Web Reinforced Question Answering (MultiText Experiments for TREC 2001)

C. L. A. Clarke G. V. Cormack T. R. Lynam C. M. Li G. L. McLearn

Computer Science, University of Waterloo, Waterloo, Ontario, Canada
mt@plg.uwaterloo.ca

1 Introduction

For TREC 2001, the MultiText Project concentrated on the QA track. Over the past year, we made substantial enhancements to our QA system in three general areas. First, we explored a number of methods for taking advantage of external resources (including encyclopedias, dictionaries and Web data) as sources for answer validation, improving our ability to identify correct answers in the target corpus. Of the methods explored, the use of Web data to reinforce answer selection proved to be particular value. Second, we made a large number of incremental improvements to the existing system components. For example, in our parsing component, the query generation and answer category identification algorithms were extended and tuned, as were the named entity identification algorithms used in our answer extraction component. Finally, we made a careful analysis of the problem of *null questions*, those that have no answer in the target corpus, and developed a general approach to the problem. A basic method for handling null questions, based on the analysis, was added to our system.

We submitted three runs for the main task of the QA track. The first run (`uwmta1`) was based on the enhanced system described above, including the full use of Web resources for answer validation. For the second run (`uwmta2`) the Web resources were not used for validation, but the system was otherwise identical. A comparison between these runs represents a major goal of our TREC experimental work and the major concern of this paper. The final run (`uwmta0`) tests a last-minute enhancement. For this run a feedback loop was added to the system, in which candidate answer terms were merged back into the query used for passage retrieval. While answer feedback was not an area of significant effort for TREC 2001, and the intial results were disappointing, it represents an area in which future work is planned.

Our other TREC 2001 runs are related to the QA track. Along with the QA runs submitted for the main task, we also submitted exploratory runs for the list (`uwmtal0` and `uwmtal1`) and context (`uwmtac0`) tasks. These runs were generated through minor modifications to the existing system, and represent preliminary attempts at participation rather than serious attempts at high performance. Our runs for the Web track (`uwmtaw0`, `uwmtaw1`, and `uwmtaw2`) are related to our QA runs. These runs were generated by our QA system by treating the topic title as a question and using the ranked list of documents containing the best answers as the result. Finally, the runs submitted by Sun Microsystems (`mtsuna0` and `mtsuna1`) were generated using our system as the backend and the Sun parser as the frontend. However, the integration between Sun and MultiText was performed in a short period of time, and these runs should also be viewed as preliminary experiments that point toward future work.

In the remainder of the paper we focus on our primary runs for the main task of the QA track. In the next section we provide an overview of the QA system used for our TREC 2001 experiments, including a discussion of our technique for Web reinforcement. In section 3 we present our approach to the problem of null questions. Section 4 details our experimental results.

2 The MultiText QA System

The MultiText QA system was introduced in our TREC-9 paper [1]. and described in further detail in a related SIGIR paper [2] This section presents an updated summary of the system.

Figure 1 provides an overview the version of our QA system used in our TREC 2001 experiments. In our approach, question answering consists of three major processing steps: question parsing, passage retrieval and answer selection.

The MultiText QA parser has two main functions: 1) to generate queries so that the retrieval engine can extract the best candidate passages, and 2) to generate answer selection rules so that the post-processor can select the best 50-byte answer fragment from the passages. The answer selection rules generated by the parser include a category for the question (`<name>`, `<place>`, etc.) and patterns that may be matched in the extracted passages to identify possible answer locations.

Queries generated by the parser are fed to the passage retrieval engine. For question answering, we have developed a passage retrieval technique that can identify small excerpts that cover as many question concepts as possible. Unlike most other passage retrieval techniques, our technique does not require predefined passages, such as paragraphs, sentences or n-word segments, but can retrieve any document substring in the target corpus. The score of substring depends on its length, the number of question concepts it contains and the relative weight assigned to each of these concepts. Once the k highest-scoring substrings from distinct documents are identified, the centerpoint of each substring is computed and a 1000-word passage centered at this point is retrieved from the corpus. These 1000-word passages were then used by the answer selection component to determine the final answers fragments.

The answer selection component identifies possible answers ("candidates") from the passages and then ranks these candidates using a variety of heuristics. These heuristics take into account the number of times each candidate appears in the retrieved passages, the location of the candidate in the retrieved passages, the rank of the passages in which the candidate appears, the likelihood that the candidate matches the assigned answer category, and other special-case information provided by the selection rules.

Since the goal of the TREC 2001 QA experiments was to select 50-byte answer fragments from the retrieved passages, the answer selection technique used to generate our experimental runs does not attempt to identify candidates that are exact or complete answers. Instead, candidates are single terms, where the nature of these terms depends on the category of the question. For example, if a question asks for a proper noun, the candidates consist of those terms that match a simple syntactic pattern for proper nouns; if a question asks for a length, the candidates consist of those numeric values that precede appropriate units; and if a question cannot be classified, the candidates simply consist of all non-query and non-stopword terms appearing in the retrieved passages.

After identification, each candidate term t is assigned a weight that takes into account the number of distinct passages in which the term appears, as well as the relative frequency of the term in the database:

$$w_t = c_t \log(N/f_t),$$

where N is sum of the lengths of all documents in the database, f_t is the number of occurrences

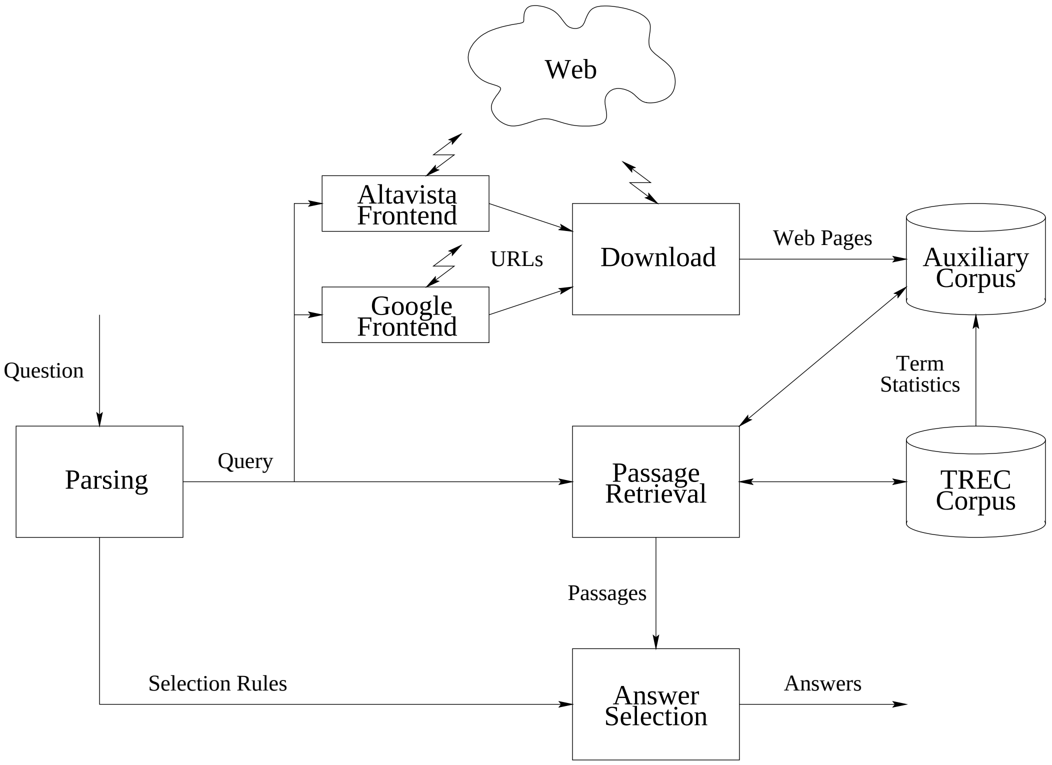

Figure 1: QA System overview.

of t in the database, and $1 \leq c_t \leq k$ is the number of distinct passages in which t appears. The value c_t, which represents the "redundancy" associated with the candidate, is a critical element of the answer selection process [2].

The weights of the candidates are used to select 50-byte answer fragments from the retrieved passages. Each 50 byte substring of the retrieved passages that starts or ends on a word boundary is considered to be a potential answer fragment. A score for each of these fragments is computed by summing the weights of the candidate terms that appear within it. Given a text fragment F and a set of candidates K, each term that appears in both F and K has its weight w_t temporarily modified to a position-specific weight w'_t using heuristics that take into account the rank of the passage in which the fragment appears, the location of the fragment relative to the centerpoint of the passage, and the selection rules generated by the parser. The resulting score for a fragment is

$$\sum_{t \in F \wedge t \in K} (w'_t)^\alpha$$

where a value of $\alpha = 3$ was used for all our TREC 2001 experiments.

Once the highest-scoring fragment is selected, the weights of the candidates appearing in that fragment are reduced to zero. All fragments are re-scored and the highest-scoring fragment is again selected. This process is repeated until five answer fragments have been selected.

At the level of detail given above, our QA system is little changed from TREC-9. However, for TREC 2001 we expanded and enhanced many of the heuristics in the parser and answer selection components. The number of question categories was increased from eight to 22, and these categories were arranged hierarchically. We extended and improved the pattern matching process for recognizing candidates corresponding to these categories. This extended matching process relies heavily on external resources, such as large lists of countries and cites, dictionaries, and the WordNet lexical database. If insufficient candidates are identified using the question category assigned by the parser, the selecion component considers patterns matching categories farther up the hierarchy. Finally, position- and rank-specific adjustments to candidate weights were modified to take the question category into account.

Another addition for TREC 2001 was the use of Web data to reinforce the scores of promising candidates by providing additional redundancy. As shown in figure 1, each question is used to generate appropriate queries for two commercial search engines. The contents of the top 200 documents returned by each engine were used to create an auxiliary database. The passage retrieval component then extracts 20 passages from the target database and 40 passages from the auxiliary database, recording the source of each passage. Since the contents of the auxiliary database is heavily biased by the query, term statistics from the target corpus are used during passages extraction from the auxiliary database.

All 60 passages are passed to the answer selection component. The answer selection component then proceeds to select answer fragments as usual, except that fragments cannot be selected from passages extracted out of the auxiliary database. The Web data influences the answer selection process only by increasing the redundancy factor c_t for particular candidates.

3 Null Questions

Null questions are those questions which have no answer in the target corpus. For TREC 2001, a corresponding null ("no answer") response was treated as a legitimate answer that could be included at any rank. If a question was judged to have no answer in the target corpus, null responses were marked as correct.

Given a question Q, a small number of parameters must be estimated to determine the best rank (if any) to place a null response. The first of these parameters is $p_0(Q)$, the probability that Q has no answer. In addition, for any question, our system will produce five ranked answer fragments. For each question, we have $p_i(Q)$, the probability that the highest-ranked fragment containing the correct answer is located at rank i, with $1 \leq i \leq 5$.

Given these parameters we can compute the expected mean reciprocal rank (MRR) for Q ($E_i(Q)$) under the assumption that the null response is placed at a particular rank i, pushing down the answer fragments appearing at the same rank and lower. For example,

$$E_3(Q) \;=\; \frac{p_0(Q)}{3} + p_1(Q) + \frac{p_2(Q)}{2} + \frac{p_3(Q)}{4} + \frac{p_4(Q)}{5}.$$

In addition we define $E_0(Q)$ as the expected MRR if a null response is not included:

$$E_0(Q) \;=\; \sum_{i=1}^{n} \frac{p_i(Q)}{i}.$$

The optimal location for a null response is then simply the value of i for which $E_i(Q)$ has maximum value:

$$\arg\max_{0 \leq i \leq 5} E_i(Q).$$

Estimating $p_i(Q)$ for a specific Q proved to be a difficult problem, and for TREC 2001 we simply used fixed estimates $p_i(Q) = p_i$ $(1 \leq i \leq 5)$ for all questions. These fixed estimates were derived

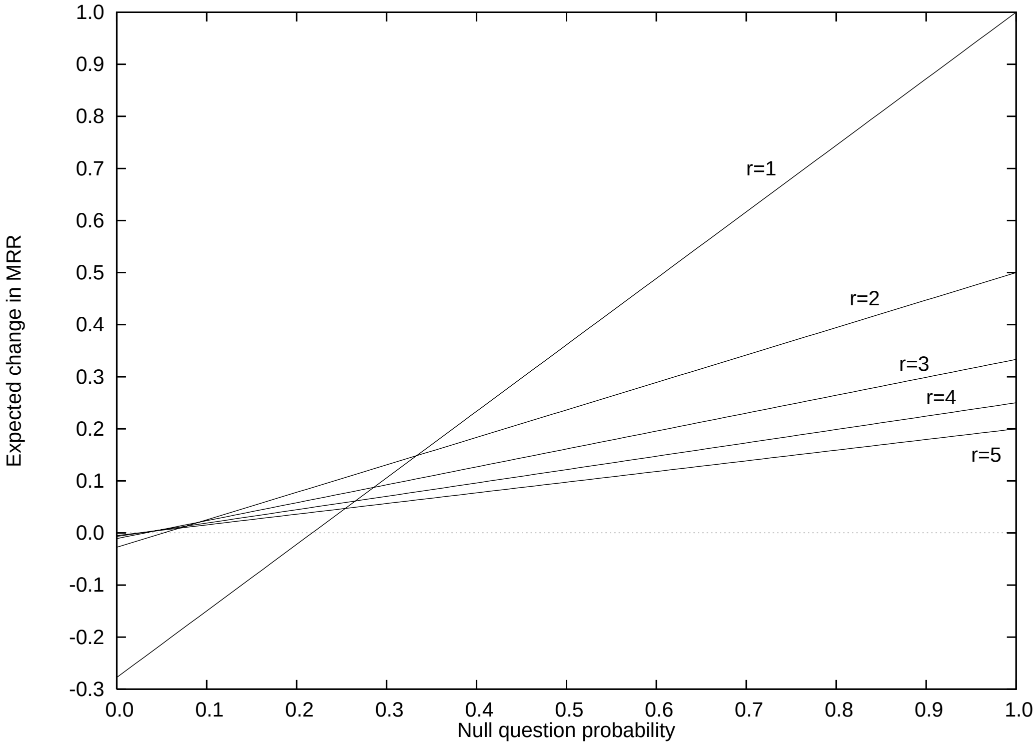

Figure 2: Effect of placing a null response at various ranks (r).

from the actual performance of our TREC 2001 system on the TREC-9 questions as judged by the NIST-supplied judging script. The values of the estimates used for TREC 2001 submissions are:

p_1	p_2	p_3	p_4	p_5
0.500	0.100	0.050	0.033	0.025

In a similar fashion we used a fixed estimate $p_0(Q) = p_0$ for the probability that a question has no answer. Fixing the value of the $p_i(Q)$ for all questions fixes the values for estimated MRR $E_i(Q) = E_i$ $(0 \leq i \leq 5)$ and implies that a null response should always be placed at fixed rank r or always omitted. The precise action taken depends on the values of the estimates.

Deriving a estimate for p_0 that was anything but a guess proved to be impossible. The value of p_0 represents the ratio of questions in the test set that have no known answers in the target corpus. The selection of this value was entirely the choice of NIST, and was not released to participants in advance.

Since a meaningful estimate of p_0 could not be obtained, we treated the value as a free parameter and examined the impact of its value on the expected change in MRR $(E_r - E_0)$ for the possible values of r. The results are shown in figure 2.

After some discussion between members of the group, we agreed that the proportion of questions with no known answer was unlikely to fall below 10% and unlikely to be greater than 20%; our best guess was 10%. A small minority felt that the value would be very small (1–2%). For p_0 in the range $(0.10, 0.20)$ values of $r = 2$ and $r = 3$ both produce small positive improvements to MRR. In the end, a value of $r = 3$ was selected to minimize the consequences of an extremely small p_0.

Run	MultiText Judgment	NIST Judgment (strict/lenient)
MultiText baseline (`uwmta2`)	0.379	0.346/0.365
+ Web re-inforcement (`uwmta1`)	0.483	0.434/0.457
+ feedback (`uwmta0`)	0.482	0.404/0.450
Sun baseline (`mtsuna1`)	N/A	0.307/0.322
+ Web re-inforcement (`mtsuna0`)	0.416	0.405/0.418
Web data only	0.608	N/A
TREC-9 method	0.317	N/A

Figure 3: QA main task results

null response rank	1	2	3	4	5
adjusted MRR	0.311	0.440	0.434	0.426	0.424

Figure 4: Effect of relocating the null response

4 Experimental Results

Our main task question answering results are presented in figure 3. The figure includes both the results of our own submissions and the results of the Sun submissions, which used our system backend, including its passage retrieval and answer selection components. The third column lists official NIST judgments or results derived from them. The second column lists unofficial judgments made by one of the authors (Clarke) immediately after the runs were submitted to NIST. Although creation of these unofficial judgments required less than two hours of total effort, their relative values appear to correlate well with the official judgments, with slightly higher absolute values. In this discussion below, we use these unofficial numbers to support comments that cannot be supported by the official numbers.

The use of Web reinforcement produced a 25% improvement on our own results (`uwmta2` vs. `uwmta1`) and a 30% improvement on the Sun results (`mtsuna1` vs. `mtsuna0`). Considering that the Web data can influence the answer selection process only through modifications to the candidate redundancy parameter c_t, the magnitude of the improvement is surprising and provides substantial support for our view that candidate redundancy is a key factor in question answering [2]. To provide an additional comparison, the top five answers were selected from the Web data used to reinforce the MultiText runs and were judged by Clarke ("Web data only").

For the TREC 2001 questions, p_0 was 10%. As a result, the decision to always place a null response at rank 3 had a small but positive impact. In reality, our performance estimates for our system (p_i, $1 \leq i \leq 5$) were somewhat optimistic. Nonetheless, rank 3 proved to be a good choice. Figure 4 shows the change to the strict MRR for our best run (`uwmta10`) if other ranks were chosen for the null response. Rank 2 would have been a slightly better location, but the potential improvement is less than 2%. If the null response had been omitted, we estimate that the MRR for `uwmta10` would have been 0.421. Thus, our choice to always place a null response at rank 3 gave a performance improvement of roughly 3%.

As a final experiment, we executed our TREC-9 system on this year's questions. Based on the judgments made by Clarke, the total effect of our efforts this year was an overall performance improvement of more than 50%.

5 Conclusion and Future Work

We are continuing to enhance and extend our question answering system. The performance of all aspects of the system is currently under review and many of the components will be heavily modified or replaced over the coming year.

If the approach taken to null questions in TREC 2001 is continued in future TREC conferences we plan to improve our technique by taking question-specific information into account. For example, we intend to consider question category when estimating values for p_i $(1 \leq i \leq 5)$. Also we hope that NIST will release a prior probability p_0 that a question will have no answer, since this information is critical for placing null responses and in practice could be readily estimated from query logs.

Finally, we are actively experimenting with Web-based question answering, both as a method of reinforcing question answering from closed collections and as an end in itself. We are presently in the process of creating a $> 1TB$ collection of Web explicitly to support question answering, replacing the commercial search engines used in our TREC 2001 experiments

References

[1] C. L. A. Clarke, G. V. Cormack, D. I. E. Kisman, and T. R. Lynam. Question answering by passage selection. In *9th Text REtrieval Conference*, Gaithersburg, MD, 2000.

[2] Charles L. A. Clarke, Gordon V. Cormack, and Thomas R. Lynam. Exploiting redundancy in question answering. In *24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 358–365, New Orleans, September 2001.