Statistical Selection of Exact Answers (MultiText Experiments for TREC 2002)

C. L. A. Clarke G. V. Cormack
G. Kemkes M. Laszlo T. R. Lynam E. L. Terra P. L. Tilker

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

1 Introduction

For TREC 2002, the MultiText Group concentrated on the QA track. We also submitted runs for the Web track.

Building on the work of previous years, our TREC 2002 QA system takes a statistical approach to answer selection, supported by a lightweight parser that performs question categorization and query generation. Answer candidates are extracted from passages retrieved by an algorithm that identifies short text fragments containing weighted combinations of query terms. If the parser is able to assign one of a predetermined set of question categories to a question, the system employs a finite-state pattern recognizer to extract answer candidates. Otherwise, one- to five-word n-grams from the passages are used. Our system assumes that an answer to every question appears in the TREC corpus, and it produces a NIL result only in a few rare circumstances. Despite the simplicity of the approach, our best QA run returned correct answers to 37% of the questions.

Our basic question answering strategy is an extension of the technique we used for both TREC 2000 and 2001 [5]. In past years, our system ranked individual terms appearing in retrieved passages and selected 50-byte responses from the passages that included one or more of the highest ranking terms. Since exact answers are required for TREC 2002, much of our effort this year was focused on the extension of this technique to multi-term exact-answer candidates.

Last year, a novel feature of our QA system was the use of commercial Web search services to reinforce answer candidates. This year we reduced our dependence on these commercial services by generating our own collections of structured and unstructured data for use in question answering. The structured data collection consists largely of tables containing answers to questions of frequently occurring types, such as the names of capital cities, the names of world leaders, and the names of baby animals. The unstructured data collection consists of a terabyte of Web data gathered from the general Web in mid-2001. For comparison, half of our submitted runs use a commercial Web search service (Altavista) in addition to our own collection.

As a supplement to our basic question answering strategy, we developed an "early answering" strategy using our structured collection. Under this strategy, if a question can be answered directly from the structured data, the problem is reduced to one of answer justification, in which our system attempts to locate a document in the TREC corpus where question and answer keywords appear in close proximity. If no acceptable justification can be found, or if the question cannot be answered with the structured collection, our basic question answering strategy is invoked. This combination of two strategies — a strategy that searches a federated collection of structured data with a statistical strategy that searches a large collection of unstructured text – is essentially the approach to question answering advocated by Lin [9]. Our experimental runs examine the impact of our early answering strategy. Half our submitted runs use the strategy and half do not.

In the next section we provide an overview of each of the main components of our QA system, with a particular focus on the components which are new to our system for TREC 2002. Section 3 gives the results of our QA track experiments. For the Web track we submitted runs that took advantage of anchor text and link information in addition to document content. In section 4 we describe the approach used for our Web track experiments and discuss the results.

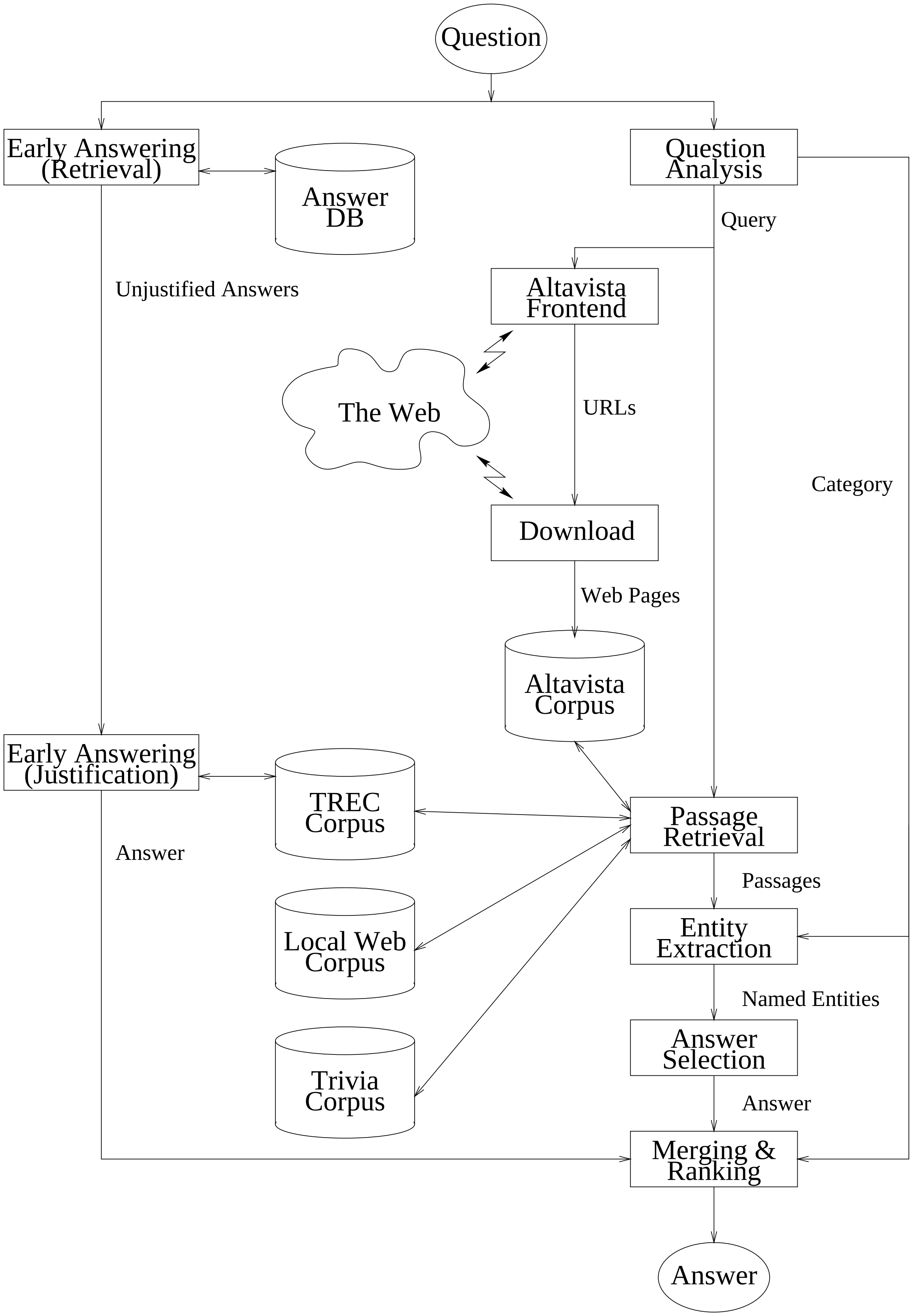

Figure 1: QA System overview.

2 The MultiText QA System

Figure 1 provides an overview of our QA system, showing the processing steps from question to answer. Not all the paths through this diagram are used in all runs. Half of our runs omit the Altavista path through the center of the diagram, and half of our runs omit the early answering path down the left.

2.1 Question Analysis

The question-analysis component takes a natural language question as input and yields a set of terms amenable for input to the passage retrieval system. Question analysis also yields answer categories used by the entity extraction component.

To achieve its objectives, the question analyzer looks for textual elements characteristic of some question form. To this end, we use a context-free grammar generating the forms of interest, and find the most likely derivation of a question using a probabilistic version of Earley's algorithm. The context-free grammar has been augmented to form an attribute grammar and the attributes are evaluated based on the most probable derivation.

Query generation has changed little from TREC 2001 [3] and is based primarily on part-of-speech and quoted-string attributes. Queries are term vectors. Our passage retrieval system provides a rich language for term expressions that includes exact matching of words and phrases, matching under stemming, matching of term disjunctions, and other less-standard operators. For query generation, part of speech is used to determine the role of each word as a query term. Nouns, adjectives, and adverbs are used directly as query terms. Verbs are stemmed if they are regular, and expanded if they are irregular. Articles and prepositions are discarded unless otherwise specified in the grammar rules. Quoted strings are used directly as query terms, as are the individual words contained in quoted strings.

Along with a query, the attribute evaluator outputs a set of global attributes and binary relationships among words in the question. This information was processed further using Prolog. As an example, the parser output for TREC 2001 question 1383 is shown in figure 2.

Global attributes are expressed as the unary relations `qno` for question number, `inst` for instance-of and so on. The binary relationships are expressed as 4-tuples:

`e(`*qno*`,` *source*`,` *sink*`,` *rel*`)`,

where *qno* is the question number, *source* and *sink* are words in the question, and *rel* is the relationship between *source* and *sink*. For example,

```
e(1382,"shepard","make","subj")
```

indicates that "`shepard`" is the subject of "`make`", and

```
e(1383,"flight","make","obj")
```

indicates that "`flight`" is the object of "`make`". More importantly, we see that "`spacecraft`" is the object of an adverbial phrase modifying "`make`", and that "`what`" (the question word) modifies "`spacecraft`". Our Prolog analyzer is able to deduce from these relations that the answer being sought is the spacecraft involved in Shepard's flight.

We use WordNet to determine if a word named in the instance-of attribute is a person, place, thing, etc. and use this information in assigning the question a category. We use word relationships as further evidence in determining the category (e.g. authors write books, inventors invent inventions, and so on). Our original intent was to use the analysis to generate better queries and to aid in recognizing answer contexts. Time permitted us to use the analysis only for categorization.

2.2 Passage Retrieval

We have developed a passage-retrieval algorithm for question answering that can identify small text fragments that cover as many question terms as possible. This retrieval algorithm has been applied in all our TREC question-answering experiments to date. A detailed description of the algorithm may be found elsewhere [3, 5].

Unlike most other passage-retrieval algorithms, ours does not retrieve predefined document elements, but can retrieve any document substring within a corpus. Usually, these fragments are considerably smaller than the documents that contain them. The score of a fragment depends on its length, the number of question terms it contains and the relative weight assigned to each of these terms.

Fragments are often only a few words in length, and may cover only the question terms. To provide the context necessary for entity extraction and answer selection, each fragment is expanded by n words on each side, and this larger passage is retrieved from the corpus. The location of the original fragment within each larger passage is marked as a "hotspot". The answer-selection algorithm takes into account the location of answer candidates relative to this hotspot.

```
qno(1383).
inp(1383,"In in what spacecraft did U u S s astronaut Alan alan Shepard
         shepard make his historic 1961 flight").
cat(1383,"what").
cat1(1383,"what").
e(1383,"what","spacecraft","adj").
e(1383,"in","spacecraft","prep").
e(1383,"alan","shepard","adj").
e(1383,"astronaut","shepard","adj").
e(1383,"us","shepard","adj").
e(1383,"1961","flight","adj").
e(1383,"historic","flight","adj").
e(1383,"his","flight","adj").
e(1383,"flight","make","obj").
e(1383,"spacecraft","make","adv").
e(1383,"shepard","make","subj").
```

Figure 2: Parser output for TREC 2001 question 1383.

Passages are retrieved from at least three — and in some cases four — corpora. The first of these is the TREC corpus itself. The second corpus is our own local terabyte Web corpus. The third is a small 27MB corpus containing 330,000 trivia questions and answers, with each question/answer pair indexed as a separate document and treated as unstructured text by the passage-retrieval component.

Half of our runs use a fourth corpus generated by querying the Altavista search engine. In these runs, we download the top 200 documents returned by Altavista to form a small question-specific corpus. Since its contents are biased by the query, term statistics from the TREC corpus are used during passage retrieval from this small corpus.

The top 20 passages are retrieved from the TREC corpus, the top 40 from the local Web corpus and up to 10 from the trivia corpus. In runs using an Altavista corpus, the top 40 passages are retrieved. The system merges the retrieved passages and passes them to the entity-extraction component.

2.3 Entity Extraction

An entity-extraction component, with responsibility for identifying answer candidates, is a major addition to our QA system for TREC 2002. Our overall approach to entity extraction and answer selection is similar to the "n-gram mining" technique described by Dumais et al. [8]. An answer candidate is always a word n-gram appearing in a retrieved passage. If the question-analysis component assigns a category to a question, simple pattern matching is used to extract n-grams corresponding to the category. But when a category cannot be assigned to a question, n-grams of one to five words within a fixed window of the hotspot become the answer candidates. The entity extractor records the passage and document in which each candidate appears, and its location relative to the hotspot. All this information is passed to the answer selection component.

The primary purpose of the entity extractor is to eliminate unacceptable or unlikely answer candidates from the set of all n-grams. Thus, we prefer to retain questionable candidates and rely on the answer-selection component to give a low score to spurious candidates. As an example, for the PERSON category the entity extractor will accept any capitalized word surrounded by uncapitalized words, provided that it is not a stopword or question term and provided it appears capitalized in the corpus more than 50% of the time.

Pattern matching is achieved with a tool that allows the results of finite-state matching to be merged, filtered and cascaded, similar to the tool described by Abney [1]. The tool incorporates an algebra for structured text search [4], which, along with other capabilities, allows the context of a match to be considered. Finite-state automata may be specified by regular expressions or by lists of terms, such as the names of countries or states. In addition to finite-state automata, hand-written matching code may be called from the matching tool as needed.

A total of 48 categories are matched; a full list is given in figure 3. Most of the categories are self-explanatory, and many are standard in

AGE	AIRPORT	ANNIVERSARY	AREA
BIRTHSTONE	CODE	COLOUR	CONSTELLATION
CONTINENT	CONVERSION=unit0,unit1	COUNTRY	CURRENCY
DATE	ELEMENT	LAKE	LARGE
LENGTH	LONG	MASS	MEASURE
MONEY	MONTH	MOON	MOUNTAIN
NATURAL	NUMBER	OCEAN	PERSON
PHONE	PLACE	PLANET	PROPER
PROVINCE	QUANTITY=unit	RATE	RIVER
SEASON	SPEED	STATE	TEMPERATURE
THING	TIME	URL	VOLUME
WEEKDAY	YEAR	ZIPCODE	ZODIAC

Figure 3: Question categories.

question answering systems (e.g. DATE, CITY, PERSON, TEMPERATURE). A few (AIRPORT, BIRTHSTONE, SEASON) are inspired by previous TREC evaluations. Two categories, CONVERSION and QUANTITY are parameterized by units. After matching, we normalize candidates corresponding to time, measurement, and numeric categories to standard formats to assist the answer selection process.

Matching of the generic proper name categories (PROPER, PERSON, PLACE, THING) is a two-stage process that depends on corpus statistics. In the first stage, we identify lexically acceptable candidates using a longest-match approach. For example, the first stage would identify only the string "`U.S. President Bill Clinton Thursday`" as a candidate for the PERSON category in the passage:

```
...U.S. President Bill Clinton
Thursday proposed a five-year,
over 6 billion U.S. dollar package
to raise the ante on his nearly
fulfilled pledge to put 100,000 new
officers on the beat nationwide...
```

The second phase uses corpus statistics to propose substrings of the first-stage candidates as additional candidates. To generate these corpus statistics, we applied the longest-match patterns for the generic proper name categories to a concatenation of the TREC 2001 and 2002 QA corpora and recorded a count of each matching string. Any substring of a first-stage candidate that appears more frequently than the candidate itself is proposed as an additional candidate. However, single terms are not proposed if they are capitalized less than 50% of the time in the combined corpus. In the example above, the strings "`U.S.`", "`Bill Clinton`", "`U.S. President Bill Clinton`" and "`Thursday`" would be proposed as additional candidates. but not "`S. President`" or "`Clinton Thursday`".

The patterns for PERSON, PLACE and THING match similar sets of strings. The pattern for THING accepts acronyms ("`I.B.M`") and uncapitalized word combinations ("`The Lord of the Rings`") that would not be accepted by the PERSON pattern. The PLACE pattern attempts to extend matches by appending state and country names ("`Waterloo, Ontario, Canada`"). The PROPER pattern is a union of the PERSON, PLACE and THING patterns.

When the question analyzer cannot assign a category to a question, the entity extractor generates a set of all 1- to 5-grams within within 30 words of the hotspot. From this set, the entity extractor eliminates n-grams that only appear in a single passage, n-grams that begin or end with prepositions, and n-grams that consist primarily of stopwords and question terms. The remaining n-grams are treated as answer candidates and are passed to the answer-selection component.

2.4 Answer Selection

From the entity extractor, the answer-selection component receives a set of n-grams, a list of the passage and document identifiers where each n-gram is found, and the location of each n-gram within these passages. Along with corpus statistics, this information is used to rank the n-grams. The highest ranking n-gram is returned as the exact answer.

Candidate *redundancy*, the number of distinct passages in which an candidate occurs, has been an important factor in our QA system since TREC-9 [5]. To rank n-gram answer candidates, our TREC 2002 ranking formula combines redundancy with an idf-

table	# elements
Biographies	25,000
Trivia Question and Answers	330,000
Airports(code, names, location)	1,500
Country Locations	800
Country Capitals and Populations	300
Currency by Country	235
Landmark Locations	2,000
Rulers (location,period,title)	25,000
Acronyms	112,000
University and College (name, location)	5,000
Major World Cites (name, location)	21,000
State and Province (name, population, date, capital, bird, flower)	63
Holidays	171
Previous TREC questions and answers	1393
Animal Name (baby, male, female, group)	500

Figure 4: Structured data for early answering.

like weight and information about the location of the candidate relative to the hotspot in passages where it occurs.

The distance of a candidate from a passage hotspot is measured in token positions, with candidates occurring in the hotspot itself treated specially. Given $\mathcal{P}$, an ordered set of m passages, we use the notation $\mathcal{P}_i$ $(1 \leq i \leq m)$ to refer to the ith passage in $\mathcal{P}$. Each passage is split into tokens, where tokens are sequences of alphanumeric characters separated by non-alphanumeric characters.

For each passage P_i containing one or more occurrences of a candidate x $(x \in P_i)$ we determine $loc(P_i, x)$, the distance from the hotspot to the closest occurrence of x If x is contained entirely in the hotspot then $loc(P_i, x) = 0$. Otherwise, $loc(P_i, x) > 0$. For multi-token candidates, the distance is measured from the closest token in the hotspot to the furthest token in the candidate. Thus for candidates occurring before the hotspot, the distance is measured from the start of the candidate to the start of the hotspot, and for candidates occurring after the hotspot the distance is measure from the end of the hotspot to the end of the candidate.

Candidates are then scored using the following formula:

$$\sum_{1 \leq i \leq m,\ x \in P_i} \log \left(\frac{N}{f_x \cdot (loc(P_i, x) + 1)} \right) \qquad (1)$$

where N is sum of the lengths of all documents in the corpus and f_x is the number of occurrences of the candidate in the database.

2.5 Early Answering

The "early answering" subsystem answers questions by referencing a database of structured knowledge gathered from the Web. The subsystem comprises two components: a retrieval component that determines when questions can by answered from the structured knowledge and proposes possible answers, and a justification component that uses IR techniques to identify documents that support the proposed answers.

We gathered Web pages from standard sources like the CIA Factbook and from other sources identified by searching the Web. The Web pages were parsed into tables and manually filtered to remove irregular information. We supplemented the data gathered from the Web with a table of acronyms extracted from the TREC 2001 and 2002 QA corpora and a table of past TREC questions and answers. Figure 4 gives a summary of the tables in the database.

One table in the database contains a collection of trivia questions with their answers. This is the same collection accessed by the passage retrieval component as unstructured text. Here, the trivia collection is treated as structured data and an exact match with the normalized text of a trivia question is required for the associated answer to be proposed. The text of a question is normalized by converting to lower case, removing punctuation and a few stopwords, sorting the words, and eliminating duplicate words.

The early-answering subsystem is geared to answer specific question types. Regular expressions are used to match question forms corresponding to these types

0) *results from the early-answering subsystem*
1) CONTINENT, CURRENCY, LAKE, OCEAN, PLANET, PROVINCE
2) COLOUR, COUNTRY, SEASON, STATE, YEAR
3) ANNIVERSARY, DATE, LENGTH, MOUNTAIN, PERSON, PLACE, PROPER, THING
4) LONG, TIME
5) CODE, LARGE, SPEED
6) NUMBER, RATE, TEMPERATURE
7) MONEY
8) *all other categories*
9) *uncategorized questions*
10) *unanswered questions ("*`NIL`*")*

Figure 5: Confidence ranking by category.

(ie. "What is the capital of X?"). Once the question type is identified, the answer is retrieved directly from the corresponding table. When possible, we order the tables so that the first occurrence is the most likely answer. When the time-frame of a question is important, the question is answered in the time-frame of the corpus rather than the time-frame of the gathered data. For example, the question, "Who is the president?" should be answered with "Bill Clinton" rather than "George W. Bush."

To meet the requirements of the QA track, our system not only must return an exact answer, but also must identify a document in the TREC corpus that supports this answer. Given a question and a proposed answer, the justification component searches the TREC corpus for a document containing the answer and keywords from the question in close proximity. If no supporting document can be found, the proposed answer is rejected.

2.6 Merging & Ranking

The final component of our system merges the output of the statistical answer-selection component with the answers generated by the early-answering subsystem and applies a confidence ranking to the result. The decisions made by this merging-and-ranking component are based on our experience with 1393 training questions drawn from the QA tracks for TREC 1999-2001.

Whenever the early-answering subsystem produces an answer for a question in this training set, we judged it to be correct (but not necessarily justified) 96% of the time. Since this performance is superior to that of statistical answer selection for all question categories, answers generated by the early-answering subsystem are always given precedence and are ranked first in the system output. Unfortunately, over the training set, the early-answering subsystem produces an answer for only 12% of the questions.

The ranking of the answers produced by statistical answer selection is entirely based on the question category, with the answers for uncategorized questions ranked lower than categorized questions. The ranking order by category is shown in figure 5. Each line of the figure gives an equivalence class for confidence ranking purposes.

In the rare cases where neither early answering nor statistical answer selection produces an answer, a no-answer ("`NIL`") result is generated and ranked last. Statistical answer selection fails to produce an answer only when the entity extraction component fails to identify any candidates from the TREC QA corpus. We took this failure as an indication that an answer may not be present in the corpus. This situation is only one in which a "`NIL`" result is produced by our system.

3 QA Track Results

Our QA track results are summarized in figure 6. The figure reports results for two judgment sets: the official NIST judgments and our own judgments, which were made immediately after the runs were submitted in August. The NIST judgments for `uwmtB0` were generated from the answer list provided by NIST, since this run was not officially judged. All the answers we marked correct in this run are exact and supported according to the NIST answer list, but it is possible that some correct answers from this run do not appear in the NIST answer list. In judging our own runs, we were far more forgiving than the NIST judges. Nonetheless, the relative differences between runs under each judgment set is remarkably consistent.

The use of Altavista as a supplement to our own QA resources had an impact of less than 4% on both

	run tag	uwmtB3	uwmtB2	uwmtB1	uwmtB0
	uses early answering?	y	y	n	n
	uses Altavista?	y	n	y	n
NIST judgments	percent correct	36.8%	36.6.X%	33.4%	32.4%
	confidence-weighted score	0.512	0.511	0.441	0.429
MultiText judgments	percent correct	44.4%	44.0%	39.4%	38.6%
	confidence-weighted score	0.614	0.610	0.509	0.493

Figure 6: QA track results.

the number of correctly answer questions and the confidence-weighted score. The use of early answering provided a 10-14% improvement in the number of correctly answered questions and a 16-24% improvement in the confidence-weighted score.

When it was used in a run, the early-answering subsystem generated answers for 65 questions. Of these, 44 (67.7%) were correct. In `uwmtB2`, where early answering was not used, only 27 of these same 65 questions (41.5%) were answered correctly.

In `uwmtB3`, our best run, 126 questions could neither be answered by the exact-answering subsystem nor assigned a category from figure 3 by the question analyzer. For these questions we took a purely statistical approach, using n-grams of 1- to 5-words in length as the answer candidates. Of these 126 questions, 27 (21.4%) were answered correctly.

As an additional experiment, we disabled the question categorization and exact answering, and applied purely statistical answer selection to the entire question set. Of the 500 questions, 73 (14.6%) were correct.

4 Web Track Experiments

Our Web track runs were the result of an intensive 48-hour effort intended to resurrect some the older MultiText work in this area and to provide a preliminary evaluation of a few new ideas.

We submitted five runs. One run (`uwmtBW0`) used only document content for retrieval, one (`uwmtBW1`) used pagerank in addition to document content, one (`uwmtBW2`) used anchor text along with document content, one (`uwmtBW3`) used a combination of document content, pagerank and anchor text, and one (`uwmtBW4`) used anchor text only.

For content retrieval, we used the current implementation of our *cover-density ranking* algorithm, which is essentially the same technique used in our MultiText TREC-8 Web track runs [6] and is related to the passage-retrieval algorithm used in our QA track runs. The cover-density ranking algorithm locates hotspots that contain combinations of query terms, and bases a document's score on the number and score of the hotspots it contains.

For anchor text retrieval, we applied the cover-density ranking algorithm in a novel way. Anchor points were indexed and hotspots were required to contain an anchor point in addition to the query terms. The score associated with each hotspot was then applied to the score of the document referenced by the anchor, rather than the document where the anchor appears. This approach is similar to other anchor text retrieval techniques [7], but takes advantage of the text surrounding the anchor, as well as the anchor text itself.

Our version of pagerank is a direct implementation of the original pagerank algorithm described by Brin and Page [2]. To apply pagerank to the Web track task, we re-ranked the top 1000 documents retrieved by document content or anchor text according to their pagerank values.

To combine the results of pagerank and/or anchor text retrieval with document content retrieval we simply added the rank of each document in each run and re-sorted them in ascending order. For the purposes of combining runs, a document not appearing in a particular run is treated as if it has a rank one greater than the number of documents retrieved by the run.

Our Web track results are summarized in figure 7. In comparison with content-only retrieval, the combination of anchor text and content retrieval provided a 5% improvement in average reciprocal rank and a 7% improvement in precision at 10 documents. Alone, anchor text retrieval exhibited very poor performance. The inclusion of pagerank had a devastating impact on performance. Since our implementation of pagerank has been throughly tested in other projects, we suspect the problem lies in the approach used to incorporate pagerank into our runs.

run tag	uwmtBW0	uwmtBW1	uwmtBW2	uwmtBW3	uwmtBW4
uses document content?	y	y	y	y	n
uses pagerank?	n	y	n	y	y
uses anchor text?	n	n	y	y	n
average reciprocal rank	0.509	0.150	0.535	0.103	0.106
precision at 10	0.747	0.320	0.800	0.260	0.307

Figure 7: Web track results.

5 Conclusions & Future Work

We continue to improve all aspects of our QA system. In future, we intend to expand the number of question categories and improve the recall of the entity extractor on the existing categories. The current version of our early answering system was two-person effort undertaken in the week preceding the question download date; it could be improved with additional answer sources and more a careful answer justification algorithm. In contrast, we expended considerable effort over several months attempting to exploit syntactic information, by parsing retrieved passages and unifying the result with a parse of the question. While we made considerable progress along this path, we ultimately were not able to develop the approach to the point where it made a positive impact on our QA system.

We are encouraged by the results of our Web experiments. In particular, we plan further experiments to explore our approach to using the text surrounding anchors. Our use of pagerank was a failure. In future, we hope to determine how to exploit pagerank more effectively in our system.

References

[1] Steven Abney. Partial parsing via finite-state cascades. *Journal of Natural Language Engineering*, 2(4):337–344, December 1996.

[2] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. In *Seventh International World Wide Web Conference*, pages 107–117, Brisbane, Australia, April 1998.

[3] C. L. A. Clarke, G. V. Cormack, T. R. Lynam, C. M. Li, and G. L. McLearn. Web reinforced question answering. In *2001 Text REtrieval Conference*, Gaithersburg, MD, 2000.

[4] Charles L. A. Clarke, G. V. Cormack, and F. J. Burkowski. An algebra for structured text search and a framework for its implementation. *Computer Journal*, 38(1):43–56, 1995.

[5] 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.

[6] G. V. Cormack, C. L. A. Clarke, C. R. Palmer, and D. I. E. Kisman. Fast automatic passage ranking. In *Eighth Text REtrieval Conference (TREC-8)*, Gaithersburg, Maryland, November 1999.

[7] Nick Craswell, David Hawking, and Stephen Robertson. Effective site finding using link anchor information. In *24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 250–257, New Orleans, September 2001.

[8] Susan Dumais, Michele Banko, Eric Brill, Jimmy Lin, and Andrew Ng. Web question answering: Is more always better? In *25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Tampere, Finland, August 2002.

[9] Jimmy Lin. The Web as a resource for question answering: Perspectives and challenges. In *3rd International Conference on Language Resources and Evaluation*, May 2002.

Acknowledgments

Our thanks to Ed Fox and Virginia Tech for providing us with the computing resources and network bandwidth to generate the TB Web crawl used in our QA track runs.