Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 2004 Proceedings

Americas Conference on Information Systems (AMCIS)

December 2004

Web Question Answering: Technology and Business Applications

Dmitri Roussinov
Arizona State University

Jose-Antonio Robles-Flores
Arizona State University

Web Question Answering: Technology and Business Applications

Dmitri Roussinov
Arizona State University
Dmitri.Roussinov@asu.edu

José Antonio Robles-Flores
Arizona State University / ESAN
Jose.Robles@asu.edu

ABSTRACT

While being successful in providing keyword based access to web pages, commercial search portals, such as Google, Yahoo, AltaVista, and AOL, still lack the ability to answer questions expressed in a natural language. We have explored the feasibility of a completely trainable approach to automated question answering on the Web for the purpose of business intelligence and other practical applications. We introduce an entirely self-learning approach that does not involve any linguistic resources. It can be easily implemented within various information awareness systems. The performance of our approach was found comparable to (and even relatively better than) many other more complex and expensive approaches. We also present the design of our empirical study and the qualitative observations from our pilot experiments.

Keywords

Question-answering systems, Information Retrieval, fact seeking, pattern matching, information triangulation, WWW.

INTRODUCTION

Virtually any science fiction work depicting the future, from Spielberg to Asimov, includes scenes where people converse with a machine in natural language to get answers to their questions. This interaction has been a dream of artificial intelligence (AI) since the invention of computers. Recent advances in Natural Language Processing (NLP) and AI in general have approached this dream world to the point where it mixes with reality. Several known futurists (people who make their living entirely by trying to predict "future") believe that computers will reach capabilities comparable to human reasoning and understanding of languages by 2020 (Lempert, Popper, Bankes, 2003; Deen, Jhingran, Navathe, Neuhold, Wiederhold, 2000).

Business Intelligence refers to the "ability to understand the business environment in order to make decisions." (Prior, 2003). From the IT perspective, it is defined as "use of high-level software intelligence for business applications" (Bernstein, Grossof, Provost, 2001) Examples of BI activities include, but are not limited to, data mining and visualization, data warehousing, machine learning and knowledge discovery, recommendation and reputation systems, automated contracting/brokering/negotiation, and intelligent information retrieval (Bernstein et al. 2001).

The goal of Question Answering (QA) is to locate, extract, and represent a specific answer to a user question expressed in natural language. A QA system would take as input a question like "What is mad cow disease?" and it should get as output "Mad cow disease is a fatal disease of cattle that affects the central nervous system. It causes staggering and agitation."

In the business environment, we may have the following scenarios:

- Find who the senior managers of a competitor firm are (e.g. Who is the CEO of IBM?)
- Find geographical facts for marketing purposes (e.g. What is the longest river in the U.S.?)
- Find companies that manufacture a particular product (e.g. Who makes rod hockey games?)
- Find definitions to unknown terms (e.g. What does audit committee mean?)

Thus, QA fits naturally into the set of tools desired by business analysts.

While being quite successful in providing keyword based access to web pages, commercial search portals, such as Google, Yahoo, AltaVista, and AOL, still lack the ability to answer questions expressed in a natural language. Recent studies (Radev,

Fan, Qi, Wu, Grewal,, 2002) indicated that the current WWW search engines, especially those with very large indexes like Google, offer a very promising source for open domain question answering. Numerous efforts are now under way which port and adapt existing QA techniques to the much larger context of the World Wide Web. The first commercial company offering QA services, Ask Jeeves, supports natural language queries and provides the technological support for Ford and Nike (Maybury, 2004). However, their questions and answers are manually constructed and lack the automatic answer identification capability.

In contrast to the NLP-based approaches that rely on laboriously created linguistic resources, "shallow" approaches that use only simple pattern matching and inherent redundancy of large text repositories have been recently tried successfully. A recent work by (Dumais, Banko, M., Brill, E., Lin, J., and Ng, 2002) presented an 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 semantic filters to achieve a striking accuracy: Mean Reciprocal Rank (MRR) of *0.507*, which can be roughly interpreted as "in average" the correct answer is the second answer found by the system.

As we are working on several projects that include QA systems, we are interested in approaches that rely on machine learning techniques, rather than on manually crafted rules or expensive linguistic resources. Our approach expands the work by Dumais et al. (2002) by exploring the feasibility of automated identification and training of simple patterns for question answering purposes. There are several advantages of a pattern-based approach over a "deep" linguistic approach that we hope can render the former attractive for technology investors: 1) *Simplicity*: patterns can be automatically learned without extensive manual development effort 2) *Objectivity*: results of the studies can be easily replicated by other researchers 3) *Speed*. The approach can also be combined with "deeper" technologies, for example for quick identification of sentences that can be later processed by "deeper" techniques.

The purpose of our studies was not to develop commercial software competing against the other similar systems. Instead, we designed and implemented our prototype specifically to test our approach and the research hypothesis associated with it. That is the reason why we deliberately left out several heuristics that can speed up the processing or slightly increase the accuracy. Another important distinction of our work is that we evaluate our approach empirically through controlled experiments, which is, up to our knowledge, a first open domain empirical evaluation of a QA system.

TECHNOLOGY INVOLVED

While searching for an answer to a question (e.g. *"Who is the CEO of IBM?"*), our approach looks for matches to certain patterns. For example *"The CEO of IBM is Samuel Palmisano."* matches the pattern "*\Q is \A ."* where \Q is a *question part* ("The CEO of IBM") and *\A = "Samuel Palmisano"* is the text that forms a *candidate answer*. We automatically create and train up to *200* patterns for each type of a question (examples of types of questions are *what is*, *what was*, *where is*, etc.) based on a training set of given question-answer pairs. Through training, each pattern is assigned the 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, \p* (punctuation mark) and * (a wildcard that matches any words) are the only special symbols used in our pattern language. Figure 1 summarizes our proposed approach.

Question Answering Steps

Answering the question *"Who is the CEO of IBM?"* demonstrates the steps of our algorithm.

Type Identification. The question itself matches the pattern *who is \Q ?*, where *\Q = "the CEO of IBM"* is the question part and "*who is*" is the type identifier.

Query modulation converts each answer pattern (e.g. "*\A became \Q \p"*) into a query for a general-purpose search engine (GPSE), e.g. *"became the CEO of IBM"*. The answer pattern *"\Q is \A"* would be converted into *"the CEO of IBM is"* etc.

Answer Matching. The sentence *"Samuel Palmisano recently became the CEO of IBM."* would result in a match and produce a candidate answer "*Samuel Palmisano recently*".

Answer Detailing produces more candidate answers by forming sub-phrases from the initial candidate answers. Our sub phrases do not exceed 3 words (not counting "stop words" such as *a, the, in, on*) and do not cross punctuation marks. In our example, the detailed candidate answers would be *Samuel*, *Palmisano*, *recently, Samuel Palmisano, Palmisano recently*.

The Triangulation process establishes that *Samuel Palmisano* is the most likely answer as explained in the next section.

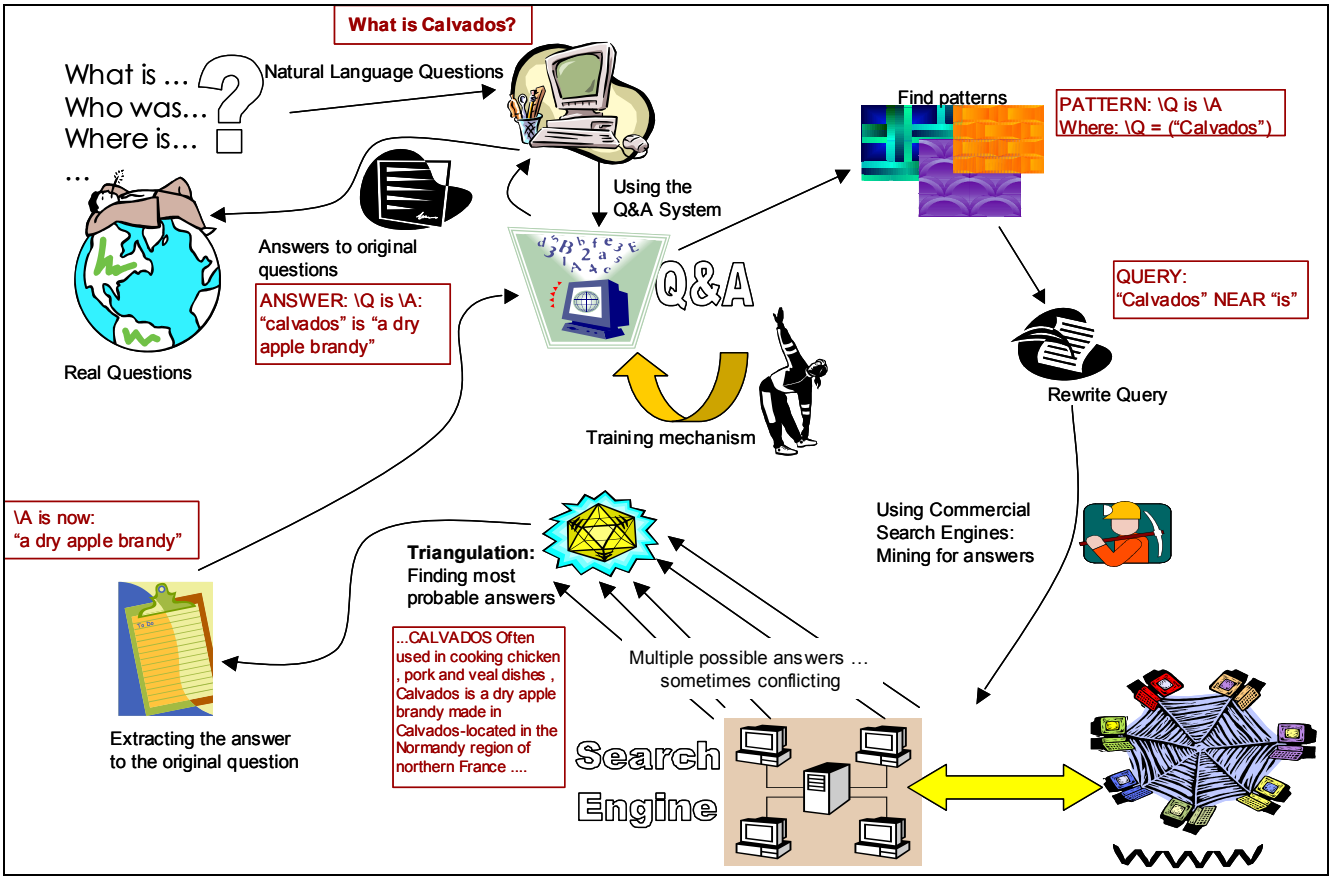

Figure 1. The general Web QA approach

The algorithm stops querying the GPSE when a specified number of web pages has been scanned (1000 in this study). If there are fewer than expected (200 in this study) candidate answers found, the algorithm resorts to a "fall back" approach: it creates candidate answers from each sentence of the snippets returned by GPSE and applies answer detailing to them. If there are still not enough candidates, the system automatically relaxes the modulated query by removing the most frequent words on the Web until enough many hits are returned by the GPSE. This way, it can simplify a question *"Who still makes rod hockey games?"* to *"Who still makes rod hockey?"*

Triangulation

Triangulation, a term widely used in intelligence and journalism, stands for confirming or disconfirming facts using multiple sources. Our triangulation algorithm can be demonstrated by the following intuitive example. Imagine that we have two candidate answers for the question *"What was the purpose of Manhattan Project?"* 1) *"To develop a nuclear bomb"* and 2) *"To create a nuclear weapon."* Those two answers reinforce (triangulate) each other since they are semantically similar. The advantage of triangulation over simple frequency counting (Dumais et al., 2002) is even stronger for less "factual" questions, those that may allow variation in the correct answers. This includes such frequent types as definitions and "how to" questions. Although, there are many known measures of semantic similarity between words and phrases, for simplicity, we used *relative significant overlap* in the current implementation:

sim (a1, a2) = so(a1, a2)/(length(a1)+length(a2)), where *so(a1, a2)* is the number of words that are present in both *a1* and *a2*, that are not stopwords or not the words present in the question part, and *length(a1)* is the count of words excluding stopwords in *a1*. In the above example *sim = 1/(3+3)*. The resulting score for each candidate answer $s^t(a)$ after triangulation

is computed by summation: $s^t(a) = \sum_{a_i \in O} s(a_i) \cdot sim(a, a_i)$, where O is the set of all original (before detailing) answers and $s(a)$ is the original score.

Pattern Training

For each training pair (Q, A), the system requests the web pages from the GPSE that have both the question Q and the answer A, preferably in proximity. Each sentence containing both Q and A is converted into a candidate pattern by replacing the question phrase with $\backslash Q$ symbol and the answer with $\backslash A$. Once a specified number of candidate patterns is identified (200 in our study), additional patterns are generated through a recursive "generalization" process of replacing words with wildcards and forming substrings containing both $\backslash Q$ and $\backslash A$. The obtained top most frequent 500 patterns are trained for the probability of matching the text that includes a correct answer by the modulation and matching processes similar to the described above.

Scalability and Responsiveness

Since our objective was to explore the feasibility of the approach, we were not that much concerned with real-time responsiveness. Our proof of concept prototype finds an answer within minutes. The bottleneck is fetching the contents of the web pages, which can be parallelized on multiple workstations that would send to the central server only the identified candidate answers, as, for example, has been successfully demonstrated in (Surdeanu, Moldovan, and Harabagiu, 2002). Another solution is to have direct access to the GPSE index and cache, which may be, for example, possible when the QA system is an internal part of it.

EMPIRICAL EVALUATION

Simulation

Due to the variety of question types that people may ask, a task of compiling a test collection with the distribution of questions matching real world needs is still a daunting unaccomplished task. TREC collections (Voorhees and Tice, 2000) have been constructed mostly by its participants. As a result, TREC organizers never have made any claims about representativeness of their collections. Another potential problem is that not all, or too many, of the answers to TREC questions can be found on the Web. The most representative list of questions used in prior studies is Excite data set (Agichtein, Lawrence, Gravano, 2001) which is a list of queries submitted to Excite search engine. Approximately 8.4% of the queries are natural language questions (Radev, Qi, Zheng, Blair-Goldstein, Zhang, Fan, and Prager, 2001). Unfortunately, Excite data set does not include answers thus using it in batch mode evaluations would be very laborious since all the correct answers would need to be added manually.

For comparison with the prior study (Dumais et al., 2002), that is the closest to our experiment, we used TREC Q/A data sets (Voorhees and Tice, 2000) from 1999 to 2002 for training, except the year 2001, which we used for testing. Similarly, we had to add answers that seemed correct but were not included in the original set. We followed the following procedure for this: We reviewed log files after preliminary runs and any of the correct answers among the top 10 were added. This procedure was surprisingly not as time consuming as we expected and took approximately 10 hours for the entire study. We made the questions and expanded the answers (available online) (url hidden for blind review) for possible replication and follow up studies.

Although various metrics have been explored, we used mean reciprocal rank of the answer (MRR) as in Dumais et al. (2002) again for comparison. The drawback of this metric is that it is not very sensitive since it only considers the first correct answer, ignoring what follows. Also, longer answers have a higher chance of including a correct answer, a case that we counted as correct since we did not apply any penalties to additional information in the answer (e.g. "President Bush" not "Bush").

We achieved mean reciprocal rank of the answer (MRR) of *0.314*, which is smaller than reported in Dumais et al. (2002) , but still generally better than results reported with the other approaches (Agichtein et al., 2001; Radev et al., 2002) that were trainable (not entirely relying on hand-crafted rules).

We believe that the lower results are due to our use of AltaVista instead of Google and due to the fact that we did not use any manually crafted semantic filters. Without them, Dumais, et al. (2002) reported MRR of *0.416*. It included mostly clearly defined answers, the scenario when our triangulation mechanism does not have much of an advantage over simple frequency count. We have not yet comprehensively tested each component separately.

Empirical Study Design

We have designed an experiment in order to test the Web QA approach by using a prototype developed for the objectives of this research. We have already pilot tested our design.

The scenario for the experiment is a set of general questions that the subjects are asked to answer by providing both, the actual answer and a link to the document where the answer was found (or derived). Subjects are graduate and undergraduate student volunteers. We involve subjects with various degrees of experience (self-assessed), using computers with access to a web searching tool. Each subject is given a set of questions and the specific amount of time to find the answers. The subjects alternate the use of a major commercial general-purpose search engine (GPSE) and our prototype of the QA system. The GPSE will be filtered in order to test the popular approach that uses a search window leaving apart other features like answers to definitions and access to pre-compiled databases. We do this filtering because we want to test the GPSE approach with our proposed self-learning pattern-based QA approach that includes triangulation. Questions are rated according to two categories: easy and difficult. The categorization of questions is performed in two steps: First, the researches did a categorization by searching the web to find answers. If the task requires more than two queries and either too many or few answers were found then the question is categorized as difficult; if the opposite happens, then it is an easier question. The second step is to confirm (or disconfirm) the initial categorization by using the comments and the results (in terms of time spent for each question) of the pilot studies since these pilots are performed by graduate students who rate themselves as advanced users of search engines and the web.

After the first pilot studies, we hypothesize that the automated QA approach is more helpful with the more difficult questions. We also hypothesize that novice users will gain more from using the QA tool (this was derived from the qualitative evaluation and comments of the pilot-subjects). The experiment is followed by a brief survey of the subjects' experience with both tools.

The set of questions that is used in the experiment is a subset of the Excite set. There are several reasons to use this set as the basis of our test questions:

- It is a publicly available set of questions.
- To avoid researcher's bias.
- It is being used by a community of researchers in QA.

Because the Excite list of questions was automatically generated, we had to filter the questions from the set in order to avoid questions that could be offensive by nature or language. This procedure has been used previously in this kind of experiments (Agichtein et al., 2001). Finally, a random set is generated from the pre-filtered set.

		User skill level (F3)	
Technology/Approach (F1)	**Task difficulty (F2)**	S1 :Advanced	S2: Novice
A1: Search engine	T1: difficult	X	X
	T2: easy	X	X
A2: Web QA	T1: difficult	X	X
	T2: easy	X	X

Table 1. The Experimental Design for the Empirical Study

The resulting experiment is a 2x2x2 factorial design. Table 1 summarizes the design. The first factor is the tool being used as representative of the approach being tested. The second factor is the level of difficulty of the task, in this case the category of

the question (easy or difficult). The third and last factor is the degree of experience of the user which could be either advanced or novice user.

The scores to be compared are the amount of good answers found under each of the conditions.

Qualitative Observations

Preliminary results of the pilot test are encouraging. Subjects who have participated in these pilots have offered invaluable feedback which will be used to correct errors and enhance the experiment. We also obtained positive comments regarding the easiness of use of the Web QA tool and that it is more appropriate for finding answers than the commercial general-purpose search engine. What is interesting is that the subjects liked the idea of having the actual answer presented. They also liked to have the links to web pages presented in order of probability as the GPSE typically does.

However, it was no easy task to set the minds of the subjects to ask questions in natural language. When they saw the questions in a word-processing document on the screen they immediately jumped to the QA tool and tried to type a simple pattern in place of the question. We had to make adjustments to make them understand that they could actually enter the questions or just copy and paste them from the original word processing document. This is a clear indication of the widespread use of general purpose search engines.

CONCLUSIONS AND FUTURE RESEARCH

Although the performance of our system tested in batch mode (simulation) is worse than of those relying on hand-crafted rules, they are still in the same "ballpark" and are better than results reported with the other trainable approaches. Thus, our results are encouraging considering that our approach does not require any manual tuning and is replicable by follow-up studies. It can be easily implemented in other open domain knowledge management systems with large amount of redundancy. Our future work will improve the system by introducing completely trainable filters for the semantic types of the answers. We are also planning to conclude our experiment by the time of the conference presentation.

Since QA systems described in this work utilizes the vast amounts of documents stored in web servers, the answers to many questions are publicly available. Thus, it makes this technology a legal and ethical method within Business Intelligence (Vedder, Vanecek, Guynes, and Cappel, 1999). Once deployed in organization, QA technology increases employee information awareness and reduces information overload. It is significantly more precise than keyword based retrieval still commonly used. Other future applications of automated QA that we plan to work on include:

Automated federating heterogeneous database. E.g. by answering a question "What is SSN?", the "SSN" attribute in one schema can be automatically matched with "Social Security Number" attribute in another.

Deception detection by automated triangulation (confirming/disconfirming) of the statements made in a specific document, e.g. if the document states that the capital of China is Taipei, and the system finds "Beijing" as the answer to the question "What is the capital of China?" then the document should be treated with suspicion.

REFERENCES

1. Agichtein, E., Lawrence, S., Gravano, L. (2001) Learning Search Engine Specific Query Transformations for Question Answering. *Proceedings of the Tenth World Wide Web Conference,* Gaithesburg, MD.
2. Bernstein, A., Grosof, B. and Provost, F. (2001) Business Intelligence: The Next Frontier for Information Systems Research?. Panel Description, *Proceedings of the Workshop on Information Technologies and Systems*, New Orleans, LA.
3. Dumais, S., Banko, M., Brill, E., Lin, J., and Ng, A. Web Question Answering: Is More Always Better? (2002) *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Tampere, Finland.
4. Lempert, R. J., Popper, S. W., Bankes, S. C. (2003). Shaping the next one hundred years: new methods for quantitative, long-term policy analysis, RAND, Santa Monica, CA.
5. Maybury, M. (2004) Editor. New Directions in Question Answering, AAAI/MIT Press, Cambridge. Forthcoming.
6. Portsmouth, I. (2002) 20 bold predictions for the next 20 years. PROFITGuide.com, Rogers, Inc. Electronic version at http://www.profitguide.com/shared/print.jsp?content=928
7. Prior, V. (2003) BI - Business Intelligence, e-News, Malaysian Institute of Management. Electronic version at http://www.mim.edu/news/MA1030.htm

8. Radev, D., Fan, W., Qi, H., Wu, H., Grewal, A. (2002) Probabilistic Question Answering on the Web. *Proceedings of the 11th World Wide Web Conference*, Honolulu, HI.
9. Radev, D., Qi, H., Zheng, Z., Blair-Goldstein, S., Zhang, Z., Fan, W., and Prager, J. (2001) Mining the Web for Answers to Natural Language Questions. In the *Proceedings of the ACM CIKM 2001: Tenth International Conference on Information and Knowledge Management*, Atlanta, GA.
10. Surdeanu, M., Moldovan, D. and Harabagiu, S. (2002) Performance Analysis of a Distributed Question Answering System', *IEEE Transactions on Parallel and Distributed Systems,*13, 6, 579-596.
11. Vedder, R. G., Vanecek, M. T., Guynes, C. S., and Cappel, J. J. (1999). *Communications of the ACM*, 42, 8; 108-116.
12. Voorhees, E. M., Tice, D. M. (2000) Building a question answering test collection, *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, Athens, Greece.