AnswerBus Question Answering System

Zhiping Zheng
School of Information
University of Michigan
Ann Arbor, MI 48109
zzheng@umich.edu

ABSTRACT

AnswerBus[1] is an open-domain question answering system based on sentence level Web information retrieval. It accepts users' natural-language questions in English, German, French, Spanish, Italian and Portuguese and provides answers in English. Five search engines and directories are used to retrieve Web pages that are relevant to user questions. From the Web pages, AnswerBus extracts sentences that are determined to contain answers. Its current rate of correct answers to TREC-8's 200 questions is 70.5% with the average response time to the questions being seven seconds. The performance of AnswerBus in terms of accuracy and response time is better than other similar systems.

Keywords

open-domain question answering,, QA specific dictionary, information retrieval

1. INTRODUCTION

Automated question answering (QA) research can be dated back to the 1960s, nevertheless has often been confined to domain-specific expert systems ([12], [9]). Researchers have experimented with QA systems based on closed, pre-tagged corpora (e.g., [8], [13], [3]), or knowledge bases (e.g. [11], [5], [4]). Many of these systems focus on Text REtrieval Conference (TREC) tasks. Researchers also have attempted to build QA systems on large collections of documents on the Web by combining information extraction and most advanced information retrieval technology (e.g., [7], [12], [1]). Recently, researchers have been attracted to the task of developing open-domain QA systems based on collections of real world documents, especially the World Wide Web. Examples of such systems include LCC[2]([7]), QuASM[3], IONAUT[4]([1]), START[5]([11]) and Webclopedia[6]([10]). At the current stage, these systems seem to typically have long response times and/or accuracy rates that may not be acceptable to users.

AnswerBus is an open-domain question answering system based on sentence level information retrieval. It accepts users' natural-language questions in English, German, French, Spanish, Italian and Portuguese and extracts possible answers from the Web. It can respond to users' questions within several seconds. Five search engines and directories (Google, Yahoo, WiseNut, AltaVista, and Yahoo News) are used to retrieve Web pages that potentially contain answers. From the Web pages, AnswerBus extracts sentences that are determined to contain answers. The current rate of correct answers to TREC-8's 200 questions is 70.5%. AnswerBus demonstrates that practical question answering on the Web is highly feasible.

Figure 1 describes the working process of AnswerBus. AnswerBus takes a user question in natural language. A simple language recognition module will determine

Proceedings of HLT 2002, Second International Conference on Human Language Technology Research, M. Marcus, ed., Morgan Kaufmann, San Francisco, 2002.

[1] http://www.answerbus.com/

[2] http://www2.languagecomputer.com/qa/head.html

[3] http://ciir.cs.umass.edu/~reu2/

[4] http://www.ionaut.com:8400/

[5] http://www.ai.mit.edu/projects/infolab/

[6] http://www.isi.edu/natural-language/projects/webclopedia/research.html#QUESTION

whether the question is in English, or any of the other five languages. If the question language is not English, AnswerBus will send the original question and language information about the question to AltaVista's translation tool BabelFish[7], and obtain the question that has been translated into English.

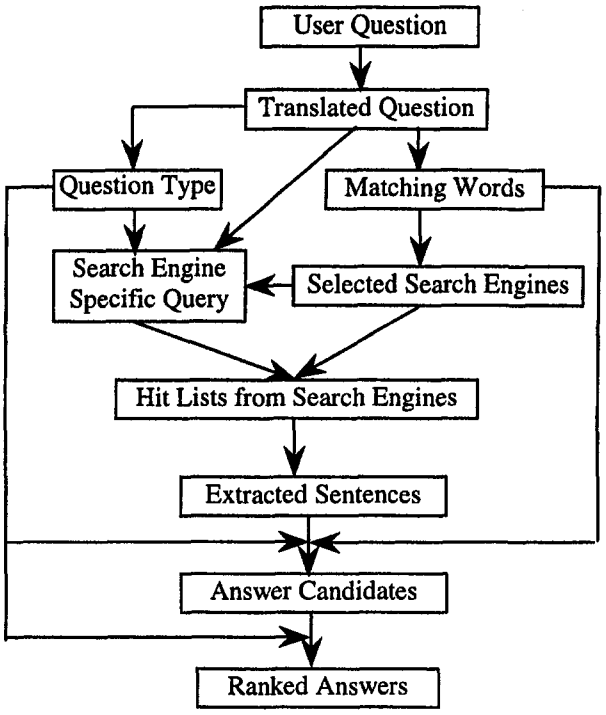

Figure 1 Working process of AnswerBus

The rest of the process is comprised of mainly four steps: 1) select two or three search engines among five for information retrieval and form search engine specific queries based on the question; 2) contact the search engines and retrieve documents referred at the top of the hit lists; 3) extract sentences that potentially contain answers from the documents; 4) rank the answers and return the top choices with contextual URL links to the user. Instead of returning a snippet of fixed length of text, AnswerBus returns sentences, which can provide users with some contextual information for the answers.

2. RELEVANT DOCUMENTS RETRIEVAL

AnswerBus aims to retrieve enough relevant documents from search engines within a response time that is acceptable to users. The main tasks in this stage are to select one or more appropriate search engines for a specific user question, and then form queries that are tailored to the question as well as the selected search engines. The formation of the queries is an essential procedure because it can largely influence the recall and accuracy of question answering and the speed of the system operation.

2.1 Search engine selection

Different search engines or directories may suit different types of questions. For example, for questions about current events, Yahoo News may be a better choice than Google. Thus, for a specific question, AnswerBus chooses to use two to three most appropriate search engines among the five ones.

In order to determine which search engines are best suited for a specific question, AnswerBus pre-answered 2000 sample questions, including TREC 8 and TREC 9 questions, and also questions typed in by testing users. It sent the queries based on these questions to all of the five search engines. For each question, it recorded the number of possible answers that came from the different search engines. All the words used in queries are indexed. For example, for Word 1, Google may return 8 answers, AltaVista returns 4 answers, and Yahoo returns 7 answers; for Word 2, Google 6 answers, AltaVista 6 answers and Yahoo 5 answers. For a query with Word 1 and Word 2, AnswerBus will choose to use Google (8+6) and Yahoo (7+5). If a query contains words not included in the indexed list, AnswerBus uses the search engines' average returns for all the indexed words to determine which search engines are most appropriate.

2.2 Search engine specific query formation

Most search engines are not designed for natural language questions. [2] shows that QA systems using good queries will significantly outperform the underlying web search engines and a commercial search engine specializing in question answering. [6] argues that using extra information to direct the search process provides more valuable results than by considering only the query.

Search engine specific query here refers to the query formed from a user's natural- language question that tailed to particular search engines, and will generate optimal search outcomes. "Optimal" does not necessarily mean recall of the largest number of relevant documents. Instead, it means the best outcomes in terms of both recall of documents and time to retrieve the documents for a QA system. For example, a user wants to find out "How tall is Mount

[7] http://babel.altavista.com/

Everest." If a full question is sent to a commercial search engine like Google, normally the user will be able to harvest the documents that contain the answer. Nevertheless, more web pages in the returned hit list will be irrelevant to the question, thus leads to lower precision. It certainly takes a much longer time to retrieve and process the documents. Thus it is necessary to form a *search engine specific query*, in this case, "Mount Everest", that will best fit the search engine and yield optimal search results.

However, the task of forming the search engine specific query is not always as simple as this example, given that each question has its unique structure and content while different search engines have different rules about the queries they accept.

Some approaches expand one query to several queries, for example, by adding synonyms, then either concatenate these queries together with operator "OR" or keep them as separate queries. These approaches can increase the possibility of getting the correct answers, however, may also make the search more complicated and significantly prolong the search response time, and overall may be detrimental to the QA outcomes.

The computing speed has been regarded as very important throughout the development of AnswerBus. For a scalable Web-based QA system, trade-offs need to be made between speed and recall of answers. Thus AnswerBus does not try to find the best query; instead, it tries to locate the good enough query that will conduct the search task very fast. The focus has been laid on generating one simple query instead of an expanded one.

Several approaches are combined to form queries, including functional words deletion, use of word frequency table, special words deletion, and word form modification.

Functional words deletion

Functional words include prepositions, determiners/ pronouns, conjunctions, interjections, and discourse particles. Functional words deletion, which often can make the query short enough, can be used as a baseline of search engine specific query formation.

Some words are not functional words but are acting as functional words either logically or structurally for example, "kind of," "name the designer of" can also be deleted from the query:

For long questions, the query made by this step is still too long. For example, after deleting all the functional words in question "*What is the name of the rare neurological disease with symptoms such as: involuntary movements (tics), swearing, and incoherent vocalizations (grunts, shouts, etc.)?*", we get "*name rare neurological disease symptoms involuntary movements tics swearing incoherent vocalizations grunts shouts*". The length of this query is 13 words and is not good enough to be a query for QA tasks.

Use of word frequency table

Another way to make a query shorter is to delete frequently used words in the query. The basic idea is that the more frequently a word is used in a language, the less discriminating the word is. Thus AnswerBus implements a word frequency table. For a long query, AnswerBus sorts all the words in the query and deletes one or more words that are identified as frequently used by the frequency table.

Word form modification

Some words in the original question are converted to another form then put in the query. Usually they are verbs, for example,

end → *ended* in question "When did the Jurassic Period end?"

have → *has* in question "How many hearts does an octopus have?"

3. CANDIDATE ANSWER EXTRACTION

At this stage, AnswerBus downloads and processes the documents referred at the top of search results returned by different search engines. It first parses the documents into sentences and then determines whether a sentence is an answer candidate through a process of word matching.

The sentence segmentation tool used in AnswerBus is designed to process complicated Web documents. In addition to deleting HTML tags, it excludes non-contextual content; regards some special HTML tags as sentence boundary indications; and takes different formatting exceptions into consideration.

In order to determine whether a retrieved sentence is potentially an answer to the question, AnswerBus classifies all words in the original question or sentences in retrieved documents into two categories: matching words and non-matching words. All words

that are used to form the search engine specific query are matching words. The rest are non-matching words.

The following formula is used to filter retrieved sentences.

$$q \geq \left\lfloor \sqrt{Q-1} \right\rfloor + 1$$

In this formula, q is the number of matching words in the sentence; Q is the total number of matching words in the query. For example, if a query contains three words, then an answer candidate sentence should have at least two of them. When a sentence meets the condition as indicated by the above formula, it will receive a primary score based on the number of matching words it contains. Otherwise, it will receive a score of "0."

4. ANSWER RANKING

After the extraction of answer candidate sentences, each sentence has received a primary score. Those sentences with a score of "0" are dropped. Nevertheless, the primary scores are not robust enough for the judgment whether a sentence is a real answer. AnswerBus uses several techniques to refine the primary scores, including the determination of question type, use of a QA specific dictionary, named entities extraction, coreference resolution. The final score that is used to determine the rank of an answer is a combination of the primary score and the influence of all the different factors.

4.1 Question type and QA specific dictionary

Almost all QA systems use question type to judge the answer. They classify the question types on the basis of types of answers users are expecting. For example, a "*Who is ...?*" will be assigned a "PERSON/ORGANIZATION" type; while "*When did ... happen?*" will be classified as "DATE/TIME" question.

AnswerBus also uses question type as an important piece of information to judge whether a sentence can be an answer to a question. AnswerBus classifies questions into different question types together with some parameters. For example, AnswerBus classifies both of "*How far ...?*" and "*How close ...?*" questions as DISTANCE question, but it also differentiate these two types of questions: the unit of the answer to "*How far ...?*" most likely will be "mile," "kilometer," "light year" and other related bigger unit, it has small chance to be "inch," "centimeter" etc. For "*How close ...?*" question, the unit of the answer to this question could be any of above, depending on the context in the question, it even could be "nanometer" or others.

AnswerBus uses a QA specific dictionary, a database containing this kind of information about the relationship of words between questions and answers. For example, for the entry of word "far," the definition provided in the dictionary contains "miles," "kilometers," "light years;" for the word of "high," the definition contains "feet," "meters." The dictionary is used to distinguish question types, and determine whether a sentence is the right answer.

4.2 Dynamic basic named entities extraction

Named entities extraction has been widely used in closed QA systems to provide support for answer selection. It is achieved through pre-tagging of the corpus or knowledge base on which a QA system is based. According to [13], the speed of tagging is approximately 100M/hour. With the assumption that, for one question, a Web-based QA system needs to process 50 HTML documents of average length, with the total size of 1M bytes, the time needs to be extended for tagging these documents will be 36 seconds. For such a system, there are still other time cost such as network response, network transfer, and other textual processing. Thus, normal tagging techniques are not unfeasible for real-time QA systems.

AnswerBus conducts dynamic named entities extraction while processing sentences from Web documents. AnswerBus only extracts the named entities that match question types. For example, when a question is asking about "how much money", a sentence with the entity of "CURRENCY" will receive a higher rank.

4.3 Coreference resolution

Some sentences contain words, such as "he," "they," that coreference to other objects described in the document. They are not as good as a real noun phrase to be answers to the questions, but they can direct users to the contextual URL links that contain the full answers. Thus, AnswerBus gives lower but still valid scores to these sentences. Coreference resolution is also a heavy NLP task like named entities extraction. Instead of using full coreference resolution, AnswerBus only solves the coreferences in the adjacent sentences. When this type of coreference is

detected, the later sentence receives part of score from its previous sentence.

4.4 Hit position and search engine confidence

The ranking score of a sentence can also be related to the position that its source document is located in the hit list returned by a search engine. A sentence extracted from the first hit receives the highest score associated with hit positions and the score decreases as the position moves down. Documents returned by different search engines may also receive different scores.

4.5 Redundancy

Different search engines can retrieve a same document for a question. Same or very similar sentences can be extracted from either a same document or different documents. This leads to candidate answer sentence redundancy.

To detect the sentence redundancy, AnswerBus compares so-far highly ranked sentences against one another. However, punctuations, spaces, other special characters, and the words with very high frequency are not considered in order to make comparisons among the sentences.

5. EVALUATION

TREC 8's 200 questions were used to evaluate AnswerBus's question answering performance and also compare its performance to that of four other similar systems. A computer program was written to send questions one by one to AnswerBus, START[11], LCC, IONAUT, and QuASM; and also retrieve the answers from the systems. After the answer to a question was received, a time interval of 10 seconds was provided before next question was sent to ensure that a system's performance would not be affected by its work on the previous one. The response times to each question by the systems and the lengths of the returned answers were recorded. In order to minimize the impact of network performance on the variations of the systems' response time, the answers from these systems were retrieved at the same time using a same computer.

Except for IONAUT, the top five answers to each question from the systems were then evaluated manually to determine the answers' accuracy. IONAUT returned long passages of information as the answers to the questions. The evaluation of the information was beyond the resources of this project. Thus, for the rest of the systems, the answers were first compared to answer keys provided by TREC. Since these systems are based on the Web, which is different from the large corpus of newswires on which the TREC questions are based, some answers that were different from answer keys were also judged as being correct. The answers were also carefully examined to ensure contextually correct answers are located.

Table 1 presents the performance of AnswerBus and other four systems. It provides the numbers of correct answers in the systems' top five and top one answers, the standard NIST scores, the maximal, minimal and average response times measured in seconds, standard deviation of response times, and the average lengths of returned answers.

Table 1 demonstrates that AnswerBus outperforms other similar systems, in terms of both accuracy and response time. AnswerBus also returned more concise answers than other systems.

Table 1 The Performance of Online Question Answering Systems

Systems	Correct TOP 5	Correct TOP 1	NIST Score	*Tmax (s)*	*Tmin (s)*	*Tmean (s)*	*Tstd dev*	*Lmean (byte)*
AnswerBus	141	120	64.18%	15.06	3.79	7.20	3.07	141
IONAUT				44.88	2.78	12.51	6.81	1312
LCC	97	75	41.73%	342.52	4.30	44.24	32.63	178
QuASM	13	7	4.45%	284.29	2.61	20.72	33.92	1766
START	29	29	14.50%	62.07	2.02	9.84	7.45	

Table 2 Performance of AnswerBus As Compared to Mulder

Systems	Correct Top 5 Answers	Correct Top 1 Answers	NIST Score
AnswerBus	70.5%	60%	64.2%
Muldera		34%	

Note: [a] Source [12].

No similar data for other two famous QA systems, Mulder and Webclopedia, were obtained because of their off-line status when the experiment was taken. According to [12], Mulder correctly answered 34% of TREC-8 questions. Table 2 gives some comparison between AnswerBus and Mulder.

References

[1] Steven Abney, Michael Collins, and Amit Singhal. Answer Extraction. *Proceedings of ANLP 2000.* Seattle, WA. April 29 - May 3, 2000.

[2] Eugene Agichtei, Steve Lawrence, Luis Gravano. Learning Search Engine Specific Query Transformations for Question Answering. *Tenth World Wide Web Conference.* May 1-5, 2001, Hong Kong, China.

[3] Eric Breck, John Burger, Lisa Ferro, David House, Marc Light, and Inderjeet Mani. A sys called Qanda. *Eighth Text REtrieval Conference (TREC-8).* Gaithersburg, MD. November 17-19, 1999.

[4] Jay Budzik and Kristian J. Hammond. Learning for Question Answering and Text Classification: Integrating Knowledge-Based and Statistical Techniques. *AAAI Workshop on Text Classification.* Menlo Park, CA, 1998

[5] Peter Clark, John Thompson, and Bruce Porter. A knowledge-based approach to question-answering. *AAAI'99 Fall Symposium on Question-Answering Systems.* Orlando, Florida. 1999.

[6] E. J. Glover, S. Lawrence, M. D. Gordon, W. P. Birmingham, and C. L. Giles, "Web Search -- Your Way," *Communications of the ACM*, To appear.

[7] Sanda Harabagiu, Dan Moldovan, Marius Paşca, Mihai Surdeanu, Rada Mihalcea, Roxana Girju, Vasile Rus, Finley Lacatusu, Paul Morarescu and Razvan Bunescu. Answering Complex, List and Context Questions with LCC's Question-Answering Server. *Tenth Text REtrieval Conference (TREC-10).* Gaithersburg, MD. November 13-16, 2001.

[8] Sanda Harabagiu, Marius Pasca, and Steven Maiorano. *Experiments with open-domain textual question answering. COLING-2000. Association for Computational Linguistics/Morgan Kaufmann*, Aug 2000.

[9] Lynette Hirschman and Robert Gaizauskas. Natural Language Question Answering: The View from Here. *Natural Language Engineering,* 2001.

[10] Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Michael Junk and Chin-Yew Lin. Question Answering in Webclopedia. Ninth Text REtrieval Conference(TREC-9). Gaithersburg, MD. November 13-16, 2000.

[11] Boris Katz, From Sentence Processing to Information Access on the World Wide Web. *AAAI Spring Symposium on Natural Language Processing for the World Wide Web.* Stanford, California. 1997.

[12] Cody C. T. Kwok, Oren Etzioni and Daniel S. Weld. Scaling Question Answeringto the Web. *Tenth World Wide Web Conference.* Hong Kong, China. May 1-5, 2001.

[13] Rohini Srihari and Wei Li. Information Extraction Supported Question Answering. *Eighth Text REtrieval Conference (TREC-8).* Gaithersburg, MD. November 17-19, 1999.