# Getting reliable answers by exploiting results from several sources of information

Jean-Baptiste Berthelin, Gaël de Chalendar, Faïza Elkateb-Gara, Olivier Ferret, Brigitte Grau, Martine Hurault-Plantet, Gabriel Illouz, Laura Monceaux, Isabelle Robba, Anne Vilnat

# Getting reliable answers by exploiting results from several sources of information

J.-B. Berthelin, G. de Chalendar, F. Elkateb-Gara,
O. Ferret, B. Grau, M. Hurault-Plantet, G. Illouz,
L. Monceaux, I. Robba, A. Vilnat

*LIMSI-CNRS Orsay France*
{*name@limsi.fr*}

**Abstract**

A question answering system will be more convincing if it can give a user elements concerning the reliability of its propositions. In order to address this problem, we chose to take the advice of several searches. First, we search for answers in a reliable document collection, and second on the Web. When both sources of knowledge allow the system to find common answers, we are confident with it and boost them at the first places.

*Key words:*
answering system, information retrieval, answer reliability

## 1 Introduction

Open-domain Question-Answering (QA) is a growing area of research whose aim is to find precise answers to questions in natural language, unlike search engines that return documents. When those engines also return snippets, as Google [1], they aim at providing a justification of documents rather than just giving the potential answer. One challenge in this field consists in finding only one answer while being sufficiently confident in it.

Evaluating the reliability of an answer leads either to prove its truth or to estimate it. A formal proof is possible if the answer comes from a reasoning on a formal representation of knowledge. LCC (Moldovan et al. 2002) developed such an approach and their system builds inference chains from a logic

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

representation of WordNet knowledge (Fellbaum 1998). It requires a complete representation of all the knowledge necessary to answer and when dealing with open-domain questions, such a hypothesis cannot be warranted. The second possibility, corresponding to the approach we developed in QALC, our question answering system, consists in estimating the reliability of an answer by scoring it according to the kind of knowledge or the kind of process used. We found that providing just an endogenous estimation was not sufficient. Thus, we decided to apply our system on another source of knowledge in order to confront the results provided by both sources. We chose then to favour common propositions over unique ones, even if these latter had a high score. As such reasoning better applies if the sources of knowledge are different enough, we chose the Web as second source. Moreover, the diversity of the Web and its redundancy both lead to find a lot of answers, as we can see it in (Magnini et al. 2002a), (Magnini et al. 2002b), (Clarke et al. 2001) and (Brill et al. 2001).

## 2 Overview of QALC

In the spirit of the TREC[2] evaluations, the QALC system (see Figure 1) was designed to find answers to factoid questions in a large collection of documents. First, its question analysis module aims at deducing characteristics helping to find possible answers in selected passages and to reformulate questions in a declarative form that is given to the search engine (Google). These characteristics are the question focus, the main verb and syntactic relations for modifiers. The analysis is based on the results of IFSP, the robust syntactic parser of (Aït-Mokhtar and Chanod 1997). Queries are not the same for the Web search and for AQUAINT search (AQUAINT is the reference TREC collection). In the latter case, we use MG[3] for retrieving passages from a query made with AND and OR operators. For querying the Web, we chose to send a nearly exact formulation of the answer assuming that the Web redundancy will always provide documents.

Retrieved documents, 1500 passages with MG (on AQUAINT) and 20 documents from the Web, are then processed. They are re-indexed by the question terms and their variants, reordered according to the kind of terms found in them, so as to select a subset of them in the case of MG, the Web documents being all kept. Named entity recognition processes are then applied. The answer extraction process relies on a weighting scheme of the sentences, followed by the answer extraction itself. We apply different processes according to the kind of expected answer, each of them leading to propose answers

[2] TREC evaluations are campaigns organised by the NIST: http://trec.nist.gov
[3] MG for Managing Gigabytes http://www.cs.mu.oz.au/mg/

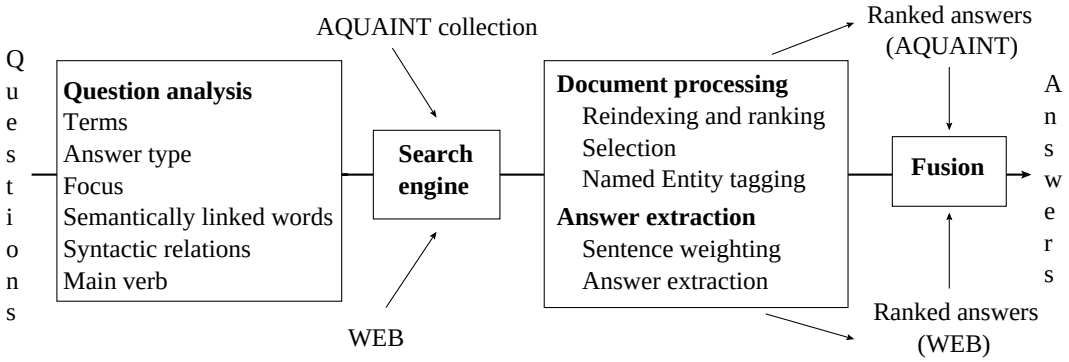


Fig. 1. The QALC system

with a weight. The final step consists in comparing the results issued from AQUAINT and from the Web and computing a final score. Its principle was to boost an answer if both chains ranked it in the top 5 propositions, even with relatively low scores.

## 3 Query formulation

Due to the Web redundancy, we state that it is possible to find documents even with a very specific query, and that a precise query will lead to find relevant documents, i.e. documents containing the searched answer, among the first ones. Thus, our choice was to reformulate the question in an affirmative form with as few variations as possible. For instance, for the question *When was Wendy's founded?*, we expect to find a document containing the answer in the form: *Wendy's was founded on ....* We first query the Web for strings with exact match in documents as in (Brill et al. 2001), and not for the different words of the query even linked with AND, OR or NEAR operators as in (Magnini et al. 2002a) or (Hermjacob et al. 2002). By this way, we can select only few documents (20 in our experiments). On the other hand, this kind of approach is not efficient on the AQUAINT collection, so we chose the opposite strategy that consists in retrieving a large amount of passages, using the possibility given by MG to rank the documents. The query is made of the lemmatised and exact content words of the question, with reinforcement of the question focus and the search process uses stemming.

Rewriting of question is based on handmade reformulation patterns resulting from the study of TREC9 and TREC10 questions. We first categorised the questions according to their answer type and their question type. Searching for a person name or a location does not lead to the same reformulation, even if the syntactic form of the question is similar. *Who is the governor of Alaska?* and *Where is the Devil's Tower?* do not expect an answer provided exactly in a same way: the query *the governor of Alaska* stands for the first question as a

name is often given in an apposition and the *Devil's Tower is located* stands for the second question as we account for an answer in the exact affirmative form. We then tried reformulations of a subset of them (about 50 questions with their answers) on the Web, using Google, to find the most frequent kinds of query patterns. These tests showed us the necessity of defining an additional criterion in order to be more precise in the rewriting. This characteristic is either the word introducing a modifier of the object in the minimal form of a question or a word of the question leading to add a word (often a preposition or a verb) in the affirmative form to introduce the searched information. For instance, the word *when* acting for introducing a modifier in a question is kept. The presence of *year* in a question asking for a date leads QALC to add the preposition *in*.

We associate each kind of question one or several patterns; these supplementary patterns often correspond to relaxation of constraints. A pattern is built according to syntactic characteristics of the questions: their focus, their main verb, their modifiers plus possibly relations introducing verb or object modifiers. In the simplest way, the affirmative form is built with all the question words, without the interrogative pronoun and the auxiliary, as for questions of the *WhatBe* type. For example, *When was Lyndon B. Johnson born?* leads to the query *Lyndon B. Johnson was born on*, by applying the pattern *[focus] [main verb] born on*. Google finds *Lyndon B. Johnson was born on August 27th, 1908* in the top 9 documents. In order to avoid a query that would be too restrictive, we submit Google the queries with quotes and the queries without quotes.

For evaluating our re-writing module, we searched for the patterns given as answers to the TREC11 questions in the 20 first documents retrieved by Google. We found that 372 questions were potentially solvable, corresponding to 74.4% of answers, 360 with more than one relevant document.

## 4 Fusion of several sources of information

For TREC11 evaluation, the participants had to provide a unique answer per question and the set of 500 answers had to be ordered according to the confidence score of the system in each of these answers. The evaluation metric, which is also the confidence weighted score, is the following:

$$\frac{1}{Q}\sum_{i=1}^{Q}\frac{Number\ of\ correct\ answers\ at\ i\ first\ ranks}{i}$$

Q being the overall number of questions.

Thus TREC11 evaluation took into account not only the rightness of the answers but also the confidence the system has in its answers. As it was said in section 2 (overview of QALC), we elaborated a strategy based on the comparison of the results of our system from two different sources of knowledge: AQUAINT collection and Web. The knowledge source explored by the Web search is obviously really much larger than AQUAINT collection search. Using such source brings our system a relevant way to confirm some of its answers and to reinforce its confidence score. However, among the answers provided by the Web search, some are not corresponding to any document of AQUAINT. So, it is to be noted that the Web answers also have to be located in AQUAINT collection; otherwise they are not retained as right answer in the TREC evaluation.

These two applications of QALC supply for each question a set of answers which are ordered according to the score they received during the answer extraction process. Hence, the role of the final selection is to choose a unique answer between these two ordered sets.

Before describing the algorithms we wrote for the final selection, we will describe the way QALC attributes a confidence score to each potential answer.

### *4.1 Answer weighting*

All the sentences provided by the document processing are examined in order to give them a weight that reflects both the possibility that the sentence contains the answer, and the possibility that the QALC system locates the answer within the sentence. The criteria that we used are closely linked with the basic information extracted from the question. The resulting sentence ranking should not miss obvious answers. Our aim is that the subsequent modules of answer extraction and final answer selection are able to raise a lower weighted answer to an upper rank thanks to added specific criteria. The criteria that we retained use the following features within the candidate sentences:

- question lemmas, weighted by their specificity degree [4] ,
- variants of question lemmas,
- exact words of the question,
- mutual closeness of the question words,
- presence of the expected named entity type.

[4] The specificity degree of a lemma depends on the inverse of its relative frequency computed on a large corpus.

First we compute a basic weight of the sentence based on the presence of question lemmas or variants of these lemmas (the two first criteria). The basic weight is relative. We subsequently add an additional weight to this basic weight for each additional criteria that is satisfied. Each additional criteria weight cannot be higher than about 10% of the basic weight. We obtained for each question an average of 543 ranked weighted sentences. About 71% of the 500 questions have a correct answer in those sentences, 84% of which are in the top 30 sentences.

During answer extraction this weight is still refined. If the expected answer type is a named entity, then selected answers are the words of the sentence that correspond to the expected type. In order to extract the answer, the system first computes additional weights taking into account:

- the precise or generic named entity type of the answer,
- the location of the potential answer with regard to the question words within the sentence,
- the redundancy of an answer in the top ten sentences.

When the expected answer type is not a named entity, we use extraction patterns. Each candidate sentence provided by the sentence selection module is analysed using the extraction pattern associated with the question type that has been determined by the question analysis. Extraction patterns are composed of a set of constraint rules on the candidate sentences. Rules are made up of syntactic patterns that are used to locate potential answers within the sentence, and of semantic relations that are used to validate answers. Potential answers are weighted according to the satisfied constraints. More detail can be found in (de Chalendar et al. 2002).

Finally after the extraction and weighting procedure, the five best weighted answers are retained for the final selection module.

### *4.2 Final selection algorithms*

For the selection we tested two algorithms that explore the whole sets of potential answers searching for common answers. For TREC11 evaluation the size of both sets was limited to five answers; but this could be easily modified. Table 1 contains an example of these sets corresponding to the question: *Who defeated the Spanish armada?*

The first algorithm examines each couple ($answer_i$, $answer_j$), $i$ being the position of an AQUAINT answer, $j$ being the position of a Web answer, its score being the best of both scores. When both answers of the couple are exactly equal, the algorithm attributes a bonus to the couple score, which is

Table 1
Answer set example

| AQUAINT answers | Web answers | Final score |
|---|---|---|
| 0) Queen Elizabeth (score 1205) | 0) Elizabeth I (score 1299) | |
| 1) England (score 1202) | 1) Elizabeth I (score 1297) | |
| 2) **Francis Drake** (score 982) | 2) Philip II (score 1282) | |
| 3) Spain (score 872) | 3) **Francis Drake (score 1252)** | **1852** |

calculated according to both positions: $i$ and $j$: $(11-(i+j))*100$. The answer that is finally returned belongs to the couple obtaining the best score. The additive bonus was chosen in order to push the confirmed answers before the unconfirmed ones.

Looking at Table 1, we see that the answer at rank two in AQUAINT and the answer at rank three in Web answers are the same. The received bonus is 600, and the answer *Francis Drake* is returned with its final score: 1852.

The second algorithm is more constrained and consists in increasing the score of the first answer of AQUAINT results, provided that this first answer is also present in the Web results. If the first answer is not found in the Web results, the other way round, the first answer of the Web results is searched in AQUAINT results. If the two searches fail the AQUAINT first answer is returned with its original score.

In both algorithm, the underlying idea is to compare results obtained from diverse sources of knowledge. In the case of the first algorithm, this comparison allows us to reinforce the score of answers belonging to both result sets, thus allowing a significant number of right answers to get at the first rank (see section 5). In the case of the second algorithm, the answer returned is in any case one of the first rank. However, on the one hand, this answer can be issued from the Web results; on the other hand, its score can be increased if, here again, this answer belongs to both result sets. Thus the confidence score, which has also a great importance in the evaluation, may be improved.

Table 2
Results of both final selection algorithms

| | Right answers | Confidence weighted score |
|---|---|---|
| Algorithm 1 | 165 | 0.587 |
| Algorithm 2 | 159 | 0.574 |

Table 2 gives the results of both algorithms applied on the set of TREC11 questions. Even if the results of the first algorithm are not far better, we think that this approach is more interesting and can be refined and improved. Indeed, the comparison could be carried out between three strategies (or more)

rather than two; it could be extended to more than only five answers and it could be more flexible by taking into account answers included one in the other, instead of exactly equal answers.

## 5 Results

Table 3 presents the results we obtained for AQUAINT and Web documents. In AQUAINT + Web *strict* column, we give the evaluation given by the NIST, it corresponds to a 9th position over the 35 participants. In AQUAINT + Web *lenient* column, we give the evaluation we computed thanks to the patterns also provided by NIST. Since this evaluation does not reject neither the unsupported answers nor the inexact ones, these results are quite better than our official results but we take them as a reference for our following comparisons.

Table 3
Results of AQUAINT, Web and AQUAINT+Web

| | AQUAINT | Web | AQUAINT + Web | |
|---|---|---|---|---|
| | | | lenient | strict |
| Number of right answers | 128 | 122 | 165 | 133 |
| Confidence weighted score | 0.402 | 0.436 | 0.587 | 0.497 |

We note that the score of AQUAINT + Web is increased of 46%, compared to the one in AQUAINT alone. This important increasing may be either due to answers found on the Web and not in the selected set of AQUAINT documents, or due to the ranking algorithm described above.

We first studied where the answers were coming from in the AQUAINT + Web results, and if they were common to both collections or not (Table 4). Among the 165 right answers, 106 were found in both of them, 42 in AQUAINT documents alone, 17 in Web alone. Then, we evaluated the benefit given only by the ranking process. To do so, we take away the 17 questions for which the right answers were only found by Web search, and Table 5 gives the results obtained on the 483 remaining questions. The score is still increased of 37%, even if this evaluation is more in favour of AQUAINT than AQUAINT + Web, because it takes away a bad answer of the first and a right one of the second.

We then examined the positions of the right answers to confirm the hypothesis of a good ranking of the right answers. Figure 2 illustrates this study. It is worth noticing the 46 right answers in the 50 first answers.

We finally proceeded to a TREC10 evaluation (5 answers were given for a

Table 4
Where do right answers come from ?

| | |
|---|---|
| AQUAINT | 25% |
| Web | 10% |
| AQUAINT + Web | 65% |

Table 5
Results on 483 common answers

| | Right answers | Score |
|---|---|---|
| AQUAINT | 128 | 0.414 |
| AQUAINT + Web | 148 | 0.568 |

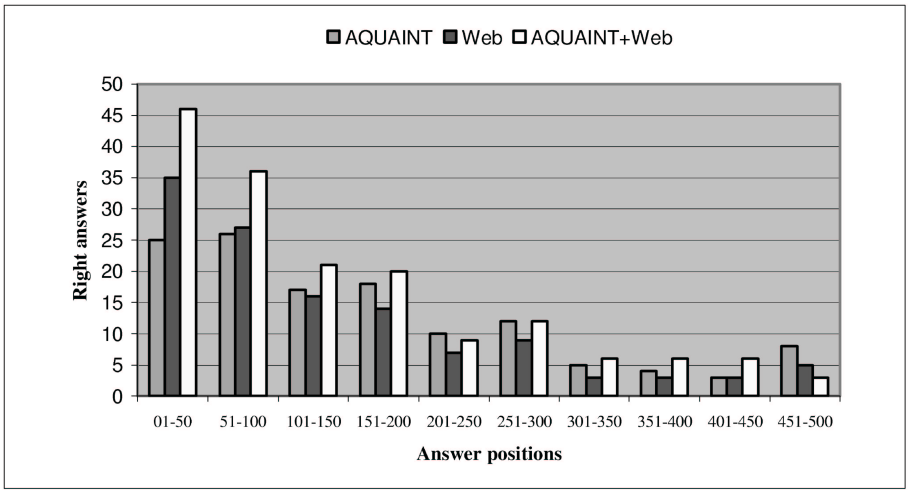


Fig. 2. Positions of the right answers

question and the answer set was not ordered) to determine how many right answers where in the ranks 2 to 5, both for AQUAINT and Web searches. The results are in Table 6. It illustrates the fact that it is important to look for answers which are ranked in the first 5, and not only in rank 1.

Table 6
Right answers in ranks 1 to 5

| | Rank 1-5 | % | Rank 1 | % | Rank 2-5 | % |
|---|---|---|---|---|---|---|
| AQUAINT | 177 | 100% | 128 | 72% | 49 | 28% |
| Web | 177 | 100% | 122 | 69% | 55 | 31% |

As it is important that the system knows if the answer it proposes is correct or not, (Chu-Carroll et al. 2002) proposed a score for measuring the ranking ability of a system. The ranking ability is evaluated as follows:

$$\textit{Ranking ability} = (cw - cwm) / (cwM - cwm)$$

*cw*: actual TREC score

*cwm*: precision (number of correct answers / number of questions)
*cwM*: TREC score with all correct answers ranked first

The ranking ability thus represents the part of the gain achievable by ranking over an average ranking. We find in (Chu-Carroll et al. 2002) an evaluation of the ranking ability of the top 15 TREC systems. In this evaluation, our best run obtained a ranking ability of 0.66, which puts our system at the first rank of these top systems. Before the final answer selection the ranking ability of the AQUAINT run was 0.42.

## 6 Related works

(Magnini et al. 2002a) make use of the Web for answer validation. They interrogate the Web with a query elaborated from the question and the answer. Queries are made of boolean operators between question words plus a proximity operator; they do not search for an exact match. Answer validity is evaluated according to the number of retrieved documents. With this approach, they find that the Web brings an improvement of 28% above their baseline (the right answers in the top ten documents provided by NIST). In TREC11, (Magnini et al. 2002b) apply the same process and 40 answers per question are searched for validation. Their ranking strategy is based on the coefficient of validity and the reliability of the type of expected answer. (Clarke et al. 2001) select 200 documents and 40 passages from the Web. The Web is only used to increase the redundancy factor for candidates, and not to add answers. Its contribution to the results is 25% to 30% .

For question rewriting, (Brill et al. 2001) keep the question words in the same order and put the main verb in all positions. They work with string matching, as in QALC. (Hermjacob et al. 2002) generate a lot of variants corresponding to syntactic and semantic paraphrases. Web boolean queries are then generated from these paraphrases and it leads to about 3 paraphrases per question.

## 7 Conclusion

Evaluating the reliability of an answer is not an easy task when all the successive processes produce approximate results. A way of deciding on the validity of an answer is to confront the results of the same processes on another source of knowledge, the Web. According to its large size and its redundancy, we opted for a strategy that pushed common answers at the top, provided they were in the 5 best answers. This strategy was profitable as our system obtained better results at TREC than systems that found more answers.

## References


[Aït-Mokhtar and Chanod 1997] S. Aït-Mokhtar and J.-P. Chanod, 1997, IFSP, Incremental finite-state parsing *Proceedings of Applied Natural Language* Washington, DC

[Brill et al. 2001] E. Brill, J. Lin, M. Banko, S. Dumais and A. Ng, 2001. Data-Intensive Question Answering. *TREC 10 Notebook, Gaithersburg, USA*

[de Chalendar et al. 2002] G. de Chalendar, T. Dalmas, F. Elkateb-Gara, O. Ferret, B. Grau, M. Hurault-Plantet, G. Illouz, L. Monceaux, I. Robba, A. Vilnat, 2002, The Question Answering System QALC at LIMSI, Experiments in Using Web and WordNet. *Trec 11, Notebook, Gaithersburg, USA* pp. 457-467

[Chu-Carroll et al. 2002] J. Chu-Carroll, John Prager, Christopher Welty, Krzysztof Czuba and David Ferruci. 2002. A Multi-Strategy and multi-source Approach to Question Answering. *TREC 11 Notebook, Gaithersburg, USA* pp. 124-133

[Clarke et al. 2001] C. L. Clarke, G. V. Cormack, T. R. Lynam, C. M. Li and G. L. McLearn, 2001, Web Reinforced Question Answering (MultiText Experiments for Trec 2001), *TREC 10 Notebook, Gaithersburg, USA*

[Fellbaum 1998] C. Fellbaum. 1998. WordNet: An Electronic Lexical Database. Cambridge, MA: MIT Press

[Hermjacob et al. 2002] U. Hermjakob, A. Echihabi and D. Marcu. 2002, Natural Language Based Reformulation Resource and Web Exploitation for Question Answering, *TREC 11 Notebook, Gaithersburg, USA*

[Magnini et al. 2002a] B. Magnini, M. Negri, R. Prevete and H. Tanev. 2002a. Is It the Right Answer? Exploiting Web redundancy for Answer Validation, *Proceedings of the 40 th ACL* pp. 425-432

[Magnini et al. 2002b] B. Magnini, M. Negri, R. Prevete and H. Tanev, 2002b, Mining Knowledge from Repeated Co-occurrences: DIOGENE at TREC-2002, *TREC 11 Notebook, Gaithersburg, USA*

[Moldovan et al. 2002] D. Moldovan, S. Harabagiu, R. Girju, P. Morarescu, F. Lacatusu, A. Novischi, A. Badalescu and O. Bolohan, 2002, LCC Tools for Question Answering, *TREC 11 Notebook, Gaithersburg, USA*