The University of Amsterdam at QA@CLEF 2004

Valentin Jijkoun Gilad Mishne Maarten de Rijke
Stefan Schlobach David Ahn Karin Müller

Informatics Institute, University of Amsterdam
Kruislaan 403, 1098 SJ Amsterdam, The Netherlands
E-mail: {jijkoun,gilad,mdr,schlobac,ahn,kmueller}@science.uva.nl

Abstract

This paper describes the official runs of our team for the CLEF 2004 question answering tasks. We took part in the monolingual Dutch task and in the bilingual English to Dutch task.

1 Introduction

To address the question answering (QA) task one has to address a challenging *recall* problem. As with all language processing tasks, in QA we face a vocabulary gap — the phenomenon that the question and its answer(s) may be phrased in different vocabularies. For QA the vocabulary gap can be especially challenging as systems have to return highly relevant and focused text snippets as output, given very short questions as input. To address the recall problem that QA confronts us with, we advocate a *multi-stream* architecture which implements multiple ways of identifying candidate answers, complemented with elaborate filtering mechanisms to weed out incorrect candidate answers. In 2003 we completed a first version of this architecture, of which we made good use for the QA tracks both at CLEF [8] and at TREC [9]. For the 2004 edition of the QA@CLEF task we built on the same architecture.

This year, we took part in the monolingual Dutch task, and in the bilingual English to Dutch task. Our main aim with our monolingual work was to extend and improve our QA system following an error analysis after the 2003 edition of the task. The bilingual English to Dutch task was new for us. Our main aim here was to evaluate the applicability of our system in a cross-language setting, and to check whether correct results obtained by the bilingual run are a subset of the monolingual one — or whether something can be gained by combining them.

The paper is organized as follows. In Section 2 we describe the architecture of our QA system. Section 3 describes our official runs. In Section 4 we discuss the results we have obtained and give a preliminary analysis of the performance of different components of the system. We conclude in Section 5.

2 System Description

Many QA systems share the following pipeline architecture. 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 based on the question, and an information retrieval engine is used to identify a list of documents that are likely to contain the answer. Those documents 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 [10] to ones that rely heavily on the redundancy of information available on the World Wide Web [4].

In essence, our system architecture implements multiple copies of the standard architecture, each of which is a complete standalone QA system that produces ranked answers, but not necessarily for all types of questions; the overall system's answer is then selected from the combined pool of candidates through a combination of merging and filtering techniques. For a reasonably detailed discussion of our QA system architecture we refer to [8, 9]. A general overview of the system is given in Figure 1. This year, we improved our question classifier by incorporating Dutch WordNet to deal with questions such as *Which X... ?*, where *X* represents a person, animal, agent etc. This year's system contains 8 streams, organized in four groups, depending on the main data source from which they try to answer questions. We now provide a brief description of these four groups.

Streams that Consult the Dutch CLEF Corpus. Four streams generate candidate answers from the Dutch CLEF corpus: *Lookup*, *Pattern Match*, *Ngrams*, and *Tequesta*. The *Table Lookup* stream uses specialized knowledge bases constructed by preprocessing the collection, exploiting the fact that certain information types (such as country

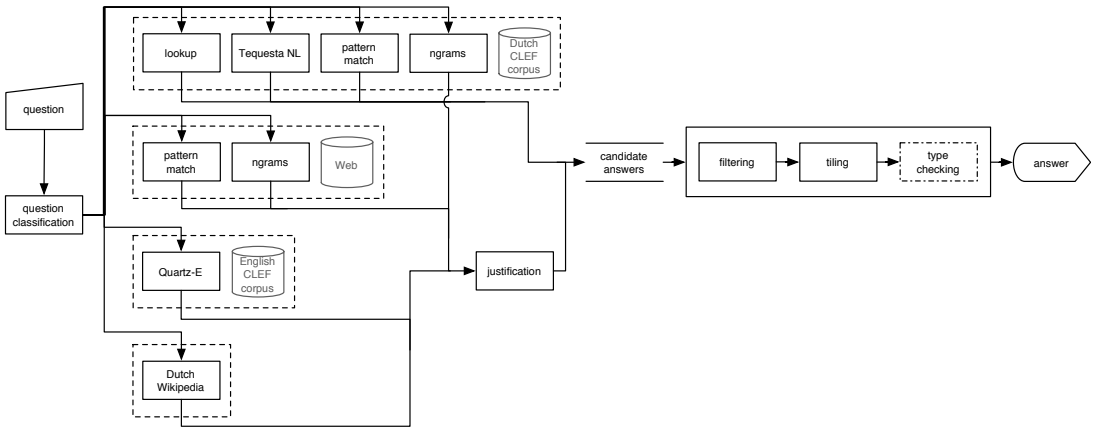

Figure 1: Quartz-N: the University of Amsterdam's Dutch Question Answering System.

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 a detailed overview of this stream, see [7]. In addition to the knowledge bases used in CLEF 2003, we built new ones (such as AWARDS and MEASUREMENTS, storing facts about winners of various prizes, and information about dimensions of objects, respectively). Furthermore, we enriched our previous knowledge bases with information extracted not with surface patterns but with syntactic patterns on top of a version of the Dutch CLEF collection that was parsed using the Alpino parser, a wide coverage dependency parser for Dutch [1]. Earlier experiments on the AQUAINT corpus had suggested that offline extraction using syntactic extraction patterns can substantially improve recall [6].

The *Dutch Tequesta* stream is a linguistically informed QA system for Dutch that implements the traditional architecture outlined above. Amongst others, it uses a Part-of-Speech tagger (TNT [2] trained on the *Corpus Gesproken Nederlands* [12]), a home-grown named entity tagger for Dutch, as well as proximity-based candidate answer selection [11].

In the *Pattern Match* stream, zero or more regular expressions 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.

The *Ngram* stream, similar in spirit to [3], constructs a weighted list of queries for each question using a shallow reformulation process, similar to the Pattern Match stream. These queries are fed to a retrieval engine (we used our home-grown FlexIR, with the `Lnu.ltc` weighting scheme), and the top retreived documents are used for harvesting word ngrams. The ngrams were ranked according to the weight of the query that generated them, their frequency, NE type, the proximity to the query keywords and more parameters; the top-ranking ngrams were considered candidate answers. The output of this stream is piped to the *Justification* module (see below).

As mentioned earlier, we aim at higher recall at the earlier stages, relying on various filtering mechanisms to "clean" the results and achieve also high precision. Therefore, for both the *Ngram* and the *Pattern Match* streams, we extended the generated regular expressions and queries, compared to our system at CLEF 2003 — sometimes creating ungrammatical ones, but we assumed that the few results they would produce would be filtered out later.

Streams that Consult the Web. Quartz-N has two streams that attempt to locate answers on the web: *Ngram mining* and *Pattern Match*. For Web searches, we used Google, and ngrams were harvested from the snippets provided by it. Pattern matching was done against *full* documents returned by Google.

Streams that Consult the English CLEF Corpus. One of the streams used by Quartz-N is the English language version of our QA system, which consults the English CLEF corpus instead of the Dutch version (but which is otherwise similar to the Dutch version). The answers found by Quartz-E are also piped to the *Justification* module.

Streams that Use Other Resources. One of the new streams this year was the Wikipedia stream. Similarly to the streams that consult the Web or the English document collection, this stream also uses an external corpus — the Dutch Wikipedia (`http://nl.wikipedia.org`), an open-content encyclopedia in Dutch (and other languages). However, since this corpus is much "cleaner" than news paper text, the stream operates in a different manner. First, the *focus* of the question is identified; this is usually the main named entity in the question. Then, this entity's encyclopedia entry is looked up; since Wikipedia is standardized to a large extent, this information has a

template-like nature. Finally, using knowledge about the templates used in Wikipedia, information such as DATE-OF-DEATH and FIRST-NAME can easily be extracted.

While each of the above streams is a "small" QA system in itself, many components are shared between the streams, including an *Answer Justification* and a *Filtering, Tiling and Type Checking* module, both of which we will now describe.

Answer Justification. As some of our streams obtain candidate answers *outside* the Dutch CLEF corpus, and as answers need to be supported, or *justified*, by a document in the Dutch CLEF corpus, we need to find justification for externally found candidate answers. To this end we construct a query with keywords from a given question and candidate answer, and consider the top-ranking document for this query to be the justification, using an Okapi model as this tends to do well on early high precision in our experience. Additionally, we use some retrieval heuristics such as marking the answer as boolean terms in the query (requiring them to appear in retrieved documents).

Filtering, Tiling, and Type Checking. A detailed error analysis carried out after the 2003 edition of QA@CLEF revealed that the two most important sources of errors were answer selection and named entity recognition [8]. For this year's task we used a new final answer selection module (similar to that described in [5]) with heuristic candidate answer filtering and merging, and with stream voting. To compensate for named entity errors made during answer extraction, our type checking module (see [13] for details) uses several geographical knowledge bases to remove candidates of incorrect type for location questions.

3 Runs

We submitted two runs for the monolingual Dutch question answering task: `uams041nlnl` and `uams042nlnl`, and one run for the bilingual English to Dutch task: `uams041ennl`. All runs return exact answers, and combine answers from all streams. The `uams042nlnl` run is identical to `uams041nlnl`, with additional filtering and sanity checks on the candidate answers before selecting the final answer. These checks included zero-count filters (assuming that answers which do not appear as a phrase on the web are incorrect, and questions for which the focus does not appear in the local collection have no answer), and type-checking for location questions [13]. Our bilingual run included a simple translation of the questions from English to Dutch using a publicly-available interface of Systran (`http://www.systranet.com`), and then using Quartz-N for for the translated questions.

4 Results and Discussion

The following table shows the evaluation results of our CLEF 2004 submissions. Beside the standard *Right*, *Wrong*, *Inexact*, and *Unsupported* measures, we also list various accuracy figures.

					Overall	*Accuracy*	*Accuracy*	*NIL accuracy*	
Run	*Right*	*Wrong*	*Inexact*	*Unsupported*	*accuracy*	*over F*	*over D*	*precision*	*recall*
`uams041nlnl`	88	98	10	4	44.00%	42.37%	56.52%	0.00	0.00
`uams042nlnl`	91	97	10	2	45.50%	45.20%	47.83%	0.56	0.25
`uams041ennl`	70	122	7	1	35.00%	31.07%	65.22%	0.00	0.00

The run `uams042nlnl` scored slightly better than `uams041nlnl`. Interestingly, the gain is in the factoids only: the run `uams042nlnl` actually scored worse on definitions than `uams041nlnl`. Had we combined `uams042nlnl`'s answers to factoids with `uams041nlnl`'s answers to definition question, we would have obtained on overall accuracy of 46.5%. This suggests that factoids benefit from additional checks and filters (that work well on short candidate answers), while definition questions benefit from a more lenient approach. Additionally, our filters prove useful for detecting questions with no answers: 5 out of the 9 NIL answers returned (as part of the run `uams042nlnl`) were correctly identified using the filters, while none were identified without them.

The overall accuracy of the bilingual run `uams041ennl` is less than that of the monolingual runs; this can largely be attributed to the imperfect machine translation used. However, it is interesting to note that the correct answers provided in this run are not a subset of the correct answers provided by the monolingual runs; while 44 questions (22%) were answered correctly by `uams041nlnl` and not by `uams041ennl`, there are 25 questions (12.5%) that were answered correctly by the bilingual run and not the monolingual one.

To analyze the contribution of different answering streams to the performance of the whole system, we carried out a number of experiments, disabling each stream individually and evaluating the resulting "smaller" systems using the assessors' judgements available for our official runs. The *Lookup* stream proved to be the most essential

(the system answers 19 more questions with the *Lookup* on), followed by the *Ngrams* streams (11 and 4, for the Web and Collection Ngrams, respectively) and the *Collection Pattern Match* stream (3 more answers). Moreover, our final answer selection module makes use of the essential redundancy of the multi-stream architecture: 70% of the correct answers come from two or more answering streams.

5 Conclusions and Future Work

We presented our multi-stream question answering system and the runs it produced for CLEF 2004. 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 ongoing work on the system is focused on additional filtering and type checking mechanisms, and on exploiting high-quality external resources such as the CIA world fact book, Wikipedia, and WordNet.

Acknowledgments

We are grateful to Gertjan van Noord for supplying us with a dependency parsed version of the Dutch CLEF corpus. This research was supported by the Netherlands Organization for Scientific Research (NWO) under project number 220-80-001. In addition, Maarten de Rijke was also supported by grants from NWO, under project numbers 365-20-005, 612.069.006, 612.000.106, 612.000.207, 612.066.302, and 264-70-050.

References

[1] G. Bouma, G. Van Noord, and R. Malouf. Alpino: Wide-coverage computational analysis of Dutch. In *Computational Linguistics in The Netherlands 2000*. 2001.

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

[3] S. Dumais, M. Banko, E. Brill, J. Lin, and A. Ng. Web question answering: Is more always better? In P. Bennett, S. Dumais, and E. Horvitz, editors, *Proceedings of SIGIR'02*, pages 291–298, 2002.

[4] M. Banko et al. AskMSR: Question answering using the Worldwide Web. In *Proceedings EMNLP 2002*, 2002.

[5] V. Jijkoun and M. de Rijke. Answer selection in a multi-stream open domain question answering system. In S. McDonald and J. Tait, editors, *Proceedings 26th European Conference on Information Retrieval (ECIR'04),*, volume 2997 of *LNCS*, pages 99–111. Springer, 2004.

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

[7] 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)*, 2003.

[8] V. Jijkoun, G. Mishne, and M. de Rijke. How frogs built the Berlin Wall. In *Proceedings CLEF 2003*, LNCS. Springer, 2004.

[9] V. Jijkoun, G. Mishne, C. Monz, M. de Rijke, S. Schlobach, and O. Tsur. The University of Amsterdam at the TREC 2003 Question Answering Track. In *Proceedings TREC 2003*, pages 586–593, 2004.

[10] D. Moldovan, S. Harabagiu, R. Girju, P. Morarescu, F. Lacatusu, A. Novischi, A. Badulescu, and O. Bolohan. LCC Tools for Question Answering. In Voorhees and Harman [14].

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

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

[13] S. Schlobach, M. Olsthoorn, and M. de Rijke. Type checking in open-domain question answering. In *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI 2004)*, 2004.

[14] E.M. Voorhees and D.K. Harman, editors. *The Tenth Text REtrieval Conference (TREC 2002)*. National Institute for Standards and Technology. NIST Special Publication 500-251, 2003.