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. This paper starts by describing Esfinge, a general domain Portuguese question answering system that uses the redundancy available in the Web as an important resource to find its answers. The paper also presents the strategies employed to participate in CLEF-2004 and discusses the results obtained. Three different strategies were tested: searching the answers only in the CLEF document collection, searching the answers in the Web and using the CLEF document collection to confirm these answers and finally searching the answers only in the Web. The intriguing question of why the system performed better when joining the two information sources, even though it was designed for the Web is discussed; in this connection, different language varieties and some problems of Google are mentioned. The paper concludes describing some of the work planned for the near future.

1 What Is Esfinge?

For a given question a question answering system returns answers with the help of an information repository. This task requires the processing of the question and of the information repository. Existing systems use various linguistic resources like taggers, named entities extractors, semantic relations, dictionaries, thesauri, etc. to do this.

Esfinge (http://acdc.linguateca.pt/Esfinge/) is based on the architecture proposed by Eric Brill [1]. 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, being the biggest free information repository that we know, is the best candidate for these experiments. Brill's approach was never tried for Portuguese and this language is quite used in the Web [2]. The motivation to start developing Esfinge was to check the results that could be obtained by applying Brill's approach to Portuguese.

Brill's architecture has four modules:

1. Question reformulation
2. N-grams harvesting
3. N-grams filtering
4. 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. For example, a plausible pattern for the question *In which year did Vasco da Gama arrived in India?* would be *Vasco da Gama arrived in India in*.

It is 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 how good it can help to find correct answers. The patterns were initially scored according to my intuition with scores ranging from 1 to 20.

The linguistic information of this module is encapsulated in a text file using the regular expression syntax of the Perl programming language. Each triple (question pattern, answer pattern, score) is defined in a line separated by a slash (/). Follows a sample of the referred text file (simplified for clarity's sake).

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

The rule states that, for a question starting with *O que X Y?* (*What X Y?*), answers with the pattern *"Y X"* should be granted a score of 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.

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 (Google[1] for the moment).

The next step is to extract and measure the frequency of word N-grams from the resulting snippets (considering the first 100 snippets), using the Ngram Statistics Package (NSP) [3] for that purpose.

For example, from the query *"a antiga capital da Polónia"* (*the former capital of Poland*), one gets the following N-gram distribution (16 most frequent N-grams):

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

[1] http://www.google.com/help/index.html

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

The correct answer is expected to 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, 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 N-grams harvesting module, analysing the N-grams' particular features.

For some questions, even if we do not 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, for instance a year (like *1973*) or an extended date (like *11/10/1973*), but such answers as *Lisboa* or *George W. Bush* do not make any sense in this context.

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

In analysing the N-grams as regards the presence of digits, capitalization and typical patterns may allow to reclassify those N-grams or even discarding them. Also, the PoS information provided by a morphologic analyser or tagger may be used to enhance the scorings of N-grams with interesting sequences of PoS categories.

1.4 N-grams Composition

This module tries to cope 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* and *John Deacon*, that

can be expected 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

Although Esfinge is still in its early stages of development, participating in the CLEF-2004 QA track seemed a good way of evaluating the work done so far, experimenting some of the difficulties in this field and getting 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, the ID of one document in the document collection that supported it. As said above, Esfinge 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 one, the system searched the answers in the CLEF document collection (Run 1). In the second one, it searched the answers in the Web and used the CLEF document collection to confirm these answers (Run 2). Finally, in the third strategy Esfinge searched the answers only 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 Corpus Workbench [4] as well as using its query capabilities. So, it seemed a good idea to use it to encode the CLEF document collection and to use its query capabilities to search for desired patterns.

Another important decision concerned the size of the text unit to be searched for patterns, i.e. whether to consider the entire text of each document or only a passage. 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:

1. Considering 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.

2. Dividing 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::PLNbase freely available at CPAN. The collection had in average 28 words per sentence.

3. Dividing 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 proceeded by the following steps:

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

Passage extraction. Searching each of these patterns in the document collection and extracting 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 discard the 22 most frequent words in the CETEMPúblico corpus [5]. At this stage the system retrieved a set of document passages {P1, P2 … Pn}.

N-grams harvesting. Computing the distribution of word N-grams (from length 1 to length 3) of the document excerpts. Ordering 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, computed using the formula above. At this stage, the system has an ordered set of possible answers {A1, A2 … An}.

N-grams filtering. Discarding some of these possible answers using a set of filters, namely:

- 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* [6] to check the PoS of the various words in each answer. The analyser returns a set of possible PoS tags for each word. 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 did not belong to one of these "interesting" PoS were dis-

carded. Example: before this filter, the highest scored answers for the question *Quem é Andy Warhol?* (*Who is Andy Warhol?*) were:

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 are:

artista: nc
cola em garrafa: nc prep nc

For the CLEF runs, I erroneously assumed that the order in which the PoS tags were returned was related to their frequency. With that in mind, I used only the first PoS for each word. Recently, I found out that this assumption was wrong. It is fair to say that most probably my misinterpretation of the analyser's results led to a poor performance of this filter.

The final answer was 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 was NIL (meaning the system is not able to find an answer in the document collection).

From the three previous experiments, I selected to send to the organization the one considering sets of three sentences as the text unit, because it seemed the one with (slightly) best results.

2.2 Run 2

Since it was possible to send two sets of results to the organization, I did some experiments using also the Web as source since that is the line of work where I expect to get better results.

The next experiment used the strategy described in another paper by Brill [7]. 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 $\{S_1, S_2 \ldots S_n\}$ were extracted from Google's results pages. These snippets are usually composed by frag-

ments 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 in 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.

2.3 Brazilian Portuguese. A Problem?

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

For example, for the question *Qual é a capital da Rússia?* (*What is the capital of Russia?*), the system returned the answer *Moscou* (in the Brazilian variant). Since we were checking in a European Portuguese collection, 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 pages published in Portugal can obviate this problem, but will diminish the corpus to search into.

Yet another example can be illustrated by the query: "a antiga capital da Polónia" presented above. 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 does not differentiate between accentuated and non-accentuated characters, so the characters *ó*, *ô* and *o* are considered exactly the same thing by this search engine. This can be a serious problem when one is processing a language with the variety and heavy use of accentuation as Portuguese. One way to solve this problem is to develop a post-Google filter to discard non-interesting documents, thus overcoming Google's limitations regarding Portuguese.

2.4 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 experiment 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

Table 1. Results by type of question

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

Table 1 shows that the results in Run 2 (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, Run 1 had better results for questions of type "Qual" (Which). There are also some relatively frequent question types without any right answer in either run (like "Como", "Quando", "De que"). This probably means that there is something in these types of questions which Esfinge does not deal properly within the answer-finding procedure.

Both Run 1 and Run 2 were evaluated by the organization. The Web-only experience is in some aspects a different task from the one proposed in CLEF. For example, CLEF's guidelines [8] stated that some questions might have no answer in the document collection (NIL answer), but it is much more difficult to say such thing when using the Web as the document collection. For this reason, I considered not answered questions as wrong when evaluating this experience. Since Esfinge was not recording the addresses of the documents it used to get the answers in the Web, it was not possible to check whether 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 am testing was designed to take advantage of the redundancy in larger corpora. Possible explanations for this are:

- Esfinge 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.

- Some documents in the Web, rather than helping to find answers, do the exact opposite (jokes, blogs, …). Discarding some kinds of pages could be of help [9].

- 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.

Table 2. Results by question length

# words in question	# questions	#right (Run 1)	#right (Run 2)
3 words	8	1	3
4 words	27	3	2
5 words	37	1	6
6 words	37	4	6
7 words	26	4	3
8 words	32	4	3
9 words	15	1	2
10 words	8	1	2
11 words	2	1	1
12 words	2	1	1
13 words	4	1	1
16 words	1	0	0
Total	199	22	30

Table 2 displays the influence of the question length in the results of Run 1 and Run 2.

In order to determine the length of the questions, I used the Perl Module Lingua::PT::PLNbase to tokenize the questions.

In Run 1 the most significant results are obtained in questions from length 6 to 8, while in Run 2 the system gets better results in questions from length 5 to 6. This slight difference can be explained by the different length of the passages recovered from the Web and from the document collection. These passages contain the question patterns and hopefully the answers. Being the passages recovered from the Web shorter, they may be more suitable for shorter questions, while passages retrieved from the document collection are usually longer, therefore more suitable to answer longer questions, as the following examples show:

- It is more likely to find the question pattern and an answer to the question *What is the name of the widow of Samora Machel, the deceased Mozambican president?* in a three sentence context than in a Google snippet.

- Conversely, extracting N-grams related to the question *Who is Christo?* in a three sentence context can provide too many N-grams, making the task of finding the right answer very difficult.

It would be interesting to do a similar study regarding the answer length, since the question and answer lengths are not directly related. One can have a long question with a short answer and vice versa. Classifying the answers is, however, more problematic, since a question may have a short and a long answer and both can be considered correct. For CLEF, Esfinge extracted only up to trigrams, so the system was unable to answer correctly questions which required an answer longer than 3 words. Such limitation was due to efficiency constraints: longer N-grams require longer processing time and I assumed that for most of the questions, a three word answer would suffice.

Table 3. Causes for wrong answers

Problem in...	#wrong Answers (Run 2)	%wrong (Run 2)
Document recovery	86	43 %
Filter "discard answers contained in questions"	8	4 %
Filter "interesting PoS"	20	10 %
Filter "documents supporting answer"	23	12 %
Answer scoring algorithm	75	37 %
Answer length >3	21	11 %

A log file was used to find out why the system produces wrong answers. In this file was possible to check an ordered list (best scored first) of all the word N-grams analyzed for each question. The reasons why they were discarded or not is also registered in this file. In any case, this evaluation takes some time, so I started with the run with best results (Run 2). For some questions I counted more than one reason for failure.
Table 3 provides a detailed error analysis. 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

The results gathered in table 3 (Causes for wrong answers) show that the main problems in Esfinge at the moment are in the document recovery and in the answer scoring algorithm stages. Now, if the first component (document recovery) is not working properly, it is very difficult to evaluate the other components of the system.

With that in mind, work in Esfinge will mainly address the two following areas in the near future:

1. Checking the questions with wrong answers due to "Document recovery", grouping then by their type (ex: Quem/Who, Qual/Which, Onde/Where). Understanding why the patterns used for the document recovery are not recovering the right documents. Changing the patterns, and testing the new patterns with the questions of a particular type (usually a pattern is closely related to a particular type of question).

2. Using the log file, I will compute a frequency list of all the solutions provided by Esfinge to the CLEF QA track questions (not only the best answer, but all the answers that managed to go through all system's filters). With this frequency list and some common sense, I plan to build a list of 'undesired answers' that will be used in an extra filter. The words in this list will be frequent words that do not really answer questions in isolation (like anos/years, mesmo /same, dia/day, maior /bigger, tempo/time).

4.1 Other Improvements

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 we intend to include professionals in that area to improve the question reformulation patterns at a more advanced stage of development, it would be better to use a friendlier syntax. As an example, the patterns could be automatically generated from real examples of questions and answers.

N-grams Harvesting. I plan to experiment 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…

Machine Learning Techniques. An interesting experiment/refinement 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;
- A set of questions posed by real users (from Esfinge's logs);
- A set of questions with answers, created and validated by myself.

5 Acknowledgements

I thank my colleague Diana Santos for all the valuable suggestions and for helping me to write this paper in a more understandable way. I also thank Nuno Cardoso for revisions on previous versions of this paper and Alberto Simões for the hints on using the Perl Modules "jspell" [6] and "Lingua::PT::PLNbase". This work is financed by the Portuguese Fundação para a Ciência e Tecnologia through grant POSI/PLP/43931/2001, co-financed by POSI.

References

1. Brill, E.: Processing Natural Language without Natural Language Processing. In: Gelbukh, A. (ed.): CICLing 2003. LNCS 2588. Springer-Verlag Berlin Heidelberg (2003) 360-9
2. Aires, R. & Santos, D.: Measuring the Web in Portuguese. In: Euroweb 2002 conference (Oxford, UK, 17-18 December 2002) 198-199
3. Banerjee, S. & Pedersen, T.: 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 (Mexico City, February 2003) 370-381
4. Christ, O., Schulze, B.M., Hofmann, A. & Koenig, E.: The IMS Corpus Workbench: Corpus Query Processor (CQP): User's Manual. University of Stuttgart, March 8, 1999 (CQP V2.2)
5. Santos, D. & Rocha, P.: "Evaluating CETEMPúblico, a free resource for Portuguese". In: Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (Toulouse, 9-11 July 2001) 442-449
6. Simões, A. M. & Almeida, J.J.: Jspell.pm - um módulo de análise morfológica para uso em Processamento de Linguagem Natural. In: Gonçalves, A. & Correia, C.N. (eds.): Actas do XVII Encontro da Associação Portuguesa de Linguística (APL 2001) (Lisboa, 2-4 Outubro 2001). APL Lisboa (2002) 485-495
7. Brill, E., Lin, J., Banko, M., Dumais, S. & Ng, A.: Data-Intensive Question Answering. In: Voorhees, E.M. & Harman, D.K. (eds.): Information Technology: The Tenth Text Retrieval Conference, TREC 2001. NIST Special Publication 500-250. 393-400
8. Magnini et al.: "Overview of the CLEF 2004 Multilingual Question answering track". This volume.
9. Aires, R., Manfrin, A., Aluísio, S.M. & Santos, D.: What Is My Style? Stylistic features in Portuguese web pages according to IR users' needs. In: Lino, M.T., Xavier, M.F., Ferreira, F., Costa, R. & Silva, R. (eds.): Proceedings of LREC 2004 (Lisboa, Portugal, 26-28 May 2004) 1943-1946