Off-line answer extraction for Dutch QA

Jori Mur

Humanities Computing, University of Groningen

Abstract

In Jijkoun et al. [2004] we showed that off-line answer extraction using syntactic patterns is a successful method for answering English factoid questions. In this paper I will discuss the results of applying this method for Dutch question answering using the CLEF text collection parsed by the Alpino parser, a wide coverage dependency parser for Dutch. We defined syntactic patterns to extract answers to frequently occurring question types, such as function questions (*Who is the president of the US?*), location questions (*Where is Groningen located?*) and inhabitant questions (*How many people live in Amsterdam?*).

1 Introduction

Open domain question answering (QA) is a research area that receives much attention nowadays. A QA system takes as input a natural language question, it analyzes the question and tries to find the correct answer in a large text collection. Several strategies have turned out to be quite successful. However, the main part of studies in this domain is focused on QA for English. It would be interesting to see whether similar results can be achieved for other languages for which less resources and less accurate tools are available.

In order to stimulate the development of non-English QA-systems the Cross Language Evaluation Forum (CLEF) organized a question answering track in 2003 and 2004. The QA Track offered tasks to test monolingual and cross-language question answering systems. Resources have been built for several European languages such as Italian, Spanish, Dutch, French and Finnish.

The University of Groningen is working on a Dutch QA-system. One of the techniques we implemented in our system is called *off-line answer extraction*. This technique has proved to be successful (Fleischman et al. [2003]; Jijkoun et al. [2003]). Before actual questions are known, a corpus is exhaustively searched for potential answers to specific question types (`capital, abbreviation, inhabitants, year of birth, ...`). The answers are extracted from the corpus off-line and stored in tables for quick and easy access.

Jijkoun et al. [2004] compared two methods for extracting answers from an English newspaper text collection. They have shown that using syntactic patterns based on dependency relations can lead to significant improvements in recall over methods that are based on regular expression pattern matching.

We implemented the method based on syntactic pattern matching in our Dutch QA system. We parsed a Dutch text collection consisting of two years of newspaper text from two different newspapers (the NRC Handelsblad and the Algemeen Dagblad) using the Alpino Parser (van Noord et al. [2001]). Subsequently, we defined syntactic patterns to extract answers to predefined question categories.

This paper presents the off-line answer extraction module based on syntactic pat-

terns for our Dutch question answering system. In section 2 we discuss related work on answer extraction. In section 3 we describe the experimental framework of our research and in section 4 we discuss our experiments and results. In section 5 you can find conclusions and ideas for future work.

2 Related work

Several QA systems for English have investigated the use of text patterns. Soubbotin and Soubbotin [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.

Ravichandran et al. [2003] have investigated the use of surface text patterns for a Maximum Entropy based QA system. The text patterns were collected automatically in an un-supervised way. They have shown that the patterns help to answer more questions correctly.

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] have also evaluated their system on "Who is ..." questions. In contrast to Fleischman et al. they 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.

3 Experimental Setting

3.1 The QA system

This section describes our open domain QA-system *Joost*. Figure 1 shows an outline of the system architecture. The right branch represents the off-line strategy. We have called this part of the system 'Qatar'. We use a corpus parsed with the Alpino parser, a wide-coverage dependency parser for Dutch. Section 3.2 gives a more detailed description of the parser and its output. Our QA-system scrolls through the parsed corpus to find dependency relations of answer phrases that match the predefined syntactic patterns. The answers found are extracted and stored in tables. More details on the extraction process are given in section 3.3.

Once the tables are created the question answering process can start. An incoming question is analysed. The keywords from the question are selected and the question type is determined. It depends on the question type which step the system will take next. If there is a table associated with the question type, the next step will be to look up the answer in that particular table using the keywords. Answers found by table look-up are ranked according to their frequency. The highest ranked answer is considered as the term with the highest probability to be correct. Candidate answer

which are incorrect, because the author made a mistake or because they were modified, for example by a year ('Amsterdam had 269,000 inhabitants in 1869), most likely have a low frequency and therefore will end up low in the ranking. If no answer is found or if no table was associated with the question type in the first place, the system will follow the other path in the process. In that case, it will use the keywords to retrieve relevant documents, apply natural language processing techniques to re-rank the documents and select the sentence that has the highest probability to contain the answer. Note that answers returned by the online module are complete sentences containing the exact answer phrase. The answers found with the off-line module are exact answers.

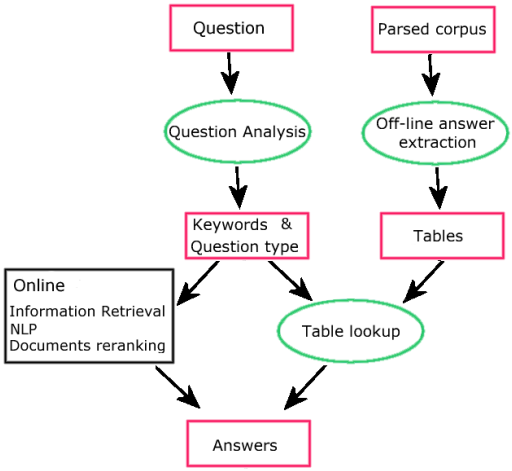

Figure 1: Architecture of our QA-system JOOST

3.2 Alpino

The Alpino system is a linguistically motivated wide-coverage grammar and parser for Dutch. The constraint-based grammar follows the tradition of HPSG Pollard and Sag [1994]. It currently consists of over 500 grammar rules (defined using inheritance) and a large and detailed lexicon (over 100.000 lexemes). To ensure coverage, heuristics have been implemented to deal with unknown words and ungrammatical or out-of-coverage sentences (which may nevertheless contain fragments that are analyzable). The output of the system is a dependency graph, compatible with the annotation guidelines of the Corpus of Spoken Dutch. See an example in figure 2. Malouf and van Noord [2004] show that the accuracy of the system, when evaluated on a test-set of 500 newspaper sentences, is over 88%.

For the QA task the disambiguation model was retrained on a corpus which con-

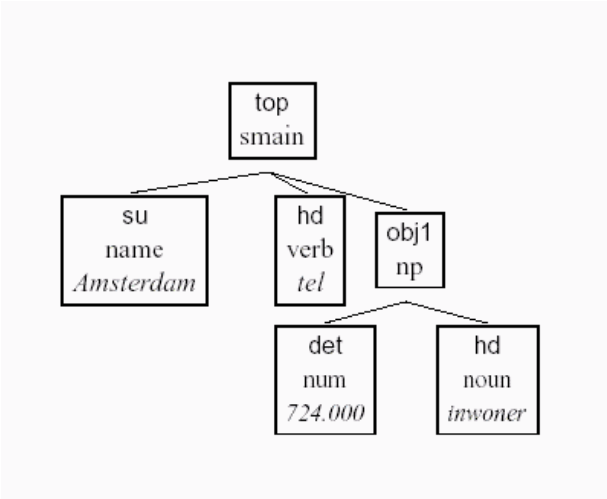

Figure 2: Output for the sentence: Amsterdam telt 724.000 inwoners (*Amsterdam counts 724,000 inhabitants.*)

tained the (manually corrected) dependency trees of 650 quiz questions.[1] The retrained model achieves an accuracy on 92.7% on the CLEF 2003 questions and of 88.3% on CLEF 2004 questions. We have used the Alpino-system to parse the full text collection provided by CLEF for the Dutch QA task.

A second extension of the system for QA, was the inclusion of a Named Entity Classifier. The Alpino system already included heuristics for recognising proper names. Thus, the classifier needed to classify strings which have been assigned a NAME part of speech by grammatical analysis as being of the subtype PER, ORG, GEO or MISC.[2] For unknown names, a maximum entropy classifier was trained, using the Dutch part of the shared task for CONLL 2003.[3] The accuracy on unseen CONLL data of the resulting classifier (which combines dictionary look-up and a maximum entropy classifier) is 88.2%.

3.3 Extraction

We derived patterns based on the structures Alpino produced for potential answers to frequently occurring question types. Table 1 shows how many patterns we created for each question type. The selected question types were in large numbers present in the training set. Answers for the question types listed here tend to occur in more or less

[1] From the *Winkler Prins spel*, a quiz game. The material was made available to us by the publisher, *Het Spectrum, BV.*

[2] Various other entities which sometimes are dealt with by NEC, such as dates and measure phrases, can be identified using the information present in POS tags and dependency labels.

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

fixed patterns in the corpus.

Table 1: Question types with # of patterns	
Abbreviations	1
Capitals	2
Currencies	4
Functions	1
Inhabitants	11
Locations	2
Manner of Death	22

For every type an example of a pattern is shown in figure 3 on page 166. The arrows denote dependency relations from head to dependent. Above the arrows are Alpino's dependency labels. *ORG, LOC* en *PER* are tags that are assigned to phrases that are identified by our named-entity tagger as organisations, locations or persons respectively . *X* and *Y* are variables. They can be replaced by every possible word. The phrase *adj* stands for an adjective and *num* for a number.

I will briefly discuss some issues we encountered during the pattern creation process. To find abbreviations we looked for two terms both tagged as 'names' and connected through a modifier relation. Consequently, we extracted much noise as well. The system found the pair 'Tim Burton' and 'Batman', for instance, from the following sentence: '(. . .) op voorwaarde dat Tim Burton (Batman) de regie voor zijn rekening neemt.'(*(. . .) on condition that Tim Burton (Batman) takes care of the production.*) Therefore it was required that all characters from one string also appeared in the same order in the other string.

For the question type 'capital' we extracted the capital and the accompanying country. The country could be formulated as a noun ('Amsterdam, de hoofdstad van **Nederland**' (*Amsterdam, the capital of Holland*)) or as an adjective ('De **Nederlandse** hoofdstad Amsterdam' (*The Dutch capital Amsterdam*)). The same holds for the country name in the currency table. To be able to match both forms with whatever form was found in the question, we used a list of country names along with their adjective forms.

For the function table we defined only one pattern. With this pattern alone we already extracted more than 200,000 facts. Here the problem was that with more patterns we retrieved too much noise. For instance, we wanted a pattern for phrases like 'De Italiaanse minister Agnelli van buitenlandse zaken (*The Italian minister Agnelli for Foreign Affairs*), but the pattern to extract 'Italiaanse' as well as 'van buitenlandse zaken' was too general, that is, it extracted too much noise.

We created many patterns to extract information about inhabitant numbers. It turned out that facts about the number of inhabitants were formulated in numerous different ways. For example the fact that Almere counts 100,000 inhabitants was formulated in the following ways: 'Het 100.000 inwoners tellende Almere', 'Almere (100.000 inwoners)', 'Almere telt 100.000 inwoners', 'Almere met 100.000 inwoners'. The same holds for information about how people died. There are many ways in which people can die.

- **Abbreviation**
  name $\xrightarrow{\text{mod}}$ *name*;
  Centraal Bureau voor de Statistiek (CBS)
  Central Office for Statistics (COS)

- **Capital**
  X $\xleftarrow{\text{mod}}$ `hoofdstad` $\xrightarrow{\text{app}}$ *Y*;
  de Duitse hoofdstad, Berlijn
  the German capital, Berlin

- **Currency**
  adj $\xleftarrow{\text{mod}}$ `munt/munteenheid` $\xrightarrow{\text{app}}$ *noun*;
  de Griekse munteenheid, de Drachme
  the Greek currency, the Drachme

- **Function**
  name(PER) $\xleftarrow{\text{app}}$ *noun* $\xrightarrow{\text{mod}}$ *X*;
  Delors,voorzitter van de Europese Commissie
  Delors, chair of the European Commission

- **Inhabitants**
  name $\xrightarrow{\text{mod}}$ `inwoner` $\xrightarrow{\text{det}}$ *num*;
  Amsterdam (724.000 inwoners)
  Amsterdam (724,000 inhabitants)

- **Location**
  name (ORG/LOC) $\xrightarrow{\text{mod}}$ `in` $\xrightarrow{\text{mod}}$ *name(LOC)*;
  het Rode plein in Moskou
  the Red Square in Moskow

- **Manner of death**
  X $\xleftarrow{\text{subj}}$ `pleeg` $\xrightarrow{\text{obj1}}$ `zelfmoord`;
  Adolf Hitler pleegde zelfmoord
  Adolf Hitler committed suicide

Figure 3: Examples of syntactic patterns

For the location table we searched for information about where organisations and locations where located. However, when we were doing the experiments we discovered that most of the location questions do not ask where something is located, but where something happened ('Waar explodeerde de eerste atoombom?' (*Where did the first atom bomb explode?*)) or it asked for a location with some specific feature ('Welke Europees land heeft de hoogste alcoholconsumptie?'(*Which European country has the highest consumption of alcohol?*)). Due to this fact many location questions were not answered (see section 4, table 5).

The extracted facts were stored as prolog facts. An example fact from the inhabitants table looks as follows:

```
inhabitants('Amsterdam',1,'724.000','AD19950506-0132.xml').
```

The first argument of the predicate `inhabitants` is the location name, the second argument is the number of times this fact was found in the corpus, the third represents the actual number of inhabitants for this location and the last argument gives the ID of the document from which this fact was extracted. The frequency argument and the document ID argument occur in every fact from every table, the other arguments vary depending on the question type.

The abbreviation table contains abbreviations and the associated full terms. In the capitals and currencies tables information is stored about countries and their capitals and currencies respectively. In the function table you can find information about persons and the roles they fulfil in life, for instance: *Bill Clinton, president of America* or *Ralph Fiennes, British actor*. In the location table are organisations, cities, towns and objects stored together with their location. The manner-of-death table contains facts about how people died.

3.4 Corpus and Questions

We extracted the facts from the Dutch CLEF text collection which contains two years of news paper texts from the Algemeen Dagblad and the NRC Handelsblad (1994 and 1995). The total corpus size is 540 MB (200,000 documents) (Magnini et al. [2003b]).

We trained our system on the set of questions provided by CLEF in the DISEQuA corpus (Magnini et al. [2003a]) which were used for the CLEF-2003 QA-track. This question corpus contains 450 questions, each one formulated in four languages, Dutch, Italian, Spanish and English. We only used the Dutch versions.

Some questions were known to have no answer in the text collection. For other questions none of the participating systems found an answer. Those questions were also classified as questions with no answer in the text collection. Since only one group partook in the Dutch monolingual task, a question for which their system did not find an answer was automatically classified as containing no answer. Out of the 450 questions 370 had an answer in the text collection according to this classification. We performed our experiments with the questions that are known to have an answer in the text collection.

We tested our system on the Dutch CLEF-2004 questions. For this year 200 questions were prepared. Out of these 200 questions 185 had an answer in the text collection. Differently from CLEF-2003, where there were only factoid question, twenty

definitional questions ('Wat is het IAEA?' *(What is the IAEA?)*, 'Wie is Alain Juppé?' *(Who is Alain Juppé)*) were included as well. Definition questions tend to be harder to answer then factoid questions, because they expect biographical or descriptive information as an answer rather than a named entity such as a name or a date.

4 Experiments and results

We measured the performance for training and testing by calculating the mean reciprocal rank (MRR), i.e. for each question we take the reciprocal of the rank at which the first correct answer occurs or 0 if none of the returned answers is correct. The score for the whole system is the mean of the individual scores for all the questions. This measure is often used to evaluate QA systems and it is a good score for comparing systems: If a system A ranks the correct answer at the first position for 50 out of 100 questions and at the second position for the other 50 questions, we may assume that this system comes nearer to the perfect system than a system B which also ranks the correct answer at the first position for 50 out of 100 questions, but which does not find an answer for the other 50.

Still, we were also interested in the number of questions for which the correct answer ended up at the first position. Therefore, we also used this as a measure to evaluate the whole system. We evaluated our system using the answers provided by the CLEF organisers.

Table 2: MRR for questions answered by Qatar.

Question set	# Questions	MRR Qatar	MRR Online
CLEF 2003	65/370 (18%)	0.73	0.40
CLEF 2004	24/185 (13%)	0.71	0.46

Table 2 shows how many questions were answered by the off-line module (Qatar), the MRR score for these questions and the MRR for these questions had they been answered by the online module. Remember that with the online module we returned complete sentences, and with Qatar the exact answer, which is harder. For the training set (CLEF 2003) the use of Qatar for these questions results in a very high score (0.73), significantly higher then the score the system would have received if it had used the online approach for answering these questions (0.40). For testing the results were similar: a MRR of 0.71 using Qatar vs. a MRR of 0.46 if the system uses the online module.

The positive effect of the tables is also clear from the results for the whole system on the training data. Table 3 shows the result for the CLEF 2003 data when we take all 370 questions into account. The column labelled - *tables* presents the results the system produced before we implemented Qatar, and the column labelled + *tables* shows the results after adding Qatar to the system.

The MRR increases from 0.49 without tables to 0.54 with the tables. The number of questions for which the correct answer was ranked at the first position increases with almost 30 questions from 145 to 174.

Table 3: Results on CLEF 2003 data.		
	- tables	+ tables
MRR	0.49	0.54
# Correct on 1st	145 (39%)	174 (47%)

Table 4: Results on CLEF 2004 data.		
	- tables	+ tables
MRR	0.51	0.52
# Correct on 1st	77 (42%)	81 (44%)

However, the positive effect was less clear during testing. The results in table 4 on the CLEF 2004 data seem a bit disappointing at first. The MRR increases only 0.01 points from 0.51 without the tables to 0.52 with the tables. In addition, only 4 more questions had the correct answer ranked at the first position.

We take a closer look at the data for testing by looking at each question type separately. The results per question type on the CLEF 2004 data set are shown in table 5. Each row lists the category of the table, the MRR and the number of questions in this category answered by Qatar. For example, three questions were classified as capital questions and all three of them were answered with Qatar. In brackets in the third column are the numbers of questions in the training data for these specific question types. They suggest that the training data was not representative for the test data. In particular the capital questions, the inhabitant questions, the abbreviation questions and location questions appeared not as often in the CLEF 2004 data set as in the CLEF 2003 data set.

Table 5: MRR per question type (CLEF 2004).		
Categories	MRR	# questions
Capital	1.00	3 of 3 (10)
Inhabitants	1.00	1 of 1 (13)
Abbreviation	1.00	1 of 4 (13)
Location	1.00	3 of 20 (62)
Manner of death	0.67	3 of 5 (3)
Function	0.55	11 of 24 (48)
Currency	0.50	2 of 5 (4)

The reason why we see so little effect is because only a small fraction of the questions was answered with Qatar. It is inherent in the setting of CLEF that Qatar answers only a small fraction of the questions, because it is not the intention of the organisers of CLEF to cover exactly the same kind of phenomena as the year before. Therefore some questions will be of an unseen type. Indeed the question set of CLEF 2004 contained question types that did not occur in the CLEF 2003 question set. For example questions that asked for the first name of persons ('Wat is de voornaam van rechter 'Borsellino'?'(*What is judge Borsellino's fist name?*)). We had not created a table for

these kind of questions.

So we may conclude that the trainingdata was not representative for the test data. It would have been better if we had taken part of the CLEF 2003 and part of the CLEF 2004 for training and used the rest for testing. In that case we would also have had a closer resemblance to a QA-system in a real setting. Lowe [2000] claims that Zipf's law applies to user queries. The most frequently asked questions can be classified in only a few question types, in other words, users always tend to ask the same kinds of questions. In that case Qatar would be very suitable.

5 Conclusions and Future work

We described an off-line answer extraction technique which we have implemented in our Dutch QA system Joost. The precision for questions that were answered with our off-line module Qatar was very high. In addition, the performance of the system overall increased. We used the CLEF 2003 data set for training and the CLEF 2004 data set for testing. However, it turned out that the training data was not representative for the test data. That explains why we see so little effect.

In short, the questions that were answered, were almost always correctly answered (high precision), but many questions were not answered (low recall). Therefore, we have to think how to get more questions answered by Qatar. One obvious way to answer more questions using Qatar is to define more patterns for more question types. We could create patterns to extract birth dates or birth places or we could create tables for measures to answer questions about heights and distances (*What is the distance between the sun and the earth?*), for example. Investigating machine learning techniques for finding more patterns would also help extracting more answers and consequently answering more questions. A third option we plan to work on is coreference solution to find more answers. For instance, it is no use to extract that *He* is the *president of the United States*. But if we could find that *He* referred to George Bush (or to Bill Clinton in case of a news paper corpus from 1994 or 1995) we could store that fact in our table.

References

M. Fleischman, E. Hovy, and A. Echihabi. Offline strategies for online question answering: Answering questions before they are asked. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, 2003.

J. Jijkoun, J. Mur, and M. de Rijke. Information extraction for question answering: Improving recall through syntactic patterns. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING)*, 2004.

V. Jijkoun, G. Mishne, and M. de Rijke. Preprocessing documents to answer dutch questions. In *Proceedings of the 15th Belgian-Dutch Conference on Artificial Intelligence (BNAIC 2003)*, 2003.

J.B. Lowe. What's in store for question answering? (invited talk). In *Proceedings of*

the Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC), 2000.

B. Magnini, S. Romagnoli, A. Vallin, J. Herrera, A. Peñas, V. Peinado, F. Verdejo, and M. de Rijke. Creating the disequa corpus: a test set for multilingual question answering. In Peters C., editor, *Working Notes for the CLEF 2003 Workshop*, 2003a.

B. Magnini, S. Romagnoli, A. Vallin, J. Herrera, A. Peñas, V. Peinado, F. Verdejo, and M. de Rijke. The multiple language question answering track at clef 2003. In Peters C., editor, *Working Notes for the CLEF 2003 Workshop*, 2003b.

Malouf and van Noord. Wide coverage parsing with stochastic attribute value grammars. In *IJCNLP-04 Workshop Beyond Shallow Analyses - Formalisms and statistical modeling for deep analyses*, 2004.

C. Pollard and I. Sag. *Head-driven Phrase Structure Grammar*. Center for the Study of Language and Information Stanford, 1994.

D. Ravichandran, A. Ittycheriah, and S. Roukos. Automatic derivation of surface text patterns for a maximum entropy based question answering system. In *Proceedings of the HLT-NAACL Conference*, 2003.

M.M. Soubbotin and S.M. Soubbotin. Patterns of potential answer expressions as clues to the right answer. In *Proceedings of the TREC-10 Conference*, 2001.

G. van Noord, G. Bouma, and R. Malouf. Alpino: Wide-coverage computational analysis of dutch. In *Computational Linguistics in The Netherlands 2000*, 2001.