Preprocessing Documents to Answer Dutch Questions

Valentin Jijkoun Gilad Mishne Maarten de Rijke

Language and Inference Technology Group, ILLC, U. Amsterdam

Abstract

We describe a framework for offline extraction of certain types of information from a document collection, and discuss its usage for answering factoid questions. We implemented this approach as a part of the Dutch Question Answering System developed at the University of Amsterdam. The evaluation of the system using data from the CLEF 2003 Question Answering track shows that our strategy yields a significant improvement in the performance of our overall system.

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. 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.

To address these issues, a number of recent initiatives are aimed at providing highly focused information 'pinpointing.' For instance, in the TREC question answering (QA) track [9] participants are given a large document set and a set of questions; for each question, the system has to return an exact answer to the question and a document that supports that answer.

QA has recently received attention from the information retrieval, information extraction, machine learning, and natural language processing communities [3, 6, 10]. But the field itself is not new; in an overview paper from the mid 1960s, as many as 15 implemented and working systems for question answering are described [8]. These early QA systems, however, were usually natural language front-ends to highly structured data sources, whereas modern systems aimed at addressing TREC-style QA operate on unstructured data (typically, collections of newspaper articles). In this paper we report on preliminary experiments in which we attempt to bring those two traditions together. Our motivation for this work is three-fold. First, while information retrieval techniques are relatively successful at providing near-instant access to the vast amount of data on the web, the precision required of QA systems makes on-the-fly question answering from unstructured

data sources too slow and impractical. Second, for many types of factoid questions used for evaluation purposes, the semantic information that (likely) answers these questions, occurs in very fixed patterns. For example, for questions like "Waar ligt Basra?"(that we classify as a LOCATION question), typical answer patterns are "Basra, slechts vier kilometer van de grens met Iran" and "Basra, in het zuiden van Irak." Our strategy is to exploit such regularities for offline extraction of semantic data so as to make the data available for rapid and easy access. Third, having access to such extracted data will allow us to more easily answer certain types of questions than we would be able to if we only did on-the-fly processing; examples include list questions such as "Noem Europese staatshoofden."

The remainder of this paper is structured as follows. In Section 2 we describe the experimental setting in which we evaluated our ideas, the CLEF 2003 Dutch Question Answering task. In Section 3 we describe our system architecture and contrast it with the canonical QA system architecture. Then, in Section 4 we provide details on the creation and use of various kinds of offline tabular data within the QA scenario; we also assess its effectiveness. In our final section (Section 5) we formulate conclusions and discuss future work.

2 Experimental Setting

This year, the Cross-Language Evaluation Forum (CLEF [2]), a forum dedicated to the development of information retrieval systems for European languages, featured a Question Answering track for the first time; the languages evaluated were Italian, Spanish and Dutch. For the Dutch systems, the corpus was composed of newspaper articles from 1994–1995, taken from the Dutch daily newspapers Algemeen Dagblad and NRC Handelsblad. The total corpus size was about 500MB (72 million words). The question set included 200 factoid question, out of which 10% had no known answer in the corpus.

At CLEF, systems were allowed to return three ranked answers for each question; an answer can either be a 50-byte string which contained the answer, or the exact answer phrase. Each answer is required to be accompanied by *justification*: an identifier for the document from which the answer originated. The University of Amsterdam only submitted runs with exact answers. The CLEF evaluation uses the standard MRR (mean reciprocal rank) scoring metric; however, since the official CLEF assesments have not yet been delivered at this time, we will use a simpler measure in this paper: the percentage of questions which had a correct answer in one of the three answer candidates provided by the system.

3 System Architecture

The general architecture of a QA system, shared by many systems, can be summed up as follows. A question is first associated with a *question type*, out of a predefined set such as DATE-OF-BIRTH or CURRENCY. Then, a query is formulated for the question's expected answer, and issued to an information retrieval engine, which then returns documents that are likely to contain the answer. Those documents

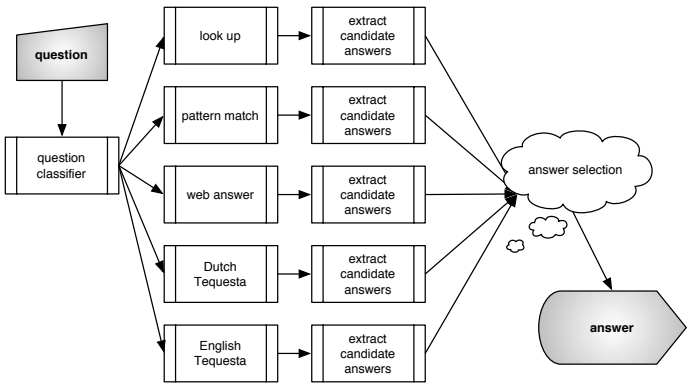

Figure 1: The University of Amsterdam's Dutch Question Answering System.

are sent to an *answer extraction* module, which identifies candidate answers, ranks them, and selects the final answer. On top of this basic architecture, numerous add-ons have been devised, ranging from logic-based methods to ones that rely heavily on the redundancy of information on the World Wide Web [9].

During the design of our QA system, it became evident that there are a number of distinct approaches for the task; some are beneficial for all question types, and others only to a subset. It was therefore decided to implement a *multi-stream* system: a system that includes a number of separate and independent subsystems, each of which is a complete standalone QA system that produces ranked answers; the system's answer is then taken from the combined pool of candidates.

Scientifically, it is interesting 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 a stream without affecting the rest of the system. A general overview of our system is given in Figure 1. The system consists of 5 separate QA streams and a final answer selection module that combines the results of all streams and produces the final answers.

Question Answering Streams. We now provide a brief description of the five streams of our QA system: Table Lookup, Pattern Match, English Tequesta, Dutch Tequesta, and Web Answer.

The *Table Lookup* stream is the focus of this paper; see Section 4 for details. It involves construction of specialized knowledge bases from the collection by preprocessing it. When a question type is determined to be one of the pre-defined types that have a possible answer in these tables, lookup is performed in the respective knowledge base and answers which are found there are assigned high confidence.

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**

Naturally, these patterns also match strings which are not answers; we therefore count the number of times each candidate answer string matched, and rely on actual answers to appear more frequently than other strings.

The *English Tequesta* stream translates the questions to English using a free web service (WorldLingo, www.worldlingo.com). The auto-translated questions are then fed to *Tequesta*, an existing QA system for English developed in the University of Amsterdam [4], using the English CLEF corpus, and extended with an Answer Justification module to anchor the answer in the Dutch corpus.

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* [5], and a named entity (NE) tagger for Dutch was also developed locally.

The *Web Answer* stream looks for an answer to a question in the World Wide Web, and then finds justification for this answer in the collection. 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 in this paper we used Google); if no relevant Web documents are found, the query is translated to English and sent again. 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 NE tagger and some heuristics were used to enhance the simple counts for the terms (e.g., terms that 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 were 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. Our *Question Classifier*, which was a shared component as well, relies on pattern matching, making use of the fact that the vast majority of questions of a certain type are formulated in a few typical ways.

4 A Closer Look at the Table Lookup Stream

Before we report on the impact of tabular data, we take a closer look at how the data was obtained.

Extraction. We hand-crafted a small number of regular expressions able to extract information about country currencies, leaders, roles, capitals, inhabitants, abbreviations, and locations. We chose these categories because likely answers to questions asking for such information tend to occur in a small number of fixed patterns. Furthermore, our main aim was to determine the viability of the idea of using knowledge bases generated from the text collection to answer questions; for this reason we opted to work with a small set of hand-crafted high precision extraction patterns. However, in a later stage of this work we plan to investigate machine learning techniques for automatically extracting patterns; such techniques have been used extensively in the field of information extraction [7].

In Table 1 we list the categories for which we created knowledge bases, plus the number of facts per category. The "Adjective-location" category concerns geographic information of the following type "Bhopal, stad, in India, NH19940125-0036", where the first field indicates a location, the second its type, the third a country or region in which it is located, and the fourth the identifier for the document from which it was extracted. "Locations" contains similar information, but without the type; "Leaders" has information of the following kind "Beieren, minister van milieu, Peter Gauweiler, NH19940217-0003", and "Roles" generalizes this to also include other roles besides government-related ones.

Type	# Facts extracted	# Unique facts extracted
Abbreviations	14575	6095
Adjective-location	2328	957
Capitals	1922	465
Currencies	41	26
Inhabitants	39	38
Leaders	10740	2456
Locations	4931	4202
Roles	9717	8954

Table 1: Facts extracted from the Dutch CLEF 2003 corpus.

Due to space limitations we can't provide details on the extraction process for each of the above classes; we briefly discuss some typical cases. Abbreviation questions include questions asking for an abbreviation and questions asking for an expansion of an abbreviation. To collect abbreviation-expansion pairs we made a single pass through the document collection to identify strings of capitals in brackets; upon finding one we extracted sequences of capitalized non-stopwords preceding it (with about as many words as the length of the capitalized string).

While extracting abbreviation-expansion information requires no background knowledge, the "adjective-location" category does. Using EU-guidelines for translators on official adjective-to-country/location mappings, we extracted phrases such as "Mexicaanse vulkaan Popocatepetl" to produce facts such as "Popocatepetl,vulkaan,in Mexico,NH19940718-0041".

We used more background knowledge to populate the roles table: from the Dutch part of EuroWordNet we compiled a list of about 900 professions; syntactic appositions of the form "`<name>, <clause-involving-a-profession>,` were

then extracted to obtain descriptions of the people identified using `<name>`. A sample fact extracted in this manner is "Ally Derks, oprichter en directeur van het International Documentary Filmfestival Amsterdam (IDFA), NH19941207-0070".

None of the regular expressions used for extraction were meant to be exhaustive of all possible varieties of patterns in which the information being extracted occurs. They are straightforward, but reasonably high precision implementations meant to extract a large proportion of the patterns in the text. After the initial harvesting step, various cleaning up steps were applied to filter out noise. Again, in future work we plan to use machine learning techniques (trained on a small sample of manually extracted facts), but for the current proof-of-concept stage of our offline extraction for QA work, we simply devised some rules by hand to reduce noise.

Look up. The generated tables are simple text files, rather than structured databases; in addition, the actual "key" for the lookup in them is not directly given, and has to be analyzed from the format of the question. We therefore use the following method for the lookup: after identifying relevant knowledge base files using the question type, we try to locate lines in tables that match the question focus. If these are not found, we look for lines that contain as many terms from the focus as possible, giving priority to capitalized words. If matching lines are found, the candidate answers are extracted and ranked according to their frequencies.

The Impact of Using Tabular Data. In order to determine the effect of using tabular data, we evaluated the performance of our system in three different ways: with all five streams, with all streams except for *Table Lookup* and with *Table Lookup* alone. An official CLEF assessment exists only for the entire five-stream system; the rest of the evaluations were done by us manually, in a compatible way to the CLEF evaluations. Like the official CLEF evaluation, we considered a question answered correctly if at least one of the top three answers given by the system was correct. We refer here to the lenient (non-strict) results (although for answers obtained from the *Table Lookup* stream it is guaranteed that the document given as justification indeed contains the answer).

Table 2 lists the number of correctly answered questions for two sets of questions: both the whole set of 200 questions, and the 187 questions that contain answers in the collection. In the first case the performance is somewhat better — mainly because our system always includes NIL (*no answer*) in the top three answers, and thus according to our evaluation scheme, questions without answer in the collection are always answered correctly.

# Questions	Only *Table Lookup*	Without *Table Lookup*	All five streams
200 (all)	54 (27%)	64 (32%)	89 (45%)
187 (with answer)	41 (22%)	51 (27%)	76 (41%)

Table 2: Evaluation: the number of questions answered correctly.

Looking at the bottom row in Table 2, we see that the *Table Lookup* stream provides correct answers for 22% of the questions, and, more importantly, 14% of

the questions was only answered correctly by the *Table Lookup* stream (i.e., not by any of the other four streams).

So where did the *Table Lookup* stream prove to be especially helpful? And how significant was the contribution? Table 3 gives a breakdown in terms of the question categories for which the system with the *Table Lookup* stream gives correct answers, while the system without *Table Lookup* fails.

# Questions	Category	Example
8	Capitals	Wat is de hoofdstad van Zuid-Afrika?
7	Inhabitants	Hoeveel inwoners heeft Sydney?
5	Roles/Leaders	Wie is de voorzitter van de Europese Commissie?
4	Abbreviations	Waar staat GATT voor?
3	Locations	In welke stad is het Europese Parlement?
1	Currencies	Hoe heet de Chinese munteenheid?

Table 3: Categories of the questions for which the *Table Lookup* stream helps.

Summing up the entries in column 1 of Table 3, we see that the *Table Lookup* stream correctly answers 28 questions not answered correctly by the other streams. In total, 72 questions looked for answers that could (in principle) be present in our tables. Table 4 gives the performance analysis for these questions: the number of questions that were correctly vs. incorrectly answered by the *Table Lookup* stream alone and all five streams vs. the four streams without *Table Lookup*.

		Table Lookup		*All five streams*	
		correct	incorrect	correct	incorrect
four streams	correct	13	8	19	2
	incorrect	29	22	28	23

Table 4: Breakdown of *Table Lookup* results.

On the types of questions on which it could potentially make a difference, the *Table Lookup* stream made a significant difference (using the sign test, with $p < 0.01$). It is worth noting that the *Table Lookup* stream still leaves lots of room for improvement: only 42 out of 72 relevant questions were answered correctly by the *Table Lookup* stream (58.3%); note that 6 of the missing answers were found by the other four stream (at the expense of 1 previously correct answer), leading to a total of 47 out 72 relevant questions answered correctly (65.3%).

The most common errors encountered in the incorrect answers produced by the *Table Lookup* stream were:

- We failed to extract the required information. E.g. the sentence "...Australië's formele staatshoofd, de Britse koningin Elizabeth..." didn't produce the entry (Australië, formele staatshoofd, de Britse koningin Elizabeth) in the Leaders table, because there was no pattern for this type of phrase.
- Some of the heuristics used for retrieving information from the tables did not always work. For this reason, e.g., for question "Wie is de president van

Rusland?" the system answered "Rusland."

Addressing these errors is part of our ongoing work.

5 Conclusion

In this paper we explored the use of offline generated lookup tables for answering certain types of factoid questions. The idea was implemented in a *Table Lookup* stream as part of our participation in the CLEF Dutch QA evaluation exercise. We found that the *Table Lookup* stream made a significant difference, providing correct answers for 58% of all relevant questions. Our ongoing and future work concerns adapting the ideas described here to the AQUAINT corpus used for the TREC QA evaluation, applying machine learning techniques to identify suitable patterns for populating tables and to clean up the output of those patterns. In addition, we want to extend our patterns to include additional question categories, such as age and date of birth or death.

Acknowledgements

This research was supported by the Netherlands Organization for Scientific Research (NWO) under project number 220-80-001. Maarten de Rijke was also supported by NWO under project numbers 612-13-001, 365-20-005, 612.069.006, 612.000.106, and 612.000.207.

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] M. Maybury, editor. *AAAI Spring Symposium on Future Directions in Question Answering*, 2003.

[4] C. Monz and M. de Rijke. Tequesta: The University of Amsterdam's Textual Question-Answering System. In *Notebook papers TREC-10*, 2001.

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

[6] M. de Rijke and B. Webber, editors. *EACL 2003 Workshop on Natural Language Processing for Question Answering*, 2003.

[7] E. Riloff. Automatically generating extraction patterns from untagged text. In *Proceedings AAAI-96*, pages 1044–1049, 1996.

[8] R. Simmons. Answering English questions by computer: A survey. *Communications of the ACM*, 8:53–70, 1965.

[9] E.M. Voorhees. Overview of the TREC 2002 question answering track. In Voorhees and Harman [10].

[10] E.M. Voorhees and D.K. Harman, editors. *The Eleventh Text REtrieval Conference (TREC 2002)*. National Institute for Standards and Technology, 2003.