Semantic Verification for Fact Seeking Engines

Dmitri Roussinov
Department of Information Systems
Arizona State University, Tempe, AZ, 85287
1-480-965-8488
dmitri.roussinov@asu.edu

Weiguo Fan
Department of Information Systems, Virginia Tech
3007 Pamplin Hall, Blacksburg, VA 24061
1-540-231-6588
wfan@vt.edu

Fernando A. Das Neves
Department of Computer Science
Virginia Tech
3007 Pamplin Hall, Blacksburg, VA 24061
1-540-231-6588
fdasneve@vt.edu

ABSTRACT
We present the architecture of our web question answering (fact seeking) system and introduce a novel algorithm to validate semantic categories of the expected answers. When tested on the questions used by the prior research, our system demonstrated the performance comparable to the current state of the art systems. Our semantic verification algorithm has improved the accuracy of answers of the affected questions by 30%.

Categories & Subject Descriptors: H.3.3 [**Information Storage And Retrieval**]: Information Search and Retrieval – *query formulation, search process.*

General Terms: Algorithms, Experimentation

1. INTRODUCTION
The goal of Question Answering (QA) is to locate, extract, and represent a specific answer to a user question expressed in a natural language. A QA system would take a sentence like "What is mad cow disease?" as an input and would produce an answer output like "Mad cow disease is a fatal disease of cattle that affects the central nervous system. It causes staggering and agitation." Many natural language questions expect the answer to belong to a certain semantic category, e.g. *What color is the sky?* Those questions prove to be difficult for current QA systems since the correct answer is not guaranteed to be found in a simple form such as in the sentence *The color of the sky is blue,* but rather needs to be extracted from a less straightforward sentence such as *I saw a vast blue sky above me*, in which a wrong answer "vast" has grammatically the same role as the correct answer "blue" and represents a property of the sky. However, "vast" refers to size, while we are expecting a color.

TREC competition-like conference [3] has been a driving force behind many recent innovation in QA and many good reviews of the technology break throughs can be found in its proceedings. A high proportion of TREC test questions involve very well known categories (states, countries, cities, organizations) and thus can be handled by manually compiled lists when the significant amount of manual effort is invested. However, handling more rare categories is still an unsolved problem even for QA systems that involve deep semantic parsing. Answering them requires combining multiple information sources. E.g. the candidate answer "Harrison Ford" should be rejected for the last question since he is an *actor*, not an *actress*, although may show up in a the following text: *"The Lion in Winter", starring Harrison Ford and Katherine Hepburn*.

In this paper, we present our novel algorithm for semantic verification of the candidate answers to fact seeking engines (question answering). We have implemented and empirically evaluated the suggested algorithm within our engine, which has been available in a demo version online and participated in TREC competition with encouraging results. Our system takes advantage of the redundancy of the Web, obtains the candidate answers by pattern matching, and then performs probabilistic triangulation of them to rank according to the final score. This paper also presents a formal evaluation of our system by following the methodology from the prior research reviewed in the next section, followed by descriptions of our system modules. Then, we describe the proposed semantic verification algorithm, followed by the empirical evaluation section.

2. ARCHITECTURE AND ALGORITHMS
The general idea behind our and the related approaches is pattern matching. For example, an answer for the question *"What is the capital of Taiwan?"* can be found in a sentence *"The capital of Taiwan is Taipei.",* which matches a pattern *\Q is \A,* where *\Q* is a question part (*"The capital of Taiwan"*) and *\A = "Taipei"* is the text that forms a *candidate answer*. We automatically create and train up to 200 patterns for each question type (such as *what is*, *what was*, *where is*, etc.), based on a training data set consisting of open domain QA pairs, e.g. those available from the past TREC conferences. Through training, each pattern is assigned a probability that the matching text contains the correct answer. This probability is used in the triangulation (confirming/disconfirming) process that re-ranks the candidate answers. *\A*, *\Q*, *\T*, *\p* (punctuation mark), *\s* (sentence beginning), *\V* (verb) and * (a wildcard that matches any words) are the only special symbols used in our pattern language so far. More details about our approach can be found in [2].

The intuition behind our approach to semantic verification is the following. If we need to answer a question *What soft drink contains the largest amount of caffeine?* then we expect a candidate answer (e.g. *coffee*) to belong to a specific category (*soft drink*). That means that one of the possible answers to the question *What is coffee?* should be *soft drink*. Thus, we can ask this question automatically, perform the same necessary steps as outlined above, and check the certain number of top answers if they contain the desired category. For the tests reported in this paper, we implemented this approach in a simpler and more efficient way in order to keep the process quick. We manually selected a subset of patterns from those automatically identified and trained for "what is" type of a question. Instead of full QA process outline above, for each candidate answer we only query GPSE for the total number of pages that match the modulated query (e.g. +"Coffee is a soft drink", matches 4 pages from Google; +"Coffee is a type of a soft drink" matches 0, etc.) and aggregate the total number of matches (denoted below as *M*). We explored the following family of heuristic scoring formulas for our semantic verification algorithm:

$$semantic\_fit\_score = \log(1 + M) \,/\, \log(2 + DF)^{a} \,/\, K^{Nw - 1}, \qquad (1)$$

where *DF* (document frequency) is the number of pages on the web that contain the candidate answer (e.g. *62,000,000* for "coffee"). The log *(DF + 2)* function is used since logarithmic scale is more appropriate for the wide range of values of *DF* (from *0* to *3,000,000,000* in our experiments). We would like to discount frequent candidate answers since they have higher chance of creating spurious matches. The parameter *a* in the power function is introduced to control how steep we would like to discount candidate answers as *DF* grows. *Nw* is the number of words in the candidate answer. The exponent function of *Nw-1* term is necessary to demote long answers since their typically small *DF-s* would otherwise result in them being overly promoted. We empirically set the parameter *K* to *400* early in the experimenting process.

We also experimented with combining formula (1) with another normalizing term: *log (2 + JoinDF)*, where *JoinDF* is the number of web pages containing both the candidate answer (coffee) and the expected category (soft drink). This number captures how frequently those two words (phrases) co-occur, thus how likely that some of the matches inside M are just due to a chance. This heuristic formula has been inspired by other works by automated semantic mining and Question Answering mentioned in our introduction. We believe that in future a more formal derivation can be performed using probabilistic framework or learned automatically from training data.

Our primary goal in this study was to show that sun semantic verification is viable. Before semantic verification, we first pre-sort the candidate answers according to their current score multiplied by the result of formula (1) excluding the *log (1 + M)* term in order to make a guess what candidate answers have a reasonable chance to be positively validated. They are validated in the obtained order until at least 30 candidate answers are obtained that have non zero scores (number of matches *M)*. Document frequencies (*DF*) are obtained by sending a query to the underlying search engine (GPSE), which slows our current real time performance. This issue can be later addressed by having a previously indexed sufficiently large corpus (e.g. TREC collections) or having direct access to the GPSE index and cache, which may be possible when the QA system is an integral part of it. Finally, the current score of each candidate answer is multiplied by its *semantic_fit_score* and the answers are re-ranked.

3. EMPIRICAL EVALUATION

First, we evaluated the overall performance of our system since we wanted to make sure it was comparable with the current state of the art in order for testing our semantic validation (or any other introduced component) to be convincing. The work by Dumais et al. [1] is the best candidate for comparison since they methodologically tested their system (AskMSR) on TREC questions and achieved much better performance than the other prior studies in similar settings. For a more meaningful comparison, we used the same set of test questions from TREC 2000 conference (Voorhees and Tice, 2000) and Google as our underlying general purpose search engine. We used the test sets from the other years (from 1999 to 2002, excluding the year 2000) in order to train our system. Since the systems participating in TREC competition are tested on the given local (non-web) collections, similarly to Dumais et al. (2002), we had to add more correct answers keys (patterns) after our preliminary runs. Since doing so required some manual effort, we decided to randomly sample 100 questions for our evaluation experiment instead of using all 500. Although various metrics have been explored in the past, we used mean reciprocal rank of the first correct answer (MRR) as in Dumais et al. (2002). E.g. if the first answer is correct, the reciprocal rank is 1. If only the second is correct, than it is ½, etc. The drawback of this metric is that it is not the most sensitive since it only considers the first correct answer, ignoring what follows.

The table 1 summarizes the results of our evaluation. The results indicate that the performance of our system is comparable with the one from Dumais et al. (2002). Again, we were only interested in verifying "in the same ballpark" performance and would like to caution again comparing two systems numerically for the following reasons: 1) Different size of the web at the times of the evaluations. 2) Possibly different ranking algorithms employed by the underlying general purpose search engine. 3) Differences in the criteria for the expansion of the correct answer sets. We also compared the simplified configurations of our system. The most basic configuration did not include any semantic filtering, semantic adjustment or pattern matching. This corresponds to the most basic configuration of AskMSR (no semantic filtering, no answer tiling or using re-writes) only approximately since our implementations of the corresponding components slightly differ. Our pattern matching mechanism plays approximately the same role as re-writes in AskMSR. The fact that our basic performance was found to be higher may be explained by the increase of size of the Web which could make it easier to mine for the exact answers. In spite of the differences in the time of evaluation and implementation details, the results clearly indicate that the performances of our systems are in the same "ballpark" and the impacts of similar components on the overall performance are also similar. We concluded that the obtained results indicated comparable performance, which allowed us to proceed to the next step: evaluating our novel semantic verification algorithm within our system.

Table 1. Summary of overall evaluation. AskMSR and Our QA System. Mean Reciprocal Rank (MRR) for various system configurations

Configuration	Our System		AskMSR	
	Features	MRR	Features	MRR
Complete	All	0.570	All	0.507
Basic	no semantic verification, no semantic filtering, no patterns	0.345	no semantic filtering, no re-writes, no tiling	0.266
Intermediate	no patterns	0.517	no re-writes	0.450

Since only 16 of the questions in the test set mentioned in the previous section have specified semantic category and in order to provide more variety in evaluation tests, we performed the evaluation of our verification algorithm on the TREC 2003 set. First, we experimented with the parameters in the formula (1) trying to achieve best possible results. For this, we only used all the 113 questions that have a defined semantic category. Table 2 lists the formulas that we tried and their resulting performance. The optimal was reached for *alpha = 7* in the formula (1). The performance was sensitive to the shapes (degree of steepness) of the discounting terms *log (2+DF)* and *log(2+JoinDF)*. However, the best performing formula did not need to include *JoinDF* term.

Finally, we evaluated the impact on the overall performance: due to our semantic verification the MRR increased from *0.394* to *0.425*, which corresponds to *8%* relative improvement overall, which we believe is a significant impact considering that only *20%* of the questions were affected and that the impacts of the other components are of the comparable magnitude.

Table 2. Summary of overall evaluation. AskMSR and Our QA System. Mean Reciprocal Rank (MRR) for various system configurations

Formula	MRR
1 / (1+JoinDF)	0.157
1	0.352
log(1+M) / (log(2+JoinDF) + log (2+DF))	0.400
log(1+M) * log (1+DF) / log(2+JoinDF)	0.389
log (1+M)	0.394
log(1+M) / $\log^4$ (2+DF)	0.437
log(1+M) / $\log^7$ (2+DF)	**0.441**
(1+M) / (2+DF) / (2+JoinDF)	0.168
log(1+M) / $\log^4$ (2+DF) * log(2+JoinDF)	0.343

4. ACKNOWLEDGMENTS

Weiguo Fan's work is supported by NSF under the grant number ITR0325579. Roussinov's work is supported by Dean's Award of Excellence, W.P. Carey School of Business, summer 2005.

5. REFERENCES

[1] Dumais, S., Banko, M., Brill, E., Lin, J., and Ng, A. (2002). Web Question Answering: Is More Always Better? *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, Tampere, Finland, August 11-15.

[2] Roussinov, D. and Robles, J., Ding, Y., Experiments with Web QA System and TREC2004 Questions. *In the proceedings of 2004 TREC conference*. November 16-19, 2004, Gaithersburg, MD.

[3] Voorhees, E. and Buckland, L.P., Eds. (2004). Proceedings of the Eleventh Text Retrieval Conference TREC 2004. Gaithersburg, Maryland, November 16-19.