How NLP Can Improve Question Answering

Olivier Ferret, Brigitte Grau, Martine Hurault-Plantet, Gabriel Illouz,
Christian Jacquemin, Laura Monceaux, Isabelle Robba, Anne Vilnat

How NLP can improve Question Answering

O. Ferret, B. Grau, M. Hurault-Plantet, G. Illouz, C. Jacquemin, L. Monceaux, I. Robba, A. Vilnat

LIMSI-CNRS
BP 133
91403 Orsay Cedex

[ferret, grau, mhp, illouz, jacquemin, monceaux, robba, vilnat]@limsi.fr

Contact : grau@limsi.fr

How NLP can improve Question Answering

Abstract

Answering open-domain factual questions requires Natural Language processing for refining document selection and answer identification. With our system QALC, we have participated to the Question Answering track of the TREC8, TREC9 and TREC10 evaluations. QALC performs an analysis of documents relying on multi-word term search and their linguistic variation both to minimize the number of documents selected and to provide additional clues when comparing question and sentence representations. This comparison process also makes use of the results of a syntactic parsing of the questions and Named Entity recognition functionalities. Answer extraction relies on the application of syntactic patterns chosen according to the kind of information that is sought for, and categorized depending on the syntactic form of the question. These patterns allow QALC to handle nicely linguistic variations at the answer level.

Keywords

Question answering system, terminological variant, extraction pattern, linguistic variation, information retrieval

1 Introduction

In 1999, the first Question Answering task in TREC 8 (Text REtrieval Conference), revealed an increasing need for more sophisticated search engines able to retrieve the specific piece of information that could be considered as the best possible answer to the user question. Such systems must go beyond documents selection, by extracting relevant part of them. They should either provide the answer if the question is factual, or yield a summary if the question is thematic (i.e. Why there was a war in the Gulf?).

The problem intersects two domains: Information Retrieval (IR) and Natural Language Processing (NLP). IR is improved by integrating NLP functionalities at a large scale, *i.e.* independently of the domain, and thus necessarily having a large linguistic coverage. This integration allows the selection of the relevant passages by means of linguistics features at the syntactic or even semantic level.

In this paper, we present how the NLP processing incorporated in QALC improves question answering. In TREC8, QALC had to propose for each of the 200 questions, 5 ranked answers to be found in a 1.5 GigaOctets (Go). The next evaluation, TREC9, changed its scale, with 700 questions and 3 Go of documents. The documents came from American newspapers articles, such as *Wall Street Journal*, and Financial *Times*, etc. The TREC10 evaluation required to give only a short answer (limited to 50 characters).

QALC relies on NLP modules dedicated to question type analysis, named entities recognition, simple and complex terms extraction from the question (to retrieve them, possibly as a variant, in the documents), and finally answer extraction realized by matching syntactic patterns which correspond to possible linguistic formulations of the answer. We focus in this article on the gain brought by taking into account linguistic variation in documents post-selection and in matching possible answers with a question. More precisely, from a set of documents selected by a search engine, QALC reduces the number of documents to be analyzed by NLP modules to reduce processing time. This second selection privileges documents containing complex terms or their variants against documents containing words of these terms spread in the text. Linguistic analysis can also improve answer selection, since the basic units of evaluation chosen are sentence and short answer. The choice of a short answer does not suppress the need to provide the context (the surrounding text) to the users, to let them judge the correctness of an answer without having to return to the original document. In that respect, the sentence level seems to be a more interesting solution than a predetermined maximum-size window around the answer, which does not constitute a semantic unit.

In this article, we present our approach in section 2, and the general architecture of QALC is exposed in section 3. The modules are then detailed in sections 4 to 9. Section 10 shows the results of QALC at TREC evaluations, these results lead to a discussion on the need for even

more ambitious, but still realistic, NLP modules. Finally, we describe more precisely existing approaches addressing this issue, before concluding on future work. As most of the questions answering systems participating to TREC have similar approaches, we decided to detail our solution, to be able to present fine-grain differences in contrast to other techniques.

2 What does question answering mean?

In the question answering paradigm (without domain-specific limitation), it is not possible to develop a only-NLP approach, as it existed in the 70s. The problem of automatic question answering is not recent. Since the first studies in language understanding, the problem was addressed mainly in the context of stories understanding. The most representative system named QUALM was developed by Lehnert (Lehnert 1977, 1979). It analyzed small stories on very specific topic (travelling by buses, going to restaurant, etc.), memorized a conceptual representation, and then answered questions by consulting this memory and by reasoning with its general knowledge base. The typology of question relied on 13 categories. It has strongly inspired some recent systems. To each category was associated a search strategy of the answer in the knowledge base. For example, a question on the cause of an event leaded to search for knowledge having a causal responsibility type. Zock and Mitkov (Zock et al 91) showed that some categories should be refined to better define the answer type expected, and thus the associated search strategy. For example, the category "*concept completion*" groups all the WH[1] questions, without distinctions between the concept types to be found. Lehnert participated to the development of another system, *Boris* (Dyer, 1983), which differs in the type of knowledge used. It introduced more pragmatic knowledge. These systems (cf. (Zock et al 91) for a more complete list) rely on much elaborated generic knowledge allowing to describe prototypic situations and to interpret characters behaviors.
This kind of approach exists as well in a psychological model of question answering, QUEST (Graesser et al 94), tested with experimental methods and which considers a large variety of questions. Its authors define 4 factors for a question answering model:

- **Question categorization**. A classification in 18 categories is proposed. it differentiates questions leading to short answers (one word or a group of words, for example WH question where concept clarification is expected) from the one leading to long answers (one or more sentence, for example, "Why" question where the answer explains a cause);
- **Identification of required information source to answer.** Use of knowledge on the episodes and generic knowledge are found there;
- **A convergence mechanism** allowing to compute a subset of known propositions, representing facts and events;
- **Answer formulation** according to pragmatics aspects, such as goals and common culture with the participants.

This kind of approach cannot be completely applied when realizing automatic systems without being domain-specific otherwise the definition and formalization of the pragmatic knowledge required are impossible. Nevertheless, a purely NLP approach can be realized for a limited domain of application, as done in the EXTRANS system (Molla et al 2000). It answers to questions about UNIX commands. It relies mainly on a syntactico-semantic analysis of the UNIX manual combined with logical inferences. Since the domain is closed, a precise knowledge base can be build to represent semantic knowledge and lexicon where

[1] WH questions are interrogative pronouns (Who, Whom, Where, What, etc.)

ambiguities are limited. It is worth noticing that this system proposes a degraded mode, based on keywords, when the full approach failed.
The shift from questions about stories to factual and encyclopedic ones about reported events leads to a new way to address the issue. In such a context, the answer searched could be explicitly present in a large set of text, provided it is big enough.
The textual database replaces the knowledge base in the previous works. An information-retrieval based system exploiting only statistics knowledge on the corpus leads to the elaboration of a system able to answer less than half of the questions (Morton 1999). TREC evaluations show that the more NLP there is in the systems, the better they perform.
For all TREC systems, passage selection containing the answer (for 250 characters in this assertion) is based on the relative proximity of the excerpt with the question. This proximity is based not only on the common words, but it takes into account the type of the expected answer to characterize the text sought, and the linguistic variation between the question formulation and the passage extracted. Since first TREC, the best systems were the one having a module analyzing the question in order to identify an answer type correlated to the output of a named entity recognition module. For the second evaluation, most of the systems introduced the use of WordNet (Fellbaum, 1998), to take into account synonymic and hyperonimic variations between question and answer formulations. In these systems, information retrieval methods are in use to select potentially relevant passages in a large corpus. After this first selection step, NLP is applied to further refine the selection. Such approaches remain possible as long as we are not addressing a domain-specific problem.
Accordingly, the NL processes used in QALC allow for typing the expected answer during question analysis, and for using named entity recognition with a tag set corresponding to these types. The originality of our approaches lies in management of the terminological variation between question and answer terms, and in the construction of the extraction patterns done according to question categories. These patterns are based on the notion of the question focus. This notion has been introduced by Wendy Lehnert (79) and then broadly refined and redefined. For Lehnert, the question focus is the concept which covers the information expectations expressed by the question. For us, it is the word (or nominal group) of the question representing the unit on which information is wanted, and which is generally found in the sentence containing the answer.

3 QALC presentation

In Figure 1, we present the QALC architecture. The question analysis is based on the results of a syntactic robust parse. It deduces information that is useful for selecting candidate sentences and extracting answers from them (see 4.1. section). For example, from the question, "Who is the 17th president of the USA", the module predicts that the answer will be a person name (the answer type), determines the syntactic category of the question, "WhoBeNP", and its focus, "president". The question analysis also extracts terms (see section 4.2), single or compound with several lemmas that will be searched in the selected documents, either as they appear in the questions, or with linguistic variations. In the preceding example, the terms are "17th president, president and USA". The documents are first selected by a search engine (see section 5 for a comparison between different engines), and afterwards indexed anew with the question terms or term variants, allowing QALC to operate a more precise selection among them (see section 6). Term recognition is realized by FASTR (Jacquemin, 1999).

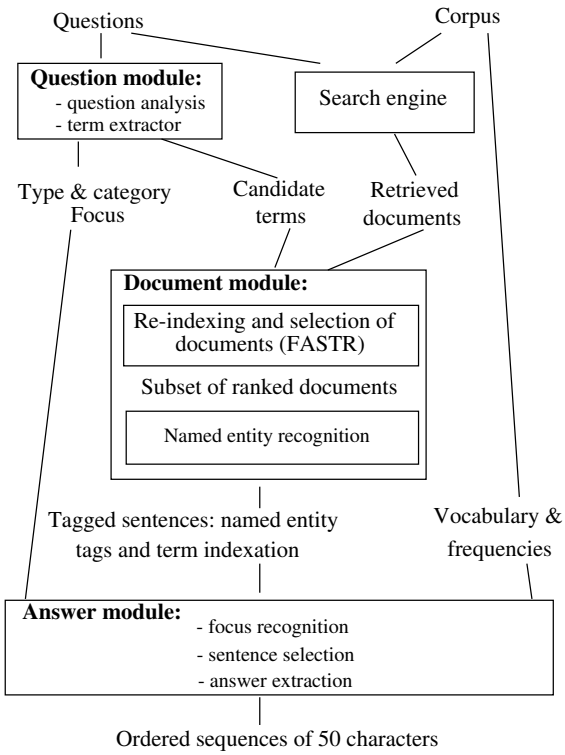

Figure 1 QALC architecture

Selecting only a minimal number of documents finds all its justification in the subsequent processes. Named entity recognition (see section 7) and question pairing with sentences (see section 8) are applied on the remaining documents. Lastly, answers are extracted from selected sentences, relying on two strategies, depending on whether the answer has to be a named entity or not (see section 9).

4 Question analysis

Question analysis is conducted with the two following objectives: the first is to extract some characteristics which will likely be used in the answer extraction module; the second is to extract the terms which will re-index the selected documents in order to retain only a subset of them and to supply further evidences during the final matching. The extracted characteristics are the expected type of the answer, the focus and the question category. Moreover syntactic and semantic knowledge are used to find out these characteristics. A syntactic robust parser (Ait-Mokhtar et al. 1997) analyses the question set and supplies syntactic information. This parser provides not only the syntactical segmentation of the question but also a set of syntactical relations between these segments. The focus, the expected type of the answer and the question category are obtained thanks to the application of rules that have been written using these syntactic representations. The semantic knowledge, which is obtained from Wordnet (Fellbaum 1998), allows us to complete the named entity lexicons in order to improve the choice of the expected answer type.

4.1 Indices for finding precise answers

4.1.1 Answer type

In a first step, the analysis module determines whether the expected type of the answer corresponds to one or several named entities listed by order of importance. The detected

named entities belong to a hierarchy, which is organized in semantic classes. Figure 2 presents the semantic hierarchy; the tags used in our module are the leaves of this hierarchy, to which we add the *Proper_Noun* and the *Number* tags. These tags are similar to the types defined in MUC (Grisham et al. 1995) for the evaluation task concerning named entities extraction. The presence of some particular features allows the question analysis module to attribute one or several tags to each question. The following examples, in which the relevant features have been underlined, illustrate this association *question - tags*.

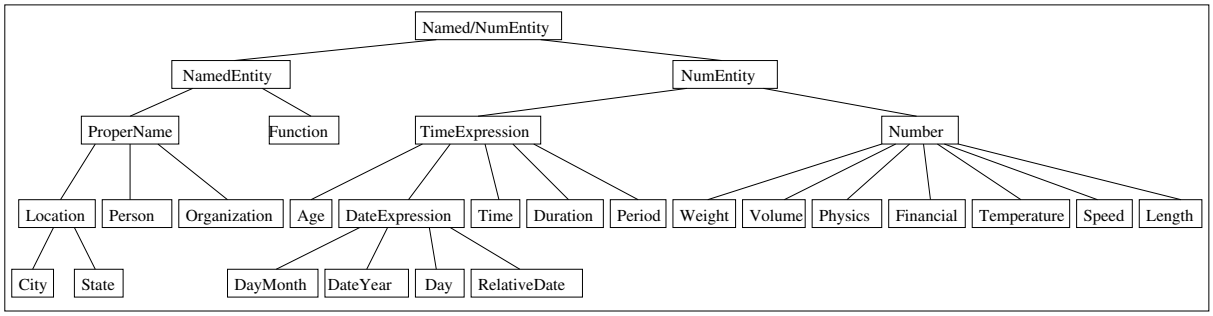

Figure 2 Hierarchy of answer types and semantic categories

PERSONNE: Who was the first President of the USA?
ORGANIZATION: What laboratory discovered the AIDS virus?
PERSON, ORGANIZATION: Who developed the Macintosh computer?
LOCATION: What is the longest river in Asia?
What is the name of the highest mountain in the world?
CITY, LOCATION: Where is Taj Mahal?
PERIOD: During which period did the dinosaurs vanish?

The recognition rules for the expected type of the answer are essentially using the form of the interrogative pronoun and the noun this pronoun is referring to. The rules look whether this noun belongs to one of the lists constituted for each named entity. During TREC10 evaluation conference, our module found about 90.5 % of right expected answer types.
If the expected type of the answer does not correspond to any named entity, QALC tries to determine a more general type corresponding to the noun to which the interrogative pronoun is referring. For TREC10, our module found 87 % of right expected semantic types.

Example: Question: What metal has the highest melting point?
General expected type of the answer: *metal*

4.1.2 Question focus

The question focus corresponds to a noun of the question that ideally should also be present in the answer sentence. Knowing the question focus provides a criterion for selecting sentences and extracting just the answer. We make the hypothesis that answer sentences will bring precisions about this focus by making it explicit. For each question, we determine not only the focus but also its modifiers (adjective, noun complement...), which will play also an important role in the search for the answer. The rules, which determine the focus, depend essentially of the syntactic form of the question, and very often the focus corresponds to the subject itself. In the question corpus of the TREC 10 conference, our question module found 85 % of right focuses.

Example: Question: Who was the first governor of Alaska?
Focus: governor
Noun Group containing the focus: the first governor of Alaska
Focus modifiers: ADJ first, COMPLEMENT Alaska

4.1.3 Question category

The detection of the question category has an important role; indeed it allows the system to differentiate the syntactic patterns that will be applied to the candidate answers considering that the answer will be a partial reformulation of the question with the focus as pivot. The question category corresponds to the syntactic form of the question, more or less detailed. For example:

Example: Question: What do bats eat?
Question category: What-do-GN-VB

Question: When was Rosa Park born?
Question category: When-do-NP-born

The study of the question set of TREC8 and TREC9 conferences along with the answer sentences conducted us to find 82 question syntactic forms.

4.2 Term extraction

In order to work out an automatic acquisition of the question terms, we use a simple filtering technique thanks to syntactic categories patterns. Indeed a statistical filtering is not possible here because the question set does not contain enough questions to allow for such a process. First of all, we use the *TreeTagger* (Schmid, 1999) to tag each question. Then, we use some patterns of syntactic categories to extract terms from the tagged questions. These patterns are not different from those of Justeson and Katz (Justeson et al., 1995) except that they do not take into account the post-posed prepositional phrases. The acquisition patterns are the following:

(((((JJ | NN | NP | VBG)) ? (JJ | NN | NP | VBG) (NP | NN))) | (VBD) | (NN) | (NP) | (CD))

The longest string is acquired first and the sub-strings cannot be acquired if they do not begin with the same term as in the superstring. For example, in the following sentence: *name of the Hawaii state flower* (NN PREP DET NP NN NN), three terms are acquired: *Hawaii state flower, state flower and flower.*

The retained acquisition mode amounts to considering only the sub-structures where the noun modifiers are attached to the leftmost constituent (the closest one). For example, from *Hawaii state flower* we obtain *state flower* and *flower* thanks to the extraction of the sub-constituents in the structure [*Hawaii [state [flower]]].*

5 Comparison of search engines

The first module dedicated to document processing is the retrieval of documents answering to the query elaborated from the questions. We tested three search engines on the 200 questions given to the participants of the QA track at TREC8. The first engine is Zprise, developed by the NIST. It is based on a vectorial representation of query and documents. We used it with the following parameters: weighting function bm25idf of Okapi (Robertson et al., 1999),

stemming by the Porter algorithm, no relevance feedback. The second engine is Indexal, relying on a pseudo-boolean research, that was given us by Bertin Technologies, a French company. Indexal results from their collaboration with the Laboratoire d'Informatique d'Avignon (LIA). Indexal is usually employed for searching in small database of documents with an indexation of the paragraphs. We applied the Indexal capacity for stemming and its particularity that consists of favoring affinities between words (de Loupy et al., 1998). This principle leads Indexal to consider a query as a set of words having to be contained in a fixed text window. The last engine is used by ATT for the AdHoc task of TREC. In this latter case, only the results returned by the engine were in our possession, given us by the TREC organizers. They consisted in a list of 1000 ranked documents with their weight, for each question.
The three search engines give back a list of documents, ordered by their decreasing relevance. Our tests were dedicated to two purposes: first determining the optimal number of documents we had to keep. Too many documents lead to a too extensive processing time, too few documents decrease the possibility of finding an answer. Secondly, we obviously were interested in knowing which search engine gave the best results, i.e. a maximum number of documents containing answers.

5.1 Document selection threshold

We first realized tests with Zprise in order to determine how many documents represented the best compromise. We evaluated the 50, 100, 200 and 500 first documents by comparing their identifier with those contained in the list given by NIST, list of documents found by all participants as containing the answer. Table 1 shows the results of this evaluation.

Selection Threshold	**Questions *with* relevant documents**	**Questions *with no* relevant documents**
50	181	19
100	184	16
200	193	7
500	194	6

Table 1 Zprise performances according to the number of returned documents

Table 1 shows that there is a tendency for the improvement of Zprise performances to be stationary after 200 documents. Thus, we opted for this threshold, which offers us the best compromise between a minimum lost answers and a reasonable processing time for a question.

5.2 Selection of the best search engine

We compared the results of the three engines for the 200 best documents they return. A query was built for each question by keeping all their meaningful words, selected according to their POS tag (noun, verb, adjective, adverb) and a list of stop words. Table 2 shows the performances of each engine.

Search Engine	Indexal	Zprise	ATT
Number of questions *with* relevant documents retrieved	182	193	194
Number of questions *without* relevant documents retrieved	18	7	6
Total number of relevant documents that were retrieved	814	931	1021

Table 2 Comparison of the 3 search engines on 200 documents

ATT search engine obtained the best results, according to the 3 criteria we kept: the maximum number of questions for which at least one correct document was returned, the minimal number of questions without correct documents, and the greatest number of retrieved documents, all questions being merged.

6 Re-indexing and selection of documents

The selection of the most relevant documents in relation to a question is based on a specific indexing using *FASTR*, a NLP tool for term and variant recognition. The resulting indexes are the single and multi-word terms extracted from the questions (see Section 4.2) that occur in documents under their initial form or under a variant form. A score is then computed from these indexes for each document returned by the search engine used by QALC and is finally exploited for selecting the documents in which an answer is likely to be found. By strongly reducing the amount of text to process, this selection enables the answer extraction module of QALC to make use of elaborate NLP methods that are also very time-consuming.

6.1 Indexing by FASTR

The automatic indexing of documents is performed by *FASTR* (Jacquemin, 1999), a transformational shallow parser for the recognition of term occurrences and variants. Terms are transformed into grammar rules and the single words building these terms are extracted and linked to their morphological and semantic families.

The *morphological family* of a single word *w* is the set *M*(*w*) of terms in the CELEX database (CELEX, 1998) which have the same root morpheme as *w*. For instance, the morphological family of the noun *maker* is made of the nouns *maker*, *make* and *remake*, and the verbs *to make* and *to remake*.

The *semantic family* of a single word *w* is the union *S*(*w*) of the *synsets* of WordNet 1.6 (Fellbaum, 1998) to which *w* belongs. A synset is a set of words that are synonymous for at least one of their meanings. Thus, the semantic family of a word *w* is the set of the words *w'* such that *w'* is considered as a synonym of one of the meanings of *w*. Taking the synonyms for all senses of *w* results from the fact that no sense tagging is achieved for documents. The semantic family of *maker*, obtained from WordNet 1.6, is composed of three nouns: *maker*, *manufacturer*, *shaper* and the semantic family of *car* is *car*, *auto*, *automobile*, *machine*, *motorcar*.

Variant patterns that rely on morphological and semantic families are generated through metarules. They are used to extract terms and variants from the document sentences in the TREC corpus. For instance, the following pattern, named NtoSemArg, extracts the occurrence *making many automobiles* as a variant of the term *car maker*:

VM('*maker*') RP? PREP? (ART (NN|NP)? PREP)? ART? (JJ | NN | NP | VBD | VBG)$^{0\text{-}3}$ NS('*car*')

where RP are particles, PREP prepositions, ART articles, and VBD, VBG verbs. VM('*maker*') is any verb in the morphological family of the noun *maker* and NS('*car*') is any noun in the semantic family of *car*.

Relying on the above morphological and semantic families, *auto maker*, *auto parts maker*, *car manufacturer*, *make autos*, and *making many automobiles* are extracted as correct variants of the original term *car maker* through the set of metarules used for the QA-track experiment. Unfortunately, some incorrect variants are extracted as well, such as *make those cuts in auto* produced by the preceding metarule.

6.2 Document Selection

The output of NLP-based indexing is a list of term occurrences composed of a document identifier *d*, a term identifier — a pair *t(q,i)* composed of a question number *q* and a unique index *i* —, a text sequence, and a variation identifier *v* (a metarule). For instance, the following index:

LA092690-0038 t(131,1) making many automobiles NtoVSemArg

means that the occurrence *making many automobiles* from document *d*=LA092690-0038 is obtained as a variant of term *i*=1 in question *q*=131 (*car maker*) through the variation NtoVSemArg given in Section 6.1.

Each document *d* selected for a question *q* by the search engine is associated with a weight. The weighting scheme relies on a measure of quality of the different families of variations described by Jacquemin (1999). Thus, the weight *w(v)* of an occurrence of a term is all the more high that it results from a small variation of the initial term: non-variant occurrences are weighted 3.0, morphological and morpho-syntactic variants are weighted 2.0, and semantic and morpho-syntactico-semantic variants are weighted 1.0.

Since proper names are more reliable indices than common names, especially in a corpus such as the TREC corpus that is mainly composed of newspaper articles, each term *t*(*q,i*) receives a weight *P*(*t*(*q,i*)) between 0 and 1.0 corresponding to its proportion of proper names. For instance, *president Cleveland's wife* is weighted 1/3=0.33. Since another factor of reliability is the length of terms, a factor |*t*(*q,i*)| in the weighting formula denotes the number of words in term *t*(*q,i*). The weight $W_q(d)$ of a query *q* in a document *d* is given by the following formula [1]. The products of the weightings of each term extracted by the indexer are summed over the indices *I*(*d*) extracted from document *d* and normalized according to the number of terms |*T*(*q*)| in query *q*.

$$W_q(d) = \sum_{(t(q,i),v)\in I(d)} \frac{w(v)\times(1+2P(t(q,i)))\times|t(q,i)|}{|T(q)|} \qquad [1]$$

For each query *q*, the 100 best ranked documents retrieved by the search engine are processed. Our studies show that 100 is a minimum number such that almost all the relevant documents are kept. Mainly two types of weighting curves are observed for the retrieved documents: curves with a plateau and a sharp slope at a given threshold (Figure 3.a) and curves with a slightly decreasing weight (Figure 3.b). Questions in these figures come from the TREC8 data.

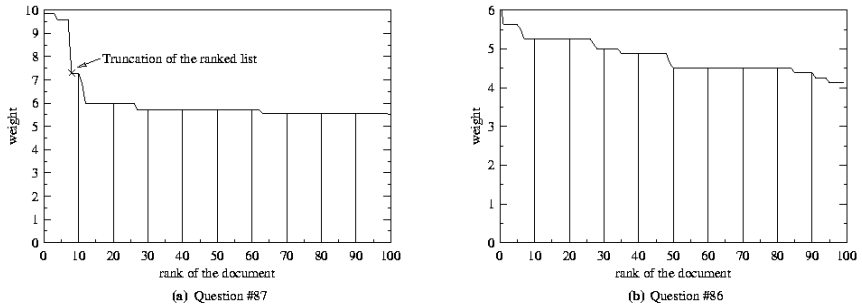

Figure 3 Two types of weighting curve for the documents retrieved by a search engine

The edge of a plateau such as the one of Figure 3.a is detected by examining simultaneously the relative decrease of the slope with respect to the preceding one, and the relative decrease of the value with respect to the preceding one. The following algorithm is used for calculating the cut-off threshold i_0 associated with the weighting scheme W of a given query q:

$$\text{If } \frac{\text{Wq(d2)}}{\text{Wq(d1)}} \leq 0.5 \text{ then } i_0 = 2$$

$$\text{else } i_0 = \min\left\{ i \in \{3 \ldots 100\} \middle| \left(\frac{Wq(di) - Wq(di-1)}{Wq(di-1) - Wq(di-2)} \right) \geq 2 \wedge \frac{Wq(di)}{Wq(di-1)} \leq 0.8 \right\} \cup \{100\} \qquad [2]$$

This decision function selects a restricted number of documents when a significant shift (when its slope is below 0.5 for the first documents and for the next ones, when its second-rate variation exceeds 2 and its slope is below 0.8) occurs in the curve of documents' weights or arbitrarily keeps the first 100 documents according to their ranking.

Through this method, the threshold i_0 is 8 for question #87 (*Who followed Willy Brandt as chancellor of the Federal Republic of Germany?*, Figure 3.a) and 100 for question #86 (*Who won two gold medals in skiing in the Olympic Games in Calgary?*, Figure 3.b). As indicated by Figure 3.a, there is an important difference of weight between documents 8 and 9. The weight of document 8 is 9.57 while the weight of document 9 is 7.29 because the term Federal Republic only exists in document 8. This term has a higher weight because it is composed of two proper names.

Finally, the system retains the i_0 best-ranked documents with a minimum number set to 20 and a maximum number set to 100.

6.3 Evaluation

In order to evaluate the efficiency of the selection process, we performed several measures. We applied our system on the material given for the TREC8 evaluation, one time with the selection process, and another time without it. At each time, 200 documents were returned by the search engine for each of the 200 questions. When selection was applied, at most 100 documents were selected and subsequently processed by the matching module; otherwise the 200 documents were processed. The system scored 0.463 in the first case, and 0.452 in the second case. These results show that the score increases when processing less documents mainly because the ration of relevant document over non-relevant ones is higher, thus reducing the residual noise in the answers.

Number of documents selected by ranking	100	<<100
Distribution among the questions	342 (50%)	340 (50%)
Number of correct answers	175 (51%)	200 (59%)
Number of correct answer at rank 1	88 (50%)	128 (64%)

Table 3 Evaluation of the ranking process

The benefit from performing such a selection is also illustrated by the results given in Table 3, computed on the TREC9 results. We see that the selection process discards a lot of documents for 50% of the questions (340 questions are processed from less than 100 documents). QALC finds the correct answer more often and in a better position for these 340 questions than for the 342 remaining ones. The average number of documents selected, when there are less than 100, is 37. These results are particularly interesting when applying such time-consuming processes as named entity recognition or question/sentence matching. Document selection will also enable us to apply syntactic and semantic sentence analysis later on.

7 Named entity recognition

Named entities are supplementary indices when searching for a sentence likely to contain an answer, and are really fundamental when only extracting the word phrase that corresponds to the answer. Names entities are tagged, and their type is one of the labels presented Figure 2, corresponding to a person name, an organization name, a place or a value. Our types are to the same as those of the MUC evaluations (Grishman *et al.*, 1995) and are recognized by means of:

- lexicons look-up to find morpho-syntactic and semantic features associated with simple words, and the application of rules that exploit these features;
- named-entity lists.

Our resources are CELEX (Celex, 1998), a database of 160,595 word forms with their lemma and their POS tag, a list of 8070 first names (6763 coming from the CLR archive (CLR, 1998)), a list of 211,587 family names (also from the CLR archive), a list of 22,095 companies coming from the Wall Street Research Network and 649 organization names acquired from the Web (Jacquemin et al., 2000). For place names, we use two lists, one for towns (7813) and another for states (1144) elaborated from CLR. For values, we manually built unit lists, for physics and monetary values.

7.1 Numeric entities

Numeric entities category group all the value and time expression, even if they are not expressed by numbers. The recognition is made in three steps:

- ordinal and cardinal number recognition, either expressed with words or numbers;
- complex expression recognition by applying rules. These expressions are a number plus a unit that gives them a type, such as length, monetary value, etc.
- recognition of time expressions from numbers that does not belong to complex expressions

Numbers that do not participate to a specialized named entity are marked by the tag *Number*.

7.2 Organizations, persons and places

When no word belongs to one of the lists, QALC tries to apply rules dedicated to each type. For organizations, QALC look for NP expressions beginning with a specific modifier such as Federal, Democratic, etc. or containing a headword such as Academy, Administration, Association, etc. Person names have to be composed of at least two words with specific

constraints. Constraints can be typographic criteria as a majuscule at the beginning of a word that follows a first name, or specific introducers as for titles (Mr., President, Ayatollah, etc.) before a word with a majuscule at its beginning. An endogenous learning mechanism is applied in order to recognize all the occurrences of a person name that appears alone in a document that first introduced her with an expression such as <*first name family name*>.

8 Question/Sentence Pairing

The Question/Sentence pairing module selects for each question submitted to the QALC system a small list of N_a ranked sentences[2] extracted from the documents that were kept after the filtering step (see Section 6.2) and in which the answer to the question is likely to be present. This selection relies on the features extracted by the linguistic modules presented in the previous sections from both the questions and the documents.
We present here two versions of this module. The first one (see Section 8.2) was used for the 250 characters task[3] of the TREC8 and TREC9 evaluations. The second one (see Section 8.3) was used for the 50 characters[4] task of the TREC10 evaluation. Theses two modules are globally based on the same principles (see next section) but some of the linguistic features they exploit are different and they do not combine them in the same way.

8.1 Principles of the pairing module

The pairing module takes the sentence as the basic unit for answering a question. This choice is the middle course between the necessity to provide a short answer and the necessity to give a context to the user to let him judge if an answer is valid without showing him the overall document. Giving a context to the user is necessary for ergonomic reasons but also because the error rate of question answering systems is still high when the size of the answer is small, even if their results in TREC evaluations are encouraging. Hence, the possibility for a user to judge an answer is important.
The overall principle of the selection process is the following. The pairing module scans the selected documents for a question sentence after sentence and constantly keeps in a buffer the N[5] sentences that are the best candidates as an answer to the question. Each new sentence is compared to the elements of this ranked list according to their similarity with the question: if the new sentence is more similar to the question than a sentence of the list, it is inserted in the buffer and the last sentence of it is removed; otherwise, the new sentence is discarded.
The similarity between a sentence and a question is based on the linguistic features they share. We considered three kinds of features in our two pairing modules:
— terms;
— named entities;
— focus.
The pairing module for long answers (250 characters task) only relied on terms and named entities while the module for short answers (50 characters task) used focus on top of that. A similarity score is computed for each kind of linguistic features and shows how close from the question the sentence is from the viewpoint of that kind of features. Finally, a combination function gathers these scores for setting a global similarity between the sentence and the question.

[2] N_a was equal to five for the evaluations TREC8, TREC9 and TREC10.
[3] The length of an answer was limited to 250 characters.
[4] The length of an answer was limited to 50 characters.
[5] N is at least equal to the final number of answers.

8.2 A pairing module for selecting a sentence as an answer

Globally, this module selects sentences that contain a named entity fitting the expected kind of answer for the considered question and a significant percentage of its mono and multi-terms, with or without morphological, syntactic or semantic variations.

8.2.1 Scores of features

From the named entity viewpoint, the score of a sentence is a binary one: it is equal to 1.0 if the sentence contains a named entity whose type corresponds to one of the types determined by the question analysis module; otherwise, it is equal to 0.0.

The sentence score for terms is made of two sub-scores. The first sub-score is tied to the mono-terms of the question that are present in the sentence without any terminological variation. The second sub-score concerns the multi-terms of the question found in the sentence with no variation or as a variant and the mono-terms of the question found in the sentence as a variant.

More precisely, the mono-terms taken into account in the first term score are the adjectives (including comparative and superlative ones), verbs, nouns and proper nouns, adverbs, acronyms and numbers of the question under their lemmatized form. The lemmatization and the morpho-syntactic tagging are achieved by the *TreeTagger* tool (Schmid, 1999). Each occurrence of these terms in the sentence is given a weight according to the *idf* (inverse document frequency) policy. Finally, the score of the sentence for this kind of terms is equal to the sum of their weights.

The terms considered for computing the second term score are those found by *FASTR* (see Section 6.1). We take as their weight for this score the weight computed by the document selection module (see Section 6.2). When two of these terms overlap each other (one term is a subpart of another one), we only keep the term with the highest weight as the representative of the underlying concept. The second term score of the sentence is given by the sum of all the selected *FASTR* terms.

8.2.2 Integration of feature scores

In this first pairing module, the scores that characterize the similarity of a sentence and a question from the viewpoint of a kind of linguistic features are combined by computing their weighted sum. More precisely, the resulting score for a sentence S is given by:

$$\text{score}(S) = \text{monoterms\_score}(S) \bullet \alpha + \text{variants \& multiterms\_score}(S) \bullet \beta + \text{named\_entities\_score}(S) \bullet \gamma \quad [3]$$

where α, β and γ are the coefficients that set the relative importance of the different types of linguistic features. They were empirically set to $\alpha = 1.0$, $\beta = 1/27$[6] and $\gamma = 0.5$.

The next part gives an example of the pairing operations for the TREC8 question Q16 *What two US biochemists won the Nobel Prize in medicine in 1992?* First, the question is turned into the following set of reference features:

two	US	biochemist	Nobel	prize	medicine
win	1992	<PERSON>	16.01	16.04	

where <PERSON> is the expected type of the answer, 16.01 is the identifier of the *US biochemist* term and 16.04 is the identifier of the *Nobel Prize* term.

The same kinds of features are extracted from each sentence of the document FT924-14045 that was selected for the question Q16. For instance, the sentence (after its processing by the named entities tagger) *<NUMBER> Two </NUMBER> US biochemists, <PERSON> Edwin Krebs*

[6] 27 is the highest possible weight of a term found by *FASTR*.

</PERSON> and <CITY> Edmond </CITY> Fischer, jointly won the <NUMBER> 1992 </NUMBER> Nobel Medicine Prize for work that could advance the search for an anti-cancer drug is turned into the following set of features:

two (1.0)	US (1.0)	biochemist (0.9)	Nobel (1.0)	prize (0.6)
medicine (0.5)	win (0.3)	1992 (1.0)	Edwin (0.0)	Krebs (0.0)
Edmond (0.0)	Fischer (0.0)	work (0.0)	advance (0.0)	search (0.0)
anti-cancer (0.0)	jointly (0.0)	drug (0.0)	<PERSON> (1.0)	<NUMBER> (0.0)
<CITY> (0.0)	16.01 (6.0)	16.04 (12.0)		

where the weight 0.0 is given to elements that are not part of the question. The question term *US biochemist* is found with no variation and *Nobel Prize* appears as a syntactic variant. Finally, according to [3], the score of this sentence is equal to 7.5, which is very high since the maximum score of a sentence for this question is 7.7.
When the scores of two sentences are very close, *i.e.* the difference between the two scores is below a fixed threshold, another criterion is used for ranking sentences. As these ones may be very long[7] and the expected answers are rather short, we favor sentences in which the content words of the question are the least scattered. We evaluate this scattering by taking the size of the shortest part of the sentence that gathers all these words.

8.3 A pairing module for finding a short answer

The main characteristic of this second pairing module, in relation to the first one, is the fact that it takes into account a new kind of linguistic features, *i.e.* the focus of questions, for selecting the sentences from which the final answer extraction is performed. The way this focus is recognized in sentences is presented in 8.3.3. When the expected answer for a question is not a named entity, the extraction of a short answer achieved by the module of QALC dedicated to this task (see Section 9) is based on patterns that are defined in relation to the question focus. As a consequence, when several sentences are equivalent according to features such as terms or named entities, a short answer is more likely to be extracted by QALC from the ones in which the focus of the question is present.

8.3.1 Term score

In this pairing module, the term score of a sentence in relation to a question only takes into account the mono-terms *T* of the question that are present in the sentence without any terminological variation. We will discuss this point more specifically in Section 8.4. These mono-terms are weighted according to the following scheme:

$$term\_weight(T) = np\_score(T) + \frac{np\_score(T) \bullet significativity(T)}{2} \qquad [4]$$

where *np_score(T)* is equal to 2.0 when *T* is a proper noun and equal to 1.0 otherwise. As Kozima (Kozima, 1993), we use the significativity of a term as an estimation of its degree of specificity:

$$significativity(T) = \frac{\log_2(f_T / Sc)}{\log_2(1/Sc)}, \text{with } significativity(T) \in [0,1] \qquad [5]$$

[7] It is frequent in the newspaper articles that represent the main part of the TREC corpus.

where f_T is the number of occurrences of T in a reference corpus and Sc, the number of words of the reference corpus[8].
Finally, the term score of the sentence is given by the sum of the weights of the mono-terms T.

8.3.2 Named entity score

As for the first pairing module, the named entity score of a sentence in relation to a question estimates to what extent the sentence contains a named entity that fits the expected type of the answer to the question. It takes into account two factors:

— the presence in the sentence of named entities that correspond to the expected type of the answer;

— the distance of these named entities to the terms of the question that were recognized in the sentence.

The evaluation of the first factor relies on the named entity hierarchy presented in Figure 2. Questions are always tagged with the more specific types of this hierarchy. But our named entity tagger cannot always be so accurate. Hence, we not only search in the document sentences the named entities having the type of the answer but also named-entities having a more general type. Of course, the score of a named entity decreases as its distance from the expected type, *i.e.* the number of levels that separate them in the hierarchy, increases. If several named entities in a document sentence are found to be compatible with the answer type, we only keep the one having the greatest score.
The second factor is justified by the length of the sentences in the *Question Answering* corpus. We suppose that a correct answer is more likely to be found if the named entity that fits the type of the answer is close to the recognized terms of the question. We take as reference for these terms the part of the sentence that is delimited for the computation of the answer length score. We consider that if a named entity occurs more than 4 words away from the beginning or the end of this part of sentence, it has few chances to be related to the question.
Finally, the named entity score of a sentence integrates these two factors by the following formula:

$$\begin{aligned}\text{NE\_score(S)} = {} & \text{NE\_hierarchy\_size} - \text{distance(question\_type, best\_NE(S))} \\ & + \text{answer\_proximity(best\_NE(S))}\end{aligned} \qquad [6]$$

with
NE_hierarchy_size: number of levels of the named entity hierarchy. In our case, it is equal to 3;
distance(question_type,best_NE(S)): number of levels between the answer type and the type of the best compatible named entity found in the document sentence;
answer_proximity(best_NE(S)): proximity between the best-named entity and the terms of the question. In our case, this is a binary value: 1 if the best-named entity fulfills the proximity conditions; 0 otherwise.

8.3.3 Focus recognition and focus score

For each question, the analysis module produces a focus which is composed of (a) the noun which is its head and (b) a list of its modifiers. The QALC system attempts to locate them in the sentences composing the pre-selected documents. It first detects the head of the focus, and then it identifies the noun phrase that contains it. We defined a local grammar to detect the noun phrase frontiers. This grammar is based on the tagging done by the TreeTagger (Smid, 1999).

[8] In the present case, the reference corpus is the part of the TREC corpus that corresponds to two years of the *Los Angeles Times* newspaper.

Example: Question: Who is the creator of the Muppets?

Focus = creator
Focus modifier = COMP Muppets

For this question, one of the selected documents contains the noun phrase "late Muppets creator Jim Henson", corresponding to the following expression:

Adjective + Plural Noun + Noun + Proper Noun + Proper Noun

QALC also looks for noun phrases containing synonyms of the focus, these synonyms being identified by FASTR. When the focus is absent from all the sentences selected for the current question, QALC looks for noun phrases containing the Proper Nouns present in the question, if any. The underlying hypothesis is that proper nouns are elements which are very often part of the answer.

QALC gives a score to each of the found NP. This score is function of (a) the reason why it has been selected: it contains the exact focus, one of its synonym or one of the question proper noun and (b) the comparison with the list of modifiers of the question. When the selected NP contains one or more of the modifiers, the score is proportionally increased. The best score is obtained when all the modifiers are present. In the preceding example, the score is maximal: the NP has been selected from the head of the focus "creator", and the modifier "Muppets" is part of this NP. When the NP is selected from a synonym of the focus, the score is decreased. And it is slightly more decreased when it comes from a proper noun. For example the score given to the NP:

their copy of the 13th century Magna Carta

selected for the question

Which king signed the Magna Carta?

will not be maximal, because it has not been obtained from the focus *king*, but from the proper noun *Carta*, even if the modifier *Magna* is also present in this NP.

More generally, the scoring algorithm takes into account the ratio of significant words of the question which are also present in the selected NP.

For each sentence of the selected documents, QALC tags all the pertinent NPs following the preceding algorithm. Only the best-scored NP is kept, and its score becomes one of the criteria used to evaluate the sentence in the modules in charge of the sentence selection and the answer extraction.

In order to evaluate the relevance of this criterion, we applied this recognition on all the sentences given as correct candidates for the questions of TREC 9. There were 13310 sentences answering to the 693 questions. 57.16 % of them contained a NP similar to the question focus. Overall, for 89 % of the question collection, at least one NP similar to the question focus is found in the list of sentences validated as correct answers.

8.3.4 Integration of feature scores

In this pairing module, the integration of the scores computed for each kind of linguistic features was designed to have a clear and a fine control of their respective influence. First, the term score is added to the focus score. This one is modulated by a coefficient that decreases its influence[9] when its reliability is too low. The resulting score constitutes the main criterion for comparing two document sentences *S1* and *S2*: if *S1* has a combined score much higher than *S2*, *S1* is ranked on top of *S2*. "Much higher" means that the difference of the scores for

[9] This coefficient is equal to 0.8 in such a case and to 1.0 otherwise.

S1 and *S2* is higher than a fixed threshold (equal to 0.1 in the present case). Otherwise, the named entity score is used according to the same principle as for terms. As there are not many possible values for that score, this criterion may also be ambiguous. In such a case, the first criterion is used once again but with a smaller threshold (0.05) for the difference of scores between two sentences. Finally, if there is still an uncertainty, QALC ranks first the sentence that has the shortest matching interval with the question. This interval corresponds as for the first pairing module to the shortest part of the sentence that gathers all the mono-terms of the question that were found in the sentence.

8.4 Discussion

One of the differences between the two pairing modules we have presented is the fact that the second one, used for extracting short answers, does not exploit the terms recognized by *FASTR*. This choice comes from our experiments, which show that taking multi-word terms or variant terms into account does not significantly improve the results of the pairing module. This is quite surprising as multi-word terms are considered as more meaningful as mono-terms and recognizing variants is *a priori* a way to increase recall. One of the reasons of these results could have been *FASTR* itself. This tool was designed for recognizing complex variants of terms in the field of terminology but not as a robust tool for information retrieval. Thus, the errors of the morpho-syntactic tagger it uses have a great impact on its results. We found that for single and multi-word terms without variation the recall of *FASTR* is equal to 71%. This evaluation was done with the documents selected for the 500 questions of the TREC11 evaluation. The reference was set by a basic term matcher working from the results of the *TreeTagger* tool. But we also found that using such a basic term matcher together with *FASTR* does not improve the results of our pairing modules either.
Globally, these results tend to show that the sentences from which an answer can be extracted are not necessarily those that contain the complex terms extracted from the questions. However, we still think that using these complex terms for selecting a short passage of a document containing an answer is probably an interesting approach. But in such a case, the passage should certainly be larger than a sentence.

9 Answer extraction

The QALC system first selects a set of sentences which are supposed to contain the answer. Answers with 50 characters length are then extracted from those sentences. The extraction process differs depending on whether the expected answer type is, or is not, a named entity. Indeed, when the answer type is a named entity, the extraction consists in the localization of the named entity within the sentence-answer. Thus, it mainly relies on the results from the named entity recognition module. On the other hand, when the answer type is not a named entity, the extraction process mainly relies on the recognition of the question focus, as it consists in the recognition of focus-based answer patterns within the sentence-answer.

9.1 Named-entities as answers

When the question allows the system to predict the kind of expected answer in term of a named entity type, the extraction of the answer is based on this information. This process looks for all the expressions tagged with the searched type. If several such expressions exist, we choose the noun phrase the closest to the focus, if it was recognized in the sentence, otherwise the first one. When there is no named entity of the right type, QALC generalized

the searched type using our own hierarchy. By this way, when looking for a person, QALC will look for a proper name, or look for a number instead of a length, etc.

9.2 Common noun or verb phrases as answers

When the expected answer type is not a named entity, the QALC system locates the exact answer within the candidate sentence through grammatical patterns.
The different patterns, as well as the different question types, were empirically determined from corpus analysis. The corpus consisted of the set of questions and answers provided by the TREC8 and TREC9 evaluation campaigns. Starting from this corpus analysis, we observed that the syntactic structure of the question determines the possible syntactic structures of the answer. The structure of the direct answer to a question gives a typical example of such possible structure of the answer to a question. For instance, the following questions and answers come from our corpus:

Question: Who is William Wordsworth?
Answer: William Wordsworth is a poet.

Question: What is Jane Goodall famous for?
Answer: Jane Goodall is famous for her 28 years of chimpanzee research.

In these two examples, the syntactic structure of the answer is closely derived from the syntactic structure of the question. Such direct answers are not often found in documents, but we rather find syntactic variants of them. For instance, let us consider the question *Who is William Wordsworth?*. The focus of the question is *William Wordsworth*. The answer may be an attribute of the focus like in the example above, or in apposition with the focus like in the phrase *the poet William Wordsworth*. Both syntactical structures represent a characterization of the focus by the answer. As we can see, the answer may be differently located with regard to the focus, and in fact the semantic relation between them remains the same (in this example, the characterization relation). In the same way, two other types of answer syntactic structures as reply to the question *What is Jane Goodall famous for?* were found in the documents: … *chimpanzee researcher Jane Goodall…*, corresponding to an answer-focus apposition, and *Jane Goodall, a leading chimpanzee specialist* …, corresponding to a focus-answer apposition with a separating comma.

Thus, the criteria for keeping as answer pattern some particular syntactic structures found in the corpus, was the identity of the semantic relations between the answer and the focus, underlying the syntactic structures. Finally, we distinguished three different pattern structures, with various connecting elements:

(1)	NPfocus	Connecting-elements	NPanswer
(2)	NPanswer	Connecting-elements	NPfocus
(3)	NPanswer-within-NPfocus		

Connecting elements are for instance expressions, *such as*, *called*, *known as*, or verbs with or without relative pronoun, *be*, *which be*, *that be*, punctuation such as comma and quotation marks, or preposition, *and*.
Extraction patterns of answer include the noun phrase of the focus and the noun phrase of the very answer, which can be connected by other elements such as comma, quotation marks, a preposition or even a verb. This answer structure is both syntactical, as it relies on syntactical

categories (noun phrase, verb, preposition) and grammatical structure elements (apposition order, comma, quotation marks), and lexical as it contains specific words (anyhow the focus of the question and, when necessary, particular verbs or prepositions). Thus, a grammatical pattern of an answer always includes the focus of the question. As a result, the focus has to be determined by the question analysis module in order to enable the QALC system to find an answer of common noun or verb phrase type.

Let consider the following question: *What does Knight Ridder publish?*
The focus of this question, determined by the rules of the question analysis module, is *Knight Ridder*. This question pertains to the question type *What-do-NP-VB*, with *Knight Ridder* as NP and the verb *publish* as VB. One answer pattern applying to this category is named *focusbeforeanswerVB* and consists of the following generic grammatical sequence:

NPfocus Connecting-elements NPanswer

The NPfocus is the noun phrase of the question focus within the sentence-answer. It is followed by the connecting elements, then by a noun phrase which is supposed to contain the noun phrase of the very answer, the NPanswer. In the pattern named *focusbeforeanswerVB*, the connecting elements mainly consist of the question verb (VB in the question type), possibly with adverbs and auxiliary verbs, according to the following pattern:

Have{0,1} Adverb{0,2} VB Preposition{0,1}

This means that the connecting elements pattern will contain at most once the auxiliary verb *have*, then at most two adverbs, then necessarily the verb of the question (named *VB* in the question type), possibly followed by a preposition.

The answer *30 daily newspapers*, that was found in the documents corpus, matches with the *focusbeforeanswerVB* pattern:
Knight Ridder publishes 30 daily newspapers,
This answer was extracted from the following sentence:
Knight Ridder publishes 30 daily newspapers, including the Miami Herald and the Philadelphia Inquirer, owns and operates eight television stations and is a join venture partner in cable television and newsprint manufacturing operations.

We saw in Section 7 that about 80 question types were determined from the corpus. Among them, about 45 do not expect a named entity as answer, and thus need answer patterns. For each of those question types, we built a number of answer patterns. We considered 24 answer patterns. The number of patterns for each question type varies from 2 to 20, with an average of ten patterns for each question type. Thus, several question types share a same answer pattern.

The difficulty in finding answer patterns varies according to the question type. This difficulty is partly due to the small number of some question types within the corpus, and, for the most part, to the grammatical diversity of the answers. For example, there is few *Why* questions (only 4) and few *How verb* questions (also 4), such as *Why can't ostriches fly* and *How did Socrates die?*. Moreover, answers to those questions can hardly be reduced to a pattern. We also hardly found grammatical regularities in the answers to the *What-GN-be-GN* questions, such as *What format was VHS's main competition?* or *What nationality was Jackson Pollock?* for instance. Indeed, depending on the situation, it is the first NP (more often *format*

in the first question) or the second NP (more often *Jackson Pollock* in the second question) which plays the main role in the pattern.

10 Evaluation

10.1 Sentences as answers

Participants to TREC9 evaluation proposed up to 5 ordered answers to 682 questions. The size of the answer should be 250 characters, answer corresponding to a sentence size, or 50 characters. The score of a run is the mean reciprocal rank of the correct answers (0 otherwise) over all the questions.
We sent TREC9 two runs which gave answers of 250 characters length. The first run used ATT as search engine, and the second one, Indexal. Results are consistent with our previous analysis (see Section 5.2). Indeed, the run with ATT search engine gives slightly better results (0.407 strict) than those obtained with the Indexal search engine (0.375 strict). With this score, the QALC system was ranked sixth over 28 participants at TREC9 QA task. Table 4 sums up the number of answers found by our two runs.

Rank of the correct answer retrieved	Run using ATT	Run using Indexal
1	216	187
2 to 5	159	185
Total of correct answers retrieved	375/682	372/682
Score	0.407	0.375

Table 4. Number of correct answers retrieved, by rank, for the two runs at 250 characters

Globally the two runs are more similar in term of the number of correct answers found (375 vs. 372) than in their scores. The run using ATT engine comprises more answers at rank 1 than the run using Indexal. However, it is difficult to explain the answer ranking difference: right documents are similarly retrieved by the two engines, although Indexal maybe returns more documents close to the expected answer that lead to interferences when selecting the sentences to be kept.

10.2 Short answers

The TREC-QA track provides an evaluation of the overall system. But we wanted to evaluate our system from different points of view. First, we wanted to know for which categories QALC obtains good or weak results, and secondly, we wanted to evaluate separately each module of our system. So, we used the evaluation reference data provided by the NIST after each TREC conference. The NIST provides the list of correct answers found by participants to TREC, and the documents where they were retrieved. The NIST also provides the patterns of correct answers and a program that computes the score of the system[10]. The score is computed as for long answers. We present here the results we sent to the TREC10 evaluation.
The categories that we obtain from the question analysis module are of very dissimilar size, from 2 questions for the smallest to 182 for the biggest. For more clearness, we brought together categories which had a number of shared characteristics so as to create 12 sufficiently homogeneous categories. Nevertheless, we have to keep in mind that these

[10] The score result is then slightly better than the one which is given at the track, as it is an automatic approximate process that decides on the correctness of the answer and not human judges as in the TREC-QA track.

categories include more specific sub-categories that we study with more detail in particular cases. Table presents the score of the system attached to each question category, in decreasing order of the scores. The *WhatbeNP* category has the biggest size. It includes a number of low size categories (such as *WhatbeNPforNP* for instance) that do not appear in the table. Only the *WhatbeNPofNP* appears because of its bigger size. Among the 492 questions from TREC10, only one is not taken into account in our statistics: *What causes gray hair?*, the only one instance in its category (*WhatverbNP*).
In this first approach, we note that the best scores are obtained by categories corresponding to a named entity. Either categories whose expected answer is, with very few exceptions, a named entity, (*Where*, a location, *When*, a date, *Who*, a person or an organization), or for which a great part of the questions expect a named entity as answer (*WhatNPdoNP*, *WhatbeNPofNP*, *How*). This result seems coherent in that the knowledge of the answer type allows a more precise location of the answer within the documents whose named entities are tagged.

Category	Example	Number of questions	Score
Where	Where is the Holland Tunnel	27	0.316
When	When did Hawaii become a state?	26	0.280
WhatNPdoNP	What year did the U.S. buy Alaska?	24	0.272
Who	Who discovered x-rays?	46	0.254
WhatbeNPofNP	What is the melting point of copper?	47	0.247
How	How long did Rip Van Winkle sleep?	33	0.192
WhatbeNP	What is acupuncture?	199	0.189
WhatNPbeNP	What precious stone is a form of pure carbon?	47	0.182
WhatNPverbNP	What strait separates North America from Asia?	6	0.167
Which	Which president was unmarried?	10	0.100
WhatdoNP	What does a barometer measure?	22	0.027
Why	Why does the moon turn orange?	4	0.000
All categories		491	0.205

Table 5: Evaluation of the overall system for each question category

In order to evaluate the sentence selection module, we computed the score on the top 5 sentences that the module returns, as defined by TREC. Table 5 shows the results of the evaluation for each category. We also computed the number of correct answers retrieved per category, with no regard to their rank. The answer extraction is evaluated using these data: the evaluation measure is then the percentage of correct short answers extracted from the sentences which contain a correct answer.
Items in Table 6 are ranked according to the decreasing order of sentence scores. We then obtain a ranking rather different from the one which results from the decreasing order of final scores. Obviously, this is due to large differences in extraction ratio according to the categories.

Category	Sentence score	Answer score	Sentence-answer extraction
WhatNPdoNP	0.425	0.272	53%
Where	0.415	0.316	93%
WhatNPvrbNP	0.333	0.167	50%
How	0.298	0.192	58%
WhatbeNPofNP	0.288	0.247	100%
Who	0.286	0.254	94%
WhatbeNP	0.281	0.189	69%
When	0.280	0.280	100%
WhatNPbeNP	0.274	0.182	63%
WhatdoNP	0.205	0.027	25%
Which	0.175	0.100	33%
Why	0.00	0.00	
All categories	0.286	0.205	73%

Table 6: Evaluation of sentence selection and answer extraction processes

Category *Why* is very small, only 4 questions that we do not answer even in terms of a sentence. As a matter of fact, this category has a bad recall though it has a good precision. It means that very few documents among the corpus contain an answer to these questions.
Results of Table 6 show that the answer is correctly extracted when categories correspond to a named entity. Therefore, the good overall results of these categories are mostly due to successful extraction than to sentence selection correctness. Then, the *WhatbeNP* category has rather good results concerning the answer extraction process. Contrariwise, *WhatNPdoNP* and above all *WhatdoNP* category get lower extraction ratio. Questions from *WhatbeNP* and *WhatdoNP* categories, and partly from *WhatNPdoNP*, do not expect a named entity as answer. Thus, the answer extraction process uses extraction patterns in these cases.
In order to better evaluate the answer extraction module, we applied the last module that selects sentences and extract the answer on all the sentences judged as correct for the 682 questions of TREC9. 590 sentences were kept and 464 correct answers were extracted from this set, thus QALC succeeded for locating 78.6% of the correct answers.

11 Related work

The question answering systems architecture is generally similar to the QALC one: determination of the expected answer type, selection of a restricted set of relevant documents by a search engine, and finally, search of this set to find the possible answers in this set.

We have already underlined the importance of the parsing of the question to find the expected answer type. As done by QALC, most systems first determine the expected answer type by seeking pre-defined patterns in the question. So, Prager et al.(2000) use 400 different patterns to identify about 50 answer types. On the opposite, the system developed by IBM (Ittycheriah et al.,2000) is based on a maximum entropy model for classifying the answer types. The FALCON system authors (Harabagiu et al.,2000) have defined a taxonomy of the answer types following the synset hierarchies in WordNet.

All the question-answering systems participating in TREC9 use a search engine selecting a subset of relevant documents in a basis composed of about one million of documents. In the QALC system, we keep each retrieved document as a whole. Some systems only keep the paragraph (or paragraphs) considered as the most relevant. For example, the Kwok et al.

(2000) system uses the information retrieval system PIRCS, a home-made system, which first selects a set of 300 relevant sub-documents composed of about 300 to 550 words. The FALCON system (Harabagiu et al., 2000), also proceeds to a selection of paragraphs by using a Boolean engine. Litkowski (Litkowski, 2000 and 2001) proceeds to a syntactico-semantic parsing of the 20 first documents retrieved by the ATT search engine. But, the 20 first documents include the correct answer only for 78% of the questions, and in the 200 first ATT documents for 92.5% of the questions. The limit on the number of considered documents prevents this system to obtain a high score.

In the Kwok et al. (2000 and 2001) system, a great number of documents are kept and the matching is done on the basis of a rich set of criteria: words stemming, synonyms (a dictionary of 300 terms manually extracted from WordNet), value of the document score given by PIRCS, weighting by the inverse of the word frequency inside the collection, presence of the exact word when it is important (some superlatives for example), proximity of the words in the sentence, compound words, presence of words which are capitalized or quoted in the question.
The FALCON system, which also deals with selected paragraphs, is the one which most widely uses syntactic and semantic parsing techniques. In this system, they try to unify the semantic representation of the question and the semantic representations of the selected paragraphs. When the unification succeeds, these semantic representations are expressed in a logical form in order to infer an answer justification. If the unification fails, or if the answer cannot be justified, the question formulation is extended for a new step of paragraphs selection. The question expansion is made by a constrained search of WordNet. It is the only system participating to TREC conference which backtracks on the question formulation at the input of the system.
Focusing on the paragraphs which are potentially interesting allows the system to consider a greater number of documents in the following steps, or to apply further syntactico-semantic analyses which give good results but are time consumers. The difficulty is also to find a robust parser, that means a parser able to deal both with the sentences of the documents and with the questions.
Hovy et al. (2001) use CONTEXT, a machine learning based grammar parser, but they have to extend its grammar to the question forms. Their system Webclopedia identifies the candidate answer passages and CONTEXT parses their sentences. Then, a matcher module compares the parse trees of the question and of the candidate sentences; but a second and independent matcher uses a word window on the answer text and seems to be useful when the answer parse tree is not complete. Buchholz (2001) system also combines a basic keyword matching component with a high-level NLP component Shapaqa that uses chunking and grammatical relations to impose constraints on what it extracts as an answer. She obtained less answers, but a much higher precision.

12 Conclusion

An open domain question-answering system has to find an answer in a sufficiently short time in order to be usable. As this answer is extracted from a very large corpus, it is attractive to apply mainly simple numeric processes. However, various experiences show the necessity to integrate natural language processes, able to reason on syntactic, semantic or even pragmatic knowledge, in order to reach quality levels comparable to those of humans. The future orientations of the TREC QA task point in this direction. In a future of 5 years, its organizers foresee large improvements in response time, answer justification, multi-document answer merging and the capacity to develop a dialogue for chaining up several questions on a same

topic. These evolutions will of course require more semantic natural language processes in order to better emulate human understanding.
The improvements we are now working on concern the integration of more semantic constraints to better define the set of possible anwers and to estimate more precisely their correctness. We will continue to use WordNet to find semantic relations having to be verified between the answer and the question focus and eventually other question terms. This way, we will reduce to a smaller set the potential answers returned by our syntactic patterns. WordNet, by its general coverage, is the most suitable semantic resource for answering open domain questions. However, it is advisable to develop mechanisms allowing to adapt its sometimes too high generality level: many synonyms, a classification that is not always homogenous, etc. Another source of knowledge lies in the Web since the redundancy that it provides leads to a better confidence level for retrieved answers.
However, the efficiency of the question answering system is dependent on the robustness of the processes involved, and the more we add precise specific processing or semantic knowledge, the more our system runs the risk to become brittle. We think that an answer can come from the strategy we will use to apply the different processes. We first developed general robust processes, and then we tried to improve their precision by adding more specific constraints, or by using more precise sub-processes like syntactic parsing or linguistic variation normalization with FASTR. A general strategy to solve the problem will then consist of trying precise resolution modules first, and then if they fail, applying more robust ones. This leads to work on auto-evaluation of the modules performance.

13 References

Aït-Mokhtar, S. & Chanod, J-P. (1997). Incremental finite-state parsing. *Proceedings of Applied Natural Language, Washington, DC.*

Buchholz S. (2001) Using Grammatical Relations, Answer Frequencies and the WorldWide Web for TREC Question Answering, Proceedings of the Text retrieval conference, TREC 10, Gaithersburg, MD. NIST Eds. pp.496-503

CELEX, http://www.ldc.upenn.edu/readme_files/celex.readme.html, UPenn, Consortium for Lexical Resources, 1998.

CLR, http://crl.nmsu.edu/cgi-bin/Tools/CLR/clrcat#D3, NMSU, Consortium for Lexical Resources, 1998

Dyer M., *In-depth understanding*, MIT Press, Cambridge, MA, 1983.

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

Graesser A.C., McMahen C.L., Johnson B.K., « Question Asking and Answering », dans *Handbook of Psycholinguistics*, M.A. Gernsbcher (Eds), Academic Press, 1994, p. 517-538.

Grishman R., Sundheim B., « Design of the MUC6 evaluation », *Actes de MUC-6*, Morgan Kauffmann Publisher, Columbia, MD, 1995.

Harabagiu S., Pasca M., Maiorano J., « Experiments with Open-Domain Textual Question Answering », *Actes de COLING 2000*, Saarbrucken, Germany, 2000, p. 292-298.

Ittycheriah A., Ratnaparkhi A., Mammone R.J., « IBM's statistical Question Answering System », *Actes de TREC9*, Gaithersburg, MD, 2000, p. 229-234.

Jacquemin C., « Syntagmatic and paradigmatic representations of term variation », *Actes de ACL'99*, 1999, p. 341-348.

Jacquemin C., Bush C., « Fouille du Web pour la collecte d'entités nommées », *Actes de TALN 2000*, Lausanne, 2000, p. 187-196.

Justeson J.S., Katz S.M., « Technical terminology: some linguistic properties and an algorithm for identification in text », *Natural Language Engineering*, vol. 1, 1995, p. 9-27.

Kozima H., Text Segmentation Based on Similarity between Words, *Actes de ACL'93 (student session)*, Columbus, Ohio, 1993, p. 286-288.

Kwok K.L., Grunfeld L., Dinstl N., Chan M., « TREC9 Cross Language, Web and Question-Answering Track experiments using PIRCS », *Actes de TREC9*, Gaithersburg, MD, 2000, p. 419-429.

Kwok K.L., Grunfeld L.,Dinstl N. & Chan M.(2001) TREC 2001 Question-Answer, Web and Cross Language Experiments using PIRCS, Proceedings of the Text retrieval conference, TREC 10, Gaithersburg, MD. NIST Eds. pp. 447-451

Hovy, E. , Hermjacob, U. & Lin C-Y. (2001). The Use of External Knowledge in Factoid QA. Proceedings of the Text retrieval conference, TREC 10, Gaithersburg, MD. NIST Eds. pp. 644-652.

Lehnert W., « Human and computational question answering », *Cognitive Science*, vol. 1, 1977, p. 47-63.

Lehnert W., *The process of question answering*, Lawrence Erlbaum Associates, 1979.

Litkowski K, « Syntactic Clues and Lexical Resources in Question-Answering », *Actes de TREC9*, Gaithersburg, MD, 2000, p. 157-166.

Litkowski K., CL Research Experiments in Trec-10 Question Answering, Proceedings of the Text retrieval conference, TREC 10, Gaithersburg, MD. NIST Eds. pp251-260

de Loupy C., Bellot P., El-Bèze M., Marteau P.-F., « Query Expansion and Classification of Retrieved Documents », *Actes de TREC7*, Gaithersburg, MD, 1998, p. 443-450.

Mollá Aliod D., Schwitter R., Hess M., Fournier R., « Extrans, an answer extraction system », *Traitement Automatique des Langues*, vol. 41, n°2, 2000, p. 496-522.

Morton T. S., « Using co-reference in Question Answering », *Actes de TREC8*, Gaithersburg, MD, 1999, p. 685-688.

Prager J., Brown E., Radev D. R., Czuba K., « One Search Engine or two for Question-Answering », *Actes de TREC9*, Gaithersburg, MD, 2000, p 235-240.

Robertson E., Walker S., Beaulieu M., « Okapi at TREC7: automatic ad hoc, filtering, VLC and interactive », *Actes de TREC7*, Gaithersburg, MD, 1999, p. 253-264.

Schmid H., « Improvments in Part-of-Speech Tagging with an Application To German », dans *Natural Language Processing Using Very Large Corpora*, S. Armstrong, K.W. Chuch, P. Isabelle, E. Tzoukermann et D. Yarowski (Eds), Kluwer Academic Publisher, Dordrecht, 1999.

Zock M., Mitkov R., « How to ask a foreigner questions without knowing his language? Proposal for a conceptual interface to communicate thought », *Actes de Natural Language Processing Pacific Rim Symposium*, Singapore, 1991, p. 121-130.