The QuALiM Question Answering Demo:
Supplementing Answers with Paragraphs drawn from Wikipedia

Michael Kaisser
School of Informatics
University of Edinburgh
M.Kaisser@sms.ed.ac.uk

Abstract

This paper describes the online demo of the QuALiM Question Answering system. While the system actually gets answers from the web by querying major search engines, during presentation answers are supplemented with relevant passages from Wikipedia. We believe that this additional information improves a user's search experience.

1 Introduction

This paper describes the online demo of the QuALiM[1] Question Answering system (http://demos.inf.ed.ac.uk:8080/qualim/). We will refrain from describing QuALiM's answer finding strategies–our work on QuALiM has been described in several papers in the last few years, especially Kaisser and Becker (2004) and Kaisser et al. (2006) are suitable to get an overview over the system–but concentrate on one new feature that was developed especially for this web demo: In order to improve user benefit, answers are supplemented with relevant passages from the online encyclopedia Wikipedia. We see two main benefits:

1. Users are presented with additional information closely related to their actual information need and thus of potential high interest.

2. The returned text passages present the answer in context and thus help users to validate the answer–there always will be the odd case where a system returns a wrong result.

---

[1]for *Question Answering with Linguistic Methods*

Historically, our system is web-based, receiving its answers by querying major search engines and post processing their results. In order to satisfy TREC requirements–which require participants to return the ID of one document from the AQUAINT corpus that supports the answer itself (Voorhees, 2004)–we already experimented with answer projection strategies in our TREC participations in recent years. For this web demo we use Wikipedia instead of the AQUAINT corpus for several reasons:

1. QuALiM is an open domain Question Answering system and Wikipedia is an "open domain" Encyclopedia; it aims to cover *all* areas of interest as long as they are of *some* general interest.

2. Wikipedia is a free online encyclopedia. Other than the AQUAINT corpus, there are no legal problems when using it for a public demo.

3. Wikipedia is frequently updated, whereas the AQUAINT corpus remains static and thus contains a lot of outdated information.

Another advantage of Wikipedia is that the information contained is much more structured. As we will see, this structure can be exploited to improve performance when finding answers or–as in our case–projecting answers.

2 How Best to Present Answers?

In the fields of Question Answering and Web Search, the issue how answers/results should be presented is a vital one. Nevertheless, as of today, the majority of QA system–which a few notable exceptions, e.g. MIT's START (Katz et al., 2002)–are
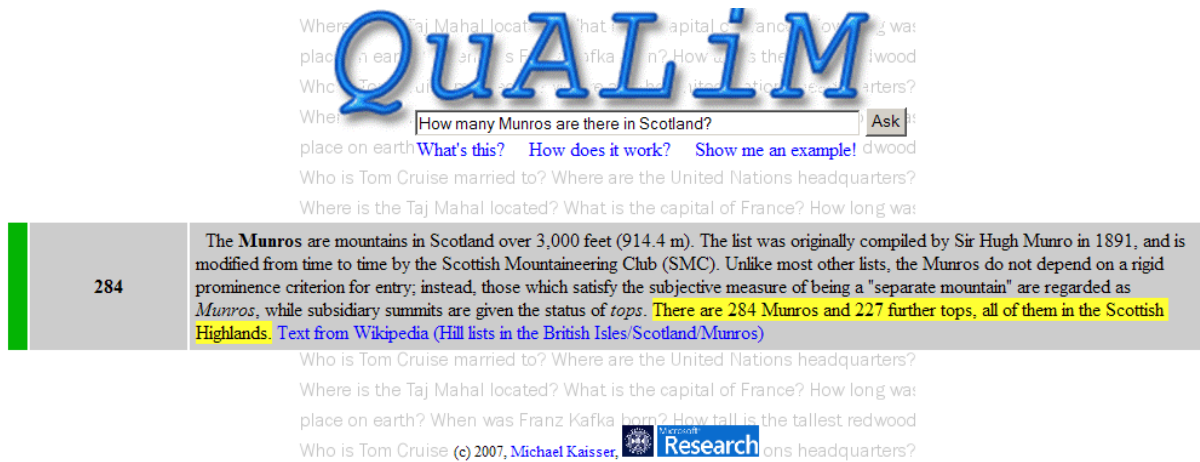
Figure 1: Screenshot of QuALiM's response to the question "How many Munros are there in Scotland?" The green bar to the left indicates that the system is confident to have found the right answer, which is shown in bold: "284". Furthermore, one Wikipedia paragraph which contains additional information of potential interest to the user is displayed. In this paragraph the sentence containing the answer is highlighted. This display of context also allows the user to validate the answer.

still experimental and research-oriented and typically only return the answer itself. Yet it is highly doubtful that this is the best strategy.

Lin et al. (2003) performed a study with 32 computer science students comparing four types of answer context: exact answer, answer-in-sentence, answer-in-paragraph, and answer-in-document. Since they were interested in interface design, they worked with a system that answered all questions correctly. They found that 53% of all participants preferred paragraph-sized chunks, 23% preferred full documents, 20% preferred sentences, and one participant preferred exact answer.

Web search engines typically show results as a list of titles and short snippets that summarize how the retrieved document is related to the query terms, often called *query-biased summaries* (Tombros and Sanderson, 1998). Recently, Kaisser et al. (2008) conducted a study to test whether users would prefer search engine results of different lengths (phrase, sentence, paragraph, section or article) and whether the optimal response length could be predicted by human judges. They find that judges indeed prefer different response lengths for different types of queries and that these can be predicted by other judges.

In this demo, we opted for a slightly different, yet related approach: The system does not decide on one answer length, but always presents a combination of three different lengths to the user (see Figure 1): The answer itself (usually a *phrase*), is presented in bold. Additionally, a *paragraph* relating the answer to the question is shown, and in this paragraph one *sentence* containing the answer is highlighted. Note also, that each paragraph contains a link that takes the user to the Wikipedia *article*, should he/she want to know more about the subject. The intention behind this mode of presentation is to prominently display the piece of information the user is most interested in, but also to present context information and to furthermore provide options for the user to find out more about the topic, should he/she want to.

3 Finding Supportive Wikipedia Paragraphs

We use Lucene (Hatcher and Gospodnetić, 2004) to index the publically available Wikipedia dumps (see http://download.wikimedia.org/). The text inside the dump is broken down into paragraphs and each paragraph functions as a Lucene *document*. The data of each paragraph is stored in three fields: *Title*, which contains the title of the Wikipedia article the paragraph is from, *Headers*, which lists the title and all section and subsection headings indicating the position of the paragraph in the article and *Text*, which stores the text of the article. An example can be seen

33

in Table 1.

Title	"Tom Cruise"
Headers	"Tom Cruise/Relationships and personal life/Katie Holmes"
Text	"In April 2005, Cruise began dating Katie Holmes ... the couple married in Bracciano, Italy on November 18, 2006."

Table 1: Example of Lucene index fields used.

As mentioned, QuALiM finds answers by querying major search engines. After post processing, a list of answer candidates, each one associated with a confidence value, is output. For the question "Who is Tom Cruise married to?", for example, we get:

```
81.0: "Katie Holmes"
35.0: "Nicole Kidman"
```

The way we find supporting paragraphs for these answers is probably best explained by giving an example. Figure 3 shows the Lucene query we use for the mentioned question and answer candidates. (The numbers behind the terms indicate query weights.) As can be seen, we initially build two separate queries for the *Headers* and the *Text* fields (compare Table 1). In a later processing step, both queries are combined into a single query using Lucene's `MultipleFieldQueryCreator` class. Note also that both answer candidates ("Katie Holmes" and "Nicole Kidman") are included in this one query. This is done because of speed issues: In our setup, each query takes up roughly two seconds of processing time. The complexity and length of a query on the other hand has very little impact on speed.

The type of question influences the query building process in a fundamental manner. For the question "When was Franz Kafka born?" and the correct answer "July 3, 1883", for example, it is reasonable to search for an article with title "Franz Kafka" and to expect the answer in the text on that page. For the question "Who invented the automobile?" on the other hand, it is more reasonable to search the information on a page called "Karl Benz" (the answer to the question). In order to capture this behaviour we developed a set of rules that for different type of questions, increases or decreases constituents' weights in either the *Headers* or the *Text* field.

Additionally, during question analysis, certain question constituents are marked as either *Topic* or *Focus* (see Moldovan et al., (1999)). For the earlier example question "Tom Cruise" becomes the *Topic* while "married" is marked *Focus*[2]. These also influence constituents' weights in the different fields:

- Constituents marked as *Topic* are generally expected to be found in the *Headers* field. After all, the topic marks *what the question is about*. In a similar manner, titles and subtitles help to structure an article, assisting the user to navigate to the place where the relevant information is most likely to be found: A paragraph's titles and subtitles indicate *what the paragraph is about*.

- Constituents marked as *Focus* are generally expected to be found in the text, especially if they are verbs. The focus indicates what the question asks for, and such information can usually rather be expected in the text than in titles or subtitles.

Figure 3 also shows that, if we recognize named entities (especially person names) in the question or answer strings, we once include each named entity as a quoted string and additionally add the words it contains separately. This is to boost documents which contain the complete name as used in the question or the answer, but also to allow documents which contain variants of these names, e.g. "Thomas Cruise Mapother IV".

The formula to determine the exact boost factor for each query term is complex and a matter of ongoing development. It additionally depends on the following criteria:

- Named entities receive a higher weight.

- Capitalized words or constituents receive a higher weight.

- The confidence value associated with the answer candidate influences the boost factor.

- Whether a term originates from the question or an answer candidate influences its weight in a different manner for the header and text fields.

---

[2]With allowing verbs to be the *Focus*, we slightly depart from the traditional definition of the term.

Header query:
```
"Tom Cruise"^10 Tom^5 Cruise^5 "Katie Holmes"^5 Katie^2.5 Holmes2.^5
"Nicole Kidman"^4.3 Nicole^2.2 Kidman^2.2
```
Text query:
```
married^10 "Tom Cruise"^1.5 Tom^4.5 Cruise^4.5 "Katie Holmes"^3 Katie^9 Holmes^9
"Nicole Kidman"^2.2 Nicole^6.6 Kidman^6.6
```

Figure 2: Lucene Queries used to find supporting documents for the "Who is Tom Cruise married to?" and the two answers "Katie Holmes" and "Nicole Kidman". Both queries are combined using Lucene's MultipleFieldQueryCreator class.

4 Future Work

Although QuALiM performed well in recent TREC evaluations, improving precision and recall will of course always be on our agenda. Beside this we currently focus on increasing processing speed. At the time of writing, the web demo runs on a server with a single 3GHz Intel Pentium D dual core processor and 2Gb SDRAM. At times, the machine is shared with other demos and applications. This makes reliable figures about speed difficult to produce, but from our log files we can see that users usually wait between three and twelve seconds for the system's results. While this is okay for a research demo, it definitely would not be fast enough for a commercial product. Three factors contribute with roughly equal weight to the speed issue:

1. Search engine's APIs usually do not return results as fast as their web interfaces built for human use do. Google for example has a built-in one second delay for each query asked. The demo usually sends out between one and four queries per question, thus getting results from Google alone takes between one and four seconds.

2. All received results need to be post-processed, the most computing heavy step here is parsing.

3. Finally, the local (8.3 GB big) Wikipedia index needs to be queried, which roughly takes two seconds per query.

We are currently looking into possibilities to improve all of the above issues.

Acknowledgements

References

Erik Hatcher and Otis Gospodnetić. 2004. *Lucene in Action*. Manning Publications Co.

Michael Kaisser and Tilman Becker. 2004. Question Answering by Searching Large Corpora with Linguistic Methods. In *The Proceedings of the 2004 Edition of the Text REtrieval Conference, TREC 2004*.

Michael Kaisser, Silke Scheible, and Bonnie Webber. 2006. Experiments at the University of Edinburgh for the TREC 2006 QA track. In *The Proceedings of the 2006 Edition of the Text REtrieval Conference, TREC 2006*.

Michael Kaisser, Marti Hearst, and John Lowe. 2008. Improving Search Result Quality by Customizing Summary Lengths. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*.

Boris Katz, Jimmy Lin, and Sue Felshin. 2002. The START multimedia information system: Current technology and future directions. In *Proceedings of the International Workshop on Multimedia Information Systems (MIS 2002)*.

Jimmy Lin, Dennis Quan, Vineet Sinha, Karun Bakshi, David Huynh, Boris Katz, and David R. Karger. 2003. What Makes a Good Answer? The Role of Context in Question Answering. *Human-Computer Interaction (INTERACT 2003)*.

Dan Moldovan, Sanda Harabagiu, Marius Pasca, Rada Mihalcea, Richard Goodrum, Roxana Girju, and Vasile Rus. 1999. LASSO: A tool for surfing the answer net. In *Proceedings of the Eighth Text Retrieval Conference (TREC-8)*.

A. Tombros and M. Sanderson. 1998. Advantages of query biased summaries in information retrieval. *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 2–10.

Ellen M. Voorhees. 2004. Overview of the TREC 2003 Question Answering Track. In *The Proceedings of the 2003 Edition of the Text REtrieval Conference, TREC 2003*.