Monolingual and Cross-language QA using a QA-oriented Passage Retrieval System

José Manuel Gómez Soriano, Empar Bisbal Asensi, Davide Buscaldi,
Paolo Rosso, Emilio Sanchis Arnal
Dpto. de Sistemas Informáticos y Computación (DSIC),
Universidad Politécnica de Valencia, Spain
{jogomez, ebisbal, dbuscaldi, prosso, esanchis}@dsic.upv.es

August 20, 2005

Abstract

This report describes the work done by the RFIA group at the Departamento de Sistemas Informáticos y Computación of the Universidad Politécnica of Valencia for the 2005 edition of the CLEF Question Answering task. We participated in three monolingual tasks: Spanish, Italian and French, and in two cross-language tasks: spanish to english and english to spanish. Since this was our first participation, we focused our work on the passage-based search engine while using simple pattern matching rules for the Answer Extraction phase. As regards the cross-language tasks, we had resort to the most common web translation tools.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: H.3.3 Information Search and Retrieval; H.3.4 Systems and Software

General Terms

Measurement, Performance, Experimentation

Keywords

Question Answering, Passage Retrieval, Query Classification, Answer Extraction

1 Introduction

The basic functionality of a Question Answering (QA) system is to allow a user to question in natural language a non-structured document collection in order to look for the correct answer. In the case of the cross-language task the collection is constituted by documents written in a language different from the one used in the query, which increases the task difficulty.

A QA system can be divided, usually, into three main modules: Question Classification (QC), document or Passage Retrieval (PR) and Answer Extraction (AE). The aim of the first module is to recognize the type or category of the expected answer (e.g. if it is a Person, Quantity, Date, etc.) from the user question. The second module obtains the passages (or pieces of text) which contain the terms of the question. Finally, the answer extraction module uses the information collected by the previous modules in order to extract the correct answer. Sometimes the QC module can provide the other modules with additional information extracted from the query. In such cases the module can be named Question Analyzer.

The most relevant part of our work is made up by the Passage Retrieval system, specifically oriented to the QA task, whereas most QA systems use classical PR methods [1, 2, 3, 4]. Our PR method is also language independent, because the question and passage processing phases do not use any knowledge about the lexicon and the syntax of the corresponding language. A SVM approach combined with pattern rules has been used for the QC module. Due to the fact that this was our first participation to the CLEF QA task, the AE module was developed using simple pattern-matching rules, and therefore resulted to be somehow coarse, due both to the small number of question categories and to the lack of time to define all the needed patterns.

2 Description of QA System

The architecture of our QA system is shown in Fig.1.

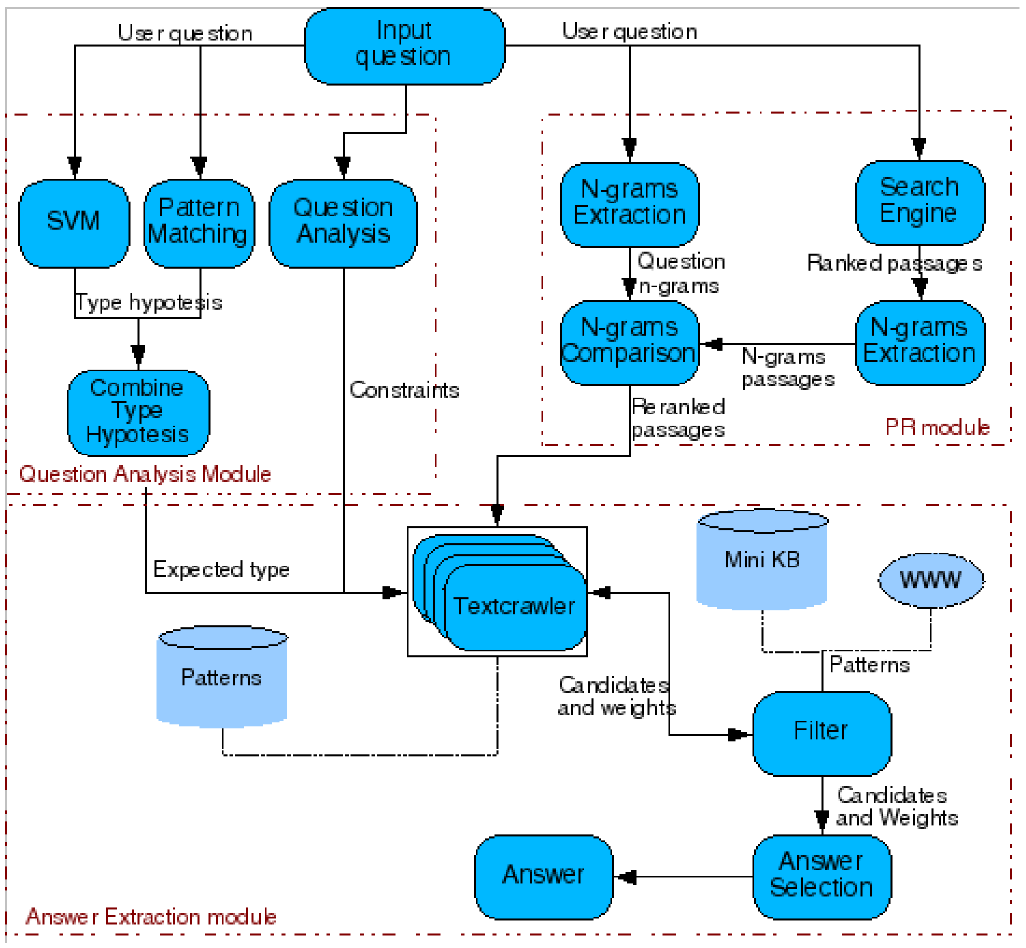

Figure 1: Main diagram of the QA system

Given a user question, this will be handed over to the *Question Analysis* (in our case it does not only classify the questions, but extracts also some constraints to be used in the Answer Extraction phase) and *Passage Retrieval* modules. Next, the *Answer Extraction* obtains the answer from the expected type, constraints and passages returned by *Question Analysis* and *Passage Retrieval* modules.

2.1 Question Analysis

The main objective of this module is to obtain the expected answer type from the question. This is a crucial step of the processing since the Answer Extraction module uses a different strategy depending on the expected answer type, and errors in this phase account for the 36.4% of the total number of errors in Question Answering as reported by Moldovan et al. [5]. The different answer types that can be treated by our system are shown in Table 1.

A SVM classifier trained over a corpus of $1,393$ questions in English and Spanish from the past TREC[1] QA test sets has been coupled with a simple pattern-based classifier. The answer of both classifiers are evaluated by a sub-module that selects the most specific category between the ones returned by the classifiers. For instance, the answer extraction module applies a specialized strategy if the expected type of the answer is "COUNTRY", that is a sub-category of "LOCATION". The patterns are organized in a 3-levels hierarchy, where each category is defined by one or more

L0	L1	L2
NAME	ACRONYM	
	PERSON	
	TITLE	
	LOCATION	COUNTRY
		CITY
		GEOGRAPHICAL
DEFINITION		
DATE	DAY	
	MONTH	
	YEAR	
	WEEKDAY	
QUANTITY	MONEY	
	DIMENSION	
	AGE	

Table 1: QC pattern classification categories.

patterns written as regular expressions. For instance, the Italian patterns for the category "*city*" are: `.*(che|quale) .*citt\'a .+` and `(qual|quale) .*la capitale .+`. The questions that do not match any defined pattern are labeled with *OTHER*. The QC system based on patterns was used stand-alone for both Italian and French, because of the unavailability of corpora for these languages.

Together with the usual Query Classification task, the module analyzes the query with the purpose of identifying the constraints to be used in the Answer Extraction (AE) phase. These constraints are made by sequences of words extracted from the POS-tagged query by means of POS patterns and rules. For instance, any sequence of nouns (such as "ozone hole") is considered as a relevant pattern. The POS-taggers used were the SVMtool[2] for English and Spanish, and the TreeTagger[3] for Italian and French.

We distinguish two classes of constraints: a *target* constraint, which can be considered the object of the question, and zero or more *contextual* constraints, keeping the information that has to be included in the retrieved passage in order to have a chance of success in extracting the correct answer. For example, in the following question: *"How many inhabitants were there in Sweden in 1989?" inhabitants* is the target constraint, while *Sweden* and *1989* are the contextual constraints. There is always only one target constraint for each question, but the number of contextual constraint is not fixed. For instance, in *"Who is Jorge Amado?"* the target constraint is *Jorge Amado* but there are no contextual constraints.

[1]http://trec.nist.gov
[2]http://www.lsi.upc.edu/ nlp/SVMTool/
[3]http://www.ims.uni-stuttgart.de/projekte/ corplex/TreeTagger/DecisionTreeTagger.html

In the case of the Cross-language task, the module works over an *optimal* translation of the input query. Four translations are obtained through the following web tools: *Google*[4], *Systran*[5], *Babelfish*[6] and *Freetrans*[7]. For each translation a *trigram chain* is obtained in the following way: let $w = (w_1, \ldots, w_n)$ be the sequence of the words in the translation, then a trigram chain is a set of trigrams $T = \{(w_1, w_2, w_3), (w_2, w_3, w_4), \ldots (w_{n-2}, w_{n-1}, w_n)\}$. Then each of the trigrams $t \in T$ is submitted to a web search engine (we opted for MSN Search[8]) as a string: "$w_i\ w_{i+1}\ w_{i+2}$", obtaining the web count $c(t)$ of that trigram. The weight of each trigram chain (and therefore of the corresponding translation) is obtained by means of Formula 1.

$$W(T) = \prod_{t \in T} \hat{c}(t) \;\; where \;\; \hat{c}(t) = \begin{cases} \log c(t) & c(t) > 1 \\ 0.1 & c(t) \leq 1 \end{cases} \tag{1}$$

The optimal translation is the one with the highest trigram chain weight.

2.2 Passage Retrieval

The user question is handed over also to the *Search Engine* and *N-grams Extraction* modules. Passages with the relevant terms (i.e., without stopwords) are found by the *Search Engine* using the classical IR system. Sets of unigrams, bigrams, ..., n-grams are extracted from the extended passages and from the user question. In both cases, n will be the number of question terms.

With the n-gram sets of the passages and the user question we will make a comparison in order to obtain the weight of each passage. The weight of a passage will be heavier if the passage contains greater n-gram structures of the question.

For instance, if we ask "*Who is the President of Mexico?*" the system could retrieve two passages: one with the expression "*... **Vicente Fox** is the President of Mexico...*", and the other one with the expression "*...**Carlo Azeglio Ciampi** is the President of Italy...*". Of course, the first passage must have more importance because it contains the 5-gram "*is the President of Mexico*", whereas the second passage only contains the 4-gram "*is the President of*", since the "*is the President of Italy*" 5-gram is not in the original question. To calculate the weight of n-grams of every passage, first the greatest relevance of n-gram in the passage is identify and we assign to this a weight equal to the sum of all term weights. Next, other n-grams less relevant are searched. These n-grams are not composed by terms of found n-brams. The weight of these n-grams will be the sum of all their weight terms divided by two. The weight of every term comes fixed by (2):

$$w_k = 1 - \frac{\log(n_k)}{1 + \log(N)} \quad . \tag{2}$$

Where n_k is the number of passages in which the associated term to the weight w_k appears and N is the number of system passages. We make the assumption that stopwords occur in every passage (i.e., n_k takes the value of N). For instance, if the term appears once in the passage collection, its weight will be equal to 1 (the greatest weight). Whereas if it is a stopword its weight will be the lowest.

Depending on the style used to submit a question, sometimes a term unrelated to the question can obtain a greater weight than those assigned to the Name Entities (NE)[9]. Therefore, the (2) is changed to give more weight to the NE than the rest of question terms and so to force its presence in the first passages of the ranking. In order to identify the NE a natural language processing is not used. We showed that in the most questions the NE start with either an uppercase letter or a number. Once the terms are weighted, these are normalized for the sum of all terms are equal to 1.

[4]http://translate.google.com

[5]http://www.systranbox.com

[6]http://babelfish.altavista.com

[7]http://ets.freetranslation.com

[8]http://search.msn.com

[9]The NE are names of persons, organizations, places, dates, etc. The NE are the most important terms of the question and it does not make sense return passages which do not contain these words.

To calculate the weight of n-grams of every passage, first the greatest relevance of n-gram in the passage is identify and we assign to this a weight equal to the sum of all term weights. Next other n-grams less relevant are searched. These n-grams are not composed by terms of found n-grams. The weight of these n-grams will be the sum of all their weight terms. A n-gram weight is divided by two in order to avoid that its weight will be the same of the complete n-gram.

The passage retrieval engine, JIRS, can be obtained at the following URL: *http://leto.dsic.upv.es:8080/jirs*.

2.3 Answer Extraction

The input of this module is constituted by the n passages returned by the PR module and the constraints (including the expected type of the answer) obtained through the Question Analysis module. A *TextCrawler* is instantiated for each of the n passages with a set of patterns for the expected type of the answer and a pre-processed version of the passage text. Some patterns can be used for all languages; for instance, when looking for proper names, the pattern is the same for all languages. The pre-processing of passage text consists in separating all the punctuation characters from the words and in stripping off the annotations of the passage. It is important to keep the punctuation symbols because we observed that they usually offer important clues for the individuation of the answer: for instance, it is more frequent to observe a passage containing *"The president of Italy, Carlo Azeglio Ciampi"* than one containing *"The president of Italy IS Carlo Azeglio Ciampi'*; moreover, movie and book titles are often put between apices.

The positions of the passages in which occur the constraints are marked before passing them to the TextCrawlers. Some spell-checking function has been added in this phase by using Levenshtein distance to compare strings. The TextCrawler begins its work by searching all the passage's substrings matching the expected answer pattern. Then a weight is assigned to each found substring s, depending on the positions of the constraints, if s does not include any of the constraint words. Let us define $w_t(s)$ and $w_c(s)$ as the weights assigned to a substring s as a function, respectively, of its distance from the target constraints (3) and the context constraints (4) in the passage.

$$w_t(s) = \max_{0<k\leq|p(t)|} close(s, p_k(t)) \tag{3}$$

$$w_c(s) = \frac{1}{|c|}\sum_{i=0}^{|c|} \max_{0<j\leq|p(c_i)|} near(s, p_j(c_i)) \tag{4}$$

Where c is the vector of contextual constraints, $p(c_i)$ is the vector of positions of the constraint c_i in the passage, t is the target constraint and $p(t)$ is the vector of positions of the target constraint t in the passage. $Close$ and $near$ are two proximity function defined as:

$$near(s,p) = \exp\left(-\left(\frac{d(s,p)-1}{5}\right)^2\right) \tag{5}$$

$$close(s,p) = \exp\left(-\left(\frac{d(s,p)-1}{2}\right)^2\right) \tag{6}$$

Where p is a position in the passage and $d(s,p)$ is computed as:

$$d(s,p) = \min_{i=0,i=|s|} \sqrt{(s_i - p)^2} \tag{7}$$

Where s_i indicates the position of the i-th word of the substring s. The proximity functions can roughly be seen as fuzzy membership functions, where *close(s,p)* means that the substring s is adjacent to the word at the position p, and *near(s,p)* means that the substring s is not far from the word at position p. The 2 and 5 values roughly indicate the range within the position p where the words are considered really "close" and "near", and have been selected after some experiments with the CLEF2003 QA Spanish test set. Finally, the weight is assigned to the substring s in the following way:

$$w(s) = \begin{cases} w_t(s) \cdot w_c(s) & if|p(t)| > 0 \wedge |c| > 0 \\ w_c(s) & if|p(t)| = 0 \wedge |c| > 0 \\ w_t(s) & if|c| = 0 \wedge |p(t)| > 0 \\ 0 & elsewhere. \end{cases} \tag{8}$$

This means that if in the passage have been found both the target constraint and the contextual constraints, the product of the weights obtained for every constraint will be used; otherwise, only the weight obtained for the constraints found in the passage will be used.

Usually, the type of expected answer directly affects the weighting formula. For instance, the "DEFINITION" questions (such as "Who is Jorge Amado?") usually contain only the target constraint, while "QUANTITY" questions (such as "How many inhabitants are there in Sweden?") contain both target and contextual constraints. For the other question types the target constraint is rarely found in the passage, and weight computation relies only on the contextual constraints (e.g. "From what port did the ferry Estonia leave for its last trip?", *port* is the target constraint but it is not mandatory in order to found the answer, since it is most common to say "The Estonia left from Tallinn", from which the reader can deduce that Tallinn is -or at least *has*- a port, than "Estonia left from the port of Tallinn").

The filter module takes advantage of some knowledge resources, such as a mini knowledge base or the web, in order to discard the candidate answers which do not match with an allowed pattern or that do match with a forbidden pattern. For instance, a list of country names in the four languages has been included in the knowledge base in order to filter country names when looking for countries. When the filter rejects a candidate, the TextCrawler provide it with the next best-weighted candidate, if there is one.

Finally, when all TextCrawlers end their analysis of the text, the *Answer Selection* module selects the answer to be returned by the system. The following strategies have been developed:

- Simple voting (SV): The returned answer corresponds to the candidate that occurs most frequently as passage candidate.
- Weighted voting (WV): Each vote is multiplied for the weight assigned to the candidate by the TextCrawler and for the passage weight as returned by the PR module.
- Maximum weight (MW): The candidate with the highest weight and occurring in the best ranked passage is returned.
- Double voting (DV): As simple voting, but taking into account the second best candidates of each passage.
- Top (TOP): The candidate elected by the best weighted passage is returned.

SV is used for every "NAME" type question, while WV is used for all other types. For "NAME" questions, when two candidates obtain the same number of votes, the Answer Selection module looks at the DV answer. If there is still an ambiguity, then the WV strategy is used. For other types of question, the module use directly the MW. TOP is used only to assign the confidence score to the answer, obtained by dividing the number of strategies giving the same answer by the total number of strategies (5), multiplied for other measures depending on the number of passages returned (n_p/N, where N is the maximum number of passages that can be returned by the PR module and n_p is the number of passages actually returned) and the averaged passage weight. The weighting of NIL answers is slightly different, since is obtained as $1 - n_p/N$ if $n_p > 0$, 0 elsewhere.

In our system, candidates are compared by means of a partial string match, therefore *Boris Eltsin* and *Eltsin* are considered as two votes for the same candidate. Later, the Answer Selection module returns the answer in the form occuring most frequently.

For this participation we developed an additional web-corrected weighting strategy, based on web counts of the question constraints. With this strategy, the MSN Search engine is initially queried with the target and contextual constraints, returning a p_c number of pages containing them. Then, for each of the candidate answers, another search is done by putting the candidate answer itself together with the constraints, obtaining p_a pages. Therefore, the final weight assigned to the candidate answer is multiplied by p_a/p_c.

3 Experiments and Results

We submitted two runs for each of the following monolingual task: Spanish, Italian and French, while only one run was submitted for the Spanish-English and English-Spanish cross-language tasks. The second runs (labelled *upv_052*) of the monolingual tasks use the web-corrected weighting strategy, while the first runs use the clean system, without the recourse to the web. In Table 2 we show the overall accuracy obtained in all the runs.

task	run	overall	factoid	definition	tr	nil	conf
es-es	upv_051	33.50%	26.27%	52.00%	31.25%	0.19	0.21
	upv_052	18.00%	22.88%	0.00%	28.12%	0.10	0.12
it-it	upv_051	25.50%	20.00%	44.00%	16.67%	0.10	0.15
	upv_052	24.00%	15.83%	50.00%	13.33%	0.06	0.12
fr-fr	upv_051	23.00%	17.50%	46.00%	6.67%	0.06	0.11
	upv_052	17.00%	15.00%	20.00%	20.00%	0.07	0.07
en-es	upv_051	22.50%	19.49%	34.00%	15.62%	0.15	0.10
es-en	upv_051	17.00%	12.40%	28.00%	17.24%	0.15	0.07

Table 2: Accuracy results for the submitted runs. Overall: overall accuracy, factoid: accuracy over factoid questions; definition: accuracy over definition questions; tr: accuracy over temporally restricted questions; nil: precision over nil questions; conf: confidence-weighted score.

It can be observed that the web weighting produced worse results, even if the 0.00% obtained for the *upv_052eses* run for definition questions can be due to an undetected problem. Definition questions obtained better results than other kinds of questions, and we suppose this is due to the ease in identifying the target constraint in these cases. Moreover, the results for the Spanish monolingual tasks are better than the other ones, and we believe this is due mostly to the fact that the question classification was performed combining the results of the SVM and pattern classifiers, whereas for French and Italian the expected type of the answer was obtained only via the pattern-based classifier. Another reason can be that the majority of the preliminary experiments were done over the CLEF2003 Spanish corpus, therefore resulting in the definition of more accurate patterns for the Spanish Answer Extractor.

In order to evaluate the impact of the answer types, we grouped the results obtained for the best run by the defined categories , as shown in Table 3. As it can be seen, the best results have been obtained for the "LOCATION.COUNTRY" category, as expected, due to the use of a customized knowledge source. The worst results have been obtained for the questions "OTHER", for which there is not a defined strategy.

4 Conclusions and Further Work

The obtained results are comparable to those we obtained over the past year corpus, and therefore are as expected. The main drawback of the system is constituted by the cost of defining patterns for the Answer Extraction module: many experiments are needed in order to obtain a satisfactory pattern, and this has to be done for each expected answer type in each category. Moreover, apart from some well-defined categories for which a pattern can be defined, in other cases is almost impossible to identify a pattern that can match with all the answers of such questions. Therefore, we plan to use in the future both machine learning approaches in order to master this problem,

category	questions	accuracy
NAME	2	0.0%
NAME.PERSON	25	28.0%
NAME.TITLE	1	0.0%
NAME.LOCATION	6	16.7%
NAME.LOCATION.COUNTRY	14	92.8%
NAME.LOCATION.CITY	2	100.0%
NAME.LOCATION.GEO	2	0.0%
DEFINITION	61	44.3%
DATE	11	36.3%
DATE.DAY	4	0.0%
DATE.YEAR	2	0.0%
QUANTITY	21	33.3%
QUANTITY.AGE	4	25.0%
TIME	4	0.0%
OTHER	41	4.8%

Table 3: Accuracy results for the upv_051eses run, grouped by answer type.

together with more knowledge bases, since the small country database allowed to obtain good results for the COUNTRY questions. In the cross-language task, the Passage Retrieval module worked well despite the generally acknowledged low quality of web translations, allowing to obtain results slightly worse than those obtained in the monolingual task.

Acknowledgments

We would like to thank CONACyT for partially supporting this work under the grant 43990A-1 as well as R2D2 CICYT (TIC2003-07158-C04-03) and ICT EU-India (ALA/95/23/2003/077-054) research projects. A special acknowledgement to Manuel Montés y Gómez for support during his stance at the Universidad Politécnica de Valencia.

References

[1] Magnini, B., Negri, M., Prevete, R., Tanev, H.: Multilingual question/answering: the DIOGENE system. In: The 10th Text REtrieval Conference. (2001)

[2] Aunimo, L., Kuuskoski, R., Makkonen, J.: Cross-language question answering at the university of helsinki. In: Workshop of the Cross-Lingual Evaluation Forum (CLEF 2004), Bath, UK (2004)

[3] Vicedo, J.L., Izquierdo, R., Llopis, F., Muoz, R.: Question answering in spanish. In: Workshop of the Cross-Lingual Evaluation Forum (CLEF 2003), Trondheim, Norway (2003)

[4] Neumann, G., Sacaleanu, B.: Experiments on robust nl question interpretation and multi-layered document annotation for a cross-language question/answering system. In: Workshop of the Cross-Lingual Evaluation Forum (CLEF 2004), Bath, UK (2004)

[5] Moldovan, D.I., Pasca, M., Harabagiu, S.M., Surdeanu, M.: Performance issues and error analysis in an open-domain question answering system. ACM Trans. Inf. Syst. **21** (2003) 133–154