# University of Alicante at TREC-10

Vicedo, Jose Luis & Ferrández, Antonio & Llopis, Fernando.

*{vicedo,antonio,llopis}@dlsi.ua.es*

*Dpto. Lenguajes y Sistemas Informáticos*

*Universidad de Alicante*

*Apartado 99. 03080 Alicante, Spain*

## Abstract

This paper describes the architecture, operation and results obtained with the Question Answering prototype developed in the Department of Language Processing and Information Systems at the University of Alicante. Our system is based on our TREC-9 approach where different improvements have been introduced. Essentially these modifications are twofold: the introduction of a passage retrieval module at first stage retrieval and the redefinition of our semantic approach for paragraph selection and answer extraction.

## 1. Introduction

Open domain QA systems are defined as tools capable of extracting the answer to user queries directly from unrestricted domain documents. Question answering systems performance is continuously increasing since recent Text REtrieval Conferences [9] [10] included a special task for evaluating and comparing this kind of systems. The analysis of current best systems [1] [3] [4] [7] allows identifying main QA sub-components:

- Question analysis
- Document / passage retrieval
- Paragraph selection
- Answer extraction

The system presented to TREC-10 QA task is based on the described structure. It departs from the system presented in last TREC conference [11] where new tools have been added and existing ones have been updated. Modifications introduced rely on several aspects. First, document retrieval stage has been changed. Instead of using first fifty documents supplied by TREC organisation, we have implemented a passage retrieval module that allows a more successful retrieval. Second, our semantic-based paragraph selection approach has been redefined in order to increase selection process performance. Finally, question analysis and answer extraction modules have been updated by including special modules for managing with definition questions.

This year, question answering task has been significantly modified. The organisation has designed three different tasks: *main task, list task* and *context task*. Main task is similar to previous years' tasks but only permitting a maximum of 50 bytes as answer length. Besides, there is no guarantee that an answer will actually occur in the document collection and participants have to measure the degree of correctness of its answers. The list task consists of answering questions that will specify a number of instances to be retrieved. In this case, it is guaranteed that the collection contains at least as many instances as the question asks for. Finally, the context task consist of

answering a set of related questions in such a way that the interpretation of a question will depend on the meaning of and answers to one or more earlier questions in a series.

Our participation has been restricted to the main task although we did not face up all the restrictions. In fact, no effort was accomplished to measure which of the returned answers is more likely to be the correct one or to detect questions without correct answers in the document collection.

This paper is structured as follows: Section 2 describes the structure and operation of our system. Afterwards, we present and analyse the results obtained for TREC-10 task we participated in. Finally, initial conclusions are extracted and directions for future work are discussed.

## 2. System Overview

Our QA system is structured into the four main modules outlined before: *question analysis, document/passage retrieval, paragraph selection* and *answer extraction*. First module processes questions expressed in open-domain natural language in order to analyse the information requested in the queries. This information is used as input by remaining modules. Document retrieval module accomplishes a first selection of relevant passages by using a new passage retrieval approach. Afterwards, the paragraph selection module analyses these passages in order to select smaller text fragments that are more likely to contain the correct answer. Finally, the answer selection module processes these fragments in order to locate and extract the final answer. Figure 1 shows system architecture.

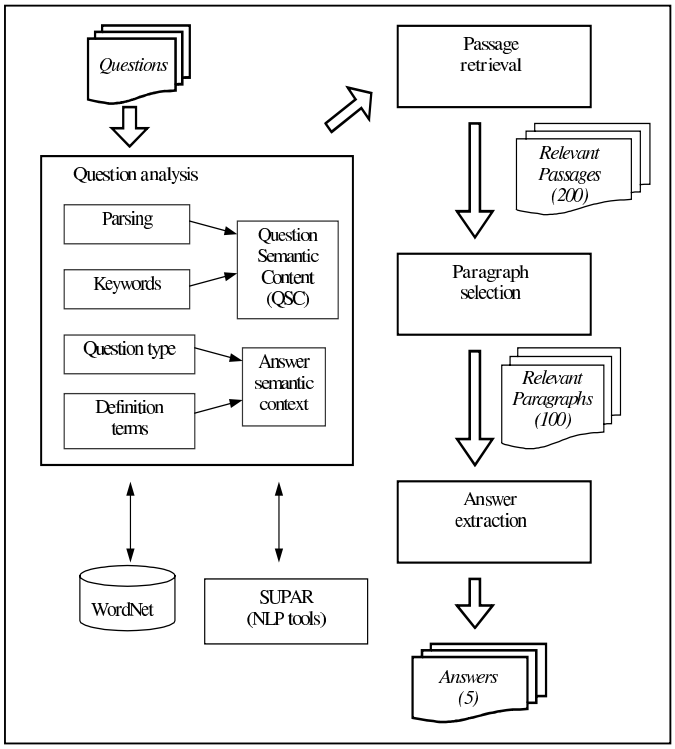


Figure 1. System architecture

Several standard natural language processing techniques have been applied to both questions and documents. These tools compose the Slot Unification Parser for Anaphora Resolution (SUPAR).

## 2.1. SUPAR NLP tools

In this section, the NLP Slot Unification Parser for Anaphora Resolution (SUPAR) is briefly described [2] [12]. SUPAR's architecture consists of three independent modules that interact with one other. These modules are lexical analysis, syntactic analysis, and a resolution module for Natural Language Processing problems.

Lexical analysis module. This module each document sentence or question to parse as input, along with a tool that provides the system with all the lexical information for each word of the sentence. This tool may be either a dictionary or a part-of-speech tagger. In addition, this module returns a list with all the necessary information for the remaining modules as output. SUPAR works sentence by sentence from the input text, but stores information from previous sentences, which it uses in other modules, (e.g. the list of antecedents of previous sentences for anaphora resolution).

Syntactic analysis module. This module takes as input the output of lexical analysis module and the syntactic information represented by means of grammatical formalism Slot Unification Grammar (SUG). It returns what is called slot structure, which stores all necessary information for following modules. One of the main advantages of this system is that it allows carrying out either partial or full parsing of the text.

NLP problems resolution module. In this module, NLP problems (e.g. anaphora, extra-position, ellipsis or PP-attachment) are dealt with. It takes the slot structure (SS) that corresponds to the parsed sentence as input. The output is an SS in which all the anaphors have been solved. In this paper, only the resolution of third person pronouns has been applied.

## 2.2. Question Analysis

Question processing module accomplishes several tasks. First, SUPAR system accomplishes part-of-speech tagging and parsing of the question. Afterwards, this module determines *question type*, classifies non-Wh terms into two categories (*keywords* or *definition terms)* and finally, concepts referred into the question are detected and processed to obtain the semantic representation of the concepts appearing in the question.

Question type is detected by analysing Wh-terms (e.g. What, Which, How, etc). This process maps Wh-terms into one or several of the categories listed in figure 2. Each of these categories is related to WordNet top concepts [6]. This module has been updated by including the definition questions as new question type. When no category can be detected by Wh-term analysis, NONE is used (e.g. "What" questions). This analysis gives the system the following information: (1) lexical restrictions that expected answer should validate (e.g. proper noun), (2) how to detect definition terms (if they exist), and (3) top WordNet concepts and related synsets that are compatible with the expected answer. Definition questions are detected by applying a pattern matching process. As example, questions such as “Who was Galileo?”, “What are amphibians?” or “What does USPS

| | | | | |
|---|---|---|---|---|
| PERSON | GROUP | LOCATION | TIME | |
| QUANTITY | DEFINITION | REASON | MANNER | NONE |

Figure 2. Question type categories

stands for?" are correctly analysed.

Once question type has been obtained, the system selects the *definition terms*. A term in a query is considered a definition term if it expresses semantic characteristics of the expected answer. Definition terms do not help the system to locate the correct answer into the document collection but they usually describe the kind of information requested by a query. Depending on question type, different patterns are used to detect definition terms. For "What", "Which", "How" and similar questions, this terms are detected by selecting noun phrases located next to the Wh-term. When questions such as "Find the number of whales…" or "Name a flying mammal ..." are analysed, noun phrases following the verb are considered definition terms.

Question type and definition terms are used to generate the *expected answer semantic context* (EASC). This context defines the lexical characteristics that the expected answer should validate to be considered a probable answer (e.g. proper noun) and the semantic context that the expected answer has to be compatible with. This context is made up by the set of synsets that are semantically related to definition terms and question type. These synsets are obtained by extracting from WordNet all hyperonyms of each definition term (its path to top concepts). These synsets are weighted depending on its level into the WordNet hierarchy and the frequency of its appearance into the path towards top concepts. Intuitively, this set of synsets defines the semantic context that has to be compatible with the expected answer semantic context. Finally, remaining question terms are classified as *keywords*.

Last question processing stage builds the semantic representation of the concepts expressed into the query (*Semantic Content of a Question - QSC*). This process consists of obtaining a general semantic representation of the concepts that appear in the questions and its main aim is to achieve concept representation in such a way that make possible to overcome term-based approach limits into the paragraph selection stage. To obtain this representation we have to deal with two basic requirements:

a) Concepts appearing in questions need to be correctly detected and extracted.
b) The different ways of expressing a concept have to be obtained and represented.

First requirement is accomplished by parsing questions. This process obtains all the syntactic structures that made up each question. Structures containing definition terms are discarded. Then, each syntactic structure (noun and verbal phrases) that contains one or more keywords defines a concept. The head of each syntactic structure represents the basic element or idea the concept refers to. Remaining terms pertaining to this structure modify this basic concept by refining the meaning represented by its head.

Accomplishing the second requirement involves obtaining and representing the different ways of expressing each of the concepts detected in a query. This process starts by associating each term pertaining to a concept, with its synonyms and one level search hyponyms and hyperonyms. These relations are extracted from WordNet lexical database. We define the semantic content of a term *t* (SC*t*) as a set of terms made up by the term *t* and all the terms related with it through the synonym and one level search hyponym and hyperonym relations. The SC of a term is represented using a weighted term vector. The weight assigned to each term pertaining to the SC of a term *t* is the 80%, 50% and 50% of the *idf* [8] value of term *t* for synonyms, hyponyms and hyperonyms respectively. As a concept is made up by the terms included into the same syntactic structure, we define the semantic content of a concept (SCC) as the set of weighted vectors (HSC, MSC) were HSC is the a vector obtained by adding the SC of the terms that made up the head of the concept and MSC is the vector resulting from adding the SC of terms that modify that head into the same syntactic structure.

The set of SCCs that stand for the concepts appearing in a question builds the semantic content of a question (QSC). This way, the QSC represent all the concepts referenced into the question and the different ways of expressing each of them. This process is widely explained in [13].

Figure 3 shows the semantic content of an example question First, the system identifies the concepts "*manufactures*" and "*American Girl doll collection*" by detecting syntactic structures that contain keywords. Afterwards, the semantic content of each concept is generated.

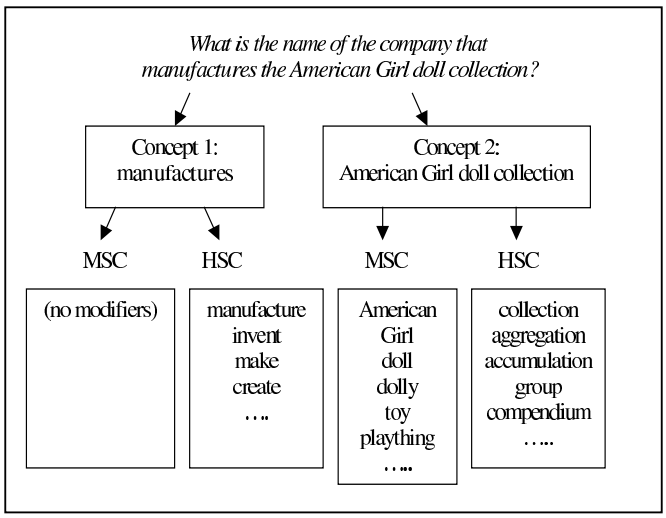


Figure 3. Example of QSC

Question keywords are used for first stage passage retrieval while QSC information will help paragraph selection module to detect the paragraphs that are more likely to contain the answer.

## 2.3. Passage retrieval module

First stage retrieval applies the passage retrieval approach described in [5]. This passage retrieval can be applied over all the document collection, but it has only been applied for the 1000 relevant documents supplied by TREC organisation. Therefore, keywords detected at question processing stage are used for retrieving the 200 most relevant passages from the documents included in this initial list. This process is intended to reduce the amount of text that has to be processed by costly NLP modules since these passages are made up by text snippets of 15 sentences length.

## 2.4. Paragraph selection

This module processes 200 first ranked passages selected at passage retrieval stage in order to extract smaller text fragments that are more likely to contain the answer to the query. As all this process is widely described in [13] we extract here the basic algorithm:

a) Documents are split into sentences.

b) Overlapping paragraphs of three sentences length are obtained.

c) Each paragraph is scored. This value measures the similarity between each paragraph and the question.

d) Paragraphs are ranked according to this score.

The score assigned to each paragraph (*paragraph-score*) is computed as follows:

a) Each SCC appearing in the question is compared with all the syntactic structures of the same type (noun or verbal phrases) appearing into each relevant paragraph. Each comparison generates a value. As result, each SCC is scored with the maximum value obtained for all the comparisons accomplished through the paragraph.

b) The paragraph-score assigned to each paragraph is obtained by adding the values obtained for all SCCs of the question as defined in previous step.

c) The value that measures similarity between a SCC and a syntactic structure of the same type is obtained by adding the weights of terms appearing into SCC vectors and the syntactic structure that is being analysed. If the head of this syntactic structure does not appear into the vector representing the SCC head (HSC), this value will be 0 (even if there are matching terms into MSC vector).

At this stage, only best 100 ranked paragraphs are selected to continue with the remaining processes.

### 2.5. Answer extraction

This process consists on analysing selected paragraphs in order to extract and rank the text snippets of the desired length that are considered to contain the correct answer. For this purpose, the system selects a window for each probable answer by taking as centre the term considered a probable answer. Each window is assigned a score (*window-score*) that is computed as follows:

$$Window\text{-}score = paragraph\text{-}score*(1+cos(EASC,PASC))$$

where *EASC* is the vector representing the semantic context of the expected answer and *PASC* stands for the vector representing the semantic context of the possible answer. PASC is computed as done for EASC but using the terms contained into the syntactic structures the probable answer appear into, as well as surrounding syntactic structures.

Intuitively, the window-score combines (1) the semantic compatibility between the probable answer and the expected answer (cos(EASC,PASC)) and (2) the degree of similarity between question and paragraphs (paragraph-score).

Finally, windows are ranked on window-score and the system returns the first five as answer.

Answer extraction manages differently with definition questions. This questions look for answers that define or explain the concept expressed in the question. From the analysis of definition questions in TREC-9 question set we derived a set of heuristics for detecting answers to definition questions. Each of these heuristics refers to a different way of expressing definition answers. The

following list shows the main ways in which answers to definition questions are more probably expressed and several examples (the answer is italicised):

- Noun phrases including the answer ("*Italian archbishop* Filippo Cune ...").
- Explanatory appositions (" Filippo Cune, the *Italian archbishop*, …").
- Explanatory conjunctions ("*Italian archbishops* Federico Pane, Filippo Cune and ...").
- Definition phrases (" Filippo Cune was the *Italian archbishop* …").
- Coreference resolution ("Filippo Cune travelled to Pisa. The *Italian archbishop* desired to renew the …").
- …

These heuristics were ordered depending on the probability of obtaining a correct answer to a question (*heuristic probability*) by applying each of them on TREC-9 definition question set. This order determines the sequence of application of each heuristic over relevant paragraphs. The following algorithm shows how these heuristics are applied:

a) Heuristics are applied over each relevant paragraph in an ordered way until one of them (or none) succeeds.
b) Answers detected by successful heuristics are extracted.
c) These answers are scored (*answer-score*) as follows:

*Answer-score = paragraph-score * heuristic probability*

d) For duplicated answers, only the highest ranked is maintained,
e) First five ranked answers are returned as final answers.

## 3. Results

This year we submitted two runs for main task. This task allowed five answers for each question and a maximum answer string length of 50 bytes. Figure 4 shows the results obtained. Applying the whole system described above has produced ALIC01M2 run. ALIC01M1 files contain results obtained applying the same strategy but without solving pronominal anaphora in relevant passages. These results were computed after the organisation decided to get rid of eight questions. Therefore, 492 questions were evaluated.

Although a detailed results analysis is a very complex task, several conclusions can be extracted.

| Run | Mean reciprocal rank | | % Answers found | |
|---|---|---|---|---|
| | strict | lenient | strict | lenient |
| ALIC01M1 | 0,296 | 0,302 | 39,2% | 40,0% |
| ALIC01M2 | 0,300 | 0,306 | 39,6% | 40,4% |

Figure 4. TREC-10 main task results

Comparison with TREC-9 results.
Our system has achieved a significant improvement since TREC-9 participation. Comparison between strict best results for 50 bytes answer length at TREC-9 (see figure 5) and TREC-10 (figure

4) shows that the mean reciprocal rank has increased 0.7 points (from 0.23 to 0.30) and besides, the percentage of correct answers found has increased 5.7 points (from 33.9% to 39.6%).

| Run | Mean reciprocal rank | | % Answers found | |
|---|---|---|---|---|
| | strict | lenient | strict | lenient |
| ALI9C50 | 23,0% | 24,5% | 33,9% | 36,1% |
| ALI9A50 | 22,7% | 24,0% | 33,9% | 35,8% |

Figure 5. TREC-9 50 bytes answer length results

Retrieving relevant documents.
Correct answer was not included into the top ranked documents supplied by TREC for 61 questions. If we discard the 49 questions with no correct answer in the collection this number falls to 12 questions. Figure 6 compares the percentage of questions that could be correctly answered between the two possible approaches: (1) processing a number of top documents and (2) selecting a number of passages.

| | Top Passages | | Top Documents | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 500 questions | 100 | 200 | 50 | 100 | 200 | 350 | 500 | 750 | 1.000 |
| Answer included | 200 | 424 | 393 | 407 | 420 | 430 | 432 | 435 | 439 |
| Answer Not included | 300 | 76 | 107 | 93 | 80 | 70 | 68 | 65 | 61 |
| % Answer Included | 40,0% | 84,8% | 78,6% | 81,4% | 84,0% | 86,0% | 86,4% | 87,0% | 87,8% |

Figure 6. Passage and document retrieval comparison

As we can notice processing 200 passages produces best results than processing 200 complete documents and besides it dramatically reduces later NLP processing costs.

Paragraph selection.
Our main objective was to inspect if our new paragraph selection method was more effective than last year proposal. As we expected, this model has achieved a better performance. Strict MRR increased 0.7 points from past results, which corroborates that precision achieved at this process has improved significantly.

Pronominal anaphora resolution
The small benefit obtained last year from applying pronominal anaphora resolution has been corroborated with TREC-10 results. This fact is mainly due to the same reasons described last year [11]. Nevertheless, although we have not participated into the context task thise kind of questions will surely take more profit form coreference resolution techniques.

## 4. Future Work

Several areas of future work have appeared while analysing results. First, passage retrieval has to be tested over the whole collection to investigate the level of benefit it can produce over current results. Besides, although our paragraph selection module has revealed to be very efficient, several aspects can be improved, especially by incorporating a validation module that could measure the inexistence of the answer. Third, it seems essential to incorporate a Name-Entity tagger to our

answer extraction module since we missed several answers that could have easily been detected. And fourth, the system needs to be adapted to manage with list and context questions.

## 5. References


1. Clarke C.L., Cormack G.V., Kisman D. and Lynam T. R. *Question Answering by Passage Selection (MultiText Experiments for TREC-9)*. In Proceedings of the Nineth Text Retrieval Conference. November 2000. Gaithersburg (US).
2. Ferrández A., Palomar M. and Lidia Moreno. *An empirical approach to Spanish anaphora resolution*. Machine TranslationSpecial Issue on Anaphora Resolution in Machine Translation. Kluwer Academic publishers. ISSN 0922-6567 .1999.vol 14(3/4) pages 191-216.
3. Harabagiu S., Moldovan D., Pasca M., Mihalcea R., Surdeanu M., Bunescu R., Gîrju R., Rus V, and Morarescu, P. *FALCON: Boosting Knowledge for Answer Engines*. In Proceedings of the Nineth Text Retrieval Conference. November 2000. Gaithersburg (US).
4. Ittycheriah A., Franz M., Zu W. and Adwait Ratnaparkhi. *IBM's Statistical Question Answering System*. In Proceedings of the Nineth Text Retrieval Conference. November 2000. Gaithersburg (US).
5. Llopis F. and Vicedo J.L. *IR-n: a passage retrieval system at CLEF-2001*. In proceedings of the second Cross-Language Evaluation Forum (CLEF2001). Lecture Notes in Computer Science. (To appear). September 2001. Darmstadt (Germany).
6. Miller G.(1995), *"Wordnet: A Lexical Database for English"*, Communications of the ACM 38(11) pp 39-41.
7. Prager J., Brown E., Radev D. and Krzysztof Czuba. *One Search Engine or Two for Question-Answering*. In Proceedings of the Nineth Text Retrieval Conference. November 2000. Gaithersburg (US).
8. Salton G.(1989). *Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer*. Addison Wesley Publishing, New York.
9. TREC-8, 1999. Call for participation Text Retrieval Conference 1999 (TREC-8).
10. TREC-9, 2000. Call for participation Text Retrieval Conference 2000 (TREC-9).
11. Vicedo J.L. and Ferrandez A. *A semantic approach to Question Answering systems*. In Proceedings of the Nineth Text Retrieval Conference. November 2000. Gaithersburg (US).
12. Vicedo J.L. and Ferrández A. *Importance of Pronominal Anaphora resolution in Question Answering systems*. In proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL2000). October 2000. Hong Kong (China).
13. Vicedo J.L. *Using semantics for Paragraph selection in Question Answering systems*. In proceedings of the Proceedings of the Eighth String Processing and Information Retrieval Conference (SPIRE'2001). November 2001. Laguna de San Rafael (Chile).