Semantic Verification in an Online Fact Seeking Environment

Dmitri Roussinov
W.P.Carey School of Business
Department of Information Systems
Arizona State University
P.O Box 873606
Tempe, AZ, 85287
+14809658488
dmitri.roussinov@asu.edu

Ozgur Turetken
School of Information Technology Management
Ryerson University
575 Bay Street, Toronto, ON Canada M5G 2C5
+14169795000 x2481
turetken@ryerson.ca

ABSTRACT
Many artificial intelligence tasks, such as automated question answering, reasoning or heterogeneous database integration, involve verification of a semantic category (e.g. "coffee" is a drink, "red" is a color, while "steak" is not a drink and "big" is not a color). We present a novel algorithm to automatically validate a semantic category. Contrary to the methods suggested earlier, our approach does not rely on any manually codified knowledge but instead capitalizes on the diversity of topics and word usage on the World Wide Web. We have tested our approach within our online fact-seeking (question answering) environment. When tested on the TREC questions that expect the answer to belong to a specific semantic category, our approach has improved the accuracy by up to 14% depending on the model and metrics used.

Categories and Subject Descriptors
H.3.3 [Information Storage and Retrieval] Information Search and Retrieval, Retrieval models

General Terms
Algorithms

Keywords
artificial intelligence, online search engines, question answering

1. INTRODUCTION

Many artificial intelligence tasks involve automated verification of a semantic category. For example, the correct answer to the question *What soft drink has most caffeine?* should belong to the category "soft drink." Automated integration of several heterogeneous databases may require matching an attribute in one database that has such values as *red, green,* and *purple* to an attribute called "color" in another database. Although our work presented in this paper was motivated by the more general task of semantic verification, we were specifically interested in its applications to online fact seeking, which is sometimes referred as open-corpus/open-domain question answering. The goal of Question Answering (QA) is to locate, extract, and represent a specific answer to a user question expressed in a natural language. Prior studies [21] have indicated that the current WWW search engines, especially those with very large indexes like Google, offers a very promising source for open domain question answering.

Answers to many natural language questions are expected to belong to a certain semantic category, e.g. *What color is the sky?* Those questions prove to be difficult for the current QA systems since the correct answer is not guaranteed to be found in an explicit form such as in the sentence *The color of the sky is blue,* but rather may need to be extracted from a sentence answering it implicitely, e.g. 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*.

The currently popular approach to solving this "semantic" matching problem is through developing an extensive taxonomy of possible semantic categories [10]. This requires the anticipation of all possible questions and, hence, substantial manual effort. Moreover, although this approach works relatively well with more common categories (e.g. cities, countries, organizations, writers, musicians, etc.), handling more rare categories (e.g. as in questions like *What was the name of the first Russian astronaut to do a spacewalk?* or *What soft drink contains the largest amount of caffeine?*) is still a challenge.

In this paper, we explore completely automated on-fly verification of a membership in a previously unanticipated category. Although, our algorithm can be used inside any other system, we have implemented and empirically evaluated it within our fact seeking engine, which has been available in a demo version online **[24].** Our inspection of the 1000+ search sessions recorded by our demo reveals that approximately 20% of questions have answers that are expected to belong to a specific semantic category, thus such systems can certainly benefit from semantic verification. The performance of our system was evaluated earlier **[24]** and found to be comparable with state-of-the-art QA systems, e.g. [6], that are based on redundancy, rather than on extensive manually codified knowledge, e.g. elaborate ontologies or rules for deep grammar parsing. Contrary to the "knowledge-heavy" commercial systems, our system is entirely transparent: all the involved algorithms are described in prior

CIKM'07, November 6-8, 2007, Lisboa, Portugal.

publications, and thus, can be replicated by other researchers. In this paper, we go past purely heuristic testing and build a model based on a logistic regression. We also explore in details 1) what variables contribute to the accuracy of answers, 2) what normalizing transformations are beneficial, and 3) what are the possible ranges of improvements. The rest of the paper is organized as follows. The next section reviews related work. Section 3 presents the description of our system followed by its empirical evaluation in section 4, and our conclusions.

2. PRIOR WORK

2.1 Closed Corpus Question Answering

The National Institute of Standards (NIST) has been organizing the annual Text Retrieval Conference (TREC) [27] since 1992, in which researchers and commercial companies compete in document retrieval and question answering tasks. The participating systems have to identify exact answers to so-called *factual* questions (or *factoids*), such as *who, when, where, what*, etc., list questions (*What companies manufacture rod hockey games?*) and definitions (*What is bulimia?*). In order to answer questions, a typical TREC QA system would: (a) transform the user query into a form it can use to search for relevant documents (web pages), (b) identify the relevant passages within the retrieved documents that may provide the answer to the question, and (c) identify the most promising candidate answers from the relevant passages. Most of the QA systems at TREC are designed based on techniques from natural language processing (NLP), information retrieval (IR), and computational linguistics (CL). For example, Falcon [10], one of the most successful QA systems, is based on a pre-built hierarchy of dozens of semantic types of expected answers (*person, place, profession, date, etc.*), complete syntactic parsing of all potential answer sources, and automated theorem proving to validate the answers.

In contrast to the NLP-based approaches, "shallow" approaches that use only simple pattern matching have recently been tried with good level of success. For example, the system from InsightSoft [26] won the 1st place in 2002 and the 2nd place in 2001 TREC competitions. The "knowledge-light" systems based on simple pattern matching and redundancy (repetitions of the answer on the Web), such as [6], also scored comparably.

Both NLP-based approaches and those that require elaborate manually created patterns have a strong advantage: they can be applied to smaller collections (e.g. corporate repositories) and still provide good performance. However, none of the known top performing systems has been made publicly open to the other researches for follow up investigations because of expensive knowledge engineering is required to build such systems and the related intellectual property issues, none of the known top performing systems has been made publicly open to the other researches for follow up investigations. As result, it is still not known what components are crucial and how well certain approaches would extend outside of the TREC test sets.

At the same time, the algorithms behind some of the systems that do not require extensive knowledge engineering, but still demonstrate reasonable performance, have been made freely available to public. We believe that from a research perspective, those systems and the approaches behind them are by no means less interesting than the top commercial systems. At the moment, only the former allow replication and independent testing by other researchers. Some of those systems are mentioned in the next section.

2.2 Web Question Answering

There are several important distinctions between QA from a closed corpus and QA from the entire Web:

1) Typically, the Web has a much larger variety in the answers that it presents. This allows the Web-based fact seeking systems to look for answers in the most simple forms (e.g. *Sky is blue*), which makes the task easier at times.

2) The users of the Web fact seeking engines do not necessary need the answers to be extracted precisely. In fact, we personally observed from the interaction with practitioners that they prefer to read the answer within its context to verify that the source is credible.

3) Web fact seeking engines need to be quick, while TREC competition does not impose any real time constraints. This places an emphasis on simple and computationally efficient algorithms and implementations such as simple pattern matching as opposed to "deep" linguistic analysis.

Web question answering has a history of development. START [2] was one of the first QA systems available online since 1993. It was primarily focused on encyclopedic questions (e.g. about geography) and used a precompiled knowledge base. Prior evaluation of START [2] indicated that its knowledge is rather limited, *e.g.*, it fails on many questions from the standard test sets (detailed below).

Mulder [1] was the first general-purpose, fully-automated QA system available on the web. It worked by querying a general purpose search engine (Google or MSN), then retrieving and analyzing the web pages returned from the queries to select answers. When evaluated by its designers on TREC questions, Mulder outperformed AskJeeves and Google by a large margin. Unfortunately, Mulder is no longer available on the Web as a demo or for download for a comparative evaluation.

In their prototype called Tritus, Agichtein et al. [1] introduced a method for learning query transformations that improves the ability to use web search engines to answer questions. Blind evaluation of this system on a set of real queries from a web search engine log showed that the method significantly outperformed the underlying web search engines as well as a commercial search engine specializing in question answering. Tritus is also not available online.

A relatively complete, general-purpose, web-based QA system, called NSIR, was presented in [22]. Dumais et al. [6] presented another open-domain Web QA system that applies simple combinatorial permutations of words (so called "re-writes") to the snippets returned by Google and a set of 15 handcrafted rules (semantic filters) to achieve a remarkable accuracy on the TREC test set: Mean Reciprocal Rank (MRR) of *0.507*, which can be roughly interpreted as "on the average" the correct answer being the second answer found by the system. This was only 20-30% below of the accuracy of the best knowledge-heavy systems and still within top 20% participating systems. The authors' experiments also indicated that semantic filtering was the component that had the greatest impact on the overall performance of the system, even though it was limited to only a few categories (people, places, dates, numbers, etc.)

2.3 Data Driven Semantic Verification

To avoid laborious creation of large knowledge bases (e.g. ontologies), researchers have been actively trying automated or semi-automated data driven techniques. The idea to count number of matches to certain simple patterns (e.g. "colors *such as* red, blue or green; soft-drinks , *including* pepsi and sprite" etc.) in order to automatically discover hyponym relationships was first introduced by Hearst [17] and tested on Grolier's American Academic Encyclopedia using WordNet as gold standard. The variations of the idea of Hearst's patterns has been adopted by other researchers: in specific domains [11], for anaphora resolution [16], for discovery of part-of [8] and causation [20] relations. Those approaches are known for a relatively high (50%+) precision but a very low recall due to the fact that the occurrence of patterns in a closed corpora are typically rare. To overcome the data sparseness problem, researchers resorted to the World Wide Web: Hearst patterns are searched for using the Google API for the purpose of anaphoric resolution in [12], enriching a given ontology in [7]. [19] and [18] used the Google API to match Hearst-like patterns on the Web in order to find the best concept for an unknown instance. Still, *none of the methods mentioned above provides a probabilistic verification of a membership in a previously non-anticipated semantic category, which is necessary for question answering and is the focus of investigation here.*

Somewhat similar to the methods and the purpose described here are the approaches within KnowItAll project [5][9], which automatically collects thousands of relationships, including the hyponymic ones, from the Web. With KnowItAll project Etzioni et al. developed a probabilistic model by building on the classic balls-and-urns problem from combinatorics. They established that the urns model outperforms those based on pointwise mutual information (PMI) metrics used earlier within KnowItAll [5][9] or by other researchers. However, the model provides the estimate of the probability of the categorical membership only in the case of supervised learning (anticipated and manually labeled categories), but *only rank-ordering in the unsupervised case* (not a pre-anticipated category). The rank ordering was evaluated by recall and precision, and only on four relations: `Corporations`, `Countries`, `CEO of a company`, and `Capital Of a Country`, while for the purpose studies here (question answering) a much wider variation of categories is typically encountered. Schlobach et al. [25] studied semantic verification for a larger number of categories, but still limited to the domain of geography and, also by empowering pattern matching statistics by knowledge intensive methods.

3. SYSTEM OVERVIEW

This section briefly overviews the principles behind our QA system used in the studies reported in this paper. The complete details of the entire QA system, including its pattern discovery and triangulation algorithms, can be found elsewhere **[24]**. They are not essential for the replication of our study reported here. The full details of our semantic verification algorithm, which are essential for replication, are presented in the next section.

The general idea behind our approach (like many others) is to apply pattern matching and take advantage of the redundancy (repeating of the same information) on the web. For example, the answer to the question *"What is the capital of Taiwan?"* can be found in the sentence *"The capital of Taiwan is Taipei.",* which matches a pattern *\Q is \A,* where *\Q* is the question part (*"The capital of Taiwan"*) and *\A = "Taipei"* is the text that forms a *candidate answer*. To our knowledge, the trainable pattern approach was first introduced in Ravichandran and Hovy [23]. We automatically create and train up to 50 patterns for each question type (such as *what is*, *what was*, *where is*, *how far, how many, how big* etc.), based on a training data set consisting of open domain QA pairs, e.g. those available from past TREC conferences [27]. 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 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, but it is easily extensible.

Answering the question *"In which city is Eiffel Tower located?"* illustrates our question answering process step by step:

Type identification: The question itself matches the pattern *in which \T is \Q \V*, where *\T = "city"* is the semantic category of the expected answer, *\Q = "Eiffel Tower"* is the so called question part (sometimes referred as "target" or "focus") , and *\V = "located"* is a verb. In order to correctly identify the question type and its components, the system uses the freely available trainable Part Of Speech (POS) tagger [3]. It applies it to the questions only, but not to the answer sources.

Query Modulation (QM): QM converts each answer pattern (e.g., *\Q is \V in \A*) into a query for a General Purpose Search Engine (GPSE). For example, the Google query would be *+"Eiffel Tower is located in"*.

Answer Matching: The sentence *"Eiffel Tower is located in the centre of Paris, the capital of France"* would result in a match and create a candidate answer *"the center of Paris, the capital of France"* with a corresponding probability of containing a correct answer (i.e. *0.5*) obtained previously for each pattern by training.

Answer Detailing (AD): AD produces more candidate answers by forming sub-phrases from the original candidate answers that: 1) do not exceed three words (not counting "stop words" such as *a, the, in, on*) 2) do not cross punctuation marks 3) do not start or end with stop words. In our example, these would be: *centre, Paris, capital, France, center of Paris, capital of France*.

If there are fewer than expected candidate answers found (e.g. 200), the algorithm resorts to a "fall back" approach as in [6]; it creates candidate answers from each sentence of the snippets returned by the search engine (Google), and applies answer detailing to those answers. If there are still not enough candidates, the system automatically relaxes the modulated query by removing the most frequent words first until enough pages are returned by the engine. Thus, this method reduces a question such as *"Who still makes rod hockey games?"* to *"Who makes rod hockey?"*

Semantic Adjustment: We do not use an extensive hierarchy of question types organized by the expected semantic category of an answer (e.g. person, celebrity, organization, city, state, etc.). Rather, we use a small set of independently considered semantic boolean (yes/no) features of candidate answers: *is number, is date, is year, is location, is person name* and *is proper name.* Although not perfectly accurate, those features are detected by

matching simple hand-crafted regular expressions. Depending on the expected semantic category of an answer, the presence or absence of those features results in applying any of approximately 20 manually tuned discounts.

Triangulation: The candidate answers are triangulated (confirmed or disconfirmed) against each other and then re-ordered according to their final score. In essence, our triangulation algorithm promotes those candidate answers that are repeated the most. It eventually assigns a score s_c in the [0,1] range which can be informally interpreted as the estimate of the probability of the candidate answer being correct. These probabilities are independent in the sense that they do not necessarily add up to 1 across all the candidate answers, which we believe does not pose a problem since multiple correct answers are indeed possible.

Semantic Verification: If the answer to a question is expected to belong to a certain semantic category (e.g. *City*), then candidate answers are re-ordered according to their semantic verification scores as detailed in the next section.

4. AUTOMATED SEMANTIC VERIFICATION APPROACH

4.1 Intuition behind the approach studied

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. *pepsi*) to belong to a specific category (*soft drink*). This implies that one of the possible answers to the question *What is pepsi?* should be *soft drink*. We can ask this question automatically, perform the same necessary steps as outlined above, and check the certain number of top answers to see if they contain the desired category. Thus, in this paper, the automated question answering on the web is not only the application of semantic verification but it also serves as the technology behind it.

We deliberately simplified the QA cycle inside our system when it is used for semantic verification in order to keep the process quick and the algorithms easier to replicate. The specific details of our QA system are not important for the semantic verification algorithm that is described here and can be easily replicated within any other QA system or even outside a QA system by just querying a commercial web search engine (e.g. Google) in a way explained below.

We selected 16 most accurate patterns from those automatically identified and trained for "what is" type of a question. During semantic verification, instead of following the full QA process outlined above, for each candidate answer, our system queries the underlying web search engine only for the total number of pages that match the modulated query for each pattern. For example, "pepsi is a soft drink", matches 127 pages from Google; "pepsi is a type of a soft drink" matches 0, etc. We denote the corresponding number of matches below as $m_1 \ldots m_{16}$, and the aggregate total number of matches is denoted as M.

4.2 The role of semantic verification in the QA process

The purpose of semantic verification is to estimate the probability of the given candidate answer to belong to the specified category. For example, since pepsi is a soft drink, that probability should be close to 1. In some less clear cases (e.g. *mosquito* is an *animal*) it can be further from *1*. An informal interpretation of this probability may be the following: if we ask 100 people the question *"Is mosquito an animal?"* and only 30 of them would answer "yes" then that probability would be .3. Since our approach draws on the aggregated opinions of all of the Web contributors, it may produce category memberships that are considered wrong according to formal taxonomies, but are still believed to be true due to common myths, misconceptions, or even jokes, which are frequently circulating on the Web. For example, a phrase "coffee is a soft drink" can be found in 7 pages, although, formally, it is not true. That is why we believe that it is important to aggregate across multiple pattern matches. We also believe it is important to consider the overall statistics of both the category and the candidate answer, for example, the total numbers of web pages that contain them. Obviously, the more frequent candidate answers need to be demoted. We combine all these variables into one model that estimates the probability of membership in a given semantic category. If we denote this probability (score) as s, then the final score according to which the candidate answers are sorted before being presented to the user can be computed as:

$$s_f = s \cdot s_c,$$

where s_c is the "initial score" (probability of being correct) of the candidate answer after all the processing steps (mentioned in the previous section) before our semantic verification is applied. This assumes that the evidence evaluated by semantic verification s is independent from the original score s_c. We believe that this is a realistic assumption since the original score s_c is primarily syntactic: it is based on the number of occurrences of the candidate answer within certain syntactic patterns trained as mentioned in section 3. Thus, we assume that syntactic and semantic indicators of correctness are not highly correlated. We are leaving a more fine-grained model for future. At the moment, we are not aware of any more formal approaches to this problem. Heuristic combining scores from various modules in a multiplicative ways typical for the current state of the art QA systems [27]. As we wrote in the literature review, the prior research has not introduced any formal (e.g. probabilistic) models for semantic verification suitable for our purpose here.

4.3 Building the predictive model

First, we approached semantic verification as a classification problem. We were specifically interested in the probability estimates. In order to train the model, we created a labeled data set as following. We took all the 68 questions from TREC 2003 set that specified the semantic category explicitly. We did not use any questions that only implied a certain semantic category. For example, "who is" question would typically imply a person. We run the questions through our QA system without semantic verification and manually labeled the correctness of the semantic category as true/false for top 30 candidate answers for each question.

We developed a number of predictive models using binary logistic regression techniques to compute s. These models simply used different combinations of the following variables that, as we intuitively conjectured, may help to discriminate between the semantic match and mis-match:

m1, m2, ...,m16: match for each of the patterns,

$M = m1 + m2 + ... + m16$: total number of pattern matches,

DF_A: number of pages on the web containing the candidate answer (e.g. *pepsi*),

DF_C: number of pages on the web containing the expected category (e.g. *soft drink*),

DF_{AC}: number of web pages containing both the candidate answer

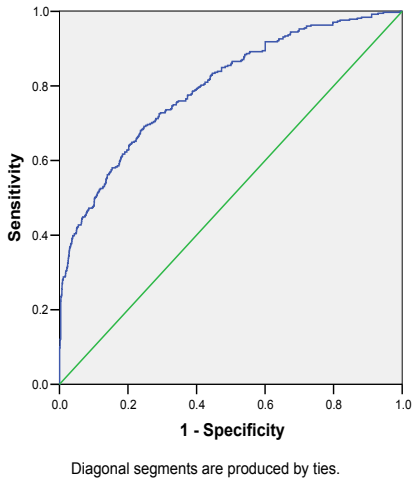

Figure 1. ROC curve for predictive model 5.

(e.g. *pepsi*) and the expected category (e.g. *soft drink*).

The intuition for this selection of variables is the following. Obviously, more frequent candidate answers would result in larger numbers of pattern matches simply due to chance (random co-occurrence), thus we would expect s to be a decreasing function of DF_A. Our model also takes into consideration that some pairs of words frequently co-occur even if one of them is not a category of another, e.g. "*cup* of *coffee"* would match one of our patterns even when where is no underlying categorical relationship exists, while "city of Rome" would be a legitimate indication that Rome is a city. For this reason, we introduced DF_{AC} into the model. Finally, the model may benefit from DF_C as a normalizing variable for m-s and DF_{AC}.

Based on the observation that the distributions of the frequencies of occurrences (DF_A, DF_C) and co-occurrences (DF_{AC}) typically look more uniform in the *log* space, we also explored the models that use natural logs of the variables M, DF_A , DF_C , and DF_{AC}. Therefore, the complete set of variables studied in our models was the following:

(m1, m2,…,m16, M, DF_A, DF_C, DF_{AC}, log(M+1), log(DF_A), log(DF_C), log(DF_{AC}).

Since M was equal to *0* for many data points, we added *1* before taking a *log* to it being undetermined. The candidates with $DF_{AC} = 0$ were assigned $s=0$.

Although this is not an exhaustive combination, in this study, we considered the following wide range of models:

Model 0) $s = const$, or simply no semantic verification.

Model 1) $s = s(m1, m2,...,m16, DF_A, DF_C, DF_{AC})$: all original variables in their non-transformed form.

Model 2) $s = s(\log(M+1), \log(DF_A), \log(DF_C), \log(DF_{AC}))$: all variables are log-transformed, aggregate variable M is used instead of separate m-s. This model subsumes the metrics based on PMI studied earlier [5][9].

Model 3) $s = s(m1, m2,...,m16, \log(DF_A), \log(DF_C), \log(DF_{AC}))$: only the document frequencies are log-transformed (since they are typically much larger than m-s; m-s are not log transformed and used independently.

Model 4) $s = s(DF_A, DF_C, DF_{AC}, \log(M+1))$: Only the total number of pattern matches is log-transformed, no separate m-s used.

Model 5) $s = s(DF_A, DF_C, \log(M+1))$: same as 4 but without DF_{AC}

Model 6) $s = s(DF_{AC}, \log(M+1))$: Only DF_{AC} (co-occurrence) and *log* of M -- the simplest model.

All of the models reported involve number of matches (m-s or M), because early in the study we found out that without them the performance was not statistically different from random guessing. It is still possible that without using patterns, and based on co-occurrence statistics only, it would be still possible to build predictive models to improve the QA accuracy since the correct answer frequently co-occurs with the words from the question. However, in this study, we wanted to isolate the improvement due to semantic verification, thus we discarded the models that did not improve the classification accuracy directly.

The numbers of pattern matches were obtained for top 30, according to initial score s_c, candidates by sending a query to the underlying search engine (MSN). Although the queries were sent in parallel, the verification would have slowed the response of our system if it were used online. However, the real-time performance can be later addressed by having a previously indexed, sufficiently large corpus (e.g. TREC collections) or having direct access to the search engine index, which may be feasible when semantic verification module is an integral part of the web search portal or connected to it through a local (or otherwise fast) network.

5. EMPIRICAL EVALUATION

5.1 Data sets

First, we approached semantic verification as a classification problem. We were specifically interested in the class membership probability estimates. In order to train the model, we created a labeled data set as following. We took all the 68 questions from TREC 2003 that specified the semantic category explicitly. We did not include any questions that only implied a certain semantic category, but not explicitly state it. For example, a question starting with "who is" would typically imply, the category of person (or organization). Similarly, our categories under investigation did not include locations (*where*) and dates (*when*). We run the questions through our QA system without semantic verification and manually labeled the correctness of the semantic category as true/false for top 30 candidate answers for each question of 68 questions.

The target semantic category was identified by applying 4 simple regular expressions to the questions. For example, the regular expression "(What|Which) (.+) (do|does|did|is|was|are|were)"

would match a question "What tourist attractions are there in Reims?" and identify "tourist attraction" as a semantic category.

For testing, we sought to find a set that we did not use before. We went back all the way to TREC 2000 and used all the 41 questions that explicitly specified semantic category of an expected answer -- same principle as we used for constructing our training set.

5.2 Category Prediction

Since the data set included disproportionate numbers of negative examples, In order to evaluate classification accuracy, we over-sampled the positive cases (correct semantic category) to built a balanced data sample. Table 1 displays the classification accuracy of our models listed above.

To test whether this classification improvement over that purely by chance is statistically significant, we examined the underlying ROC curves. For the lack of space, we display here the ROC curve for model 5 only (Figure 1). All the models predict significantly better than chance.

In order to compare models against each other we needed a specific application since some of them may be better at recall while others at precision, which results from the very well known trade-off between Type I and Type II classification errors. For the same reason, the accuracy reported here can not be compared to any of the prior research. E.g. KnowItAll [5][9] evaluated accuracy based on the specially harvested data sets, while ours are coming from and an automated QA process.

5.3 Improving Answering Accuracy

We evaluated the impact of semantic verification on our online fact seeking (QA) system using the sets of questions and correct answer regular expressions from TREC competitions. Although various metrics have been explored by researchers in the past, we used mean (across all the questions) reciprocal rank of the first correct answer (MRR) as our primary metric. E.g. if the first answer is correct, the reciprocal rank is 1. If only the second is correct, then it is ½, etc. The drawback of this metric is that it is not the most sensitive because it only considers the first correct answer, ignoring what follows. For this reason, we also used mean total reciprocal rank of the correct answers (TRDR). It aggregates the reciprocal ranks of all the correct answers, e.g., if the second and the third are the only correct ones, it would be equal to *1/2 + 1/3*. We did not use the "degree of support" of the answer within the document as part of the metric due to its known difficulty [15], and, thus, only checked if the answer was correct, which is sometimes called "lenient" evaluation, to which the concerns of Lin et al. [15] do not apply.

Table 2. Models performance on the training set. ** and * indicate statistical significance at the levels .05 and 0.1 accordingly.

Model	MRR	TRDR	Improvement over Baseline	
			MRR	TDRR
0 (base)	0.442	0.580	-	-
1	0.455	0.618	+2.94%*	+6.55%*
2	0.447	0.620	+1.13%	+6.90%
3	0.388	0.521	-12.22%**	-10.17%**
4	0.482	0.650	+9.05%**	+12.07%**
5	0.485	0.652	+9.73%**	+12.41%**
6	0.459	0.621	+3.85%	+7.07%*
Oracle	0.580	0.956	+31%	+64%

Table 1. Classification accuracy of predicting membership in a given semantic category. "0" means wrong category, "1" means correct category.

Model	Observed Category:	Predicted Category		correct (%)
		0	1	
1	0	4938	118	97.7
	1	2899	2028	41.2
	Overall			69.8
2	0	4144	912	82.0
	1	1924	3003	60.9
	Overall			71.6
3	0	4356	700	86.2
	1	2444	2483	50.4
	Overall			68.5
4	0	4568	488	90.3
	1	2262	2665	54.1
	Overall			72.5
5	0	4569	487	90.4
	1	2262	2665	54.1
	Overall			72.5
6	0	4506	550	89.1
	1	2340	2587	52.5
	Overall			71.1

First, we computed the baseline performance, which corresponded to the model 0: no semantic verification. The resulting MRR was 0.442. Next, we estimated what was the maximum possible range of the improvement and computed the "would be" performance if our semantic verification were perfect. Thus, our "Oracle" returned 1 as semantic verification score if the candidate answer belonged to the required category and 0 otherwise. This resulted in MRR and TRDR of *0.580*, and *0.956* respectively. The total proportion of questions that had a correct answer within a set of the candidate answers was *0.65*. The loss in accuracy from *.65* to *0.58* is explained by the existence of the highly scoring candidate answers that belonged to the required category but were nevertheless wrong (E.g. What is the color of the sky? – orange). By inspecting the logs we observed that there are two reasons why the improvement was limited: 1) the correct answer already dominates other candidate answers even without semantic verification 2) the correct answer was not even in set of candidate answers. To test how much our classification improves the performance, we compared the MRR and TRDR values form

Table 3. Models performance on the "unseen" testing set.

Model	MRR	TRDR	Improvement over Base	
			MRR	TDRR
0 (Base)	0.380	0.542	-	-
1	0.420	0.588	+11%**	+8.5%**
2	0.399	0.577	+5%*	+6.5%**
3	0.398	0.581	+5%*	+7.2%**
4	0.422	0.583	+11%**	+7.6%**
5	0.433	0.596	+14%**	+10.0%**
6	0.401	0.559	+6%*	+3.1%

the six models we have built with the base model (Model 0). The summary results are displayed in Table 2.

As Table 2 indicates, all the models, except for model 3, improve QA performance over the baseline. Models 4 and 5 are especially promising. The results suggest that by using our semantic verification approach, the QA performance can be improved (in terms of both MRR and TRR) significantly especially when (log-transformed) pattern matches are used (models 4 and 5). The results also indicate that treating numbers of pattern matches separately instead of as one aggregate variable was not resulting in statistically detectable additional improvement. Obviously, using larger data set may fine-grain the results reported here.

5.4 Testing on a Previously Unseen Set

Since the first two testing steps performed above constituted training (tuning) steps, in order to provide higher objectivity of the results and to avoid possible dangers of overfitting, we also performed tests on a set of questions that the system had never seen before (*testing set*).

We did not manually label the testing set into correct/wrong categories of the candidate answers and only measured the effect of our semantic verification on the answer accuracy using the same metrics or MRR and TRDR as described above. Table 3 presents the results. The improvement ranged from +6% to +16% depending on the model. All of them are statistically different (alpha < 0.05) from the baseline (no verification). The results clearly indicate that the improvement from our semantic verification is noticeable. The correlation between the performances of the models on the training set (Table 1) and on testing set is also noticeable. Future, more detailed tests may establish this with higher reliability.

6. CONCLUSIONS, LIMITATIONS AND FUTURE RESEARCH

We have introduced a novel semantic verification algorithm and demonstrated that *it improves the accuracy of answers produced by a redundancy-based question answering system.* The overall performance of our system was evaluated earlier and found comparable to that of the other current state of the art systems that do not require an elaborate knowledge base. Although, the systems that reported the best performance at TREC competitions surpass the "knowledge-light" systems, including ours, those top performing systems do not disclose their algorithms completely and do not make the underlying knowledge available to other researchers, which makes replication and follow up practically impossible. That is why we were not able to test our method within any of the best (top 3) systems, which still may be considered a limitation. However, the algorithms behind knowledge-light systems, like AskMSR or ours, have been already studied by a number of researchers and their performance, even numerically inferior to the best commercial systems, has been confirmed and analyzed. We have already indicated throughout our paper several other possible limitations that we are currently addressing in ongoing studies. Specifically, these are using larger sets of testing/training questions, more patterns, or other predictive models and transformations of its input variables.

Up to our knowledge, this is the first study on completely automated verification of the membership in a previously unanticipated semantic category for the purpose of improving question answering. For this reason, we do not have any other technique to report here for comparison. Our method is different in the terms of coverage and the manual effort necessary to those based on WordNet or other manually developed taxonomies. In our sets of questions, WordNet contained the entries only for 12% of categories and 6% of candidate answers. Thus, we would not expect a method based on WordNet to provide comparable results. Indeed, many of the categories in our test set were phrases (*American feminist, sporting event, Ridley Scott movie*, etc.), of which there were no direct entries in WordNet. Besides, our approach can be complementary to those based on WordNet since it is based on entirely different source of data: patterns of occurrence on the Web rather than manually created ontology. For those reasons we did not think that comparing our approach against those based on WordNet (or similar) sources was necessary within our study.

We believe our results to be extremely encouraging and our approach viability to be successfully applied to other problems such as database integration and ontology matching where semantic verification is necessary. Since it is commonly believed that categorization is behind many cognitive tasks, the completely automated verification of a given unrestricted semantic category has the potential to be a milestone within the broader scope of artificial intelligence research.

7. REFERENCES

[1] Agichtein, E., Lawrence, S., and Gravano, L. (2001). Learning Search Engine Specific Query Transformations for Question Answering. 10th WWW Conference. Hong Kong, China, May 1-5.

[2] Boris Katz, Jimmy Lin, Daniel Loreto, Wesley Hildebrandt, Matthew Bilotti, Sue Felshin, Aaron Fernandes, Gregory Marton, and Federico Mora. Integrating Web-based and Corpus-based Techniques for Question Answering. (2003). Proceedings of the *Twelfth Text REtrieval Conference (TREC 2003)*, November 2003, Gaithersburg, Maryland.

[3] Brill, E. (1995). Transformation-based error-driven learning and natural language processing: A case study in part of speech tagging. Computational Linguistics, 21(4):543-566.

[4] Brill, E., Dumais, S. and Banko, M., An Analysis of the AskMSR Question-Answering System, *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing*, Philadelphia, PA, USA, July 6-7.

[5] Downey, Oren Etzioni, and Stephen Soderland (2005). A Probabilistic Model of Redundancy in Information Extraction. IJCAI-05.

[6] 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.

[7] E. Agirre, O. Ansa, E. Hovy, and D. Martinez. Enriching very large ontologies using the WWW. In Proceedings of the ECAI Ontology Learning Workshop, 2000.

[8] E. Charniak and M. Berland. Finding parts in very large corpora. In Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics (ACL), pages 57-64, 1999.

[9] Etzioni, O., Cafarella, M., Downey, D., Popescu, A-M., Shaked, T., Soderland, S/, Weld, D., and Yates, A. (2005). Unsupervised Named-Entity Extraction from the Web: An Experimental Study. 2005. *Artificial Intelligence*.

[10] Harabagiu, S., Moldovan, D., Pasca, M., Mihalcea, R., Surdeanu, M., Bunescu, R., Gırju, R., Rus, V., and Morarescu, P. (2000). Falcon: Boosting knowledge for answer engines. In *NIST Special Publication 500-249: The Ninth Text REtrieval Conference (TREC 9)*, pages 479–488, Gaithersburg, Maryland, November 13-16.

[11] K. Ahmad, M. Tariq, B. Vrusias, and C. Handy. Corpus-based thesaurus construction for image retrieval in specialist domains. In Proceedings of the 25th European Conference on Advances in Information Retrieval (ECIR), pages 502-510, 2003.

[12] K. Markert, N. Modjeska, and M. Nissim. Using the web for nominal anaphora resolution. In EACL Workshop on the Computational Treatment of Anaphora, 2003.

[13] Katz, B. (1997). From Sentence Processing to Information Access on the World Wide Web. In Natural Language Processing for the World Wide Web: Papers from the 1997 AAAI Spring Symposium, pages 77-94, 1997.

[14] Kwok, Cody and Etzioni, Oren and Weld, Daniel S. (2001) Scaling Question Answering to the Web. In Proceedings International WWW Conference(10), Hong-Kong.

[15] Lin, J. (2005). Evaluation of Resources for Question Answering Evaluation. Proceedings of *ACM Conference on Research and development in information retrieval,* 2005.

[16] M. Poesio, T. Ishikawa, S. Schulte im Walde, and R. Viera. Acquiring lexical knowledge for anaphora resolution. In Proceedings of the 3rd Conference on Language Resources and Evaluation (LREC), 2002.

[17] M.A. Hearst, 'Automatic acquisition of hyponyms from large text corpora', in Proceedings of the 14th International Conference on Computational Linguistics, (1992).

[18] P. Cimiano, G. Ladwig, and S. Staab. Gimme' the context: Context-driven automatic semantic annotation with C-PANKOW. In Proceedings of the 14th World Wide Web Conference, 2005.

[19] P. Cimiano, S. Handschuh, and S. Staab. Towards the self-annotating web. In Proceedings of the 13th World Wide Web Conference, pages 462-471, 2004.

[20] R. Girju and M. Moldovan. Text mining for causal relations. In Proceedings of the FLAIRS Conference, pages 360-364, 2002.

[21] Radev, D. R., Qi, H., Zheng, Z., Blair-Goldensohn, S., Zhang, Z., Fan, W., and Prager, J. (2001). Mining the web for answers to natural language questions. In the *Proceedings of ACM CIKM 2001: Tenth International Conference on Information and Knowledge Management*, Atlanta, GA, November 5-10.

[22] Radev, D., Fan, W., Qi, H., Wu, H., and Grewal, A., (2005). Probabilistic question answering on the web, *Journal of the American Society for Information Science and Technology, 56(3)*.

[23] Ravichandran, D., and Hovy, E. (2002). Learning surface text patterns for a question answering system. In Proceedings of ACL, 2002.

[24] Roussinov, D., Chau., M., Filatova, E., Robles, J., Building on Redundancy: Factoid Question Answering, Robust Retrieval and the "Other". In proceedings of *TREC 2005*, Nov. 15-18, 2005.

[25] Schlobach, S., Olsthoorn, M., and de Rijke, M. (2004). Type Checking in Open-Domain Question Answering (Extended Abstract), In: R. Verbrugge, N. Taatgen, and L. Schomaker, editors, Proceedings BNAIC 2004, pages 367-368, 2004.

[26] Soubbotin, M. and Soubbotin, S. (2002*).* Use of patterns for detection of likely answer strings: A systematic approach. *Proceedings of the Eleventh Text Retrieval Conference TREC 2002*. Gaithersburg, Maryland, November 19-22.

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