Linguistic knowledge and question answering

Gosse Bouma — Ismail Fahmi — Jori Mur — Gertjan van Noord
Lonneke van der Plas — Jörg Tiedemann

University of Groningen, PO Box 716,
9700 AS Groningen, The Netherlands

{g.bouma,j.mur,g.j.m.van.noord,m.l.e.van.der.plas,j.tiedemann}@rug.nl

ABSTRACT. *The availability of robust and deep syntactic parsing can improve the performance of all modules of a Question Answering system. In this article, this is illustrated using examples from our QA system Joost, a Dutch QA system which has been used for both open and closed domain QA. The system can make use of information found in the fully parsed version of the document collections. We demonstrate that this improves the performance of various components of the system, such as answer extraction and selection, lexical acquisition, off-line relation extraction, and passage retrieval.*

RÉSUMÉ. *Une analyse syntaxique profonde et robuste améliore la performance d'un système de question-réponse. Dans cet article, nous le montrerons en donnant des exemples de notre système QR, appelé Joost. C'est un système néerlandais, qui a été appliqué au domaine général ainsi qu'au domaine restreint. Le système utilise l'information contenue dans une version analysée syntaxiquement du corpus des documents. Nous montrerons que l'utilisation de l'information syntaxique améliore certains modules de Joost, comme l'extraction et l'ordonnancement final des réponses, l'acquisition automatique d'information lexicale, l'extraction de faits hors ligne et la recherche de passages.*

KEYWORDS: *question answering, syntactic analysis, dependency relations, CLEF, question analysis, passage retrieval, answer extraction, information extraction.*

MOTS-CLÉS : *système de question-réponse, analyse syntaxique, relations de dépendance, CLEF, analyse de questions, recherche de passages, extraction de réponses, extraction d'information.*

1. Introduction

Joost is a monolingual question answering (QA) system for Dutch which makes heavy use of syntactic information. There are two strategies implemented in the system: a table look-up strategy and a retrieval based strategy. Most questions are answered by retrieving relevant paragraphs from the document collection, using keywords from the question. Potential answers are identified in these paragraphs and ranked using a number of clues. Apart from obvious clues such as matching keywords, we use syntactic structure to identify and rank answer strings. A second strategy is based upon the observation that certain question types can be anticipated, and the corpus can be searched off-line for answers to such questions. Whereas previous approaches have used regular expressions to extract the relevant facts, we use patterns of dependency relations.

To enable both question answering strategies, the full document collection has been analysed syntactically. In this article we describe both strategies in detail with the emphasis on the application of deep syntactic analysis in the QA modules. We focus on open-domain question answering using data provided by the Cross Language Evaluation Forum (CLEF) for the Dutch QA track. In addition, we also worked on improving QA on a closed domain task of medical questions.

In the next section we give an overview of the general architecture of our QA system. Thereafter, we discuss the building blocks of the system in detail with the focus on the incorporation of syntactic information. Finally, we present results of our system on the CLEF QA task and summarise this article with some conclusions and prospects for future work.

2. Related Work

Several researchers have attempted to use syntactic information, and especially dependency relations, in QA. Most research is done in the field of answer extraction. One approach is to look for an exact match between dependency tuples derived from the question and those present in a potential answer (Katz *et al.*, 2003; Litkowski, 2004). Attardi *et al.* (2002) and Mollá *et al.* (2005) compute the match between question and answer using a metric which basically computes the overlap in dependency relations between the two. Punyakanok *et al.* (2004) compute the tree edit distance between the dependency trees of the question and answer, and select answers from sentences which minimise this distance. They employ an approximate tree matching approach that allows one to disregard subtrees in potential answer sentences.

Other studies have shown that syntactic information is useful in other modules of common QA systems as well. Tellex *et al.* (2003) concluded after a thorough evaluation of passage retrieval algorithms that neglecting relations between words is a major source of false positives for retrieval systems based on lexical matching. Many irrelevant passages do share lexical items with the question, but the relations between these items may differ from the relations in the question. Cui *et al.* (2004) have used dependency relations for two QA modules, namely passage retrieval (Cui *et al.*, 2005) and answer selection (Cui *et al.*, 2004). For passage retrieval they propose a fuzzy relation matching based on statistical models. They show that the method using dependency relations outperforms standard passage retrieval methods by up to 78% in mean reciprocal rank. It also gives 50% improvement in a system enhanced by query expansion. Their answer extraction approach using dependency relations also produces a significant improvement over the baseline system. The improvement is strongest for questions that do not require a specific type of named entitiy as answer.

Several teams working on QA systems have investigated the use of text patterns to find answers. Soubbotin *et al.* (2001) present a question answering mechanism which uses predefined surface patterns to extract potential answer phrases. After their system achieved the best performance at the TREC-10 evaluation in 2001 more research teams working in the field of corpus-based QA became interested in this technique. Fleischman *et al.* (2003) were the first to present a strategy in which patterns are used to extract answers off-line, before the questions are asked. They evaluated their system on “Who is ..." questions (e.g. person identification: *Who is the mayor of Boston?* and person definition: *Who is Jennifer Capriati?*) against a state-of-the-art web-based QA system. Results indicated that their system answered 25% more questions correctly when it used the extracted information. Jijkoun *et al.* (2004) used dependency relations for the extraction of answers off-line. The results showed a significant improvement in recall over systems based on regular expression pattern matching.

Our work combines the results of previous work by taking a fully parsed text collection as starting point, and using syntactic information in all components of the QA system.

3. General Architecture of Joost

In this section we briefly describe the general architecture of our QA system Joost. Details about its components will be given in the next section. The architecture of our system is depicted in figure 1. Apart from the three classical components *question analysis*, *passage retrieval* and *answer extraction*, the system also contains a component called QATAR, which is based on the technique of extracting answers off-line. All components in our system rely heavily on syntactic analysis, which is provided by Alpino (Bouma *et al.*, 2001; van Noord, 2006), a wide-coverage dependency parser for Dutch. Alpino is used to parse questions as well as the full document collection from which answers need to be extracted. A brief overview of the components of our QA system follows below.

The first processing stage is question analysis. The input to this component is a natural language question in Dutch, which is parsed by Alpino. The goal of question analysis is to determine the question type and to identify keywords in the question.

Depending on the question type the next stage is either passage retrieval or table look-up (using QATAR). If the question type matches one of the table categories, it

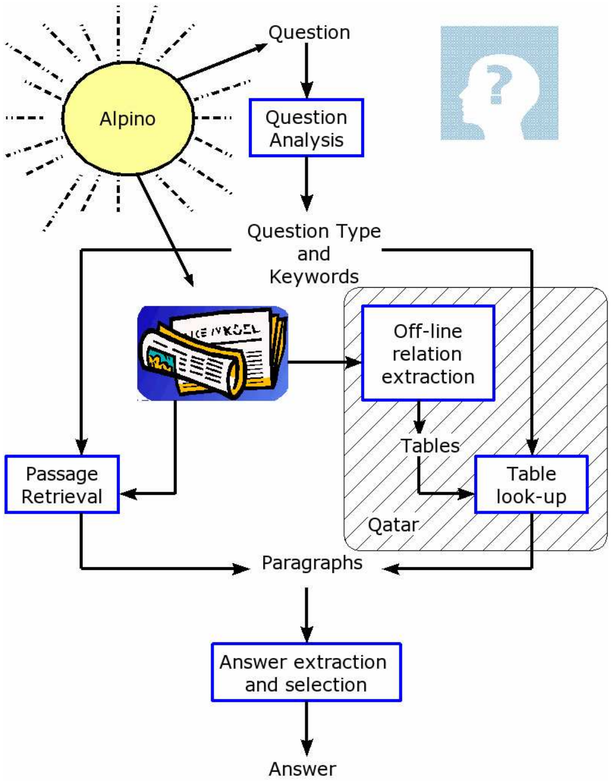

Figure 1. *System architecture of Joost*

will be answered by QATAR. Tables are created off-line for relations that frequently occur in fixed patterns. We store these relations as potential answers together with the IDs of the paragraphs in which they were found. During the question answering process the question type determines which table is selected (if any).

For all questions that cannot be answered by QATAR (either because the type of the question is inappropriate, or because there is no answer available in the relevant table), we follow the other path through the QA-system to the passage retrieval component. Our previous experiments showed that a segmentation of the corpus into paragraphs is most efficient for information retrieval (IR) performance in QA. Hence, IR passes relevant paragraphs to subsequent modules for extracting the actual answers from these text passages.

The final processing stage in our QA-system is answer extraction and selection. The input to this component is a set of paragraph IDs, either provided by QATAR or by the IR system. We then retrieve all sentences from the text collection included in these paragraphs. For questions that are answered by means of table look-up, the tables provide an exact answer string. In this case the context is used only for ranking the answers. For other questions, answer strings have to be extracted from the paragraphs returned by IR. The features that are used to rank the extracted answers will be explained in detail below. Finally, the answer ranked first is returned to the user.

4. Components of the System

In this section, we discuss the components of our QA system in detail. Subsection 4.1 gives an overview of the linguistic modules that are used in Joost. Thereafter, we describe the core components of Joost, namely question analysis, table look-up, passage retrieval, and answer extraction.

4.1. *Linguistic Analysis*

Joost incorporates several modules for linguistic analysis of Dutch text. First of all, the dependency parser Alpino has been integrated in the system. For any given input string, it provides a dependency analysis and labels for the named entities found in the input. Second, a module for reasoning over syntactic dependency relations using equivalence rules has been implemented. This module is used in several components of the QA system and accounts for the fact that there is a certain amount of systematic syntactic variation in the way information is expressed in text. Finally, automatically acquired lexical knowledge is used to improve the performance of the system on a number of specific question types. Lexical knowledge was acquired by extracting specific syntactic relations (such as appositions) from the parsed text collection, and by computing distributional similarity based on the syntactic distribution of words.

4.1.1. *Alpino*

The Alpino system is a linguistically motivated, wide-coverage, grammar and parser for Dutch in the tradition of HPSG. It consists of over 600 grammar rules and a large lexicon of over 100,000 lexemes and various rules to recognize special constructs such as named entities, temporal expressions, etc. Heuristics have been implemented to deal with unknown words and word sequences, and ungrammatical or out-of-coverage sentences (which may nevertheless contain fragments that are analysable). The grammar provides a 'deep' level of syntactic analysis. The output of the system is a dependency graph. van Noord (2006) shows that the accuracy of the system (fscore of named dependency triples), when evaluated on a test-set of 1400 newspaper sentences, is about 91%.

Alpino includes heuristics for recognising named entities. For the QA task, classification of these entities was added. To this end, we collected lists of personal names (120K), geographical names (12K), organisation names (26k), and miscellaneous items (2K). The data was primarily extracted from the Twente News Corpus, a collection of over 300 million words of newspaper text, which comes with relevant annotation. For unknown names, a maximum entropy classifier was trained, using the Dutch part of the shared task for CONLL 2003.[1] The accuracy on unseen CONLL data of the resulting classifier which combines dictionary look-up and a maximum entropy classifier is 88.2%.

The Dutch text collection for CLEF, which contains all articles from the 1994 and 1995 edition of two Dutch newspapers, was tokenised and segmented into 4.1 million sentences, and parsed in full. We used a Beowulf Linux cluster of 128 Pentium 4 processors[2] to complete the process in about three weeks. The dependency trees are stored as XML. For our closed domain medical QA system, we recently also parsed a 3 million word heterogeneous medical text collection, including reference works such as encyclopedias and manuals, and web documents, as well as the Dutch (October 2005) version of Wikipedia (25 million words).

4.1.2. *Reasoning over Dependency Relations*

We have implemented a system in which dependency patterns derived from the question must be matched by equivalent dependency relations in a potential answer. The dependency analysis of a sentence gives rise to a set of dependency relations of the form $\langle\texttt{Head/HIx, Rel, Dep/DIx}\rangle$, where `Head` is the root form of the head of the relation, and `Dep` is the head of the dependent. `HIx` and `DIx` are string indices, and `Rel` the dependency relation. For instance, the dependency analysis of sentence (1-a) is (1-b).

(1) a. Mengistu kreeg asiel in Zimbabwe
Mengistu received asylum in Zimbabwe
Mengistu received asylum in Zimbabwe

b. $$\left\{ \begin{array}{ll} \langle\texttt{krijg/2, su, mengistu/1}\rangle, & \langle\texttt{krijg/2, obj1, asiel/3}\rangle, \\ \langle\texttt{krijg/2, mod, in/4}\rangle, & \langle\texttt{in/4, obj1, zimbabwe/5}\rangle \end{array} \right\}$$

Here `su` stands for *subject*, `obj1` is used for *direct object*, and `mod` is used for *modifier*. Other relations include `vc` (*verbal complement*), `wh` (the WH-phrase in a question), `app` (*apposition*), `det` (*determiner*).

A dependency pattern is a set of partially under-specified dependency relations:

(2) $$\left\{ \langle\texttt{krijg/K, obj1, asiel/A}\rangle, \ \langle\texttt{krijg/K, su, S/I}\rangle \right\}$$

A pattern may contain variables, represented here by strings starting with a capital letter. A pattern P matches a set of dependency relations R if $P \subset R$, under some substitution of variables.

[1] `http://cnts.uia.ac.be/conll2003/ner/`

[2] Provided by the High-Performance Computing center of the University of Groningen.

Equivalences can be defined to account for syntactic variation. For instance, the subject of an active sentence may be expressed as a PP-modifier headed by *door* (*by*) in the passive:

(3) Aan Mengistu werd asiel verleend door Zimbabwe
To Mengistu was asylum given by Zimbabwe
Mengistu was given asylum by Zimbabwe

The following equivalence accounts for this:

(4) $$\{\langle\texttt{V/I,su,S/J}\rangle\} \quad \Leftrightarrow \quad \left\{ \begin{array}{c} \langle\texttt{word/W,vc,V/I}\rangle, \\ \langle\texttt{V/I,mod,door/D}\rangle, \\ \langle\texttt{door/D,obj1,S/J}\rangle \end{array} \right\}$$

Here, the verb *word* is the root form of the passive auxiliary, which takes a verbal complement headed by the verb `V`.

Given an equivalence $Lhs \Leftrightarrow Rhs$, substitution of Lhs in a pattern P by Rhs gives rise to an equivalent pattern P'. A pattern P now also matches with a set of relations R if there is some equivalent pattern P', and $P' \subset R$, under some substitution of variables.

We have implemented 14 equivalence rules, to account for, among others, word order variation within appositions, the equivalence of genitives and PPs headed by the preposition *van* (*of*), equivalence of appositions and simple predicative sentences, coordination, and relative clauses. In Bouma *et al.* (2005), we show that the inclusion of equivalence rules has a positive effect on various components of our QA system, i.e. answer analysis, off-line relation extraction and answer selection.

4.1.3. *Lexical Knowledge*

Joost employs knowledge of the meaning of words to improve question analysis, off-line relation extraction and answer selection. To this end, Joost incorporates information extracted from Dutch EuroWordNet (EWN) (Vossen, 1998). To improve the coverage, we have extended this knowledge base considerably by acquisition techniques, described in more detail below.

There are two types of lexical information which are employed to improve our QA system: hypernym relations between common nouns (e.g. the fact that *leader* is a hypernym of both *chairman* and *president*, and hypernym relations between proper names and common nouns (for instance the fact that *Seles* is the name of a tennis player). We distinguish the two types since EWN only includes information of the first type (there are no proper names in EWN), and our lexical acquisition techniques (presented in further detail in van der Plas *et al.* (2006)) are different for the two types as well.

Hypernym relations between common nouns are used for the question analysis module described in section 4.2, and for the construction of the tables of QATAR. For example, the pattern for the extraction of function roles in QATAR makes use of a list of words such as *minister, president, chairman* ... This list is taken from Dutch EWN, and consists of all words under the node *leider* (*leader*), 255 in total. However, the coverage of this list, when tested on a newspaper corpus, is far from complete. Many semantically similar words that occur frequently in newspaper text are missing (i.e. Dutch equivalents of *banker, boss, national team coach, captain, secretary-general*, etc.). We therefore employed a technique based on distributional similarity to extend this list automatically. This technique is now explained shortly as follows.

Syntactic relations have been shown to provide information which can be used to acquire clusters of semantically similar words automatically (Lin, 1998). The underlying assumption of this approach is that semantically similar words are used in similar syntactic contexts. Dependency tuples containing the target word, an accompanying word and the syntactic relation between them are used to extract the syntactic context of the target word. Apart from the commonly used subject and object relations we also apply the following grammatical relations: adjective, coordination, apposition and prepositional complement. For each word a context vector is constructed that consists of all grammatical relations a word is found in with the accompanying word attached to it. There are several similarity measures available for computing the similarity between these vectors. We have used the best scoring measures from Curran *et al.* (2002) to get a ranked list of semantically related words for each target word. We have evaluated our results against EWN using the Wu and Palmer measure (Wu *et al.*, 1994) to calculate the EWN similarity between a pair of words.. We gained a EWN similarity score of 0.60 when only taking the highest ranked similar word into account and 0.52 when taking the 10 highest ranked words. The baseline that simply outputs random words receives a score of 0.26.

To improve recall of the extraction of function roles, we extended the list of 255 words under the node *leider* (*leader*) obtained from EWN with distributionally similar words obtained with the technique explained above. After a semi-automatic selection, 644 valid nouns were merged with the original EWN list, to form a list of 899 words which was then used for the off-line relation extraction process, as well as the question analysis module.

Hypernym relations between proper names and common nouns are used to label proper names. These labels describe an IS-A relation with the proper names (e.g., Seles is_a tennis player; Estonia is_a ferry). The labels of proper names are used for instance in the answer extraction module described in section 4.5, and in the anaphora resolution component of QATAR.

Both Pasca (2004) and Pantel *et al.* (2004) describe methods for acquiring labels for proper names from large text corpora and evaluate the results in the context of web search and question answering. Pantel *et al.* (2004) use the apposition relation to find potential labels for proper names. We have used the apposition relation and recently we added the nominal predicate complement (i.e., *Guus Hiddink is a coach*). We extracted 342 K proper names with their label. More than 90% of the data is found using the apposition relation. The rest is found by scanning the corpus for the nominal predicate complement. This type of knowledge has proven useful for answering WHICH-questions and definition questions.

4.2. *Question Analysis*

Each incoming question is parsed by Alpino. To improve parsing accuracy on this specific task, the disambiguation model was retrained on a corpus which contained annotated and manually corrected dependency trees for 650 quiz questions.[3] For CLEF 2005, we used a model which was trained on data which also included manually corrected dependency trees of the CLEF 2003 and 2004 questions. It achieved an accuracy of 97.6 on CLEF 2005 questions.

On the basis of the dependency relations returned by the parser the question type is determined. Joost distinguishes between 29 different question types. 18 question types are related to the relation tuples that were extracted off-line. Note that a single relation can often be questioned in different ways. For instance, whereas a frequent question type asks for the meaning of an acronym (*What does the abbreviation RSI stand for?*), a less frequent type asks for the abbreviation of a given term (*What is the abbreviation of Mad Cow Disease?*). The other 11 question types identify questions asking for an amount, the date or location of an event, the first name of a person, the name of an organisation, HOW-questions, WHICH-questions, and definition questions.

For each question type, one or more syntactic patterns are defined. For instance, the following pattern accounts for questions asking for the capital of a country:

$$(5) \qquad \left\{ \begin{array}{ll} \langle\texttt{wat/W, wh, is/I}\rangle, & \langle\texttt{is/I, su, hoofdstad/H}\rangle \\ \langle\texttt{hoofdstad/H, mod, van/V}\rangle, & \langle\texttt{van/V, obj1, Country/C}\rangle \end{array} \right\}$$

Depending on the question type, it is useful to identify one or two additional arguments. For instance, the dependency relations assigned to the question *Wat is de hoofdstad van Togo?* (*What is the capital of Togo?*) match with the pattern in (5), and instantiate `Country` as *Togo*. Therefore, the question type `capital` is assigned, with `Togo` as its argument: `capital(Togo)`. Similarly, *Who is the king of Norway?* is classified as `function(king,Norway)`, and *In which year did the Islamic revolution in Iran start?* is classified as `date(revolution)`.

Some question types require access to lexical semantic knowledge. For instance, to determine that *In which American state is Iron Mountain located?* asks for a location, the system needs to know that *state* refers to a location; to determine that *Who is the advisor of Yasser Arafat?* should be classified as `function(advisor,Yasser`

[3] From the *Winkler Prins spel*, a quiz game made available to us by *Het Spectrum.*

`Arafat)`, it needs to know that *advisor* is a function. We obtained such knowledge mainly from EWN. As already mentioned in section 4.1.3, the list of function words (indicating function roles such as *president, queen, captain, secretary-general, etc.*) was expanded semi-automatically with words from the corpus that were distributionally similar to those extracted from EWN.

Question classification was very accurate for the CLEF 2005 questions. However, there were a few cases where the additional arguments selected by the system did not seem the most optimal choice. Two clear mistakes were found. One of them was the following: The question *What is the currency of Peru?* was classified as `currency(of)` and not as `currency(Peru)`. The other mistake was caused by the parser.

4.3. *QATAR - Question Answering by Table Look-Up and Relations*

4.3.1. *Off-line Relation Extraction*

Off-line methods (Fleischman *et al.*, 2003) can be used to improve the performance of the system on questions for which the answers frequently occur in fixed patterns. For example, for a question asking who fulfills a certain role within an organisation, the answer can often be found in appositions:

(6) W.F. Selman, voorzitter van Unilever, zei dat ...
W.F. Selman, chair of Unilever, said that ...

In off-line QA plausible answers to questions are extracted before the actual question has been asked. Jijkoun *et al.* (2004) showed that an extraction method based on a small number of simple syntactic patterns allows an off-line QA system to correctly answer substantially more questions than a method based on surface text patterns. By using dependency based patterns it becomes possible to extract instances of relations consisting of terms that are not necessarily adjacent on the surface level. Bouma *et al.* (2005) describe how syntactic patterns are used to extract answers. The following syntactic pattern serves to extract ⟨*Person,Role,Organisation*⟩-tuples from the corpus:

$$(7) \quad \left\{ \begin{array}{l} \langle \texttt{Role/R,app,Person/\_} \rangle, \\ \langle \texttt{Role/R, mod, van/V} \rangle, \\ \langle \texttt{van/V, obj1, Organisation/\_} \rangle \end{array} \right\}$$

Here, the apposition provides the *Person* argument of the relation and the object of the preposition *van* which is a dependent of *Role* provides the name of the *Organisation*. The noun *Role* has to match with one of the words in the list which consists of relevant words extracted from EWN and extended with distributionally similar words obtained using the techniques described in 4.1.3.

Off-line methods are not only used for function questions. For open-domain QA, tables are constructed for other question types such as age of a person, the location and date of birth of a person, the date and cause of death of a person, and various other question types for which the answers are likely to appear according to predictable patterns.

Other types of questions require different approaches than the ones used for factoid questions. Answers to definition questions can be extracted from sentences containing a copular verb and a nominal predicate (*X is a Y*). In Fahmi *et al.* (2006) it is shown that syntactic features of these sentences can be used to improve the accuracy of an automatic classifier which distinguishes definitions from non-definitions in the extracted data set. Medical questions asking for causes, symptoms, and treatments of diseases require answers that cannot be identified by looking for specific named entities. Answers to such questions can be extracted off-line by using syntactic dependency patterns which extract general NPs as the arguments of the relevant relation.

When the question analysis component has assigned a type to a question that is matched by the relation tuples extracted off-line, the keywords are used to look-up the answers and the paragraph IDs in the appropriate table. We select the matching answers with the highest frequency. These answers together with their paragraph IDs are passed on to the next processing stage, the answer extraction and selection component described in section 4.5.

4.3.2. *Anaphora Resolution*

We have used anaphora resolution to expand the coverage of the QATAR tables. Consider the following question:

(8) How old is Ivanisevic?

In order to extract the answer from the text provided below we have to analyse it not only at sentence level, but at discourse level as well.

(9) Yesterday, Todd Martin was the opponent of Ivanisevic in the final of the Grand Slam Cup in Berlin. The American, who defeated local hero Boris Becker a day earlier, was beaten by the 26-year old Croatian in straight sets.

Among other things, one must correctly identify *Ivanisevic*, located in the first sentence, as the denotation of *the Croatian*, located in the second sentence, in order to extract the correct answer that is stated in the second sentence. Semantic knowledge such as the IS-A relations between *Ivanisevic* and *Croatian* can help to resolve the anaphoric connection between these two entities. In section 4.1.3 we explained how such IS-A relations can be acquired automatically for proper names. Anaphora resolution supported by IS-A relations may help to extract potential answers from the text collection if they are not clearly stated with the accompanying proper name in the same sentence but in the context of the discourse.

	original	anaphora
age	17.038	20.119
born_date	1.941	2.034
born_loc	753	891
died_age	847	885
died_date	892	1.061
died_how	1.470	1.886
died_loc	642	646

Table 1. *Number of facts (types) found for the different tables for off-line relation extraction*

More specifically, we try to resolve definite NPs that refer to named entities. Our strategy is as follows: We scan the left context of a definite NP for proper names from right to left (i.e. the closest proper name is selected first). For each proper name we encounter, we check whether it is in an IS-A relation with the definite NP by consulting the lexical knowledge base. If so, the named entity is selected as the antecedent of the NP. As long as no suitable proper name is found we select the previous proper name and so on until we reach the beginning of the document. If no suitable named entity is found, i.e., no proper name is found that is in an IS-A relation with the definite NP, we use a fallback procedure. This fallback selects the proper name in the previous sentence, that is nearest to the anaphoric expression. If no proper name is present in the previous sentence, the NP is not resolved. If the NP is resolved, the fact is added to the appropriate relation table.

In order to explain our strategy for resolving definite NPs we will apply it to the example above. The left context of the NP *the 26-year old Croatian* is scanned from right to left. The proper name *Boris Becker* is selected before the correct antecedent *Ivanisevic*. The fact that *Boris Becker* is not found in an IS-A relation with *Croatian* puts it aside as an unsuitable candidate. Then *Ivanisevic* is selected and this candidate *is* found to be in an IS-A relation with *Croatian*, so *Ivanisevic* is taken as the antecedent of *Croatian*. And the fact `Ivanisevic, 26-year old` is added to the relevant QATAR table.

Using anaphora resolution in off-line relation extraction leads to improvements in terms of coverage, as can be seen in table 1. The added facts fall into two categories: they are either facts that were already present in the original table or facts that are new. In table 1 we show the number of new facts (types). It should be noted that the facts that are not new do contribute to the overall reliability of the table, as facts that are found more frequently are more reliable than facts that are found only once.

We randomly selected 400 facts that were found using anaphora resolution (including both facts that were new, and facts that were already present in the original

	correct	incorrect
new	168	128
increase freq.	95	9
total	263	137

Table 2. *Evaluation of sample of 400 facts found by anaphora resolution*

table). Two human experts determined the correctness of those facts. The results are given in table 2.

A large number of our sample (22.75%) comprises already known facts with increased frequencies. This is a positive result with regard to the reliability of the tables. The precision of the new facts however is not very encouraging (about 56.8%). Using a slightly different technique without fallback strategy did yield a high precision for new facts added by anaphora resolution (Mur *et al.*, 2006). However, the number of new facts was disappointing. Therefore, we included the fallback strategy in our current experiments.

4.4. *Linguistically Informed IR*

Information retrieval (IR) is used in most QA systems to filter out relevant passages from large document collections to narrow down the search carried out by answer extraction modules in a QA system. Accurate IR is crucial for the success of this approach. Answers in paragraphs that have been missed by IR are lost for the entire QA system. Hence, high performance of IR especially in terms of recall is essential. Furthermore, high precision is also desirable as IR scores are used for ranking potential answers. The chance of extraction errors in subsequent modules is also smaller if precision is high.

4.4.1. *Indexing with Linguistic Features*

Given a full syntactic analysis of the text collection, it becomes feasible to exploit linguistic information as a knowledge source for IR. Using Apache's IR system Lucene (Jakarta, 2004), we can index the document collection along various linguistic dimensions, such as part of speech tags, named entity classes, and dependency relations. We defined several *layers* of linguistic features and feature combinations extracted from syntactically analysed sentences and included them as index fields in our IR component. Table 3 lists the layers that we use. The table illustrates each layer with example index tokens for one sentence from the CLEF corpus.

Note that Dutch stemming and stop word removal is applied internally by Lucene for the text field. In this way, the text field corresponds to a basic plain text retrieval index. Observe furthermore that the *compound* field contains compositional compounds

layers for each word in each paragraph		
text	plain text tokens	Het embargo tegen Irak werd ingesteld na de inval in Koeweit in 1990
root	linguistic root forms	het embargo tegen Irak word stel in na de inval in Koeweit in 1990
RootPOS	root + POS tag	het/det embargo/noun tegen/prep Irak/name word/verb stel_in/verb na/prep de/det inval/noun in/prep Koeweit/name in/prep 1990/noun
RootRel	root + relation (to its head)	het/det embargo/su tegen/mod Irak/obj1 word/ stel_in/vc na/mod de/det inval/obj1 in/mod Koeweit/obj1 in/mod 1990/obj1
RootHead	root (dependent) + root (head)	het/embargo embargo/word tegen/embargo Irak/tegen word/ stel_in/word na/stel_in de/inval inval/na in/inval Koeweit/in in/inval 1990/in
RootRelHead	dependent + relation + head	het/det/embargo embargo/su/word tegen/mod/embargo Irak/obj1/tegen word// stel_in/vc/word na/mod/stel_in de/det/inval inval/obj1/na in/mod/inval Koeweit/obj1/in in/mod/inval 1990/obj1/in
layers for selected words in each paragraph		
compound	compounds	stel_in
ne	named entities	Irak Koeweit
neLOC	location names	Irak Koeweit
nePER	person names	
neORG	organisation names	
neTypes	labels of named entities	LOC LOC YEAR

Table 3. *Index layers defined and example tokens from the Dutch sentence:* Het embargo tegen Irak werd ingesteld na de inval in Koeweit in 1990. *(The embargo against Iraq has been established after the invasion of Kuwait in 1990.)*

as well as particle verbs as shown in table 3 (this information is present in the output of the parser). The latter are included in this field even in cases where particle and verb are not concatenated. On the other hand, compounds and particle verbs are always split into their component words in the *root* field. We also split on hyphens (for instance *Noord-Korea* to *Noord Korea*). In named entity layers (*ne, neLOC, nePER, neORG*) both versions are added (original root form and compounds split into their components words). The *neTYPES* layer contains labels of named entities and other special units such as temporal expressions or measurements, one for each unit in the paragraph.

```
text:(stelde Verenigde Naties +embargo +Irak)
ne:(Verenigde_Naties^2 Verenigde^2 Naties^2 Irak^2)
RootHead:(Irak/tegen embargo/stel_in)
neTypes:(YEAR)
```

Table 4. *An example IR query from the question* Wanneer stelde de Verenigde Naties een embargo in tegen Irak? *(When did the United Nations establish an embargo against Iraq?) using the following keyword selections: (1) all plain text tokens (except stop words), (2) named entities weighted with boost factor 2, (3) RootHead bigrams for all words tagged as noun, (4) the question type transformed into a named entity class, (5) plain text keywords of words in an object relation (embargo & Irak)*

Given the patterns defined in table 3 we index each paragraph in the text collection. The task is now to make use of this rich information by appropriate queries. It has been shown before that it is important to select linguistic features carefully in order to be successful in tasks like information retrieval (Katz *et al.*, 2003). Hence, we will focus on optimising keyword selection and weighting in the remaining part of this section.

4.4.2. *Query Formulation and Optimisation*

Questions are analysed using Alpino in the same way as sentences in the document collection. We extract the same features and feature combinations as was done for producing the IR index. Query keywords are produced from questions using features corresponding to fields in the index. Furthermore, we allow additional constraints to carry out a fine-grained selection of such keywords. We define constraints on part-of-speech and relation type. For example, we may select keywords of type *RootHead* for all words in the question that have been tagged as *nouns*.

Furthermore, keyword selections may be of different importance for the success of a query. Lucene's query language allows one to set weights (so-called *boost factors*) and 'required' markers (using the '+' character) to any keyword in a query.

Different keyword selections (using certain features, constraints and weights) are combined in a disjunctive way to form complex queries. Keywords from different selections that query the same index field are combined using simple heuristics (more specific constraints overwrite less specific ones and 'required' markers overwrite boost factors). In our experiments we limit ourselves to restrictions on a small sub-set of part-of-speech labels (noun, adjective and verb) and a small sub-set of dependency relation types (direct object, modifier, apposition and subject). We also stipulate that relation type constraints are more specific than part-of-speech constraints.

Finally, we also use the question type produced by question analyses. In many cases we are looking for named entities which are answers to factoid questions. Therefore, we match question types with expected answer types in terms of named entities

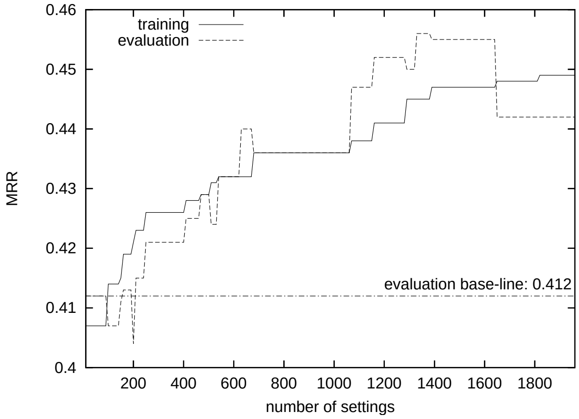

Figure 2. *IR Parameter optimisation using a genetic algorithm*

(if applicable) in order to match keywords in the *neTypes* layer of the index. Figure 4 shows an example query assembled from a number of keyword selections.

According to the definitions above we can now construct various keyword selections that can be weighted in many ways. The problem is to select appropriate constraints for possible keyword types that result in improved retrieval performance for the QA task. Furthermore, we also have to find optimal weights for our selected keywords. Keyword selection and weighting is optimised using an automatic learning technique described as follows.

For optimisation we applied a genetic algorithm that runs iteratively through randomised IR settings in order to optimise query parameters according to a given training set (taken from the Dutch QA task at CLEF). Essentially, we initialise the process with basic settings (querying one index layer per setting, using any possible constraint on keywords, using default weights but no 'required' markers) and then combine a fixed number of preferable settings (according to the fitness function used in the genetic algorithm) to test new parameters. This combination together with random variation is repeated until no improvement can be measured anymore. A fitness score is determined for each setting by evaluating the results obtained for a training set of questions. We use the mean reciprocal rank of the first five answers produced by the system. Details of the genetic optimisation process are given in Tiedemann (2006). As the result of the optimisation we obtain an improvement of about 10% over the baseline using standard plain text retrieval (i.e. the text layer only) on unseen evaluation data. Figure 2 illustrates the optimisation process for about 2000 IR settings tested. It should be noted that this improvement is not solely an effect of using root forms or named entity labels, but that many of the features that are assigned a high weight by the genetic algorithm refer to layers that make use of dependency information.

4.5. *Answer Extraction and Selection*

For questions that are answered by means of table look-up, the relation table provides an exact answer string. For other questions, it is necessary to extract answer strings from the set of paragraphs returned by the passage retrieval component. Given a set of paragraph IDs, we retrieve from the parsed corpus the dependency relations for the sentences occurring in these paragraphs.

4.5.1. *Selecting Answers with Syntactic Patterns*

Various syntactic patterns are defined for exact answer extraction. For questions asking for the name of a person, organisation, or location, or for an amount or date, a constituent headed by a word with the appropriate named entity class has to be found. As all of these occur frequently in the corpus, usually many potential answers will be identified. An important task is therefore to rank potential answers.

The following features are used to determine the score of a short answer A extracted from sentence S:

Syntactic Similarity: The proportion of dependency relations from the question which match with dependency relations in S.

Answer Context: A score for the syntactic context of A.

Names: The proportion of proper names, nouns, and adjectives from the query which can be found in S and the sentence preceding S.

Frequency: The frequency of A in all paragraphs returned by IR.

IR: The score assigned to the paragraph from which A was extracted.

The score for syntactic similarity implements a preference for answers from sentences with a syntactic structure that overlaps with that of the question. Equivalence rules as defined in section 4.1.2 are used to reason over dependency relations when matching syntactic structures. Answer context implements a preference for answers that occur in the context of certain terms from the question. Given a question classified as `date(Event)`, for instance, date expressions which occur as a modifier of `Event` are preferred over date expressions occurring as sisters of `Event`, which in turn are preferred over dates which have no syntactic relation to `Event`. These preferences are manually tuned for each of the various question types.

The overall score for an answer is the weighted sum of these features. Weights were determined manually using previous CLEF data for tuning. The highest weights are used for Syntactic Similarity and Answer Context. The highest scoring answer is returned as the answer.

Ranking of answers on the basis of various features was initially developed for IR-based QA only. Answers found by table look-up were ranked only by frequency. Recently, we have started to use the scoring mechanism described above also for answers found by QATAR. As the tables contain pointers to the sentence from which a tuple was extracted, we can easily go back to the source sentence, and apply the scoring mechanisms described above.[4] Using more features to rank an answer provides a way to give the correct answer to questions like *Who is the German minister of Economy?*. The function table contains several names of German ministers, but does not distinguish between different departments. The most frequent candidate is *Klaus Kinkel* (54 entries), who is minister of foreign affairs. The correct name, *Günter Rexrodt*, occurs only 11 times. Using Syntactic Similarity and Names as additional features, Joost gives the correct answer.

4.5.2. *Special Cases*

4.5.2.1. WHICH-questions

WHICH-questions, such as (10), are relatively difficult to answer. Whereas for most question types, the type of the answer is relatively clear (i.e. it should the name of a person or organisation, or a date, etc.), this is not the case for WHICH-questions.

(10) a. Which fruit contains vitamin C?
b. Which ferry sank southeast of the island Utö?

To improve the performance of our system on such questions, we make use of two additional knowledge sources. From EWN, we imported all hypernym relations between nouns. Question (10-a) is assigned the question type `which(fruit)`. We use the hypernym relations to assign a higher score to answers which are hyponyms of `fruit`.[5]

As EWN includes very few proper names, we also used the IS-A relations extracted from appositions and nominal predicate complements containing a proper name, as described in section 4.1.3. Consider question (10-b) above. Question analysis classifies this as a question of type `which(ferry)`. Candidate answers that are selected by our system are: *Tallinn, Estonia, Raimo Tiilikainen* etc. Apart from other heuristics, potential answers which have been assigned the class corresponding to the question stem (i.e. `ferry` in this case) are ranked higher than potential answers for which this class label cannot be found in the database of IS-A relations. Since *Estonia* is the only potential answer which IS-A ferry, according to our database, this answer is selected.

Adding ISA-relations as an additional knowledge source for answering WHICH-questions from the CLEF data set improves the MRR score by 13% and improves the CLEF score by 15% (van der Plas *et al.*, 2006).

[4] As no IR is involved in this case, the IR score is set to 1 for all answers.

[5] Unfortunately, EuroWordNet only contains two hyponyms for the synset *fruit*, neither of which could be used to identify an answer to (10-a).

4.5.2.2. Definition Questions

An important category in CLEF 2005 are questions asking for the definition of a person or organisation (i.e. *What is Sabena?, Who is Antonio Matarese?*). No less than 60 questions were of this type. Again, we used the IS-A relations extracted from appositions and nominal predicate complements to answer such questions. Frequency is important to ensure that an appropriate class is chosen. The named entity *Sabena* for instance occurs frequently in the corpus, but often with class labels that are not suitable for inclusion (*possibility, partner, company,...*). By focusing on the most frequent class label assigned to a named entity (*airline company* in this case), we hope to select the most appropriate label for a definition. A disadvantage of this technique is that the class label by itself is not always sufficient for an adequate definition. Therefore, we expand the class labels with modifiers which typically need to be included in a definition. In particular, our strategy for answering definition questions consists of two steps:

– Phase 1: The most frequent class found for a proper name is taken.

– Phase 2: The sentences that mention the proper name and the selected class are searched for additional relevant information, e.g., words in an adjectival relation or prepositional complements of the proper names.

For the example above, our system first selects *airline company* as the most frequent proper name class (phase 1) and then adds the attached adjective *Belgian* from the highest ranked answer sentence (phase 2) to produce the final answer *Belgian airline company*.

Using lexical information to provide answers to definition questions in the CLEF data set improves the MRR scores by about 11% and the CLEF score by 13% (van der Plas *et al.*, 2006).

Recently, we experimented with supervised machine learning techniques to learn the identification of medical concept definitions, such as *What is a runner's knee?*, based on syntactically analysed text. In Fahmi *et al.* (2006) several learning approaches and feature settings were explored to classify sentences taken from Wikipedia to be either a definition or not. The best performance was achieved with a maximum entropy classifier using the following features:

Text properties: bag-of words & bigrams (punctuations included)

Document properties: position of the sentence in the document

Syntactic properties: position of the subject in the sentence (initial or non-initial); type of the determiner of the subject and of the predicative complement (definite, indefinite, other)

The classifiers are trained on manually annotated data containing 1336 definitions and 963 non-definitions. The automatically trained classifier yields significantly

	# Q	# correct	% correct
Factoid Questions	114	62	54.4
Temporally Restricted Questions	26	7	26.9
Definition Questions	60	30	50.0
Total	200	99	**49.5**

Table 5. *Official CLEF results (Dutch QA@CLEF 2005)*

higher accuracy (91.67%) than the baseline that picks the first sentence in a Wikipedia document (which gives already an accuracy of about 75.9%). Other features such as named entity tags have been tested as well but the best performance is achieved without using these.

5. Evaluation

5.1. *CLEF 2005*

For evaluation we used data from CLEF. The CLEF text collection for Dutch contains 2 years of text taken from 2 Dutch daily newspapers. It comprises about 4.1 million sentences in about 190,000 documents. The question sets from the competitions in 2003 and 2004 have mainly been used for development purposes to prepare our participation in the Dutch QA track of CLEF 2005. Questions in these sets are annotated with valid answers found by the participating teams including IDs of supporting documents in the given text collection that contain the answers.

Our system performed best among the Dutch QA systems and came third in the overall evaluations of all monolingual QA systems in the CLEF competition in 2005. The official results of the CLEF 2005 evaluation are given in table 5. The scores are satisfactory for factoid questions and definitions. We can see that the system performed significantly less well on temporally restricted questions. We would like to address this problem in future work.

Of the 140 factoid questions, 46 questions were assigned a type corresponding to a fact table. For 35 of these questions, an answer was actually found in one of the tables. The other 11 questions were answered by using the IR-based strategy as fall-back. 52 of the 60 definition questions were answered by the strategy described in section 4.5.2.2. For the other definition questions, the general IR-based strategy was used as fall-back. Three definition questions received NIL as an answer.

5.2. *Error Analysis*

Parsing errors are the cause of some wrong or incomplete answers. The question *Who is Javier Solana?*, for instance, is answered with *Foreign Affairs*, which is extracted from a sentence containing the phrase *Oud-minister van buitenlandse zaken Javier Solana* (*Ex-minister of foreign affairs, Javier Solana*). Here, *Javier Solana* was erroneously analysed as an apposition of *affairs*. Similarly, the wrong answer *United Nations* for the question *What is UNEP?*, which was extracted from a sentence containing *the environment programme of the United Nations (UNEP)*, which contained the same attachment mistake.

A frequent cause of errors were answers that were echoing part of the question. Currently, the system discards answers that are literal substrings of the questions. However, this strategy fails in cases like:

(11) a. Q: Where is Bonn located? A: `in Bonn`.
b. Q: In which city does one find the famous Piazza dei Miracoli? A: `at the Piazza dei Miracoli`
c. Q: In which American state is Iron Mountain located? A: `The United States`.

It seems cases like (11-a) and (11-b) could be easily rejected as well. Cases like (11-c) are harder, as they involve equivalences on a deeper level. Note finally that not all answers which overlap with the question should be discarded, as the answer in (12) is valid, even though the word *rocket* also occurs in the question.

(12) Q: What is the name of the rocket used to launch the satellite Clementine? A: `Titan rocket`

Maybe syntactic relations can be useful to improve the filtering process. For example, we may allow answers if there is a new element in a modifier relation with the echoing part of the answer. On the other hand, answers that only contain additional prepositions attached to the echoing part are dismissed. Such strategies will be explored in future work.

Our strategy for answering definition questions worked reasonably well, although it did produce a relatively large number of inexact answers (of the 18 answers that were judged inexact, 13 were answers to definition questions). As we explained in section 4.5.2.2, this is due to the fact that we select the most frequent class label for a proper name, and only expand this label with adjectival and PP modifiers that are adjacent to the class label (a noun) in the corresponding sentence. Given the constituent *the museum Hermitage in St Petersburg*, this strategy fails to include *in St Petersburg*, for instance. We did not include relative clause modifiers, as these tend to contain information which is not appropriate for a definition. However, for the question, *Who is Iqbal Masih*, this leads the system to answer *twelve year old boy*, extracted from the

	Table look-up			IR-based			All		
Data set	# Q	MRR	% ok	# Q	MRR	% ok	# Q	MRR	% ok
CLEF 2003	84	0.879	84.5	293	0.565	52.2	377	0.635	59.4
CLEF 2004	69	0.801	75.4	131	0.530	48.9	200	0.623	58.0
CLEF 2005	88	0.747	68.2	111	0.622	57.7	199	0.677	62.3

Table 6. *Current scores on Dutch CLEF questions (2003–2005)*

constituent *twelve year old boy, who fought against child labour and was shot sunday in his home town Muritke*. Here, at least the first conjunct of the relative clause should have been included. Similarly, we did not include purpose clauses, which leads the system to respond *large scale American attempt* to the question *what was the Manhattan project*, instead of *large scale American attempt to develop the first (that is, before the Germans) atomic bomb*.

5.3. *Current Status*

Our QA system is in continuous development. Several improvements to the system have been described in the previous sections already. For example, the optimised passage retrieval component with integrated linguistic features has not been applied in the system we used for CLEF 2005. We have improved the syntactic patterns for extracting facts for QATAR in general. We also worked on improving QA on a closed domain task (medical questions). We continuously test our system on CLEF data.

Table 6 summarises the current status of the system in terms of scores on CLEF data from the recent years. We have used the same sets as used in CLEF with the addition of some valid answers that we identified during the development of our system. Most of these additional answers are due to spelling variations such as “1 miljoen” (1 million) that can be spelled as “één miljoen” or “Hiroshima” spelled as “Hirosjima”. Many variations can be found among names of persons (e.g. “Giovanni Agnelli” vs. “Gianni Agnelli”).

The scores in table 6 illustrate the improvements of our system compared to previous runs submitted to CLEF 2005. However, note that the CLEF data should be considered as the development set for our system. It remains to be shown that these improvements reflect the increasing quality of our system.

The performance of QATAR is very high in terms of precision. Evaluation on the dataset of CLEF 2003, 2004 and 2005 showed that about 75% of the questions answered by QATAR are answered correctly compared to a score of 52% for the questions answered by the technique based on passage retrieval. For the CLEF 2005 dataset, QATAR found an answer for about 85% of the questions that were classified as QATAR-questions.

The quality of the answers returned by table look-up appears to degrade over more recent CLEF runs. The reason is the amount of definition questions. These questions are typically harder, and the proportion of those questions ranged from 0 in 2003, to 20 in 2004 and 57 in 2005.

We have also carried out informal experiments to evaluate the quality of our system on closed domain medical QA. On a list of 100 manually compiled test questions, Joost returned a correct answer for 38 of them. Only 26 of the 100 test questions could be answered by table look-up. We believe that this result illustrates on the one hand that our approach is robust enough to generalize to specialised domains. On the other hand, it also illustrates that closed domain QA can be harder than open domain QA. One important reason for this is that medical questions tend to be less “factoid” in nature than typical CLEF questions.

6. Conclusions and Future Work

Joost is a QA system which incorporates various components that make use of high-quality syntactic information. The Alpino parser for Dutch is used to analyse document collections off-line as well as user questions on-line. Joost has been used for the open-domain monolingual QA task of CLEF 2005, as well as for closed domain medical QA. We have shown that deep syntactic parsing is robust enough to deal with such material and that syntactic information is useful in all components of our QA system: question analysis, passage retrieval, answer selection and off-line extraction of facts for table look-up strategies. Our system performed best among the Dutch QA systems and came third in the overall evaluations of all monolingual QA systems in the CLEF competition in 2005.

In future work we would like to continue working on exploring syntactic information for further improvements. First of all, we want to extend the strategies described here. Furthermore, we would like to experiment with other methods for matching questions with answer sentences based on syntactic structures. We would also like to optimise the combination of clues used for ranking answer candidates. We will work on the improvement of answering temporally restricted questions and we will experiment with various techniques for query expansion to improve passage retrieval. For the latter we would like to employ lexical knowledge extracted automatically as described in the paper. We will also continue working on anaphora resolution. We hope to be able to show that resolution techniques with high accuracy can boost the performance of our QA system. Finally, we would also like to improve closed domain QA by, for example, including automatically acquired domain specific terminological resources.

Acknowledgements

This research was carried out as part of the research program for *Interactive Multimedia Information Extraction*, IMIX, financed by NWO, the Dutch Organisation for Scientific Research.

7. References

Attardi G., Cisternino A., Formica F., Simi M., Tommasi A., “ PiQASso: Pisa Question Answering System”, *Text REtrieval Conference (TREC) 2001 Proceedings*, Gaithersburg, ML, p. 633-642, 2002.

Bouma G., Mur J., van Noord G., “ Reasoning over dependency relations”, *Proceedings of KRAQ*, 2005.

Bouma G., van Noord G., Malouf R., “ Alpino: Wide-coverage Computational Analysis of Dutch”, *Computational Linguistics in The Netherlands 2000*, Rodopi, Amsterdam, p. 45-59, 2001.

Cui H., Li K., Sun R., Chua T.-S., Kan M.-Y., “ National University of Singapore at the TREC-13 Question Answering Main Task”, *Proceedings of the 13th Text Retrieval Conference (TREC 2004)*, 2004.

Cui H., Sun R., Li K., Kan M.-Y., Chua T.-S., “ Question Answering Passage Retrieval Using Dependency Relations”, *Proceedings of SIGIR 05*, Salvador, Brazil, 2005.

Curran J., Moens M., “ Improvements in Automatic Thesaurus Extraction”, *Proceedings of the Workshop on Unsupervised Lexical Acquisition*, p. 59-67, 2002.

Fahmi I., Bouma G., “ Learning to Identify Definitions using Syntactic Features”, *in* R. Basili, A. Moschitti (eds), *Proceedings of the EACL workshop on Learning Structured Information in Natural Language Applications*, Trento, Italy, 2006.

Fleischman M., Hovy E., Echihabi A., “ Offline Strategies for Online Question Answering: Answering Questions Before They Are Asked”, *Proc. 41st Annual Meeting of the Association for Computational Linguistics*, Sapporo, Japan, p. 1-7, 2003.

Jakarta A., “ Apache Lucene - a high-performance, full-featured text search engine library”, , http://lucene.apache.org/java/docs/index.html, 2004.

Jijkoun V., Mur J., de Rijke M., “ Information Extraction for Question Answering: Improving Recall Through Syntactic Patterns”, *Coling 2004*, Geneva, p. 1284-1290, 2004.

Katz B., Lin J., “ Selectively using relations to improve precision in Question Answering”, *Proceedings of the workshop on Natural Language Processing for Question Answering (EACL 2003)*, EACL, Budapest, p. 43-50, 2003.

Lin D., “ Automatic Retrieval and Clustering of Similar Words”, *COLING-ACL*, p. 768-774, 1998.

Litkowski K. C., “ Use of Metadata for Question Answering and Novelty Tasks”, *in* E. M. Voorhees, L. P. Buckland (eds), *Proceedings of the eleventh Text Retrieval Conference (TREC 2003)*, Gaithersburg, MD, p. 161-170, 2004.

Malouf R., van Noord G., “ Wide Coverage Parsing with Stochastic Attribute Value Grammars”, *IJCNLP-04 Workshop Beyond Shallow Analyses - Formalisms and stati stical modeling for deep analyses*, Hainan, 2004.

Mollá D., Gardiner M., “ AnswerFinder - Question Answering by Combining Lexical, Syntactic and Semantic Information”, *Australasian Language Technology Workshop (ALTW) 2004*, Sydney, 2005.

Mur J., van der Plas L., “ Anaphora Resolution for Off-line Answer Extraction Using Instances”, 2006, submitted.

Pantel P., Ravichandran D., “ Automatically Labeling Semantic Classes”, *in* D. M. Susan Dumais, S. Roukos (eds), *HLT-NAACL 2004: Main Proceedings*, Association for Computational Linguistics, Boston, Massachusetts, USA, p. 321-328, May 2 - May 7, 2004.

Pasca M., “ Acquisition of categorized named entities for web search”, *Proceedings of the Thirteenth ACM conference on Information and knowledge management*, p. 137 - 145, 2004.

Punyakanok V., Roth D., Yih W., “ Mapping Dependency Trees: An application to Question Answering”, *The 8th International Symposium on Artificial Intelligence and Mathematics (AI&Math 04)*, Fort Lauderdale, FL, 2004.

Soubbotin M., Soubbotin S., “ Patterns of Potential Answer Expressions as Clues to the Right Answer”, *Proceedings of the TREC-10 Conference*, 2001.

Tellex S., Katz B., Lin J., Fernandes A., Marton G., “ Quantitative evaluation of passage retrieval algorithms for question answering”, *In Proceedings of the SIGIR conference on Research and development in information retrieval*, ACM Press, p. 41-47, 2003.

Tiedemann J., “ A Genetic Algorithm for Optimising Information Retrieval with Linguistic Features in Question Answering”, *in* G. A. Nicolas Nicolov, Kalina Bontcheva, R. Mitkov (eds), *Recent Advances in Natural Language Processing*, vol. IV, John Benjamins Publishing Company, 2006.

van der Plas L., Bouma G., Mur J., “ Automatic Acquisition of Lexico-semantic Knowledge for QA”, *in* C.-R. Huang (ed.), *Ontologies and Lexical Resources for Natural Language Processing*, Cambridge University Press, Cambridge, UK, University of Sinica, 2006.

van Noord G., “ **A**t **L**ast **P**arsing **I**s **N**ow **O**perational”, *TALN 2006*, Leuven, 2006.

Vossen P., “ EuroWordNet A Multilingual Database with Lexical Semantic Networks”, , available from http://citeseer.ist.psu.edu/vossen98eurowordnet.html, 1998.

Wu Z., Palmer M., “ Verb semantics and lexical selection”, *The 23rd Annual Meeting of the Association for Computational Linguistics*, p. 133-138, 1994.

ANNEXE POUR LE SERVICE FABRICATION

A FOURNIR PAR LES AUTEURS AVEC UN EXEMPLAIRE PAPIER
DE LEUR ARTICLE ET LE COPYRIGHT SIGNE PAR COURRIER
LE FICHIER PDF CORRESPONDANT SERA ENVOYE PAR E-MAIL

1. ARTICLE POUR LA REVUE :

   TAL. Volume 46 – n°3/2005

2. AUTEURS :

   Gosse Bouma — Ismail Fahmi — Jori Mur — Gertjan van Noord Lonneke van der Plas — Jörg Tiedemann

3. TITRE DE L'ARTICLE :

   Linguistic knowledge and question answering

4. TITRE ABRÉGÉ POUR LE HAUT DE PAGE MOINS DE 40 SIGNES :

   Linguistic knowledge and QA

5. DATE DE CETTE VERSION :

   March 14, 2007

6. COORDONNÉES DES AUTEURS :

   – adresse postale :
   University of Groningen, PO Box 716,
   9700 AS Groningen, The Netherlands
   {g.bouma,j.mur,g.j.m.van.noord,m.l.e.van.der.plas,j.tiedemann}@rug.nl
   – téléphone : 00 00 00 00 00
   – télécopie : 00 00 00 00 00
   – e-mail : guillaume.laurent@ens2m.fr

7. LOGICIEL UTILISÉ POUR LA PRÉPARATION DE CET ARTICLE :

   LaTeX, avec le fichier de style `article-hermes.cls`,
   version 1.23 du 17/11/2005.

8. FORMULAIRE DE COPYRIGHT :

   Retourner le formulaire de copyright signé par les auteurs, téléchargé sur :
   `http://www.revuesonline.com`

SERVICE ÉDITORIAL – HERMES-LAVOISIER
14 rue de Provigny, F-94236 Cachan cedex
Tél. : 01-47-40-67-67
E-mail : revues@lavoisier.fr
Serveur web : http://www.revuesonline.com