.2. Confidence Annotation

Once the likely answer candidates (paragraphs or sentences) are extracted for a question we need to estimate the likelihood of those being *good* answers. To do this, we have a scheme that annotates each with some level of confidence. The features that we use to estimate this confidence are:

- ▷ **N-Best Count** – This is the frequency distribution of n extracted answer strings in the top N final answer candidates. We calculate prior statistics on the probability that an answer is correct if it occurs n times in the top N candidates.
- ▷ **Named Entity Class** – The system tends to answer questions about some named entity classes more accurately than others. We calculate prior statistics for the percent correct answers for each named entity class.
- ▷ **N-gram Length** – We calculate prior statistics for the probability that an answer is correct if it contains an n-gram of length n .

These features are combined as a weighted linear combination of the individual estimates.

3. CURRENT IMPLEMENTATION AND RESULTS

In this section we will discuss the state of the system currently implemented, which was used in the Text REtrieval Conference (TREC) 2002 question answering track.

3.1. TREC-2002 Database

The text database comprises non-stemmed, case-folded indexing of the TREC/AQUAINT corpus using a modified version of the `mg` (Managing Gigabytes) search engine [9], that incorporates the Okapi BM25 ranking scheme [7], and an expanded set of characters forming an indexed unit – so as to accommodate hyphenated words, URLs, emails etc. Each indexed document is a collection of segmented sentences forming a paragraph in the original corpus. For efficiency reasons, we use another index of the same documents pre-tagged with 29 named entities using BBN's Identifinder [1] and another rule-based tagger that tags some specific named entity sub-categories inside those identified by Identifinder, and some other closed-class named entities that have been frequent answer types in the previous TREC questions, bringing the total to 54.

3.2. System Components

In this section, we will highlight the details of the current implementation.

3.2.1. Answer Identification

We currently use a rule-based question classifier that identifies the named entity and thematic role of the expected answer to each question. Keywords are extracted from the questions by removing a small set of stopwords and some punctuations that are not considered part of the indexed unit. For each query, top N ranked documents are retrieved for processing. N is currently fixed at 2500, based on an acceptable recall level of about 85% on a sample set of past TREC questions. A set of documents are carried over from the list of documents retrieved by the IR engine, using gradually diluted boolean filters, beginning with one that is formed by AND-ing all the keywords, and then dropping the smallest length non-noun, non-adjective, non-capitalized and non-date words one-by-one, until there are no more keywords to drop, or the cumulation of filtered documents exceeds a threshold of n. The value of n, is empirically set to 10. We call this the *boolean peel-off* strategy. The answer type named entity is used to filter out documents that do not contain the required named entity. For ones that do not expect a known named entity, the ranked list is kept as is. The re-ranking of documents using the features is implemented as a series of major and minor sorts. Documents with words in the same case as the query are preferred over ones with a different case – within the group of documents containing the same number of query words, while preserving the relative Okapi ranking. Moreover, words in the same sequence as in the query are preferred over ones in a different sequence, and n-grams are promoted within the former. The paragraphs are then broken down into sentences, which are ranked using the same criterion.

3.2.2. Answer Extraction

Since the answer expected of systems in the TREC-2002 evaluation was the exact phrase without any other justification, an additional module that extracts the exact answer from the text snippet was implemented. In the case of questions in which the answer is a specific named entity or a thematic role, and the top ranking sentence contains only one instance of that element, then extraction of the answer is as simple as extracting the element from the text. In many cases, there is more than one element of the predicted type and some form of selection mechanism is required. In such cases, the system selects the element with the shortest average distance from each of the query words present in that snippet. There are penalties added for some punctuations like commas, semi-colons, hyphens etc. In cases when the required answer is not a known named entity, the answer extractor tries to find a sequence of capitalized words that could be probable named entities. In case the expected answer is not a named entity, and the thematic role cannot be identified without much ambiguity, then the system tries to find an apposition near the question words, and extracts that as the answer. An example would be definition questions, of the style “What is X?”. Failing to get any of the above, the system replies that it does not have an answer to that particular question in the given corpus.

3.2.3. Confidence Annotation

The prior-probability of correct answers within a particular question type, along with the n-best count, is used to assign the confidence. The system currently goes a step further and judges whether the candidate at rank two has more counts in the top ten candidates, than candidate at rank one, and if so, promotes it to rank one. We need to tune the thresholds so that we can generalize this to promote an answer string ranked r, to a higher rank, depending on the relative counts for candidates above it, in a set of top N candidates.

3.3. Results

We present results obtained using the current system on past TREC data[4]. There was no human intervention during scoring, so the scores could be slightly higher than the ones computed by human evaluators. Two metrics were used in scoring the runs. One, total number of correct exact answers. Two, the NIST score – using the formula mentioned below.

$$\text{NIST score} = \frac{\sum_{i=1}^{N} \left(\frac{\textit{number correct up to question } i}{i} \right)}{N}$$

where, N is the total number of questions being evaluated. Answering a question correctly early in the ranking is worth more than answering a question correctly later in the ranking. In other words, for two systems that correctly answer the same number of questions, the metric gives more score to one that has a better mechanism for determining the confidence in its answers.

[4] The exact answer accuracy is determined by using the answer patterns available at the NIST webpage `http://trec.nist.gov`. More details of the evaluation procedure can also be found at the same address.

Table 1 shows the number of answers that were correctly promoted from rank 2 to rank 1 using the n-best count in top 10 candidate answers.

	TREC-9	TREC-10
Total questions graded	682	482
Number correct before n-best re-ranking	(15.6%) 106	(13.6%) 66
Number correct after n-best re-ranking	(17.6%) 120	(14.5%) 70

Table 1. Exact answer accuracy on TREC-9 and TREC-10 questions.

Table 2 shows the improvement in NIST score after confidence re-ranking of the final exact answers (one answer per question) – which includes the n-best re-ranking of top two answers.

	TREC-9	TREC-10
Total questions graded	682	482
NIST score before confidence and n-best re-ranking	0.18	0.16
NIST score after confidence and n-best re-ranking	0.25	0.27

Table 2. Improvement in NIST score after confidence re-ranking on TREC-9 and TREC-10 questions.

It can be seen that the confidence ranking scheme brings at least about 50% relative improvement in the unranked NIST score. The confidence ranker used on TREC-10 data was trained on features from TREC-9 only.

4. SUMMARY

In this paper we presented the architecture and implementation details of the CUAQ system that was used in the TREC-2002 evaluation. We plan to implement the thematic role and answer surface pattern features that we proposed, and present the results in the near future. Since the current implementation generates exact answers as required by the TREC-2002 evaluation, we cannot directly compare the results with those of other participating groups for TREC-9 and TREC-10. However, a subjective interpolation suggests that our system performance is at par with systems that do not use any semantic information.

5. ACKNOWLEDGMENTS

We would like to thank Ralph Weischedel of BBN Systems and Technologies for giving us the named entity tagger – Identifinder, and Daniel Gildea for the FrameNet parser.

6. REFERENCES

[1] Bikel, D., Schwartz, R., Weischedel, R., "An Algorithm that Learns What's in a Name", *Machine Learning*, 34, Kluwer Academic Publishers, 1999, 211–231.

[2] Baker, C., Fillmore, C., and Lowe, J., "The Berkeley FrameNet project", in Proceedings of the COLING-ACL, Montreal, Canada, 1998.

[3] Brin, S., Page, L., "The Anatomy of a Large-Scale Hypertextual Web Search Engine", *Computer Networks and ISDN Systems*, Vol. 30, No. 1–7, 1998, 107–117

[4] Fellbaum, C., *editor*, "WordNet: An Electronic Lexical Database", MIT Press, 1998.

[5] Gildea, D., Jurafsky, D., "Automatic Labeling of Semantic Roles", Technical Report TR-01-005, International Computer Science Institute, Berkeley, 2001

[6] Hovy, E., Hermjakob, U., "The Use of External Knowledge of Factoid QA", *The Tenth Text REtrieval Conference* (TREC-10), Gaithersburg, Maryland, November 13-16, 2001, 644–652.

[7] Robertson, S., Walker, S., "Okapi/Keenbow at TREC-8", *The Eighth Text REtrieval Conference* (TREC-8), Gaithersburg, Maryland, November 17-19, 1999, 151–162.

[8] Harabagiu, S., Moldovan, D., *et al.*, "Answering complex, list and context questions with LCC's Question-Answering Server", *The Tenth Text REtrieval Conference* (TREC-10), Gaithersburg, Maryland, November 13-16, 2001, 355–361.

[9] Witten, I., Moffat, A., Bell, T., "Managing Gigabytes: Compressing and Indexing Documents and Images", Morgan Kaufmann Publishing, San Francisco, May 1999.