QALC–The Question-Answering System of LIMSI-CNRS

Olivier Ferret, Brigitte Grau, Martine Hurault-Plantet, Gabriel Illouz, Christian Jacquemin, Nicolas Masson, Paule Lecuyer

QALC - the Question-Answering system of LIMSI-CNRS

Olivier Ferret* Brigitte Grau* Martine Hurault-Plantet* Gabriel Illouz*
Christian Jacquemin* Nicolas Masson **
Paule Lecuyer **

*LIMSI-CNRS BP 133, 91403 ORSAY Cedex, FRANCE

**Bertin Technologies Parc d'Activités du Pas du Lac, 78180 Montigny-le-Bretonneux

{ferret,grau,gabrieli,jacquemin}@limsi.fr

{masson, lecuyer}@bertin.fr

1 Introduction

In this report we describe the *QALC system* (the Question-Answering program of the Language and Cognition group at LIMSI-CNRS) which has been involved in the QA-track evaluation at TREC9. The purpose of the Question-Answering track is to find the answers to a set of about 700 questions. The answers are text sequences extracted from the 5 volumes of the TREC collection. All the questions have at least one answer in the collection.

The basic architecture of *QALC* is composed of seven modules, two for the questions and four for the corpora, and a last pairing module which produces the sentences ranked by decreasing order of relevance (see Figure 1).

The *QALC* system relies mainly on genuine Natural Language Processing components. Most of the components take as input a tagged version of the documents. We use the *TreeTagger* for this purpose (Stein and Schmid, 1995). The system is based on the following modules:

Natural language question analysis The analysis of the questions relies on a shallow parser which spots discriminant patterns and assigns categories to the questions. The categories correspond to the types of entities which are likely to constitute the answer to this question.

Term extraction The term extractor is based on syntactic patterns which describe compound nouns. The maximal extension of these compounds is produced along with the plausible subphrases.

Search engine We tested two kinds of search engines: the search engine from ATT, by using only its outputs, and Indexal, a search engine from Bertin Technologies.

Automatic indexing & variant conflation Automatic indexing relies on *FASTR* (Jacquemin, 1999), a shallow transformational natural language analyzer which recognizes the occurrences and the variants of the terms produced by the term extraction module. Each occurrence or variant constitutes an index to the document which is ultimately used in the process of document ranking and in the process of question/document pairing.

Named entity recognition Similarly, named entities are recognized in the documents in order to build indexes which are used for measuring the degree of similarity between the questions and the document sentences. Named entities are extracted through lexico-syntactic patterns combined with significantly large lexical data.

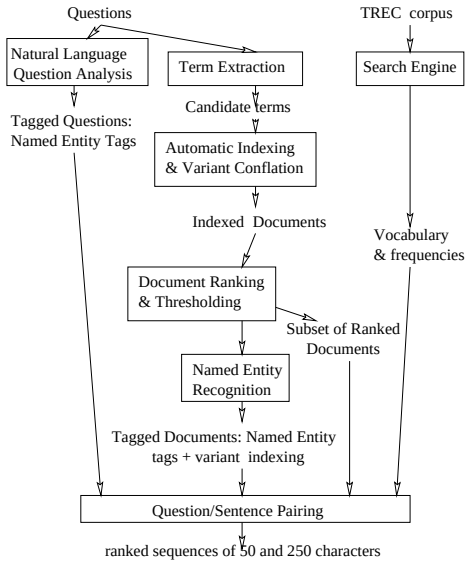

Figure 1: Flowchart of QALC.

Document ranking & thresholding Documents are ranked according to a weighted measure of the indexes produced by the automatic indexing and variant conflation module. Only the n best ranked documents are selected. A further selection of the documents is made if a plateau can be recognized in the relevance curve of the documents.

Question/sentence pairing Finally, all the data extracted from the questions and the documents by the preceding modules is used by a pairing module to evaluate the degree of similarity between a document sentence and a question. The answers are extracted from the sentences that are chosen from the documents selected by the preceding module.

Our system is very similar to the system built for TREC8 (Ferret et al., 1999). Hence, we will only focus on the differences between these two systems in the following sections.

2 Natural Language Question Analysis

Question analysis is performed in order to assign features to questions and use these features for the pairing measurement between a query (question) and potential answer sentences (answer). Basically, question analysis allows the prediction of the kind(s) of answer, called *target* (for instance, ORGANIZATION). The retrieved documents (see Section 6) are labeled with the same tagset as the questions. During the pairing measurement, the more the question and a sentence share the same tags, the more they are considered as involved in a question-answer relation. For example:

Question: *How many people live in the Falklands?* $\rightarrow$ target = NUMBER

Answer: *... Falklands population of* `<b_numex_TYPE=NUMBER> 2,100 <e_numex>` *is concentrated.*

The question analysis is based on a set of rules that uses syntactic and semantic criteria. The targets used are PERSON, ORGANIZATION, LOCATION (either CITY or PLACE), TIME-EXPRESSION (either DATE, TIME, AGE or PERIOD), and NUMBER (either LENGTH, VOLUME, DISTANCE, WEIGHT, PHYSICS or FINANCIAL). The target tags are hierarchised in order to offer more flexibility when searching for the answer. We have established 17 semantic classes, hierarchically structured as shown in Figure 2.

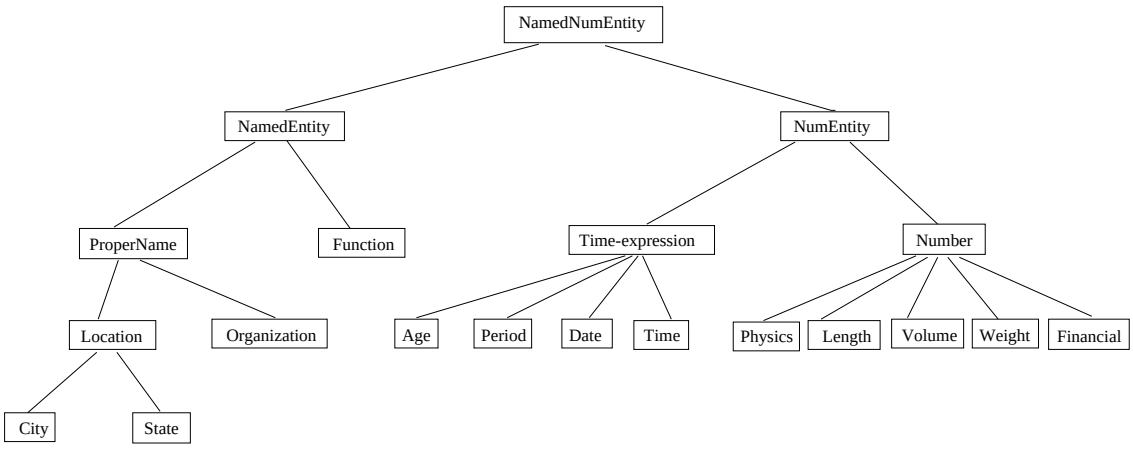

Figure 2: The hierarchy of target

3 Term Extraction

Terms are extracted from the questions and sentences through part-of-speech filtering. These terms are used by FASTR to index documents retrieved by the search engines.

The questions are first tagged with the help of the *TreeTagger*. Then, patterns of syntactic categories are used to extract terms from the tagged corpora. They are very close to those in (Justeson and Katz, 1995), but we do not include post-posed prepositional phrases. The pattern used for extracting terms is[1]:

$$(((((\mathrm{JJ}|\mathrm{N_N}|\mathrm{N_P}|\mathrm{V_{BG}})\ )?(\mathrm{JJ}|\mathrm{N_N}|\mathrm{N_P}|\mathrm{V_{BG}})\ (\mathrm{N_P}|\mathrm{N_N})))|(\mathrm{V_{BD}})|(\mathrm{N_N})|(\mathrm{N_P})|(\mathrm{CD})) \quad (1)$$

The longest string is acquired first and substrings can only be acquired if they do not begin at the same word as the superstring. For instance, from the sequence $name_{\mathrm{N_N}}$ of_{IN} the_{DT} $US_{\mathrm{N_P}}$ $helicopter_{\mathrm{N_N}}$ $pilot_{\mathrm{N_N}}$ $shot_{\mathrm{V_{BD}}}$ $down_{\mathrm{RP}}$, the following four terms are acquired: *US helicopter pilot*, *helicopter pilot*, *pilot*, and *shoot*. We also keep all the lemmas corresponding to the single words of the question : *US* and *helicopter*.

The mode of acquisition chosen for terms amounts to considering only the substructures that correspond to an attachment of modifiers to the leftmost constituents (the closest one). For instance, the decomposition of *US helicopter pilot* into *helicopter pilot* and *pilot* is equivalent to extracting the subconstituents of the structure [*US* [*helicopter* [*pilot*]]].

4 Search Engine

We tested the QALC system with the 200 questions that were proposed at the TREC8 Question-Answer task. One module of our system is the selection, through a search engine, of documents which may contain an answer to the question. We examined the results of three search engines, Zprise provided by NIST, ATT whose results for the TREC questions are provided by NIST, and Indexal placed at our disposal by Bertin Technologies, a French engineering company.

Zprise is a vectorial search engine developed by NIST. We used it with the following features : bm25idf weighting function of Okapi (Robertson E. and Beaulieu, 1999), stemming with the Porter algorithm, and no relevance feedback. Indexal is a pseudo-boolean search engine developed by Bertin Technologies. It is

[1] $\mathrm{N_N}$ are common nouns, $\mathrm{N_P}$ proper nouns and CD numeral determiners.

mostly used for information retrieval in smaller data bases. Indexal enlarges the request by use of stemming. A side effect of the retrieval method seems to be the preference given to large documents. Another feature is the use of the notion of affinity between words. The request then consists of a set of words that have to be found in a window of text of a given length.

One goal of our tests was to determine the optimal number of documents to select from the retrieved documents. Indeed, having too many documents leads to a question processing time that is too long, but conversely, having too few documents reduces the possibility of obtaining the correct answer. Obviously, the other goal was to determine the best search engine, that is to say the one which gives the highest number of documents containing the answers.

4.1 Document Selection Threshold

We carried out four different tests with the Zprise search engine, respectively with 50, 100, 200, and 500 selected documents. At the end of TREC8, the NIST provided, for each question, a list of documents which contained the answer. We based our comparisons on this list of relevant documents . Table 1 shows the test results.

Number of selected documents	50	100	200	500
Number of questions for which all the relevant documents were retrieved	75	86	111	128
Number of questions for which some of the relevant documents were retrieved	116	98	82	66
Number of questions for which no relevant document was retrieved	19	16	7	6
Total number of relevant documentsthat were retrieved (total number: 1197)	702	820	931	1018

Table 1: Comparison between the numbers of relevant retrieved documents for different thresholds of selected documents

According to Table 1, the number of questions for which all the relevant documents were retrieved increases with the number of selected documents, but increases less between 200 et 500 selected documents. In the same way, the number of questions for which no document was retrieved is almost equal for 200 and 500 selected documents. Furthermore, the total number of relevant documents that were retrieved is less increasing between 200 and 500 selected documents (7% more) than between 50 and 100 selected documents (10% more). Generally speaking, the improvement of the search engine results tends to decrease beyond the threshold of 200 selected documents. For the TREC9 questions processing, we then choose the threshold of 200 documents selected which seemed to offer the best arrangement between the number of documents in which the answer may be found and the question processing time.

4.2 Search Engine Evaluation

We compared the results given by the three search engines for the 200 TREC8 questions and for a threshold of 200 selected documents. Table 2 gives the tests results.

Search engine	Indexal	Zprise	ATT
Number of questions for which all the relevant documents were retrieved	109	111	142
Number of questions for which some of the relevant documents were retrieved	73	82	52
Number of questions for which no relevant document was retrieved	18	7	6
Total number of relevant documentsthat were retrieved (total number : 1197)	814	931	1021

Table 2 : Compared performances of the Indexal, Zprise and ATT search engines

The ATT search engine revealed itself as the most efficient one according to the following three criteria: the largest number of questions for which all the relevant documents were retrieved, the lowest number of questions for which no relevant document was retrieved, and the most of relevant documents retrieved for all the 200 questions.

The evaluation results of our whole processing chain, tested with the TREC8 questions, were respectively 0.409 with Zprise, 0.452 with Indexal, 0.463 with ATT.

These different observations led us to choose two different search engines for the TREC9 QA task. The first one is ATT, for the obvious reason that it gives us the best results. The other is Indexal because, on the one hand, we can use it more freely as we have the software at our disposal, and, on the other hand, the Bertin Technology is in the process of improving its system.

5 Automatic Indexing and Document Ranking

The selection of relevant documents among the results given by the search engine relies on an NLP-based indexing composed of both single-word and phrase indexes and linguistic links between the occurrences and the original terms. The original terms are those extracted from the questions. The occurrences or variants of these terms are extracted by *FASTR* (Jacquemin, 1999), a transformational shallow parser for the recognition of term occurrences and variants. The detection of variants in the documents is based on rules, enabling morphological and semantic transformations.

The ranking of the documents relies on a weighted combination of the terms and variants extracted from the documents. The weighting scheme relies on a measure of quality of the different families of variations described in (Ferret et al., 1999): non-variant occurrences are better than morphological and morpho-syntactic variants, and semantic and morpho-syntactico-semantic variants receive the lowest weight. We also emphasize terms with proper names, since they are more reliable indexes than common names, and long terms are preferred over single ones, according to their number of words. Thus documents containing multiple word terms, either variant of initial terms or not, are better ranked than documents that contain scattered single words. We retain a maximum of 100 documents. However, when the weight curve presents a sudden slope, we only select documents before the fall, with a minimum set to 20.

6 Named Entity Recognition

Named entities are recognized in the documents in order to build indexes which are used for measuring the degree of similarity between the questions and the document sentences. Named entities receive one of the following types: `PERSON, ORGANIZATION, LOCATION (CITY or STATE), NUMBER` (a time expression or a number expression, see Figure 2). They are defined in a similar way to the MUC task (Grishman and Sundheim, 1995) and recognized through a combination of

- lexical lookup (for syntactic or semantic tags on the single words) and rules which use these tags together with lexical elements; and
- dictionary lookup (the direct access to lists of named entities).

The three lists used for lexical lookup are CELEX (CELEX, 1998), a lexicon of 160,595 inflected words with associated lemma and syntactic category, a list of 8,070 first names (6,763 of which are from the CLR (1998) archive at New Mexico State University) and a list of 211,587 family names from the CLR archive at New Mexico State University. The recognition of location expressions was improved by integrating the work of (Illouz, Jacquemin, and Habert, 1999).

7 Question/Sentence Pairing

This section first presents the module that selects for each question a list of five ranked sentences in which the *Answer Extraction* module then tries to find the five required answers. This module relies on the results of all the preceding modules:

- each question is assigned a set of terms and one or several categories according to its focus;

- a set of documents is selected for each question. In each of them, named entities and terms extracted from the questions are tagged.

Although they share a large number of features, the *Question/Sentence Pairing* module for the 250 character task is not the same as the one for the 50 character task. The first one is identical to the *Pairing* module used in the TREC8 *Qalc* system while the second one is a new one that aims at being more precise. Nevertheless, these two modules are based on the same principle: we compare each sentence from the selected documents for a question to this question and we always keep the five sentences that are the most similar to the question.

7.1 The TREC8 *Pairing* Module

In the TREC8 *Pairing* module, questions and document sentences are transformed into vectors. Then these vectors are compared by computing a similarity measure. Such vectors contain the most significant words of the primary sentences or questions, i.e. mainly their content words (as nouns, verbs and adjectives). Each word in a vector is weighted according to its importance in relation to the *Question/Answering* corpus by using the *tf.idf* weighting policy, as it is often done in Information Retrieval. These vectors also contain two kinds of linguistic features: terms and named entities. These linguistic features are considered in the vectors as if they were significant words.

Terms Each term that has been extracted by the term extractor described in Section 3 has an unique identifier that is used for marking the occurrences and the variants of this term both in the questions and in the document sentences. In the sentences, this identifier is associated with a score that reflects the distance between the found variant and its reference form in the question vector (see Section 5). In the question vector, we add the term identifier with a default weight.

Named entities Each recognized named entity is marked with a specific tag according to its type (see Section 6). On the other hand, the kind of the answer expected for each question is determined by the *Question Analysis* module. Thus, for a question, we add the tag(s) of the expected type(s) of the answer to its vector and for a sentence, we add the tags of the named entities that have been recognized in the sentence to its vector. In both cases, each tag is given a fixed weight.

The comparison between a sentence vector V_d and a question vector V_q, both enriched with linguistic features, is then achieved by computing the following similarity measure:

$$sim(V_q, V_d) = \frac{\sum_i wd_i}{\sum_j wq_j} \tag{2}$$

with wq_j, the weight of a word in the question vector and wd_i, the weight of a word in a sentence vector that is also in the question vector. This measure evaluates the proportion and the importance of the elements (words or linguistic features) in the question vector that are present in the sentence vector with regards to all the words in the question vector.

We also take into account the difference in length between a question and a document sentence. This criterion is used as a secondary key for sorting the sentences that are selected as possible answers to a question: if two sentences have the same similarity value, the shortest sentence is ranked first.

7.2 The TREC9 *Pairing* Module

The *Pairing* module used this year for the 50 character task differs from the above one on two main points. First, it makes no distinction between words and terms. It only deals with terms and considers words as mono-terms. Second, it does not compute the similarity between a question and a sentence globally by gathering all their features in a vector. On the contrary, question/sentence similarity is evaluated for each kind of features and the results of this evaluation are then aggregated in relation to the importance of the type of the features. Three kinds of features are taken into account:

- the presence in the sentence of the terms extracted from the question;
- the presence in the sentence of named entities whose type fits the expected type of the answer;
- the length of the shortest part of the sentence that contain all the terms that are terms of the question.

The first change is intended to simplify the overall process while the second one aims at increasing the tuning capabilities of the pairing module.

7.2.1 Term Similarity

The term similarity of a sentence in relation to a question is given by the sum of the weights of the terms extracted from the question that are present in the sentence, either as they are or as variants. The weight of such a term T takes into account three factors:

- the score given by *FASTR* to the term T. This score depends on what kind of variant of T was recognized (see Section 5);
- the fact that T is or not a proper noun. We actually consider that proper nouns are more significant than the terms built with common words;
- the specificity of the term T. As Kozima (1993), we evaluate this specificity by using the significativity of T, which corresponds to its normalized information with regards to a corpus. In our case, the reference corpus is the *LA Times* part of the *Question Answering* corpus. As they are considered as very specific, multi-terms and proper nouns are given the maximum significativity value, which is equal to 1.

The weight of a term T combines these three factors into the following formula:

$$term\_weight(T) = fastr\_score(T) + (\frac{fastr\_score(T)}{2} \bullet np\_modulator(T) \bullet significativity(T)) \quad (3)$$

1. $fastr\_score(T)$: score given by *Fastr* to the term T;
2. $np\_modulator(T)$: proper noun modulator. It is equal to 2 for proper nouns and equal to 1 for the other terms;
3. $significativity(T)$: significativity of the term T.

Roughly speaking, the weight of a term T is equal to its *Fastr* score plus a modulation of the half of that score in relation to the type and the specificity of T. Moreover, if a term of a question has several occurrences in a sentence, only the occurrence that has the greatest weight is kept.

7.2.2 Answer Length Score

The length score of a sentence aims at favoring sentences in which the terms of the question are grouped in a small area. Its presence is justified by the presence in the *Question Answering* corpus of a large proportion of newspaper articles, which often contain long sentences.

The length score is simply equal to the number of words in the smallest part of the sentence that gathers all the terms of the question that were recognized for this sentence.

7.2.3 Named Entity Similarity

The named entity similarity measure of a sentence in relation to a question 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 Section 2. Questions are always tagged with the more specific types of this hierarchy. But our named entity tagger can not always be so accurate. Hence, we search in the document sentences not only the named entities having the type of the answer but also the 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 chance to be related to the question.

Finally, the named entity similarity measure is given by the following formula:

$$\begin{aligned} ne\_similarity \quad = \quad & ne\_hierarchy\_level\_number - distance(question\_type, best\_document\_ne) \\ & + answer\_proximity(best\_document\_ne) \end{aligned} \tag{4}$$

1. *ne_hierarchy_level_number*: number of levels of the named entity hierarchy. In our case, it is equal to 3;
2. *distance*(*question_type*, *best_document_ne*): number of levels between the answer type and the type of the best compatible named entity found in the document sentence;
3. *answer_proximity*(*best_document_ne*): 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.

7.2.4 Global Similarity

As in the TREC8 *Pairing* module, the global similarity measure in this module is used to rank the document sentences that may contain the answer to a question. This measure consists of aggregating the three dimensions we have presented above according to the following principles.

First, we favor the term similarity: if a sentence *S1* has a term similarity that is significantly greater than the term similarity of a sentence *S2*, *S1* is ranked on top of *S2*. By *significantly greater*, we mean more precisely that $term\_similarity(S1) > term\_similarity(S2) + similarity\_equivalence$. For the runs we submitted, *similarity_equivalence* was equal to 0.1. When the term similarity is too ambiguous, we rely on named entity similarity: if *S1* has greater named entity similarity than *S2*, *S1* is ranked on top of *S2*. As there are not many possible values for that kind of similarity, this criterion may also be ambiguous. In such a case, we come back to the term similarity criterion, but with a smaller *similarity_equivalence* value: this one was equal to 0.05 for our evaluation runs. Finally, if there is still an uncertainty, we use the answer length score: we choose the sentence that has the lowest length score.

7.3 Answer Extraction

The selection of a subpart of a sentence longer than 250 characters is simply done by reducing it by its two ends. The extraction of a short answer depends on the presence or not of a tag that may correspond to the

kind of answer, if it is known. We retain either the exact tag proposed by the question analyzer or a more general one in our hierarchy. If such a tag is found in the sentence itself, or just besides in one of the two contiguous sentences, we select the tagged expression. When there is no tag in the sentence or no requested tag found by the question analyzer, we select the longest part of the sentence that does not contain any term.

8 Results and Analysis

We sent to TREC9 three different runs. Two of those runs give answers of 250 characters and use the same processing chain except for the search engine module which uses for one run ATT, and for the other run, Indexal. The third run gives answers of 50 characters length and is the result of a different ending module. The search engine used for the third run is ATT. Results are consistent with our previous analysis. Indeed, the run with ATT search engine gives slightly better results (0,407 strict et 0,414 lenient) than those obtained with the Indexal search engine (0,375 strict et 0,382 lenient). Table 3 sums up the number of answers found by our two runs.

Rank of the retrieved answer	1	2	3	4	5	No answer retrieved
Run using ATT (strict)	216	73	48	23	15	307
Run using Indexal (strict)	187	78	50	35	22	310
Run using ATT (lenient)	221	72	50	23	16	300
Run using Indexal (lenient)	190	81	50	34	23	304

Table 3 : Number of retrieved answers, by rank, for the two runs at 250 caracters

The results given in Table 3 lead us to the following remarks : the run using the ATT search engine gives more answers at rank 1 than the run using Indexal. Conversely, this last run gives more answers in lower ranks. Therefore, the score difference between these two runs seems to result more from a better ranking of the correct answers than from the slightly different number of retrieved answers.

We then analysed the overlap between the answers of the two runs. These runs give rather different results, as they have only partial overlapping : for 246 same questions, none of them gives the correct answer at any rank (among about 310 not found for each of them). The other 60 questions for which they do not have the correct answer are different for each run. Furthermore, only 169 answers are found at the same rank for the two runs, among about 370 retrieved answers, e.g. a little less than a half. As shown in Table 3, answers given by the run using ATT have a better rank than those given by the run using Indexal: 162 questions are ranked better through ATT while only 105 question are ranked better by Indexal.

9 Conclusion and Future Developments

Our participation to the *Question Answering* track this year was guided by two purposes, mainly concerned with the support to the QA task and not with the QA task itself:

- transforming our previous set of modules into an actual QA system that can be easily used in order to test different ideas;
- integrating in our QA system a search engine that we can tune.

We also improved the *Question/Answer Pairing* and the *Answer Extraction* modules but these ones still have to be perfected as it is proved by our results for the 50 character task.

Among the future developments that we are considering for our next participation in the QA-track are:

- answer unit could be enlarged and position of indexes inside a document could be accounted for in order to focus on the units that gather the largest number of indexes and which are more likely to provide the answer;

- term acquisition could be improved through a disambiguation of long noun phrases and a better part-of-speech tagging of the questions;
- named entity recognition could be improved through machine learning techniques (Baluja, Mittal, and Sukthankar, 1999);
- we achieved some tests about using topic resources for query expansion, with some promizing results that we could not exploit this year;
- although we focused this year on architecture issues, there is still work to do in this area, especially about having coherent linguistic annotations among all our tools, such as in the ATLAS framework (Bird et al., 2000), for example.

References

Baluja, Shumeet, Vibhu O. Mittal, and Rahul Sukthankar. 1999. Applying machine learning for high performance named-entity extraction. In *Proceedings, PACLING'99*, pages 365–378, Waterloo, CA.

Bird, Steven, David Day, John Garofolo, John Henderson, Christophe Laprun, and Mark Liberman. 2000. Atlas: A flexible and extensible architecture for linguistic annotation. In *Second International Conference on Language Resources and Evaluation*, pages 1699–1706.

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

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

Ferret, O., B. Grau, G. Illouz, C. Jacquemin, and N. Masson. 1999. QALC - the Question-Answering program of the Language and Cognition group at LIMSI-CNRS. In *TEXT RETRIEVAL CONFERENCE (TREC-8)*, pages 387–404, Columbia, MD. NIST Special publication.

Grishman, R. and B. Sundheim. 1995. Design of the muc-6 evaluation. In NIST, editor, *the Sixth Message Understanding Conference (MUC-6)*, Columbia, MD. NIST, Morgan-Kauffmann Publisher.

Illouz, G., C. Jacquemin, and B. Habert. 1999. Repérage des entités nommées (REN) dans des transcriptions de documents parole. Technical Report 99-18, LIMSI-CNRS.

Jacquemin, Christian. 1999. Syntagmatic and paradigmatic representations of term variation. In *Proceedings, ACL'99*, pages 341–348, University of Maryland.

Justeson, John S. and Slava M. Katz. 1995. Technical terminology: some linguistic properties and an algorithm for identification in text. *Natural Language Engineering*, 1(1):9–27.

Kozima, H. 1993. Text segmentation based on similarity between words. In *31th Annual Meeting of the Association for Computational Linguistics (Student Session)*, pages 286–288.

Robertson E., Walker S. and M. Beaulieu. 1999. Okapi at trec7: automatic ad hoc, filtering, vlc and interactive. In *Seventh Text Retrieval Conference (TREC-7)*.

Stein, Achim and Helmut Schmid. 1995. Etiquetage morphologique de textes français avec un arbre de décision. *t.a.l.*, 36(1-2):23–36.