Building Infrastructure for Dutch Question Answering

Valentin Jijkoun Gilad Mishne Maarten de Rijke

Language & Inference Technology group, ILLC, U. Amsterdam
Nieuwe Achtergracht 166, 1018 WV Amsterdam, The Netherlands
jijkoun, gilad, mdr@science.uva.nl

ABSTRACT

We report on the construction of the first-ever open domain question answering system for the Dutch language. In addition to providing experimental results based on the CLEF 2003 QA test set for Dutch, we also identify a number of key natural language processing resources that are needed to further question answering for Dutch.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing—*linguistic processing*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*information filtering, search process*; H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*question-answering (fact retrieval) systems*; I.2.1 [**Articifial Intelligence**]: Applications and Expert Systems; I.2.7 [**Articificial Intelligence**]: Natural Language Processing

General Terms

Information retrieval

Keywords

Question answering, Dutch language resources

1. INTRODUCTION

With recent advances in computer and Internet technology, people have access to more information than ever before. Much of the information is available in free text with little or no metadata, and there is a tremendous need for tools to help organize, classify, and store the information, and to allow better access to the stored information. Over the years, research in information retrieval has made significant progress in addressing this problem. Large parts of this work have found their way into our every day world. In addition, significant progress has been made in our theoretical understanding of document retrieval methods.

DIR 2003 4th Dutch-Belgian Information Retrieval Workshop, Amsterdam, The Netherlands

Current information retrieval systems allow us to locate documents that might contain the pertinent information, but most of them leave it to the user to extract the useful information from a ranked list. This leaves the (often unwilling) user with a relatively large amount of text to consume. People have questions and they need answers, not documents. There is a need for tools that reduce the amount of text one might have to read to obtain the desired information. *Corpus-based question answering* is designed to take a step closer to *information* retrieval rather than *document* retrieval. The question answering (QA) task is to find, in a large collection of data, an answer to a question posed in natural language.

Question answering for languages other than English is relatively underdeveloped, although recently launched evaluation initiatives are bound to change this. The aim of the present paper is twofold: to report on experiments with the recently released CLEF test collection for Dutch question answering, and to list language-specific isssues and needs for resources that we identified during our experiments. The remainder of the paper is organized as follows. We briefly mention related work in Section 2. Then, in Section 3 we describe the architecture of our Dutch QA system, in Section 4 we give the evaluation results and provide a brief error analysis. Finally, in Section 5 we indicate language resources which are crucial to enable high-performance QA for Dutch.

2. RELATED WORK

QA systems have been around since the early 1960s [10], but back then they were almost exclusively used as natural language front-ends to database servers. Since 1999, the annual Text REtrieval Conference (TREC, [9]) has organized a dedicated QA track, aimed at bringing the benefits of large-scale evaluation to bear on the QA problem, with a particular emphasis on systems that can function in unrestricted domains.

Open domain QA for English received a big boost with the launch of TREC's QA-track. In part because of the lack of evaluation platforms, for a number of years QA efforts in languages other than English were largely non-existent. At its 2001/2002 edition, the NTCIR workshop featured QA on a Japanese corpus [7]. And as of 2003, the Cross-Language Evaluation Forum (CLEF) features a track dedicated to QA in three non-English European languages: Dutch, Italian, and Spannish. Our specific interest is in QA for Dutch.

In recent years, there has been a significant amount of work on document retrieval for Dutch, fueled in part by the fact that Dutch became one of the languages for which document retrieval was evaluated at CLEF [2]. In addition, various Dutch teams have worked on information extraction in Dutch. And recently, there were various efforts aimed at building language resources (both tools and corpora) for Dutch, including the *Corpus Gesproken Nederlands*, a richly annotated corpus of contemporary standard Dutch as spoken by adults in the Netherlands and Flanders, which has already proved to be a useful building block for developing taggers and parsers [8]. We believe that ours is the first attempt to build an open-domain question answering system for Dutch.

3. QUESTION ANSWERING FOR DUTCH

As with many other (open domain) natural language processing applications, the big challenge in open domain QA is to bridge the *lexical gap*, i.e., to effectively cope with the fact that a question and its answers in a given corpus may be phrased in different vocabularies:

- **Q85:** Waar explodeerde de eerste atoombom?
  (English: Where did the first atom bomb explode)
- **Answer snippet:** Op 6 augustus viel de eerste kernbom op *Hiroshima*...
  (English: On August 6 the first nuclear bomb fell on *Hiroshima*...)

Roughly speaking, two strategies exist for bridging the lexical gap: one based on language understanding and sophisticated lexical and common sense reasoning, and one based on data redundancy where the assumption is that the more data one has, the bigger the likelihood that answers are expressed in the same vocabulary as the question so that shallow extraction methods suffice to obtain an answer.

The challenge for a relatively small language such as Dutch is that neither of the above two general methods for bridging the lexical gap seem fully applicable, due to a lack of resources and a lack of data. We will come back to this issue in Section 5 below; before doing so, we describe the implementation of our Dutch QA system as well as some experimental findings.

3.1 The General Question Answering Architecture

Dozens of open domain QA systems have been described in the literature. In 2002, 34 research groups participated in the question answering track of the annual Text REtrieval Conference (TREC), each group with its own system. While these systems cover a wide spectrum of different techniques and architectures, they have a number of features in common. The prototypical system has four components: *question analysis*, *document retrieval*, *answer extraction*, and *answer selection*. Let's take a closer look at each of these components.

Given a natural language question posed by a user, the first step is to analyze the question itself. The question analysis component may include a morphosyntactic analysis of the question. The question is also classified with respect to its expected answer type, i.e., whether it is asking for a date, a location, the name of a person etc. Depending on the morpho-syntactic analysis and the class of the question, a retrieval query is formulated which is posed to the retrieval component. Some of this information, such as the question class and a syntactic analysis of the question, are also sent to the answer extraction component.

The retrieval component is generally a standard document retrieval system which identifies documents that contain terms from a given query. The retrieval component returns a set or ranked list of documents that are further analyzed by the document analysis component.

The document extraction component takes as input documents that are likely to contain an answer to the original question, together with a specification of what types of phrases should count as correct answers. This specification is generated by the question analysis component. The document analysis component extracts a number of candidate answers which are sent to the answer selection component.

The answer selection component selects the phrase that is most likely to be a correct answer from a number of phrases of the appropriate type, as specified by the question analysis component. It returns the final answer or a ranked list of answers to the user.

3.2 A Multi-Stream Architecture

In the design of our Dutch QA system we initially followed the general 4-stage architecture outlined above. However, during the design of the system, it became evident that there are a number of distinct approaches for the task, some of which are beneficial for all question types, while others benefit only a subset. For instance, abbreviations are often found enclosed in brackets, following the multi-word string they abbreviate, as in "*Verenigde Naties (VN).*" This suggests that for abbreviation questions the text corpus can be mined to extract multi-word strings with leading capitals followed by capitalized strings in brackets; the results can then be stored in a table to be consulted when an abbreviation (or an expansion of an abbreviation) is being asked for. Similar table-creation strategies are applicable for questions that ask for capitals, dates-of-birth, etc., whereas the approach seems less appropriate for definition questions, why-questions, or how-to questions. It was therefore decided to implement a *multi-stream* system for Dutch QA: a system that includes a number of separate and independent subsystems, each of which is a complete standalone QA system that produces ranked answers, but not necessarily for all types of questions; the system's answer is then taken from the combined pool of candidates.

One of the main scientific interests here is to understand the performance of each stream on specific question types and in general. On the practical side, our multi-stream architecture allows us to modify and test an individual stream without affecting the rest of the system. A general overview of our system is given in Figure 1. The system, called Quartz-d, consists of 5 separate QA streams and a final answer selection module that combines the results of all streams

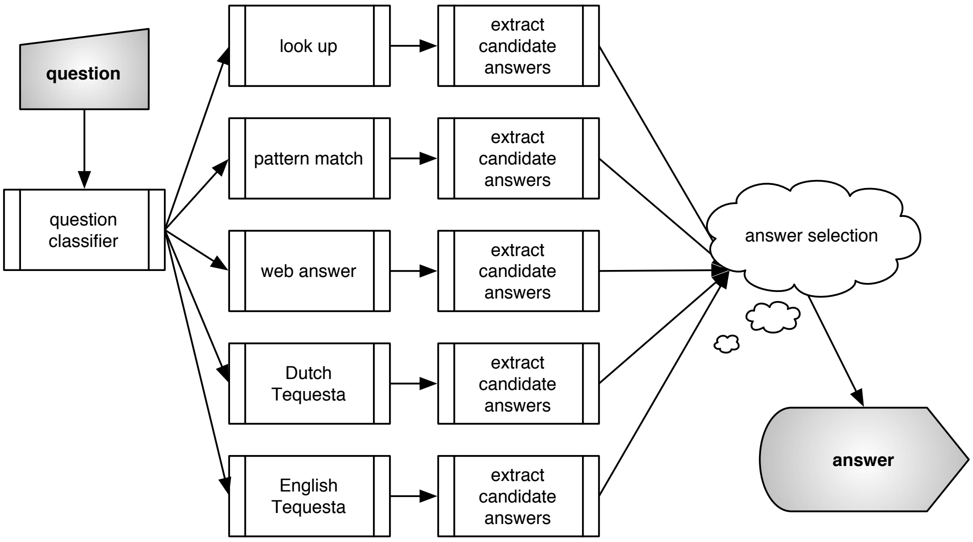
question
question classifier
look up
pattern match
web answer
Dutch Tequesta
English Tequesta
extract candidate answers
extract candidate answers
extract candidate answers
extract candidate answers
extract candidate answers
answer selection
answer

Figure 1: Quartz-d: The University of Amsterdam's Dutch Question Answering System.

and produces the final answers.[1]

We now provide a brief description of the five streams of Quartz-d: Table Lookup, Pattern Match, English Tequesta, Dutch Tequesta, and Web Answer.

The *Table Lookup* stream uses specialized knowledge bases constructed by preprocessing the collection, exploiting the fact that certain information types (such as country capitals, abbreviations, and names of political leaders) tend to occur in the document collection in a small number of fixed patterns. When a question type indicates that the question might potentially have an answer in these tables, a lookup is performed in the appropriate knowledge base and answers which are found there are assigned high confidence. For example, to collect abbreviation-expansion pairs we searched the document collection for strings of capitals in brackets; upon finding one, we extracted sequences of capitalized non-stopwords preceding it, and stored it in the "abbreviation knowledge base." This approach answered questions such as:

Question	*84. Waar staat GATT voor?*
Knowledge Base	Abbreviations
Table Entry	GATT: Overeenkomst over Tarieven en Handel
Extracted Answer	**GATT**

For a detailed overview of this stream, see [4].

In the *Pattern Match* stream, zero or more Perl regular patterns are generated for each question according to its type and structure. These patterns indicate strings which contain the answer with high probability, and are then matched against the entire document collection. Here's a brief example:

Question	*2. In welke stad is het Europese Parlement?*
Generated pattern	`Europese Parlement\s +in\s +(\S +)`
Match	. . . voor het **Europese Parlement in Straatsburg**, dat . . .
Extracted Answer	**Straatsburg**

The *English Tequesta* stream translates the questions to English using Worldlingo's free translation service available at `http://www.worldlingo.com/`. The auto-translated questions are then fed to *Tequesta*, an existing linguistically informed QA system for English developed at the University of Amsterdam [6]. The system uses the English CLEF corpus, and is extended with an *Answer Justification* module to anchor the answer in the Dutch collection.

The *Dutch Tequesta* is an adaptation of English Tequesta to Dutch and used as an independent stream, provided with the original Dutch newspaper corpus. The modifications to the original system included replacing (English) language specific components by Dutch counterparts; for instance, we trained TNT [1] to provide us with Part-of-Speech tags using the *Corpus Gesproken Nederlands* [8]. Considerable effort was spent on developing a named entity tagger for Dutch.

The *Web Answer* stream looks for an answer to a question on the World Wide Web, and then attempts to find justification for this answer in the collection. First, the question is converted to a web query, by leaving only meaningful keywords and (optionally) using lexical information from EuroWordNet. The query is sent to a web search engine (for the experiments reported here we used Google); if no relevant Web documents are found, the query is translated to English and sent again. Next, if the query yields some results, words and phrases appearing in the snippets of the top results are considered as possible answers, and ranked according to their relative frequency over all snippets. The Dutch named entity tagger and some heuristics were used to enhance the simple counts for the terms (e.g., terms that

[1] In the meantime, we have taken this idea and also implemented it for our English question answering efforts.

matched a TIME named entity were given a higher score if the expected answer type was a date). Finally, justifications for the answer candidates are found in the local Dutch corpus.

While each of the above streams is a "small" QA system in itself, many components are shared between the streams, including, for instance, an *Answer Justification* module that tries to ground externally found facts in the Dutch CLEF corpus, and a *Web Ranking* module that uses search engine hit counts to rank the candidate answers from our streams in a uniform way, similar to [5]. The overall score of an answer is the product of the confidence measure produced by the stream generating the answer and the "Web Hit Count" measure, which equals the number of hit counts produced by Google for a query made up of the answer and keywords from the question. To boost queries with words that do not occur frequently, we also calculated a "Query Value" measure, in two different ways. The query value was either calculated using the word frequencies of the query words in the CLEF English and Dutch corpora, or using the Web hit count of the answer alone. Query values were used to normalize the Web Hit Count measure. To illustrate this, Table 1 displays a simplified example, in which frequencies from the CLEF corpora produced better results (stream confidence level not displayed).

4. EVALUATION

We evaluated Quartz-d using the test set made available by the 2003 edition of the QA at CLEF evaluation exercise [2]. The document collection is composed of newspaper articles from 1994–1995, taken from the Dutch daily newspapers Algemeen Dagblad and NRC Handelsblad. The total corpus size is about 500MB (72 million words). The question set includes 200 factoid question, out of which 10% has no known answer in the corpus.

An answer can either be a 50-byte string which contains the answer, or the exact answer phrase; systems are allowed to return three ranked answers for each question. Each answer is required to be accompanied by *justification*: an identifier for the document from which the answer originated. We only created runs with exact answers. The CLEF evaluation uses the standard MRR (mean reciprocal rank) scoring metric; in this paper we also use a simpler measure: the percentage of questions which have a correct answer in one of the three answer candidates provided by the system.

In the *Strict* evaluation, answers are correct only if they answer the question, do not contain redundant information, and are indeed supported by their justification. In the *Lenient* evaluation, the last requirement is dropped. Beside the standard *Strict* and *Lenient* measures, we also evaluated our runs using a more "generous" *Lenient, Non-exact* measure that accepts non-exact answers as correct.

For the experiments on which we report below, we were particularly interested in the impact of redundancy. Table 2 shows the evaluation results of two runs: "CLEF Corpus" for which answer candidates' confidence scores were estimated using the CLEF corpus, and "Web Corpus" for which the estimation was done using the web.

The "Web corpus" run scored better than the "CLEF corpus" run: as expected, normalizing web hit counts according to the distribution of words on the web yielded a more accurate ranking than normalization using corpus word frequencies. More data helps! What is surprising, though, is that the difference between the two runs is just 2% (on the *Strict* measure). Although Web provides more reliable word co-occurence statictics than the relatively small CLEF corpora, it seems that the size of the Dutch Web (as compared to the English Web) is not enough for redundancy-based methods to significantly improve the performance.

In many respects, Quartz-d is still at its early stage. To illustrate this, we take look at the system's output on three questions:

- **Q60**: Wie heeft de Berlijnse Muur gebouwd?
  (English: Who built the Berlin Wall?)
  - Answers: Afrikanen (English: Africans)
    frogs (English: frogs)
- **Q13**: Waar ligt Basra?
  (English: Where is Basra located?)
  - Answers: in Irak (English: in Iraq)
    slechts vier kilometer van de grens met Iran
    (English: only four kilometers from the border with Iran)
- **Q104**: Who is the president of Peru?
  (English: Who is the president of Peru?)
  - Answers: Alberto Fujimori
    Guerra NEDERLANDSE VERTALING

For Q60 we return wrong answers because of errors of our named entity tagger; Q13 comes with two nice and correct answers from the Table Lookup stream; and Q104 comes with a correct answer (the first), but another named entity error occurs with the second answer.

We now turn to a brief (and incomplete) error analysis of the runs described above. Of the 200 CLEF questions we classify correctly 172 (86%). Our classifier consists of a set of hand-written rules; it would be interesting to be able to *learn* a classifier. Unfortunately, there is a glaring lack of training material (unlike for English, where several thousands of classified questions are available). Moreover, to determine the expected answer type of some questions, it is useful to consult WordNet. Think of a question such as *Which heavyweight bit off someone's ear?* — a simple look up in (the English) WordNet reveals that a heavyweight is a person, and, hence, that the answer should have type PERSON. While a Dutch WordNet exists [3], it is much smaller than the English WordNet, and much too sparsely populated to be really useful in the setting of open domain QA right now.

An error analysis of the questions which had a correct answer with incorrect document ID (i.e., those separating *Strict* and *Lenient* scores) revealed that answers with incorrect justifications did not necessarily come from external resources

Question 115. *Waar bevindt zich de Klaagmuur?*		
Candidate Answer	Jeruzalem	Joyce
Generated Query	`Klaagmuur Jeruzalem`	`Klaagmuur Joyce`
Query Hit Count	793	26
Total Word Frequency	4.48e-05	1.85e-05
Candidate Hit Count	70700	3460000
Normalized Query Value	1.0	0.413
Normalized Query Value	0.02	1.0
Final Web Score	1.0	0.0135
Final Web Score	0.02	0.033

Table 1: Example result: answer scoring using the CLEF corpora vs. answer scoring using Web hit counts.

	Strict		Lenient		Lenient, Non-exact	
Run	# correct answers	MRR	# correct answers	MRR	# correct answers	MRR
CLEF corpus	84 (42%)	0.335	87 (43.5%)	0.352	100 (50%)	0.407
Web corpus	88 (44%)	0.349	95 (47.5%)	0.375	107 (53.5%)	0.428

Table 2: Two ways of estimating the answer score: using the CLEF corpus and using the Web.

(the Web and English Tequesta streams); this suggests a local problem in our justification mechanism, rather than an inherent inability to justify externally found answers in the local corpus. Taking this into account, our 53.5% score in the table seems quite reasonable.

It is interesting to see the increase in performance with the *Lenient*, *Non-exact* measure. Most of the non-exact answers that the system produced contained noise around the correct answer strings, e.g. “Jacques Delors. Met”, “Kim Il Sung. Japan” or “1989, heeft vooral in het oostelijke deel van Berl”, due to named entity extraction errors.

An initial analysis of the contribution of the different answering streams to the system's overall performance suggests that every stream has its own strengths, that is, specific question types for which it provides correct answers with higher probability than other streams. The Web Answer stream, for example, seemed to perform better than other streams on questions for which the answer was a date; the Pattern and Table Lookup streams had very good performance on the specific (5-6) question types for which they were used. Every stream contributed some correct answers, so the total combined output of the system was better than any subsystem alone. E.g., out of the 200 questions, 54 (27%) were answered by the Table Lookup stream; of these, 26 answers (13% of the total answers) came solely from this stream.

A further analysis of the performance of our streams on different question types will allow us to give each stream a confidence weight conditioned on question type, and thus to make the answer selection more informed, in ways similar to the approach adopted by BBN for TREC 2002 [11]. A lack of training material prevents us from setting this up for Dutch in a useful manner at this time. Quartz-d's English language counterpart, Quartz-e, follows the same multi-stream strategy as Quartz-d. Quartz-e's answer selection module takes all answer candidates from all streams and selects the final answer of the system. First, we filter candidates to exclude obviously incorrect answers (e.g., for questions asking for a date we check that a candidate indeed contains temporal information). Then the voting procedure assigns a final confidence score to each valid candidate from each stream. This final score is based on the score provided by the stream that generated the candidate and on the overall performance of the stream on questions of the same type. More specifically, the final score is a product of the original score and the weight associated with the (stream, question type) pair. We used machine learning methods to calculate the weights offline so as to achieve maximal performance of the system on the training set of 2000 questions. Finally, a pool of answer candidates with adjusted confidence scores is created and similar answer strings are identified and merged (we used edit distance and string containment to detect similar candidates, e.g. “Washington DC” and “Washington”, or “Yasser Arafat” and “Yasir Arafat”). When several answer candidates are merged, their scores are added. Now, the candidate with the highest condifence score is taken as the final answer. Informal evaluations suggest that this voting mechanism yields substantial improvements.

5. NEXT STEPS

Our ongoing error analysis has identified many key components that require additional investments, mostly in terms of resource building. Just to recapitulate, QA work on Dutch would receive a big boost if the following became available:

- a large collection of sample questions, preferably classified with respect to their expected answer type;
- a large collection of Dutch newspaper data, both to boost the performance of redundancy based components and to build resources (such as the following); and
- a significantly enhanced and extended Dutch WordNet.

In addition, it is clear that our named entity tagging methods need debugging. In the near future we may also experiment with shallow parsing techniques in the answer extraction and answer selection phases of the QA process, to complement our current purely statistical selection modules with more knowledge-intensive criteria.

6. CONCLUSION

In this paper we have described recent first steps towards the creation of a Dutch question answering infrastructure. We presented Quartz-d, a multi-stream question answering system for Dutch, and evaluated it against the CLEF 2003 test collection. Running in parallel several subsystems that approach the QA task from different angles proved successful, as some approaches seem better fit to answer certain types of questions than others.

Our current work on Quartz-d is focused on extensions of the Table Lookup stream and the Web Answer stream. Future plans also include improvements of the voting mechanism between the answers provided by the different streams, and enhancing the system to support definition and list questions.

7. ACKNOWLEDGMENTS

All three authors were supported by the Netherlands Organization for Scientific Research (NWO) under project number 220-80-001. In addition, Maarten de Rijke was supported by grants from NWO, under project numbers 612-13-001, 365-20-005, 612.069.006, 612.000.106, 612.000.207, and 612.066.302.

8. REFERENCES

[1] T. Brants. TnT – a statistical part-of-speech tagger. In *Proceedings of the 6th Applied NLP Conference, ANLP-2000*, 2000.

[2] CLEF: Cross-Language Evaluation Forum. URL: `http://www.clef-campaign.org`.

[3] Building a multilingual database with wordnets for several European languages. URL: `http://www.illc.uva.nl/EuroWordNet/`.

[4] 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'03)*, To appear.

[5] B. Magnini, M. Negri, R. Prevete, and H. Tanev. Is it the right answer? exploiting web redundancy for answer validation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL)*, pages 425–432, 2002.

[6] C. Monz and M. de Rijke. Tequesta: The University of Amsterdam's textual question answering system. In E. Voorhees and D. Harman, editors, *The Tenth Text REtrieval Conference (TREC 2001)*, pages 519–528. National Institute for Standards and Technology. NIST Special Publication 500-250, 2002.

[7] NTCIR: NII Test Collection for IR Systems. URL: `http://research.nii.ac.jp/ntcir/`.

[8] N. Oostdijk. The Spoken Dutch Corpus: Overview and first evaluation. In *Proceedings LREC 2000*, pages 887–894, 2000.

[9] TREC: Text REtrieval Conference. URL: `http://trec.nist.gov`.

[10] B. Webber. Question answering. In S. Shapiro, editor, *Encyclopedia of Artificial Intelligence*, pages 814–822. Wiley, 1992.

[11] J. Xu, A. Licuanan, J. May, S. Miller, and R. Weischedel. TREC 2002 QA at BBN: Answer selection and confidence estimation.