# Combining Linguistic Processing and Web Mining for Question Answering: ITC-irst at TREC-2004

Hristo Tanev, Milen Kouylekov, Bernardo Magnini
{tanev|kouylekov|magnini@itc.it}

ITC-irst, Centro per la Ricerca Scientifica e Tecnologica
Via Sommarive, 38050 Povo (TN), Italy

**Abstract**

This paper describes the work we have been done in the last year on the DIOGENE Question Answering system developed at ITC-Irst. We present two preliminary experiments showing the possibility of integrating into DIOGENE a textual entailment engine based on entailment rules. We addressed the problem proposing both a methodology for acquiring rules from the Web and a matching algorithm for comparing dependency trees derived from the question and from documents. Although the overall results are not high, we consider this year participation at TREC as an intermediate step in view of a more complete and in depth integration of textual entailment rules into the system. We also report about the problems we encountered in maintaining the Web-based answer validation module.

## 1 Introduction

DIOGENE is the QA system we have been developing at ITC-irst in the last three years (Kouylekov et al. 2003) and (Magnini et al. 2002c). The system presented at TREC 2003 was based on a rather simple architecture, did not make use of deep linguistic processors (e.g. we do not use syntactic analysis) and implemented an answer validation module based on Web redundancy. This year work on QA at ITC-irst was mainly driven by the following considerations: (i) the system was heavily dependent from the Web and from the search engines available on the Web; the main disadvantages are that Web information could not be of help for restricted domains and that we are subject to changes in the interface to Web search engines, so that a crucial module of the system is basically out of our control; (ii) from a research point of view, we are interested in investigating and developing core language technologies, which can be applied to different application scenarios, including QA; one of the most promising approach in this direction is Textual Entailment, recently proposed by (Glickman and Dagan 2004).

Following the above considerations, this year we have been working on two parallel lines: the first is a long term research project on textual entailment; the second aims at utilizing preliminary ideas of the first activity able to produce improvements in the QA system in the short term. As for textual entailment, the long term goal is the realization of a QA system whose core component is a textual entailment engine able to produce inferences from a large database of entailment rules. This research line is been conducted in

collaboration with Bar Ilan University and resulted in a module for automatic extraction of entailment rules from the Web (Szpektor et al. 2004). However, since at the TREC 2004 time we did not have available the entailment engine and since the quality of the acquired rules was still not satisfactory, we decided to conduct a number of small-scale experiments aiming at showing the potential of the entailment approach on a subset of the TREC questions. This is our main contribution for TREC 2004.

More in detail, this year we performed two experiments. Since the textual entailment approach is based on the ability to derive textual inferences on the basis of a syntactic representation of the text (i.e. dependency trees), we have integrated into the DIOGENE both a component for dependency analysis and a matching algorithm which discovers overlapping structures among two dependency trees. The algorithm matches the syntactic tree of the question (transformed into its affirmative form) to the retrieved documents and extracts answer candidates appearing in the syntactic positions where the answer is expected. The approach is then coupled with the Web validation module in order to filter out candidates with less probabilities.

The second experiment aims at a preliminary evaluation of the contribution of textual entailment rules on a limited subset of questions. Rules for those questions as been acquired in a semi-automatic way and implemented as templates which are matched on the retrieved documents. What is significant in our long term perspective is the possibility to bypass the Web validation module without a loss of performance when an entailment rule provides enough evidence for a particular answer. In the experiment we performed, considering a subset of questions we achieved promising results in term of precision.

Finally, the paper also briefly reports about the problems we encountered in maintaining the Web-based answer validation module. Since the interface with the Altavista search engine change during the year, we were forced to move to another search engine and to adapt our program interface.

The rest of the paper is structured as follows. Sections 2 and 3 describe the first experiment, where we use a dependency parser and we look for overlapping structures in order to find possible candidate answers. In section 4 we describe the second experiment, including the algorithm for the semi-automatic acquisition of entailment rules. In section 5 we describe the changes to the Web-based answer validation approach. Results for both the experiments are presented and discussed in Section 6.

## 2. Experiment 1 : syntactic analysis of questions

The experiment aims at estimating the role of syntactic-based QA coupled with a powerful algorithm for partial matching of syntactic structures and shows a step toward a full implementation of a textual entailment engine. In concrete, the changes that we made in our question analysis module aim at transforming a question in its affirmative form in order to use this form for answer extraction.

We have integrated into the system architecture a dependency based parser. We used MINIPAR, a principle-based English parser (Lin, 1998a) because of its processing speed and good performance in terms of precision and coverage. We used the parser specific clauses and relations to identify the head of the question – a phrase from which can be identified the type of the answer and the expected sentence sub-structure that contains it. For instance, the head of the question represented in Figure 1 is the sub-structure “What

animal". The specific relation "whn" denotes a phrase that contains a "wh" word (e.g. "what", "When", "Who", …).

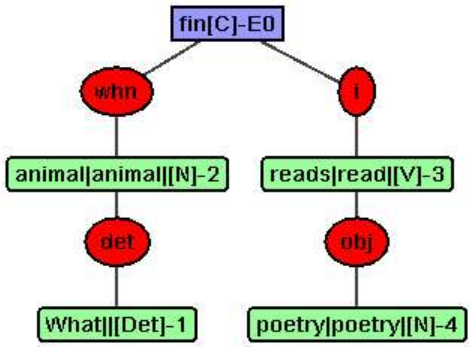


**Figure 1 Question parsed by MINIPAR**

Our goal when converting the question into the affirmative form was to use it as a basic syntactic template that can be used for extracting answer candidates. The affirmative form of the question is obtained trough transformations of the parse tree of the question. In the example of Fig. 2 the parse tree of the question (left sub-tree) is transformed in what we call *basic template* (right sub-tree). The node "X" is a variable node. It denotes the expected answer position. The variable node is attached to the main verb "read". The words "What" and "animal" are removed from the basic pattern.

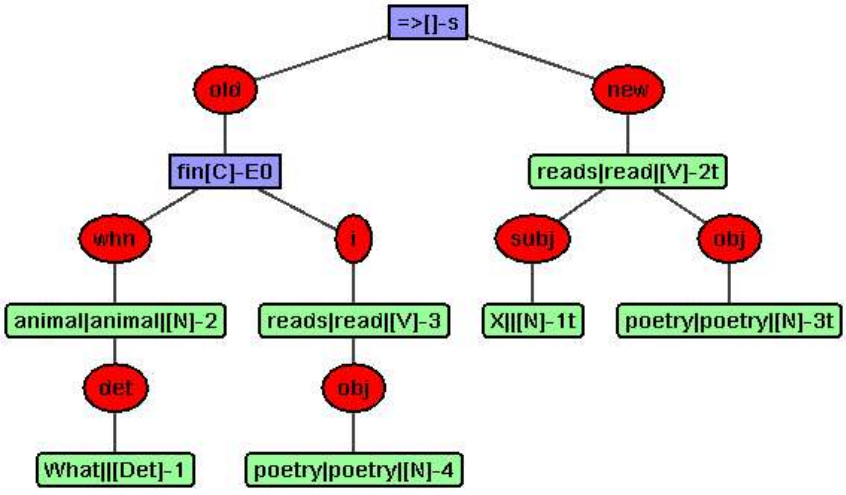


**Figure 2 Question transformation in affirmative form**

The reconstruction is done using a small number of hand crafted rules and the information from the question analysis output. The transformation consists of:

- Removing nodes and arcs from the dependency tree
- Introducing some functional words
- Attaching variables on the right positions

## 3. Experiment 1: Answer Extraction Using Syntactic Matching

To extract answer candidates we match the basic template of the question against the syntactic trees of the documents retrieved by the search engine. The underlying assumption is that in many cases we can find the answers to a question in a syntactic structure similar to the structure obtained from that question's affirmative form. When matching, we faced the following problems:

- Not every word from the basic syntactic template can be found in the syntactic tree that contains the answer: this led us to investigate algorithms for partial matching.
- Fifty percent of the extracted answers were anaphoric expressions (pronouns) which indicates the need for an anaphora resolution module.

We match the basic template of the question by searching for dependency trees in the retrieved documents that contain at least one node of the basic template. After finding such tree, starting from this node we try to map the rest of the template.

We also experimented a simple approach to address lexical variations. We used a database of dependency-based thesaurus of similar words (Lin, 98b), available on the Web at http://www.cs.ualberta.ca/~lindek/downloads.htm. When a syntactic matching is detected among dependency trees, words are also checked according to the following procedure.

Two words are matched if they have the same lemma and the same part of speech: the weight of such an exact match is 1.

Otherwise, two words are matched if they belong to the same thesaurus class, and the similarity between them is the weight of the matching.

Finally, if none of the above conditions hold, the words are matched if they share the same part of speech; in this case the weight of the word matching is zero.

We tried also to handle problems with matching caused by anaphoric expressions. If in place of a noun in the parse tree of a sentence in the document we find a pronoun, then the following procedure is adopted:

1. We search for the noun phrase present in the basic template in the previous sentences of the document.
2. If we can find it, then we can assume that an exact match is found.
3. The cost of such matching is then divided by the number of the other nouns that can be candidates for replacing the pronoun, if resolved.

We also have to estimate the weight of partially matched expressions. Consider, for instance, the following question:

"George Bush purchased a small interest, in which baseball team?"

The matched words in two documents extracted from the Aquaint collection are:

- „Bush", „purchased" and „team"
- „George", „Bush" and „baseball"

The matched words in these documents do not give an indication on which of them contain the right answer to the question. In the first document the word *Bush* is a common English name, and the fact that the person it represents *purchased* something does not provide us with enough information to conclude that the answer is contained in the document. The word *team* does not help us to reach a conclusion, because it is likely to be replaced in the document by the answer. In the second document we have some certainty about the person about which the document is speaking. Still we need additional information in order to be certain that in the document *George* is the first name of *Bush* and not a person that occurs closely to *Bush*. The word *baseball* determines in some way the domain of the document. But still this document will not be a relevant document if *Bush* is the name of a baseball player.

For the evaluation of the importance of partially matching sub-trees we use a scoring scheme defined in (Kouylekov and Tanev, 2004). The weight of the matched sub-tree of a pattern is defined by the formula:

$$f(x)=\sum \text{weight}(\text{relation}_i)*\text{weight}(w_{i1})*\text{weight}(w_{i2}),$$

where $W_{i1}$ and $W_{i2}$ are words which are in a dependency relation "*i* ". The weight of a relation is the inverse document frequency of the phrase formed by the two words that are in the relation. Using this scoring function we define the score of the document as the sum of the scores of the matched basic patterns or parts of basic patterns in it.

We extracted as candidate answers the phrases which appear in the appropriate positions in the basic templates. When we can not match an answer, but still there is a partial match of the basic template, we pick a phrase that correspond to the answer type, and appears close to the words of the basic template.

## 4. Experiment 2: Introducing Entailment Rules

A significant part of the TREC 2004 questions referred to persons and organizations. We expected this, since the guidelines and the test examples released before TREC pointed in this direction. There are important attributes and relations typical for persons and organizations. For persons these are the date and the location of birth and death, the name of his/her wife, name of college attended, etc. For an organization, it is important to know when and where it was founded, who founded it, what are its activities, how many people work for it, who are its members, etc. Studying different texts we concluded that relations of these kind can be identified using a library of syntactic templates, which represents our first attempt to introduce entailment rules into the Diogene system..

### *4.1 Rule Acquisition*

We constructed a small library of such templates. Apart of the basic template obtained directly from the question (see Section 3), we included in this library a set of templates acquired in a semi-automatic manner, using a machine learning approach similar to (Ravichandran and Hovy 2002). We acquired syntactic templates for four question types:

*When was X born?*
*Where was X born?*

*When did X die?*
*When was X founded?*

We followed the basic steps of the approach of Ravichandran&Hovy, which can be described through the following points:

1. For a relation (e.g. "X in <LOCATION>"), manually create a list of pairs of entities for which this relation holds (e.g. ("Washington", "USA") ("Sidney", "Australia"), …).
2. Query the Web with these pairs and collect a corpus of sentences with these entities (e.g. "Washington (USA) is in the eastern part …", "Sidney (Australia) hosted…")
3. Extract patterns which tend to connect the pairs of entities (e.g."X(<LOCATION>)")
4. Since these patterns connect the same entities as the original relation, it is assumed that they paraphrase it.

However, while Ravichandran&Hovy acquire linear patterns, we are interested in syntactic templates, represented through syntactic dependency structures. For this purpose we used TEA, a template extraction algorithm described in (Szpektor et.al.2004). For example, for the relation <**ORGANIZATION**> WAS FOUNDED IN <**YEAR**> the following steps are performed:

1. We created lists of 9 pairs of organizations and the years of their foundation (i.e. ("Red Cross" "1863"), ("Greenpeace", "1971"), ("NATO", "1949"),…) and collected from the Web sentences where these pairs appear.
2. We parsed the sentences with the dependency parser MiniPar (Lin…) and in the parse tree we substituted the organizations (i.e. "Red Cross", "Greenpeace", "NATO"…) with the variable "ORGANIZATION" and the year of foundation (i.e. "1863", "1971", "1949",…) with the variable "YEAR-FOUNDED". Next, using TEA, we extracted all the syntactic constructions which tend to span over pairs of syntactic nodes named "ORGANIZATION" and "YEAR-FOUNDED".
3. For the relation in this example, we got 10 syntactic paraphrases of the type (here we show their linearized form):
    - <**ORGANIZATION**> WAS ESTABLISHED IN <**ANSWER**>
    - SINCE ITS INCEPTION IN <**ANSWER**>, <**ORGANIZATION**> HAS
    - <**ORGANIZATION**> WAS FORMED IN <**ANSWER**>
4. We expanded some of the verbs and nouns in these templates with synonyms taken from WordNet to extend their coverage.

### *4.2. Rules Application*

The templates acquired for the four classes are applied to extract possible answers from documents. As an example, DIOGENE recognizes that question *61.2.* from TREC2004: *When was the Muslim Brotherhood formed?* belongs to the class of questions *When was X founded?* using the syntactic recognition template *When was X formed?*. If the question recognizer assigns the question to one of the four above mentioned classes, the set of templates specific for that questions are instantiated with the entities from the question. For example the template <**ORGANIZATION**> WAS ESTABLISHED IN <**ANSWER**> becomes **MUSLIM BROTHERHOOD** WAS ESTABLISHED IN <**ANSWER**>

Next, the lexical items from these templates are used to perform query expansion for document retrieval. For example, for question *61.2* the query to the document retrieval engine looks like:

*Muslm Brotherhood **AND** (founded **OR** formed **OR** established **OR** created ….)*

The paragraphs we obtain are next parsed using MiniPar and several syntactic normalizations take place in order to unify the different ways one and the same meaning is expressed syntactically. For example all constructions *"X's Y"* are transformed into *"Y of X"*. Currently we created manually such normalization rules, considering the possibility to use machine learning approaches for automatic or semi-automatic learning (Szpecktor et al.2004). When paragraphs are retrieved and normalized, the system matches the templates against the paragraphs. When a match is successful, the system checks if the entity matching the answer position in the template belongs to the appropriate Named Entity (NE) class, since each type of question is related to such a class. Finally, the entity which matches the template answer position and the NE constraints is returned as an answer.

## 5. Changes to the Web-based Answer Validation Module

In our previous participations at TREC we used a Web validation method based on a co-occurrence statistical formula (see (Magnini et al. 2002c) for details). The frequency information used in this formula was taken from AltaVista. We used AltaVista's proximity operator "NEAR" which allowed for identifying the number of pages in which certain words co-occur close to each other. However, AltaVista changed its interface, providing no further support for proximity searches, nor we were able to find a public available search engine which offers the same feature. Therefore, we opted for a variant of the answer validation method called *content based answer validation*, whose main ideas we described earlier in (Magnini et al. 2002b). The new method uses the AllTheWeb search engine (www.alltheweb.com) and performs the following basic steps:

1. It queries the Web with the question keywords *QK* and the answer *a*. For example, for the question *"Where was Carlos the Jackal born?"* and the (correct) candidate answer "*Venezuela"*, we have: *QK={Carlos, Jackal, born}; a="Venezuela"*
2. The top 100 hits returned by AllTheWeb are explored and for each text fragment where the answer *a* co-occurs with some of the *QK* words we calculate a score on the basis of the distance between *a* and the number of keywords present in *QK* which also appear in the snippet, according with the following formula:

$$score(snippet) = \prod_{k \in snippet \cap QK} 2^{1+|a\,k|^{-1}},$$

where |a k| is the distance in tokens between the candidate answer *a* and a question keyword *k* which appears in the snippet. In this way the closer a candidate appears to the question keywords in a snippet, the higher the score it gets from that snippet. For example in the text fragment:

> **Carlos** the **Jackal** was **born** in **Venezuela** to an affluent family whose father was a committed Marxist.

the answer *"Venezuela"* obtains score of 15, while in the text:

> He was **born** in **Venezuela**, under the more sane name of Ilich Ramirez Sanchez, but history will remember him as **Carlos** the **Jackal**

the answer obtains score 12.3, since it is more distant from the keywords.

3. The scores gained from all the snippets returned by AlltheWeb are summed up for each candidate answer.

DIOGENE returns as answer the candidate for which the answer validation returns the highest score. If the answer validation module returns zero for all the candidate answers, DIOGENE returns NIL as answer.

## 6. Experiments and Evaluation

We submitted three runs. In all of them for the factoid and list questions first we tried if syntactic templates can be applied; if not, we selected 13 candidate answers per question and used the Web based answer validation module to rank and filter them. The selection of answer candidates for Web-based answer validation was performed in three different ways: In **irst04web** we selected the candidates which appear closer to the question keywords. In **irst04parse** we selected these for which certain syntactic relation with the question keywords exists. In **irst04higher** we used both type of candidate answers. The best results was obtained from this last combined run where both candidates were used. This shows that the syntactic matching can successfully be used as a complimentary strategy together with proximity answer extraction. Selecting candidate answers through syntactic matching has the advantage that it captures answers that are not close to the question keywords in the surface structure of the sentence, but nevertheless have syntactic relations with them.

| Run | Overall F-score | Factoid accuracy | List F-score | Definition |
|---|---|---|---|---|
| **irst04web** | 0.213 | 0.278 | 0.09 | 0.207 |
| **irst04parse** | 0.195 | 0.239 | 0.1 | 0.2 |
| **irst04higher** | 0.223 | 0.291 | 0.103 | 0.207 |

**Table1: DIOGENE results on TREC2004 QA task**

Although the results we gained are not high in absolute values, our results are better than the average results calculated over all the 63 TREC2004 runs. In particular, in our best run we have factoid accuracy with 0.12 over the average.

There were 27 out of 230 factoid questions (12%) which belong to the four questions types we considered in the small syntactic template library presented in section 4. The

template answer extractor answered 7 of them (26%). All the answers returned by the template answer extractor were correct. Although the template answer extractor covers a small number of question classes and the coverage of 26% is still not sufficient, the high precision (100%) shows that for certain question types the template based answer extraction is a feasible and promising approach.

The total impact of using syntactic templates and selecting candidates through syntactic matching was 7% improvement of the overall F-score (we considered the results from Table 1 and we studied also the improvement due to the use of syntactic templates). Although this improvement is quite modest, we intend to improve further both the template answer extraction and the syntactic matching approaches and to integrate both paraphrases (Lin and Pantel 2001) and entailment rules (Szpektor et.al.2004) in order to strengthen the linguistic infrastructure of our QA system.

**References:**


Dagan I. and O.Glickman "Probabilistic Textual Entailment: Generic Applied Modelling of Language Variability" In Learning Methods for Text Understanding and Mining Workshop, 2004

Kouylekov M., B. Magnini, M.Negri, and H.Tanev "ITC-irst at TREC 2003: the DIOGENE QA System" In TREC-11 Conference Notebook Papers, Gaithersburg, MD, 2003

Kouylekov M. and Tanev H. "Document Filtering and Ranking Using Syntax and Statistics for Open Domain Question Answering", ESSLLI 2004 Workshop on Combining Deep and Shallow Linguistic Analysis for NLP, 9-13 August, 2004

Lin D., "Dependancy-based evaluation of MINIPAR" In Workshop on the Evaluation of Parsing Systems, Granada, Spain, May, 1998a

Lin D., "Automatic Retrieval and Clustering of Similar Words", COLING-ACL-98, Montreal, Canada, August, 1998b

Lin D. and P. Pantel, "Discovery of Inference Rules for Question Answering" Natural Language Engineering 7(4), 2001

Magnini B., Negri M., Prevete R. and Tanev H. "Is it the Right Answer? Exploiting Web Redundancy for Answer Validation", Association for Computational Linguistics 40th Anniversary Meeting (ACL-02), of Pennsylvania, Philadelphia, July 7 - 12, 2002a

Magnini B., Negri M., Prevete R. and Tanev H. "Comparing Statistical and Content-Based Techniques for Answer Validation on the Web", Proceedings of the VIII Convegno AI*IA , Siena - Italy, September 11-13, 2002b

Magnini B., Negri M., Prevete R. and Tanev H. "Mining Knowledge from Repeated Co-occurrences" In TREC-11 Conference Notebook Papers, Gaithersburg, MD, November 19-

22, 2002c

Ravichandran D. and E. Hovy “Learning Surface Text Patterns for a Question Answering System” In Proceedings of the 40th ACL conference, Philadelphia, 2002.

Szpektor I., Tanev H., Dagan I., and Coppola B. "Scaling Web-based acquisition of Entailment Relations" In Proceedings of EMNLP-04 - Empirical Methods in Natural Language Processing, Barcelona, July 2004