First evaluation of Esfinge – a question answering system for Portuguese

Luís Costa
Linguateca at SINTEF ICT
Pb 124 Blindern, 0314 Oslo, Norway
luis.costa at sintef.no

Abstract

In this paper I will start by describing Esfinge - a general domain Portuguese question answering system, and then the strategies I used to participate in the CLEF-2004 QA track. Then I will present and discuss the results obtained and finally describe some of the work planned for the near future.

1 Esfinge

With a question answering system we want, for a given question, that the system be able of returning answers with the help of an information repository. This task requires the processing of the question and of the information repository. Existing systems use in this processing various linguistic resources like taggers, named entities extractors, semantic relations, dictionaries, thesauri, etc…
Esfinge (http://acdc.linguateca.pt/Esfinge/) is based on the architecture described by Eric Brill in (Brill, 2003). Brill tried to check the results that could be obtained by investing less in the resources to process the question and the information repository and more in the volume of the information repository itself. The Web, as the biggest free information repository that we know is a good candidate for these experiences. Brill's approach was never tried for Portuguese and this language is quite used in the Web (Aires & Santos, 2002). The motivation to start developing Esfinge was to check the results that could be obtained by applying Brill's approach to Portuguese.

The planned architecture has four modules:

Question reformulation
N-grams harvesting
N-grams filtering
N-Grams composition

1.1 Question reformulation

In this module, patterns of plausible answers to a given question are obtained. These patterns are based on the words in the question. As an example, for the question: "In which year did Vasco da Gama arrived in India?" a plausible pattern would be "Vasco da Gama arrived in India in".
It's too optimistic to expect the existence of pages with answers in "friendly" formats for all the questions (with the exact format as the result of the question reformulation module). Therefore, patterns of plausible answers with less ambitious strings, like for example the simple conjunction of the question words are also considered. Each one of these patterns is scored according to the probability of helping to find correct answers.
The patterns were initially scored according to my intuition. At the moment the scores range from 1 to 20.

The linguistic information of this module is encapsulated in a text file using the regular expression syntax of the computer programming language Perl. Each triple (question pattern, answer pattern, score) is defined in a line separated by a slash (/).

Here follows a sample of the referred text file (it's actually a simplification for clarity sake):

O que ([^\s?]*) ([^?]*)\??/"$2 $1"/10
O que ([^?]*)\??/$1/1

The first rule says that for a question starting with "O que X Y?" (What X Y?), answers with the pattern "Y X" should be granted the score 10 (since Y and X are enclosed in double quotes, it means this is a phrase pattern – Y must appear just before X). For the question "O que é a MTV?" (What is MTV?), this rule generates the pattern "a MTV é" with the score 10.
The second rule says that for a question starting with "O que X?" (What X?), answers with the pattern X should be granted the score 1. For the question in the previous example, this rule generates the pattern "é" "a" "MTV" with the score 1. Since the words in the pattern are not all enclosed in a pair of double quotes – this means they don't need to appear in this order or even in the same sentence.

1.2 N-grams harvesting

In this module, the resulting patterns of the Question Reformulation module are queried against an information repository. For that purpose they are submitted to a web search engine (for the moment I've been using Google [6]).

In Figure 1 we can see the results of querying the pattern "a antiga capital da Polónia" (the former capital of Poland) in Google.

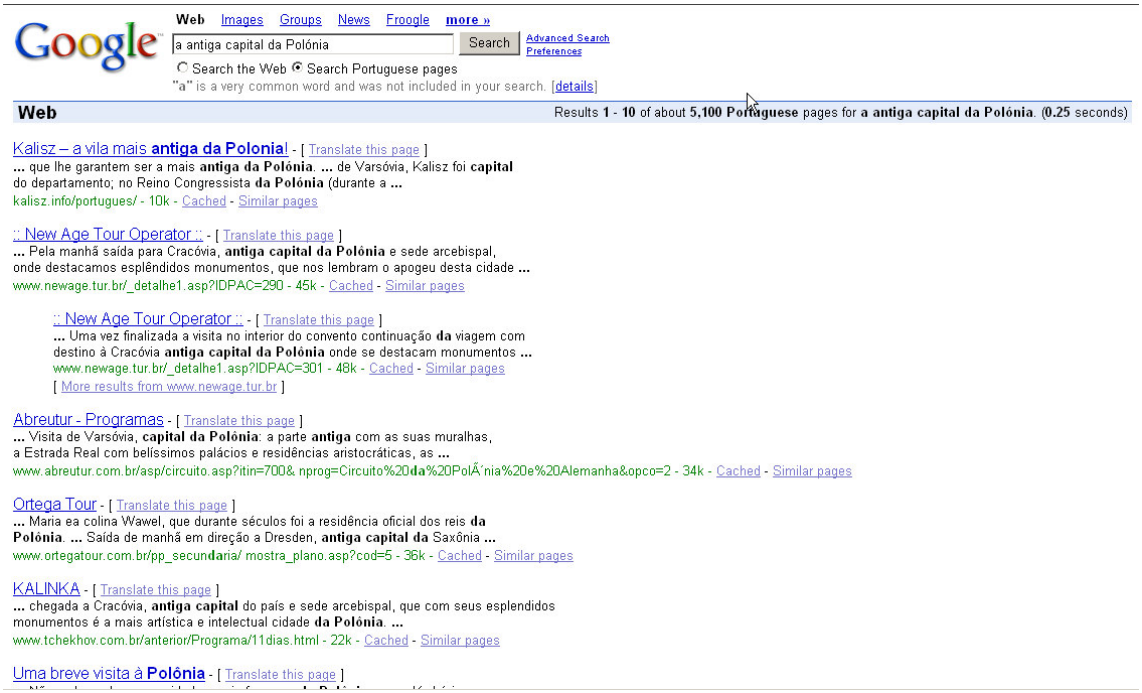

Figure 1

The next step is to extract and measure the frequency of word N-grams from the resulting snippets (I'm considering the first 100 snippets).
I've been using Ngram Statistics Package (NSP) (Banerjee & Pedersen, 2003) for that purpose.

For the 100 first document snippets of the previous query we got the following N-gram distribution (I'm presenting only the 16 most frequent N-grams):

da: 185
a: 99
antiga: 96
capital: 91
de: 78
e: 73
Polônia: 54

capital<>da: 47
do: 46
da<>Polônia: 38
em: 30
antiga<>capital: 30
o: 28
que: 28
com: 26
é: 25

There is some hope that the correct answer will be among the extracted N-grams
Next, these N-grams of different lengths will be scored accordingly to their frequency, length and the scorings of the patterns that originated them.

I'm using the following equation:

N-gram score = $\sum$ (F * S * L) through the first 100 snippets resulting from the web search

where

F = n-gram frequency
S = score of the search pattern which recovered the document
L = n-gram length

1.3 N-grams filtering

This module re-evaluates the scorings obtained in the module "N-grams harvesting". In this module the N-grams will be analysed by their particular features.
To a given question, even if we don't know the answer, we can predict the type of expected answer.
For example:

- A "When?" question implies an answer of type "date". It can be more or less precise: a year (like 1973) or a complete date (like 11/10/1973), but answers like "Lisboa" or "George W. Bush" don't make sense in this context.

- A "How many?" question implies an answer of type "number". Answers like "Oslo" or "5/8/2004" are not acceptable answers.

Analysing the N-Grams about the presence of digits, capitalization and typical patterns may allow reclassifying those N-Grams or even discarding them. The PoS information provided by a morphologic analyser may also be used to enhance the scorings of N-grams with interesting sequences of PoS categories.

1.4 N-grams composition

This module tries to deal with questions with a set of answers, like "Who were the musicians in Queen?". The complete answer to this question demands the composition of the word N-grams "Freddy Mercury", "Brian May", "Roger Taylor", "John Deacon" that can be expected to be among the top scored word N-grams obtained from the three previous modules.
The first task in this module is to determine whether the type of answer is singular (ex: "Who was the first king of Norway?"), plural with a known number of items (ex: "Which are the three largest cities in Portugal?") or plural with an unknown number of items (ex: What are the colours of Japan's flag?").
For the first type this module will return the best scored word N-gram resulting from the previous modules. For the second type it will return the required number of best scored word N-grams (three, in the example above). For the third type, it will need to decide which word N-grams will be part of the answer. This can be done using a threshold that will define which word N-grams will be part of the answer according to their scoring. The proximity of the scoring values can also be used as a decisive factor.

2 Strategies for CLEF 2004

Esfinge is still in its first stages of development, but participating in the CLEF-2004 QA track seemed a good way of evaluating the work done so far, feel some of the difficulties in this IR field and get in touch with the state-of-the-art of actual QA systems and their approaches.
For the QA-CLEF monolingual track, one had to supply, along with each answer, one document in the document collection that supported it. As said above, my system originally used Google's search results and was mainly statistical (tried to use the redundancy existing in the Web), so I knew I would need to add some extra functionalities.
I tested three different strategies. In the first, the system searched the answers in the CLEF document collection (Run1). In the second, it searched the answers in the Web and used the CLEF document to confirm these answers (Run 2). Finally, in the third strategy my system searched the answers in the Web (this one was not submitted to the organization).

2.1 Run 1

The first thing I needed was some way of searching in the document collection. I have some experience in encoding corpora using IMS Workbench (Christ et al., 1999) as well as using its querying capabilities. So, it seemed a good idea to use it to encode the CLEF document collection and to use its querying capabilities to search for desired patterns.
Another important decision to be made was the size of the text unit to be considered when searching for patterns – the entire text of each document, or a passage: a fixed number of sentences or a fixed number of words. I had not a definitive answer for this question, so I chose to do some experiments.
Since the document length seemed too big for a unit, I tried the three following strategies:

- Considered the text unit as 50 contiguous words. This is done dynamically: it is possible to query corpora encoded using IMS Workbench for the context (in terms of words) in which the required patterns co-occur.
- Divided each document into sentences. Those sentences were considered as the text unit. . To segment the document collection into sentences, I used the Perl Module Lingua::PT::Segmentador [7]. The resulting sentences had an average of 28 words per sentence.
- Divided each document into sets of three sentences. Those sets of three sentences were considered as the text unit.

For each question in the QA track, Esfinge did the following steps:

1. Question reformulation

- Submitted the question to the question reformulation module. The result was a set of pairs (answer pattern, score).

2. Passage extraction

- Searched each of these patterns in the document collection and extracted the text units (50 contiguous words, one sentence or three sentences) where the pattern was found. The system discards stop-words without context. For example in the query "a" "antiga" "capital" "da" Polónia", the words "a" and "da" are discarded while in the query "a antiga capital da Polónia" (phrase pattern) they are not discarded. Currently I'm discarding the 22 most frequent words in the CETEMPúblico corpus (Santos & Rocha, 2001). At this stage the system retrieved a set of document passages $\{P_1, P_2 \ldots P_n\}$.

3. N-grams harvesting

- Computed the distribution of word n-grams (from length 1 to length 3) of the document excerpts.

- Ordered the list of word n-grams according to a score based on the frequency, length and scorings of the patterns that originated the document excerpts where the n-grams were found (formula in section 1.2). At this stage the system had an ordered set of possible answers $(A_1, A_2 \ldots A_n)$.

4. N-grams filtering

- The next step was to discard some of these possible answers using a set of filters.

The filters used were:

- First, a filter to discard answers that are contained in the questions. Ex: for the question "Qual é a capital da Rússia" (What is the capital of Russia?), the answer "capital da Rússia" (capital of Russia) is not desired and should be discarded.

- Then, a filter that used the morphologic analyser *jspell* (Simões & Almeida, 2001) to check the PoS of the various words in each answer. The analyser returns a set of possible PoS tags for each word. I erroneously assumed that the order in which the PoS tags were returned was related to their frequency. With that in mind, I was using only the first PoS for each word. Recently I found out that this assumption was wrong. This filter considered some PoS as "interesting": adjectives (adj), common nouns (nc), numbers (card) and proper nouns (np). All answers whose first and final word didn't belong to one of these "interesting" PoS were discarded. It's worthwhile to say that most probably my misinterpretation of the analyser's results led to a poor performance by this filter.

Example:

For the question "Quem é Andy Warhol?" (Who is Andy Warhol?), the system had the following answers among the highest scored:

que
um
de Andy
por
como
pela primeira vez
sua
mais
ou
artista
que Andy
com esta dimensão
segundo andar chamado
cola em garrafa

The morphologic analyser gives the following information:

que: prel
um: art
de Andy: prep np
por: prep
como: con
pela primeira vez: cp nord nc
sua: ppos
mais: pind
ou: con
artista: nc
que Andy: prel np
com esta dimensão: prep pdem nc
segundo andar chamado: nord nc v
cola em garrafa: nc prep nc

After applying the filter the set of highest scored answers will be:

artista: nc
cola em garrafa: nc prep nc

- The final answer will be the candidate answer with the highest score in the set of candidate answers which were not discarded by any of the filters above. If all the answers were discarded by the filters then the final answer is NIL (meaning the system is not able to find an answer in the document collection).

2.2 Run 2

It was possible to send two sets of results to the organization. I wanted to do some experiments using also the Web as source since that's the line of work where I expect to get better results.
For that purpose I selected one of the previous experiences to send to the organization (my run1): the one considering sets of three sentences as the text unit because it seemed the one with best results, even though the results were quite similar in all three experiments.

The next experiment used the strategy described in (Brill et al., 2001).
First, it looked for answers in the Web, and then tried to find documents in the document collection supporting those answers. It submitted the patterns obtained in the question reformulation module to Google. Then the document snippets {S_1, S_2 … S_n) were extracted from Google's results pages. These snippets are usually composed by fragments of the different sentences in the recovered documents that contain the query words and have approximately 25 words.

The next step was to compute the distribution of word n-grams (from length 1 to length 3) existing in this document snippets. From this point the algorithm followed the one described as run 1, with an extra filter in the N-grams filtering module: a filter that searched the document collection for documents supporting the answer – containing both the candidate answer and a pattern obtained from the question reformulation module. This filter is necessary because it was stated in the task guidelines that the system should return the code of a document supporting each answer.

2.2.1 Brazilian Portuguese. A problem?

Using texts in Brazilian web pages may enlarge the corpus the system uses to find answers, but may also bring some problems. The system may return an answer in the Brazilian variant which is not possible to support in the document collection, which was built with newspaper texts written in European Portuguese.

Example: For the question "Qual é a capital da Rússia?" (What is the capital of Russia?), my system returned the answer "Moscou" (in the Brazilian variant). It would be much easier to support the answer "Moscovo" (same word in the European variant).

Another problem may occur when the scoring gets diluted by the two variants (like "Moscou" and "Moscovo" in the example), thus allowing other answers to get better scores. Searching only in Portuguese pages can obviate this problem, but will diminish the corpus to search into.

Yet another example can be illustrated by the query in section 1.2: "a antiga capital da Polónia". Even though, using the word "Polónia" (Portuguese variant) in the query, this word is not on the top 10 of harvested n-grams. On the other hand, "Polônia" (in the Brazilian variant) is third placed on the n-gram ranking. The reason for this is that Google doesn't differentiate between accentuated and non-accentuated characters, so the characters "ó", "ô" and "o" are exactly the same thing to this search engine. This can be a serious problem, when one is processing a language with the variety and heavy use of accentuation as present in Portuguese. One way to obviate this problem is to develop a post-Google filter to discard non-interesting documents, thus overcoming Google's limitations regarding the Portuguese.

2.3 Web-only experiment

For the present paper, I did an extra "run" using the Web as document collection and without crosschecking the answers in CLEF's document collection. I thought this experience could give some insight on whether there are advantages in combining two different information sources (Web and CLEF's document collection) or whether one can get better results using only one of these information sources.

3 Results

3.1 Results by type of question

	# questions	# right (run 1)	% right (run1)	# right (run 2)	% right (run2)	# right (Web-only)	% right (Web-only)
Quem (Who)	53	8	15,1 %	9	17 %	3	5,7 %
Qual (Which)	34	8	23,5 %	6	17,6 %	2	5,9 %
Onde (Where)	24	1	4,2 %	5	20,8 %	3	12,5 %
O que (What)	18	0	0 %	2	11,1 %	1	5,6 %
Em que (In which)	15	0	0 %	2	13,3 %	0	0 %
Quanto(a)s (How many)	13	2	15,4 %	3	23,1 %	1	7,7 %
Como (How)	9	0	0 %	0	0 %	0	0 %
Que (What, Which)	9	2	22,2 %	2	22,2 %	1	11,1 %
Quando (When)	9	0	0 %	0	0 %	0	0 %
De que (Of what, which)	7	0	0 %	0	0 %	0	0 %
A que (To which, what)	3	0	0 %	0	0 %	0	0 %
Mencione, Nomeie, Indique (Name)	4	1	25 %	1	25 %	0	0 %
X ... em que (... in which)	1	0	0 %	0	0 %	0	0 %
Total	199	22	11,1 %	30	15,1 %	11	5,5 %

Table 1

In Table 1 we can see that the results in run2 (the one which used the Web crosschecking the results in the document collection) are slightly better. However we can also see that the type of question is not irrelevant to the results. For example run1 had better results for questions of type "Qual" (Which). There are also some relatively frequent questions types without any right answer in either of the runs (like "Como", "Quando", "De que"). This probably means that there is something in these types of questions that Esfinge doesn't deal properly in the answer-finding procedure.

Both run1 and run2 were evaluated by the organization. To evaluate my Web-only experience I needed to know the right answers. For that purpose I created a list with my "right answers" to the question set.
The Web-only experience is in some aspects a different task from the one proposed in CLEF. For example it was stated in CLEF's guidelines [9] that some questions might have no answer in the document collection (NIL answer), but it's much more difficult to say such thing when using the Web as the document collection. For that reason I considered not answered questions as wrong when evaluating this experience. Since my system was not recording the addresses of the documents it used to get the answers in the Web, it was not possible to check if the answers were supported or not.

Globally, we can see that the best results were obtained combining the use of the document collection and the Web. The worst results are the ones obtained using solely the Web. It is somehow surprising that the results using solely the document collection are better than the ones using solely the Web, since the approach I'm trying to test was designed to take advantage of the redundancy in larger corpora. Possible explanations for this are:

- My system is not extracting efficiently text from the Web. Possibly it is getting control symbols and documents in other languages - according to Nuno Cardoso (p.c.), it is common for search engines to mistake UTF for iso8859-1 character encoding. Also, the snippets resulting from the search engine are most probably not good enough to extract good answers, since most of those snippets are formed by truncated sentences.

- Some documents in the Web, rather than helping to find answers, do the exact opposite (jokes, blogs, …).

- The text size unit of 3 sentences ≈ 90 words gives a larger context, while many Google snippets do not even include all the words in the query.

3.2 Results by question length

	# questions	# right (run 1)	% right (run1)	# right (run 2)	% right (run2)
3 words	8	1	12,5 %	3	37,5 %
4 words	27	3	11,1 %	2	7,4 %
5 words	37	1	2,7 %	6	16,2 %
6 words	37	4	10,8 %	6	16,2 %
7 words	26	4	15,4 %	3	11,5 %
8 words	32	4	12,5 %	3	9,4 %
9 words	15	1	6,7 %	2	13,3 %
10 words	8	1	12,5 %	2	25 %
11 words	2	1	50 %	1	50 %
12 words	2	1	50 %	1	50 %
13 words	4	1	25 %	1	25 %
16 words	1	0	0 %	0	0 %
Total	199	22	11,1%	30	15,1%

Table 2

In table 2, I try to study the influence of the question length in the results of "run1" and "run2".
In order to determine the length of the questions, I used the Perl Module Lingua::PT::Atomizador [7] to tokenize the questions..
In "run1" the most significant results are obtained in questions from length 6 to 8, while in "run2" the system gets better results in questions from length 5 to 6. This difference can be explained by the different length of the passages recovered from the Web and from the document collection. The passages recovered from the Web being shorter, may be more suited to answer shorter questions, while the passages recovered from the document collection being longer, needs the questions to be longer in order to get the appropriate context.

3.3 Results considering 5 answers per question

	# questions	# right (run 1)	% right (run1)	# right (run 2)	% right (run2)
Quem (Who)	53	15	28,3 %	11	20,8 %
Qual (Which)	34	9	26,5 %	7	20,6 %
Onde (Where)	24	2	8,3 %	6	25 %
O que (What)	18	0	0 %	2	11,1 %
Em que (In which)	15	0	0 %	3	20 %
Quanto(a)s (How many)	13	2	15,4 %	4	30,8 %
Como (How)	9	0	0 %	0	0 %
Que (What, Which)	9	3	33,3 %	2	22,2 %
Quando (When)	9	0	0 %	1	11,1 %
De que (Of what, which)	7	0	0 %	0	0 %
A que (To which, what)	3	0	0 %	0	0 %
Mencione, Nomeie, Indique (Name)	4	1	25 %	1	25 %
X ... em que (... in which)	1	0	0 %	0	0 %
Total	199	32 (+10)	16,1 %	37 (+7)	18,6 %

Table 3

The system can be configured to give more than one answer to a question. It returns these answers ordered by their probability of being the right answer. In this table a question is considered rightly answered if a right answer is found in the top 5 answers returned by the system.

To do this evaluation I needed also to know if the answers were supported in the document collection. For that purpose I included in my list of "right answers" the list of questions to which I couldn't find any answer supported by the document collection (NIL answers).
The results are obviously better than the ones present in the previous table, but not dramatically. This suggests that most problems are located before the scoring of the candidate answers.

3.4 Causes for wrong answers

	# wrong answers in first 30 questions (run 2)	% wrong answers in first 30 questions (run2)
Failure in document recovery	9	30 %
Filter "discard answers contained in questions"	2	6,7 %
Filter "interesting PoS"	2	6,7 %
Filter "documents supporting answer"	7	23,3 %
Answer scoring algorithm	8	26,7 %
Answer length >3	2	6,7 %

Table 4

In the table above I tried to find out why the system produces wrong answers. To find the causes takes some time, so I started with the run with best results (run 2) and did the evaluation only for the first 30 questions of the question set. For some questions I counted more than one reason for failure. This sort of evaluation can give some insight into the system modules that are causing more errors and therefore should be looked into more in detail.

4 Future work

4.1 Development of Esfinge

4.1.1 Question reformulation

In this module the linguistic information is encapsulated in a text file using Perl's regular expression syntax. This syntax is quite powerful, however it is much more suited to the thought processes of computer-scientists than to linguists' ones. In case it is intended to include professionals in that area to improve the question reformulation patterns in a more advanced stage of development, it would be better to use a friendlier syntax.

4.1.2 N-grams harvesting

There are planned experiences about extracting word N-grams not from the snippets returned by the search engine, but from the actual pages. Other planned experiences are related to the type of web pages to be considered: only European Portuguese pages, pages written in other languages, only news sites…

4.1.3 Machine learning techniques

An interesting experience/refinement that is planned is to use a set of questions associated with their answers as a training set for the system
The results of the system on the training set questions can be compared with the correct answers. The scorings of the patterns and/or the word n-grams can then be changed and the system executed again against the training set, the new results compared with the right answers and the results checked again to understand if the system is improving.

4.2 Further evaluation of Esfinge

I plan to use a multitude of sources to further evaluate Esfinge:

- The questions and answers created by QA@CLEF
- A set of real questions and answers found on the web, created by humans, using several distinct methods for collecting them (oráculo)
- A set of questions posed by real users (from Esfinge's logs)
- A set of questions with answers, created and validated by myself

Acknowledgements

I thank my colleague Diana Santos (Linguateca / SINTEF) for all the valuable suggestions and for helping me to write this paper in a much more understandable way than it was written in the preliminary versions. I thank Nuno Cardoso (Linguateca/XLDB) for the final revision of this paper. I thank Alberto Simões (Linguateca/Universidade do Minho) for the hints on using the Perl Modules "jspell" [11], "Lingua::PT:: Atomizador" [7] and "Lingua::PT::Segmentador" [7]. I also thank the Fundação para a Ciência e Tecnologia for the grant POSI/PLP/43931/2001, co-financed by POSI.

References

[1] Aires, Rachel & Diana Santos. "Measuring the Web in Portuguese". In *Euroweb 2002 conference* . Oxford, UK, 17-18 December 2002, http://www.linguateca.pt/Diana/download/AiresSantosEuroWeb2002.html

[2] Banerjee , Satanjeev & Ted Pedersen. "The Design, Implementation, and Use of the {N}gram {S}tatistic {P}ackage". In Proceedings of the Fourth International Conference on Intelligent Text Processing and Computational Linguistics, pp. 370--381, February 2003, Mexico City

[3] Brill, Eric. "Processing Natural Language without Natural Language Processing", in A. Gelbukh (ed.), *CICLing 2003,* LNCS 2588, Springer-Verlag Berlin Heidelberg, 2003, pp. 360-9.

[4] Brill, Eric, Jimmy Lin, Michele Banko, Susan Dumais & Andrew Ng."Data-Intensive Question Answering", In E.M. Voorhees & D.K. Harman (eds.), Information Technology: The Tenth Text Retrieval Conference, TREC 2001. NIST Special Publication 500-250, pp. 393-400.

[5] Christ, O., Schulze, B. M., Hofmann, A., & Koenig, E. (1999). The IMS Corpus Workbench: Corpus Query Processor (CQP): User's Manual. University of Stuttgart, March 8, 1999 (CQP V2.2).

[6] Google Help Central. http://www.google.com/help/index.html

[7] "Lingua::PT:: Atomizador", "Lingua::PT::Segmentador" , http://linguateca.di.uminho.pt/cvsinfo/modules.html, http://search.cpan.org/dist/Lingua-PT-Atomizador/, http://search.cpan.org/dist/Lingua-PT-Segmentador/

[8] Magnini B., S. Romagnoli, A. Vallin, J. Herrera, A. Peñas, V. Peinado, F. Verdejo, M. de Rijke, The Multiple Language Question Answering Track at CLEF 2003 , in Carol Peters, editor, Working Notes for the CLEF 2003 Workshop, 21-22 August, Trondheim, Norway, 2003.

[9] QA@CLEF-2004 Guidelines. http://clef-qa.itc.it/2004/guidelines.html

[10] Santos, Diana & Paulo Rocha. "Evaluating CETEMPúblico, a free resource for Portuguese". In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*, pp. 442-449. Toulouse, 9-11 July 2001

[11] Simões, Alberto Manuel & José João Almeida. "Jspell.pm - um módulo de análise morfológica para uso em Processamento de Linguagem Natural". In Anabela Gonçalves & Clara Nunes Correia (eds.), *Actas do XVII Encontro da Associação Portuguesa de Linguística (APL 2001)*, pp. 485-495. Lisboa: APL. Lisboa, 2-4 Outubro 2001