The RMIT/CSIRO Ad Hoc, Q&A, Web, Interactive, and Speech Experiments at TREC 8

Michael Fuller* Marcin Kaszkiel* Sam Kimberley* Corinna Ng*‡ Ross Wilkinson† Mingfang Wu*†
Justin Zobel*

1 Ad-hoc Task

1.1 Background

The focus of our work in TREC 8 has again been on the retrieval of documents using arbitrary passages. This year the system has been refined to include variable sized passages and pivot normalisation. Passage based automatic relevance feedback has also been explored, albeit without the use of negative feedback.

1.2 Method

As in previous years, an in-house version of the MG retrieval system was used for all experiments. For document ranking, documents and queries were matched using the Okapi similarity measure [13]:

$$sim(q,d) = \sum_{t \in q \wedge d} w_{d,t} \cdot w_{q,t} \tag{1}$$

where

$$w_{d,t} = \frac{(k_1+1)\cdot f_{d,t}}{k_1 \cdot [(1-b) + b \cdot \frac{W_d}{avr\_W_d}] + f_{d,t}}$$

$$w_{q,t} = \frac{(k_3+1)\cdot f_{q,t}}{k_3 + f_{q,t}} \cdot \log \frac{N - f_t + 0.5}{f_t + 0.5}$$

The constants k_1, k_3 and b were set to 1.2, 1000 and 0.75 respectively, as recommended by the City University group [13]. W_d is the length of the document d in bytes and avr_W_d is the average document length in the entire collection. N is the total number of documents in the collection, f_t is the number of documents in which term t occurs, and $f_{x,t}$ is the frequency of term t in either a document d, passage p or query q.

For passage ranking, queries and passages were matched using a non-normalised version of the cosine similarity function:

$$sim(q,p) = \sum_{t \in q \wedge p} w_{q,t} \cdot w_{p,t} \tag{2}$$

where the weights were as follows:

$$w_{q,t} = (\log(f_{q,t}) + 1) \cdot \log(\frac{N}{f_t} + 1)$$

$$w_{p,t} = \log(f_{p,t}) + 1$$

Multiple passage sizes were used for *mds08a1*, *mds08a2*, and *mds08a3*, necessitating the lengthwise normalisation of their passage scores before they could be compared. The passage scores were normalised using a pivot with a slope of 0.2 [11].

$$sim_n(q,p) = \frac{sim(q,p)}{(1 - slope) + slope \cdot (\frac{W_p}{avr\_W_p})} \tag{3}$$

where W_p is the passage's length and avr_W_p is the average length of all of the passages, both measured in words. The similarity score for each document is the maximum of the normalised similarity scores for its passages.

Automatic relevance feedback is based on the Rocchio formula [9]:

$$Q_{new} = \alpha \cdot Q_{orig} + \frac{\beta}{|R|}\sum_{r \in R} r - \frac{\gamma}{|R'|}\sum_{r' \in R'} r' \tag{4}$$

where Q_{orig} is a weighted term vector for the original query; R is the set of relevant documents; R' is the set of non-relevant documents; and r and r' are weighted term vectors for relevant and non-relevant documents respectively. The parameters α, β and γ determine the effect of the terms from the original query, relevant documents and non-relevant documents. These parameters were set as follows; $\alpha = 1.0$ and $\beta = 2.0$, negative feedback was not used rendering the γ parameter irrelevant.

Both the document and query terms were stopped and stemmed. Single terms were stemmed according to the Lovins algorithm [7], while the stop-list contained 368 terms and is the same as that used in our TREC 7 experiments [6].

1.3 Ad-hoc runs

For each of the submitted runs *mds08a1 – 3*, passage sizes in the range {50, 100, 150, ..., 600} terms were used. For each passage size, all of the documents with a non-zero similarity to the query, as calculated from equation 2, were recorded. These passage scores were then normalised using equation 3 and the maximum normalised passage score for a document was used as its similarity score. The 1000 highest scoring documents were chosen as candidate documents for each query.

The *mds08a4* and *mds08a5* were experimental runs that used query expansion based on passages and passage-based re-ranking. Using a passage size of 100 terms, 20 passages were retrieved and assumed to be relevant, giving set R in

* Department of Computer Science, RMIT,
GPO Box 2476V, Melbourne VIC 3001, Australia
{msf,martin,sam,cln,ming,jz}@mds.rmit.edu.au
† CSIRO, Division of Mathematical and Information Science
723 Swanston St, Carlton VIC 3053, Australia
Ross.Wilkinson@cmis.csiro.au
mingfang.wu@cmis.csiro.au
‡ On leave at Sharp Laboratories of Europe Ltd.

	Precision @10 docs	Precision @20 docs	Precision @100 docs	Average Precision	% Δ	Recall (max 4728)
title						
document	0.4120	0.3520	0.2058	0.2095	0.0	2507
passage-300	0.4180	0.3470	0.2018	0.2154	+2.8	2576
optimal passage (500)	0.4160	0.3480	0.1966	0.2187	+4.4	2577
var. passages (**mds08a1**)	0.4040	0.3490	0.2052	0.2236	+6.7	2594
query expansion (**mds08a5**)	0.3900	0.3470	0.2076	0.2324	+10.9	2804
title + desc						
document	0.4400	0.3750	0.2104	0.2395	0.0	2690
passage-300	0.4160	0.3440	0.1986	0.2308	-3.6	2683
optimal passage (200)	0.4000	0.3350	0.2002	0.2323	-3.1	2702
var. passages (**mds08a2**)	0.4160	0.3590	0.2026	0.2397	+0.1	2732
query expansion (**mds08a4**)	0.4180	0.3680	0.1986	0.2550	+6.5	2843
title + desc + narr						
document	0.4280	0.3720	0.2118	0.2366	0.0	2768
passage-300	0.4000	0.3290	0.1768	0.2191	-7.4	2478
optimal passage (150)	0.4080	0.3430	0.1844	0.2256	-4.6	2578
var. passages (**mds08a3**)	0.4000	0.3570	0.1906	0.2338	-1.2	2640
query expansion	0.3900	0.3490	0.1998	0.2534	+7.1	2809

Table 1: *The effectiveness results for the described runs on the 50 TREC 8 topics.*

equation 4. No documents were assumed to be irrelevant, i.e. set R' was empty. All of the terms that appeared in the text of these passages were stopped and stemmed using the Lovins algorithm. The number of passages that each of these terms appeared in and its total number of occurences in R was calculated. The 50 terms that appeared in the most passages were selected to be included in the expanded query. Where there was contention between terms for selection it was resolved by reference to the total number of times that the terms appeared in the top 20 passages.

The documents that contributed the 20 passages were retrieved and the weights for each of the selected terms were calculated according to:

$$w_{d,t} = \frac{1 + \log(1 + \log(f_{d,t}))}{(1 - slope) + slope \times \frac{W_d}{avr\_W_d}}$$

The original query terms were re-weighted using:

$$w_{q,t} = (\log(f_{q,t}) + 1) \cdot \log(\frac{N+1}{f_t})$$

These weights were combined using the Rocchio formula of equation 4 then normalised. With the query terms re-weighted and the new terms added to the query, the query was re-run using a fixed passage size of 300 terms, and the best 1000 documents selected.

1.4 Results and Analysis

The TREC 8 results are shown in Table 1 with the submitted runs shown in boldface. Only the *mds08a3* and *mds08a4* runs were used for pool judgments.

The *document* run is intended as a base against which the other runs can be compared; it used the Okapi similarity measure. The optimal passage runs are fixed-size-passage runs that generated the highest average precision; they are shown for comparison with the variable passage length runs. The query-expansion run for the title+desc+narr queries used the same automatic relevance feedback as *mds08a4* and *mds08a5*.

	> Median
mds08a1	14
mds08a2	18
mds08a3	13
mds08a4	26
mds08a5	24

Table 2: *The number of the 50 queries that were better than the median average precision for each submitted run.*

As noted in the TREC 7 report, passage retrieval is comparable to document-based retrieval for short queries and slightly less effective for longer queries.

It is interesting to observe that although the document based runs generally have better precision at 10, 20 and even 100 documents, the passage based runs on occasion manage comparable or even superior average precision. An example of this is provided by the title runs *mds08a1* and *document*, of table 1. This effect is caused by the less precise run having superior recall, as is shown in the recall column of table 1. In most situations, where a run retrieves more relevant documents than the document-based run it also has a higher average precision.

In contrast to previous years the use of the topics' narrative component appears to detract from the system's retrieval effectiveness. This is particularly evidenced by the Okapi document based retrieval which shows a 1.2% degradation in effectiveness when the narrative is included, compared with a 17.6% improvement in TREC 7 [6].

Table 2 compares the submitted runs against the submissions of other TREC participants, it shows the number of queries for each run that achieved a higher average precision than the median. None of the runs had the best precision for any of the queries.

The use this year of variable sized passages has been worthwhile, providing an improvement in effectiveness over the use of fixed size passages. The exception to this is for title queries where it has a lower precision at 10 documents than both the passage-300 and optimal passage runs. However, it is again greater at 20 and 100 documents. Another

Query Type	Precision @10 docs	Precision @20 docs	Precision @100 docs	Average Precision	Δ%	Recall (max 206)
title						
document	0.2579	0.1763	0.0589	0.3452	0.0	173
passage-300	0.2684	0.1684	0.0547	0.3356	-2.8	177
var. passages	0.2895	0.1763	0.0579	0.3451	0.0	177
query expansion	0.2684	0.1684	0.0605	0.3549	+2.8	181
title + desc						
document	0.2579	0.1711	0.0600	0.3305	0.0	183
passage-300	0.2368	0.1711	0.0574	0.3167	-4.2	185
var. passages	0.2684	0.1763	0.0589	0.3429	+3.8	184
query expansion	0.2789	0.1763	0.0632	0.3832	+15.9	184
title + desc + narr						
document	0.2684	0.1658	0.0621	0.3273	0.0	183
passage-300	0.2579	0.1684	0.0626	0.3490	+6.6	187
var. passages	0.3105	0.1816	0.0568	0.3810	+16.4	186
query expansion	0.3263	0.2079	0.0684	0.4330	+32.3	205

Table 3: *The results of running the TREC-8 runs on the FR collection alone, using the 19 queries that had relevant answers in the FR collection.*

benefit of this approach is that it is independent of query length; in contrast, different sizes of fixed passages are better suited to queries of different lengths, with the optimum passage size shrinking as query length increases.

The experimental *mds08a4* and *mds08a5* runs did not use variable size passages. These runs are more effective than the fixed size passage runs on which they were based. This is particularly true for longer queries, with the exception of the title+desc+narr run which registered a precision at 10 documents which was worse than both of the fixed size passage runs.

Table 3 shows how the same runs fared on the Federal Register collection (FR) alone. The documents contained in the FR collection are statutory rules and regulations of the US Federal Government, these documents are characterised by an average length that is greater than the other collections that comprise the TREC 8 collection. Only those topics that had a relevant document in the FR collection were used; this meant that there were 19 topics with 206 relevant documents.

Passage-based retrieval seems to be particularly effective for the FR collection. Whilst the effectiveness of passage-based retrieval decreased — relative to document based retrieval — for longer queries over the entire collection, it increased when only the FR collection was used. In fact, passage-based retrieval improves on document-based retrieval for virtually all of the runs. It is particularly interesting that the variable size passage runs have better recall and precision than the document-based runs, this appears to be for a number of reasons.

Firstly, running the 19 FR queries over the total collection, as opposed to the 50 queries that generated the results in table 1, gives results that are similar to the results of table 3 in that passage retrieval outperforms document retrieval. This suggests that either some of the 19 FR queries are particularly susceptible to passage retrieval or that some of the other 31 queries are poor candidates for passage retrieval.

Secondly, given the longer documents in the FR collection it would be expected for the effectiveness of passage-based retrieval to improve relative to document retrieval.

Thirdly, it is conceivable that the limited number of relevance judgments, relevant documents and candidate documents are skewing the results.

2 Question & Answer track

2.1 Introduction

We participated in the 250 byte category of the question and answer track, submitting one run, ***mds08q1***. Our objective in participating in this track was to determine the appropriateness of applying traditional document retrieval techniques to the retrieval and extraction of small, focused text segments.

2.2 Method

It was our goal to determine whether the passages of text that exhibited the closest correspondence to the question in terms of content also contained a correct answer.

Therefore, with this in mind our approach was as follows;

1. Modify the question into an acceptable query for the system.
2. Retrieve the 50 most relevant passages from the document collection using passage ranking.
3. Find the 5 best sentences from the 50 retrieved passages.
4. If a sentence was over 250 bytes in length, use a 250 byte sliding window to find the segment that best matched the query.

Query generation

Queries were generated from the official questions by stopping terms and removing all punctuation. This converted the question to a conventional query that was used as input to the MG retrieval system. Generally, the resulting query was very short, with an average length of 4.8 words in the training queries and 5.3 words in the actual queries.

Extracting the best passage

These queries were used as input to the MG passage retrieval system which used the similarity measure defined in equation 2 to retrieve the most relevant passage from the 50 most relevant documents for each query. The passages were 150 words in length with each new passage commencing at a 25 word interval from its predecessor. Because all of the passages are the same length there was no need for passage length normalisation. The document and query terms were case-folded, and stemmed using the Lovins algorithm.

Finding the best sentences

The passages retrieved by MG were segmented into individual sentences and a similarity relative to the query was calculated for each sentence. The five sentences with the highest similarity scores were then selected as the candidate answers to the question. The query and sentence terms were case-folded and stemmed by the sentence retrieval engine using the Lovins algorithm.

The similarity of each sentence was calculated by:

$$sim(q, s) = \frac{1}{w_s} \sum_{t \in q} w_{s,t} \cdot w_{q,t} \tag{5}$$

where:

$$w_{s,t} = \log(f_{s,t} + 1)$$

$$w_{q,t} = \log(f_{q,t} + 1) \cdot \log(\frac{N}{f_t} + 1)$$

$$w_s = \sqrt{\sum_{t=1}^{n} w_{s,t}^2}$$

In the above equations $f_{x,t}$ is the number of times that the term appears in x, where x is either a sentence or a query, N is the total number of documents in the collection, and f_t is the number of documents in which the term appears.

Reducing a sentence to 250 bytes

Some of the sentences that were returned were in excess of the 250 byte ceiling imposed on acceptable answers. The pruning stage described here only applied to those sentences that exceeded the 250 byte limit.

A sliding window was used to identify 250 byte segments for excessively long candidate sentences. The left edge of the window was placed at the start of the sentence, with the window extending 250 bytes into the sentence. The window was then slid across the sentence in single word increments, so that the left edge of the window was always located at the start of a word. At each position a similarity score for the window was calculated using equation 5.

When the right edge of the window reached the end of the sentence the process was stopped, and the window position that generated the highest similarity score was chosen to represent the sentence. The sliding operation did not observe word boundaries at the window's right edge.

As in the previous stages all of the query and sentence terms were case-folded and stemmed using the Lovins algorithm.

2.3 Results and analysis

This approach retrieved many short sentences with a high proportion of relevant terms but not an appropriate answer. A good example of this phenomenon were headlines, which were frequently retrieved. The role of a headline is to succinctly describe the topic of its accompanying article. It is brief and can contain a large proportion of the terms used to identify a subject. Therefore it is not suprising that a question requesting specific information about an event would retrieve the headline of an article describing the event. Unfortunately, an acceptable answer generally required more detailed information than a headline could provide.

Another concern was that sentences containing repeated occurrences of a single query term were in some cases being retrieved before sentences that referenced multiple distinct query terms. This was a concern because the sentences containing a broader coverage of the query terms were generally the more relevant sentences.

From these observations the similarity measure underwent two modifications. Firstly, the sentence normalisation procedure was modified to reduce the disproportionate number of short sentences that were being returned. Secondly, the similarity measure was changed to reward sentences that provided a good degree of query coverage, through the use of supplementary coordinate matching [16].

Floor on sentence length

The first modification to the similarity measure was to set a floor on the length of sentences for the purpose of calculating similarity. If the calculated weight for a sentence (equal to the sentence length in the case of sentences without repeated terms) was less than a floor x its weight was fixed at $\sqrt{x}$. This required that a short sentence be highly relevant before it was retrieved and also allowed the similarity of sentences to be adjusted according to their length. As shown in table 4, it was an effective mechanism for improving the performance of this task. Experimentation with the use of pivot normalisation found that the thresholding mechanism provided superior sentence length normalisation for this task.

This normalisation technique converted w_s, defined previously in equation 5, to:

$$w_s = \begin{cases} w_{s'}, & \text{if } w_{s'} \geq x^{1/2} \\ x^{1/2}, & \text{otherwise} \end{cases} \tag{6}$$

where:

$$w_{s'} = \sqrt{\sum_{t=1}^{n} w_{s,t}^2}$$

Experimentation with varying floor lengths suggested that 30 words was a reasonable length for the current document collection; see table 4.

Coordinate Matching

The information needs represented by the TREC Q&A questions require the extraction of specific information from the document collection in the form of short, precise answers. Our approach has been to attempt to identify appropriate sentences or sentence fragments. It is our contention that when using such an approach it is preferable to combine the terms in a more conjunctive manner than would be appropriate for a conventional ranked query. It is also our belief that concepts are generally only stated once within a given question, whereas a query such as a TREC topic may contain many restatements of a single concept through the use of synonyms, and so on. These hypotheses suggest that a candidate sentence's likelihood of relevancy increases with the number of distinct query terms that it contains. Therefore, a good candidate answer should provide coverage of most if not all of the query terms and the degree of this coverage should be taken into account when calculating the similarity for a given sentence.

Answer appears in	Sentence floor length, in terms					
	0	10	20	30	40	50
1st sentence	1	6	10	8	9	9
2nd sentence	2	3	2	8	3	2
3rd sentence	2	1	0	1	3	2
4th sentence	4	2	3	1	3	1
5th sentence	1	0	3	1	1	2
not found	28	26	20	19	19	22
Score	0.102	0.219	0.325	0.336	0.328	0.298

Table 4: *System performance using a range of sentence-length minimums on the 38 training questions. These judgments were not made by NIST assesors and should not be compared to the official results.*

Given this consideration, coordinate matching was used to award a sentence an extra point for each distinct query term that it contained. These coverage points are in addition to the sentence's conventional cosine similarity score. This converted the similarity score defined previously in equations 5 and 6, to the form:

$$sim(q,s) = \frac{1}{w_s} \sum_{t \in q} w_{s,t} \cdot w_{q,t} + count(t_{\in q \wedge s}) \tag{7}$$

The effect that this mechanism has on the retrieval effectiveness of the system is shown in table 7. The submitted run, ***mds08q1***, used this similarity measure with a sentence floor of 30 terms. The results of the judged run are shown in table 5, and are compared against the other judged runs in table 6.

Answer appears in	**mds08q1**
1st sentence	71
2nd sentence	22
3rd sentence	11
4th sentence	12
5th sentence	5
Not found	77
Score	0.453

Table 5: *The judged run, mds08q1. It used the similarity measure of equation 7 with a sentence floor of 30 terms.*

	mds08q1
= Best	77
> Median	73
= Median	108
< Median	17
= Worst	77
Average Rank	5.2

Table 6: *A comparison of mds08q1 with the other question and answer runs that were judged at NIST. The average rank metric is relative to the other runs, with the rank for a topic being the position at which the run's score was placed relative to all other runs.*

Subsequent to the submission of our official run, we have explored this idea further. One modification undertaken was to normalise the cosine similarity scores to a range between 0 and 1 by dividing the scores by the query weights. This meant that when the query coverage points were added, the sentences with the most distinct query terms were automatically the highest ranked sentences. The cosine score was used to differentiate between sentences that contained the same number of query terms.

This similarity score is described by:

$$sim(q,s) = \frac{1}{w_s w_q} \sum_{t \in q} w_{s,t} \cdot w_{q,t} + count(t_{\in q \wedge s}) \tag{8}$$

where

$$w_q = \sum_{t=1}^{n} (w_{q,t}^2)^{1/2}$$

As table 7 shows, supplementary coordinate matching is a very effective mechanism for increasing the precision of this task. However it is not without disadvantages.

One undesirable effect occurs when a single logical component of the query is expressed using a phrase consisting of multiple words. This effectively gives that concept a much higher weighting in the query than those that can be more briefly described.

For example, the question "The Faroes are a part of what northern European country?" after stopping becomes, "Faroes part northern European country" This embodies three separate concepts; "Faroes", "part", and "northern European country" each of which are nominally of equal importance.

However, using word based coordinate matching, the concept "northern European country" can attract up to 3 coverage points (one per word) whereas the "Faroes" concept will only attract a maximum of 1 coverage point if it appears. This biases the results towards those sentences that contain multi-word concepts. Essentially, the problem is that query coverage points are awarded for single-word terms when they should be awarded for concepts.

This phenomenon was observed on the training data, with sentences containing the correct answer to the Faroe islands question regularly retrieved before the query coverage points were awarded. However, they were replaced by sentences about northern European countries once the query coverage points were added.

This problem could be minimised through a linguistic preparsing of the query to divide it into logical concepts or common phrases, and to subsequently award query coverage points equitably across identified concepts.

2.4 Conclusion

This study has been an experiment into the effectiveness of the use of statistical IR for solving the question answering problem. It has shown that statistical IR without natural language processing can be used with some effectiveness to locate and retrieve small fragments of text as answers to questions.

Answer appears in:	conventional cosine measure	including coordinate matching	incl. cosine normalisation & coordinate matching
1st sentence	57	74	75
2nd sentence	23	18	22
3rd sentence	13	13	11
4th sentence	11	11	12
5th sentence	7	5	2
Not found	87	77	76
Score	0.389	0.460	0.470
Difference	0.0%	+18.3%	+20.8%

Table 7: *Comparison of the performance of three similarity measures on the 200 Q&A questions. All of the runs shown used a sentence floor of 30 words. These runs have not been officially judged and should not be compared to the official run scores from NIST.*

Simple question answering can be reasonably supported by traditional IR techniques however for more robust solutions to complex questions it is envisaged that natural language processing techniques will be required. Techniques such as coreference resolution, entity detection and question analysis have been demonstrated by groups to be effective in this task.

3 Web track

3.1 Introduction

The MDS group participated in the small web track, submitting three runs; a content-only run, ***mds08w1***, and two content-and-link runs, ***mds08w2*** and ***mds08w3***.

Our objective in participating in this track was twofold. Firstly, to determine whether simple manipulation of linking information would enable effective re-ranking of documents within a result set. Secondly, to examine the effectiveness of content-only retrieval on web data.

3.2 Content only run

This run, ***mds08w1***, was performed using a similar procedure to the runs in the ad hoc task. The in-house version of MG was used to retrieve the most relevant documents using passage similarities that were calculated from the similarity measure defined in equation 2. Passages of 150 words were used, with each passage starting at 25 word intervals. It was not necessary to normalise the passage similarities as all of the passages were the same length.

3.3 Content and link runs

Our retrieval system made use of the sibling relationship between documents, where sibling documents are defined as two or more documents that are linked from the same document. This is based on the hypothesis, that if document P links to document A, then other documents that are directly accessible from document P are likely to contain similar content to document A. Therefore, it should be possible to infer some degree of related content from a sibling relationship between two documents. Of course, this will not always be the case: for example, an individual's home page may have pointers to many different fields of interest that are wholly unrelated. Therefore, an additional constraint was imposed on the sibling documents before they were used to infer related content. This constraint was to only recognise a sibling relationship if both of the siblings were retrieved in the top n documents for a given query. This ensures that the two documents have reasonably similar content which, when combined with their sibling relationship, suggests that they share very similar content. Therefore, if A is relevant and B, A's sibling, has similar content to A then it is likely that B is relevant as well.

Sibling relationships were only identified if the siblings and the parent that links to them were all present in the WT2G collection.

There were two types of content-and-link runs used; a very simple sibling relationship implementation, and another version that aimed to overcome some of the simpler run's shortcomings.

Simple run

The run described in this section was submitted as ***mds08w2***, and was processed as follows.

1. Retrieve the best 1000 documents using the same mechanism as the content-only run, call this set R.
2. For each retrieved document, d:
   (a) Locate the document's siblings.
   (b) For each located sibling:
      i. If the sibling is in R add $1/k$th of its content similarity score to the document d's similarity score.
3. Re-rank the documents.

Therefore, the siblings of a document with a high similarity score receive a greater similarity increase than the siblings of documents with low similarity scores. The k parameter was set to 50, a figure that gave reasonable results in training.

The similarity measure can be expressed as:

$$sim(q,d) = sim_c(q,d) + \frac{1}{k} \sum_{d' \in sib(d) \wedge ret(q)} sim_c(q,d') \qquad (9)$$

where, $sim_c(q,d)$ is d's content similarity score for the query q, $sib(d)$ is the set of d's siblings, and $ret(q)$ is the set of documents retrieved for the query q.

A concern identified from the results of this run was that very well linked documents could have their similarity score boosted enormously. In some cases, the bonus scores derived from linking completely overshadowed the base similarity score of documents with good content. This resulted in documents with poor content that were linked to documents with good content being ranked higher than documents that themselves had good content. This is undesirable because the content similarity score is a considerably more accurate measure of relevance than the linking information score.

In *mds08w3* measures were undertaken to minimise the influence of this and other factors.

Improved Run

The run described in this section was submitted as run ***mds08w3*** and it was constructed as follows.

1. Retrieve the best 2000 documents using the same mechanism as the content-only run, call this set R.
2. Choose the top x documents from R, call this set X.
3. For each retrieved document, d:
   (a) Locate the document's siblings.
   (b) For each located sibling:
      i. if the sibling is in X add $1/k$th of its content similarity score to the document d's similarity score.
   (c) Limit the linking component of document d's score so that it cannot exceed the contribution made by the document's content.
4. Re-rank the documents.
5. Select the top 1000 documents.

Step 1 of the algorithm extracted a larger collection of documents, 2000 as opposed to 1000 in *mds08w2*, to increase both the amount of linking information available and the pool of relevant documents. It was hoped that this would allow the identification of more sibling relationships and also improve the system's depth of recall.

Step 3(c) of the algorithm is used to reduce the accumulated impact of excessive linking on similarity scores and thereby limit linking bonuses to a secondary role. This constrained the amount that linking information could contribute to a document's similarity score to no more than the document's original score derived from its content. Therefore, a document's similarity score could be at most doubled through the existence of relevant siblings.

The other modifications to the *mds08w2* process are steps 2, and 3(b)i. These modifications are aimed at reducing the effect whereby well connected, less relevant documents tend to drive each other up the rankings. If the 950th document has the 1010th, 996th and 988th ranked documents as its siblings then its ranking will probably be significantly improved as a result of these relationships. However, at these low ranks there is only a slight probability that these documents are actually relevant. The problem is that a linking relationship with 3 relatively poor documents should not be equivalent to a relationship with a single good document, if in fact it should be worth anything at all. The motivation behind the use of linking information was to boost documents that were linked to good documents rather than documents that were linked to poor documents. It was originally intended that this problem would be handled by proportionally adjusting the sibling's score relative to the retrieved document's similarity, however this mechanism was not sufficiently restrictive to eliminate the problem described previously. In response to this, a smaller subset of the higher ranked documents were used for sibling linking, rather than all of the retrieved documents. These higher ranked documents are much more likely to be relevant and presumably so are their siblings.

For the purposes of *mds08w3* the subset of retrieved documents that were considered for sibling linking were the 500 most highly ranked documents retrieved by the MG system.

From these modifications the similarity measure became:

$$sim(q,d) = sim_c(q,d) + sim_l(q,d) \tag{10}$$

where,

$$sim_l(q,d) = \begin{cases} sim_{l'}, & \text{if } sim_{l'}(q,d) < sim_c(q,d) \\ sim_c(q,d), & \text{otherwise} \end{cases}$$

$$sim_{l'}(q,d) = \frac{1}{k} \sum_{d' \in sib(d) \wedge ret(x,q)} sim_c(q,d')$$

and $ret(x,q)$ is the set of the top x documents retrieved by the retrieval engine for the query, q.

3.4 Results and Analysis

Unfortunately, both of the submitted content-and-link runs yielded disappointing results compared to the content-only run; see table 8. On average the performance was degraded when the sibling linking information was used. The *mds08w3* run outperformed *mds08w2* as anticipated, however both were inferior to *mds08w1*. Both of *mds08w1* and *mds08w3* were judged, whereas *mds08w2* was submitted but not judged. There were queries for which the content-and-link runs outperformed the content-only run, indicating that those queries fitted the sibling model well.

Run Identification	> Median	Best
Content-only		
mds08w1	39	3
Content-and-link		
mds08w2	34	2
mds08w3	40	12

Table 9: *Comparison of the submitted MDS runs against all of the submitted runs for the 50 queries.*

The content-only result was fairly pleasing given that only a single passage size was used and no attempt was made to extract meta-information from the HTML. Interestingly, from table 9 it can be seen that although the *mds08w3* was less effective than the content-only run, its performance relative to the other submitted runs is an improvement.

Perhaps the run of greatest interest is *max-sibling* which was not submitted to TREC as it was processed post-submission. Instead of summing the similarities of a document's siblings, a proportion of the highest scoring sibling's similarity was added to the document's score; for the run shown in Table 8, this proportion was 1/10th. Also, rather than using the siblings of the top 500 documents, only the top 20 were used. This led to an improvement of 2.0% in average precision over the base content-only run. Whilst far from significant, it was gratifying to improve upon the content-only run after many unsuccessful attempts. The improvement gained from the inclusion of linking informaton in the max-siblings run was similar to the greatest increases observed by any of the participating groups. An improvement was also observed on the

Run Identification	Precision @5 docs	Precision @10 docs	Precision @20 docs	Precision @100 docs	Average Precision	% Δ
Content-only						
mds08w1	0.4480	0.4480	0.3860	0.1994	0.3220	0.0
Content-and-link						
mds08w2	0.4000	0.3860	0.3330	0.1894	0.2878	-10.6%
mds08w3	0.4440	0.4120	0.3590	0.2010	0.3047	-5.4%
max-sibling	0.4680	0.4360	0.3830	0.2030	0.3284	+2.0%

Table 8: *The results for the small webtrack runs using the 50 TREC 8 topics, each webtrack topic used the title and description components of the query.*

training data, suggesting that this approach is a preferable mechanism for the combination of this type of evidence.

4 Interactive Retrieval Track

4.1 Introduction

In TREC7, we tested using clustering technology to organize retrieved documents for aspectual retrieval, but did not find a significant gain for the clustering interface over a ranked list interface. This year, we investigated a question-driven categorization. Unlike the clustering approach, which was data-driven and attempted to discover and present topic relationships that existed in a set of retrieved documents without taking users into account, the question-driven approach tries to organize retrieved documents in a way that is close to the users' mental representation of the expected answer. In our approach, the retrieved documents are categorized dynamically into a set of categories derived from the user's question. The user determines which of several possible sets of categories should be used to organize retrieved documents.

Our participation in TREC-8 was to investigate and compare the effectiveness and usability of this question-driven classification with a ranked list model. In the following sections we present a rationale for the question-driven approach, and then describe an experiment that compares this approach with a more traditional ranked list presentation. We then report and analyze the results of this experiment. Based on these findings and discussions, we conclude with some recommendations for future improvement.

4.2 A Question-driven Approach

The nature of information needs is variable. A user may need to find specific facts, to learn about a topic, to gather a variety of information, or may simply wish to explore an information set without having a well defined-goal [1, 15]. It is difficult for an information access tool to satisfy all types of information needs with equal effectiveness. Our question driven approach mainly focuses on information needs that involve seeking specific facts about a topic. In particular, we are considering aspect queries: topics whose answer consists of more than one related piece of information. For example, the question "what non-surgical alternatives exist for treating heart disease?" might have diet, exercise, meditation, and drug programs as different aspects of the answer.

When a user seeks specific facts, they are usually able to describe the type of information they are after. Consider a user who wants to know "what countries had ferry sinking that caused 100 or more people to lose their lives?"; the answer sought consists of the names of those countries that meet the stated criteria. Thus, this user will be actively focused on

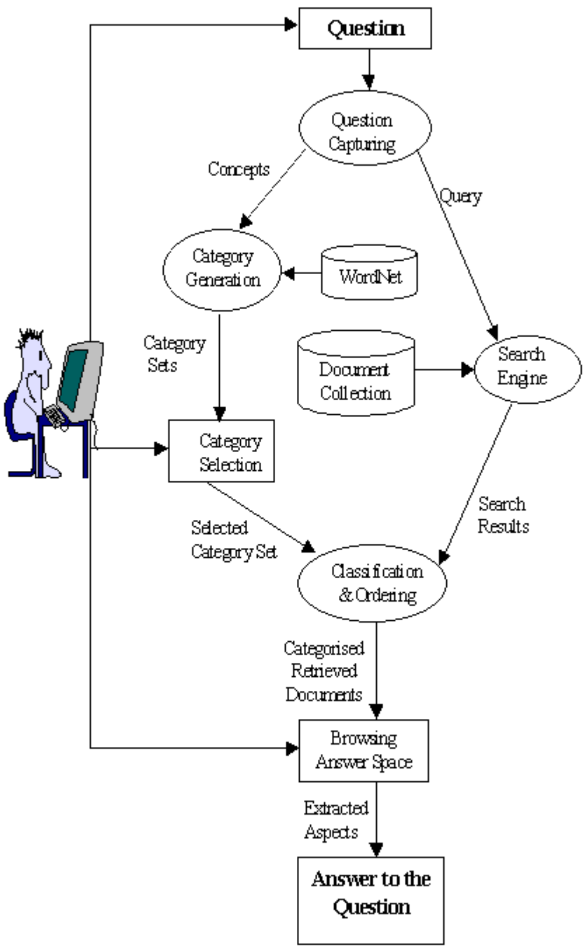

Figure 1: A Question-driven approach

searching for names of countries from retrieved documents. If the retrieved documents can be categorized by country name, this task is simplified, allowing the set of retrieved documents to be more easily analyzed.

The semantic relationship between the categories (in our example, the names of countries) and the query term from which they are derived (in our example, "country") is that of hyponym to hypernym [5]. For instance, Australia is a hyponym of country, and country is a hypernym of Australia.

An architecture for a system that categorizes retrieved document using question-driven classification is shown in Figure 1. This system contains three significant new stages: a category generation stage, a category selection stage, and a document classification and ordering stage.

Category generation. The category generator extracts keywords from each query, and uses WordNet[5] to identify a set of hyponyms for each keyword. These hyponym sets form the basis for candidate category sets. We do not attempt to distinguish between alternate senses when identifying sets of hyponym.

Category selection. For a given query, there may be multiple ways of classifying the set of retrieved documents. Automatically determining the appropriate classification axis is in itself a difficult procedure. Rather than attempting to divine users' intention, we let the user select the categorization appropriate for organizing the retrieved documents. In our implemented system, a window (shown in Figure 2) shows users the extracted keywords and their associated categories; users can consider the alternative classifications before selecting the most appropriate.

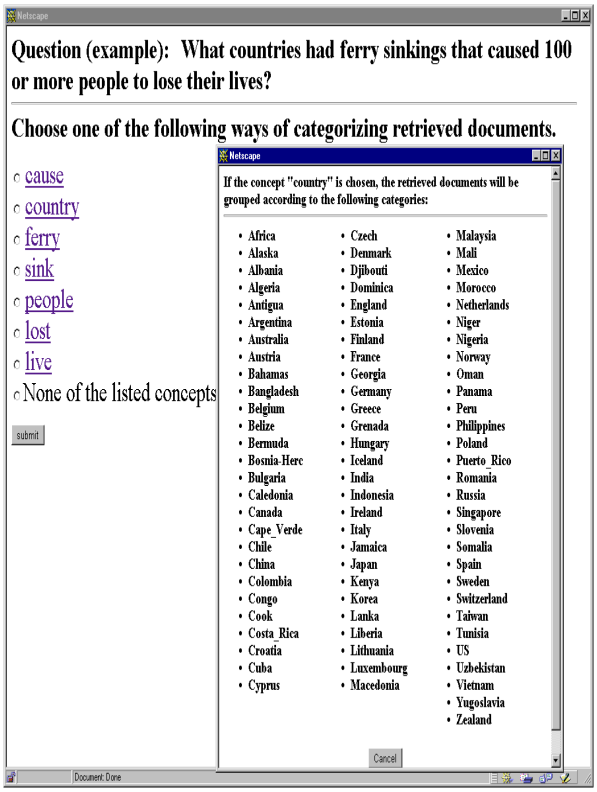

Figure 2: Interface for selecting an appropriate categorization

Classification & ordering. The retrieved documents are then matched against the selected set of categories. Classification is based on ranking the set of retrieved documents by their similarity to the terms describing each category; in our initial implementation, each category is restricted to the ten highest-ranked documents for that category. Documents may belong to more than one category; those that do not match any specific category are allocated to a category of miscellaneous documents. Within each category, the documents are ordered according to their similarity to the original query. Overall, categories are ranked by the similarity of their first-ranked document to the query.

After the search results have been categorized and ranked, they are presented to a user as shown in Figure 3. The interface is divided into halves. In the left half, the upper frame shows the document categories. Each category is expandable and collapsible; in Figure 3 the first category is shown collapsed, and the second expanded. The middle frame shows the already discovered aspects, along with the saved documents relevant to each aspect. A button in the bottom frame enables users to add new categories into which documents may be classified. When any document is selected from the upper-left or middle-left frame, its content is shown in the right half of the window. Any terms that match the currently expanded category are highlighted in red; terms that match the descriptions of other categories are highlighted in blue. This highlighting is intended to help users more easily locate potential answers from within what may be lengthy documents. When the user finds information relevant to an aspect of the topic in a document, they can click on "Save Instances" button. This causes a pop-up window to appear in which the user can note the aspects to which the document is relevant. The discovered aspects and their associated document are then added to the middle-left frame. Whereas the upper-left frame helps the user to search for information that can contribute to their answer, the information in the middle-left frame helps the user synthesize their answer.

4.3 Experimental Setup

Goal. An experiment was conducted to evaluate the usefulness of question-driven classification. The experiment was intended to investigate, for a fact finding task using aspect queries, the ability of user directed, dynamic categorization of retrieved documents to:

- help users find more aspects relevant to their question;
- enhance user satisfaction.

A system using a ranked list interface was used as a control.

Interfaces. The question-driven classification interface is shown in Figure 3. The ranked list interface is shown in Figure 4. As the experiment was focused on comparing alternative organizations of retrieved documents, the two interfaces were kept as consistent as possible, where appropriate. The interfaces varied in three ways. One, the classification-based interface contained a list of expandable categories of retrieved documents in the upper-left frame, whereas the ranked list interface contained a simple ranked list of retrieved documents. Two, the classification-based interface allowed users to interactively add additional categories. Three, no term highlighting was used when displaying documents using the ranked list interface.

Retrieval Engine. The MG[16] search engine with an implementation of passage based retrieval and a variant of the

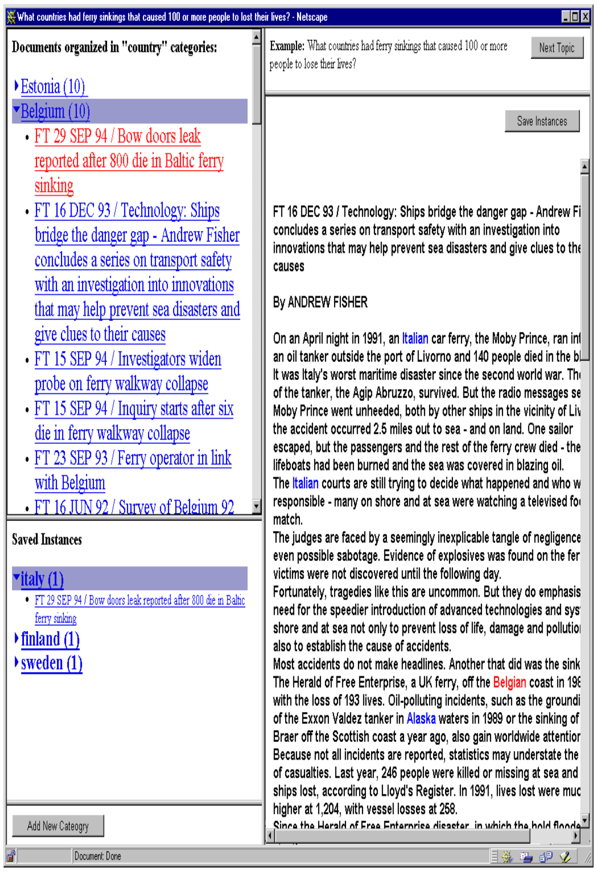

Figure 3: The interface for categorization

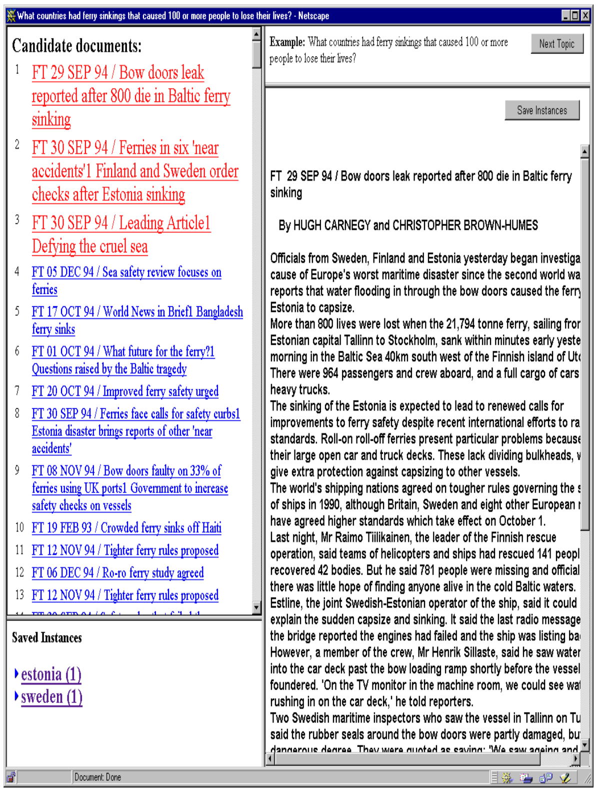

Figure 4: The interface for ranked list

Okapi similarity measure[6] were used as a retrieval mechanism. Those 300 top ranked documents for each topic were then available for display as a list or via categorization. Our previous experiments on TREC-7 aspect topics showed that the top 300 documents on average contained almost 90% of available topic aspects.

Experimental design and procedure. Twenty four subjects undertook the experiment, according to the Latin Square arrangement stipulated by the TREC-8 Interactive Track guidelines. All subjects were either undergraduate or master student from the department of computer science, RMIT, recruited via an internal RMIT newsgroup. All subjects are male, had an average age of 23, 3 years on line search experience, and average FA-1 (Controlled Associations) score of 28.6 and VZ-1 (paper folding) score 15. None of the subjects had previously participated in any TREC experiment.

When subjects arrived at the experiment site, they first filled in a pre-search questionnaire and completing two psychometric tests: FA-1 and VZ-1. Subjects were then given a quick demonstration of the main functions of each interface. During the experiment, prior to using a system for the first time, subjects attempted an example topic to familiarize themselves with its interface; they were free to ask questions about the interface at this point. Each subject was required to fill in post-topic questionnaires after completing each topic, post-system questionnaires after completing their three allocated topics on each system, and an exit questionnaire at the conclusion of the experiment. Subjects were permitted up to twenty minutes to complete each topic; at the twenty minute mark they were informed that the time allocated had expired and were directed to complete their current action, complete the appropriate questionnaire, and move on to the next topic. However, all actions time-stamped outside the allocated twenty minutes were discarded for evaluation purposes. During each search session, every "significant" event such as a user selecting a classification scheme, a category, a document, an interface button, or entering text - was automatically logged and time-stamped. Participants were aware only the differences between the two interfaces; they were not informed which interface was the control system and which was the experimental system.

4.4 Results and Discussion

Effectiveness

In the interactive track, system performance is mainly measured in terms of aspectual precision and aspectual recall, where the aspectual judgements is made by independent assessors. The assessors' judgement was taken as an objective assessment of the quality of the documents that were chosen for viewing. Given that users are involved, aspectual judgement can also made by users (when they select/save an aspect). The subjects' judgement reflects their subjective conception of document's relevance to the topic in terms of their own understanding of the information need. We intend to compare the performance of two interfaces using both subjects' and assessors' judgements.

Subjects' judgement. We started our initial evaluation based on the number of aspects saved by subjects. Here, the relevance of the aspects was solely determined by each individual subject. Table 10 shows the average number (mean) of saved aspects for each topic.

	408	414	428	431	438	446	Average
L	8.5	6.6	8.3	8.9	13	7.6	8.8
C	10.1	7.3	8.8	11.3	11.6	6.7	9.3

Table 10: The average number of saved aspects for each topic in Subjects' view (L: ranked list interface; C: classification-based interface)

Table 10 shows that, overall, subjects saved more aspects by using the classification-based interface than the list interface. The mean number of saved aspects is 9.3 ($SD = 5.2$) for the classification-based interface, and 8.8($SD = 4.1$) for the ranked list interface; this difference is not statistically significant.

Before the experiment, we anticipated that for the four "country" topics (414, 428, 438, and 446), where instances or aspects can be differentiated by country, a classification-based interface would work better than a ranked list, as the retrieved documents are exactly organized under possible aspects; for the other two "non-country" topics (408 and 431), the performance of the classification-based interface would be uncertain, as the categorizations available to the users did not match well with a reasonable division of the topics into aspects. Table 10 shows that, for four topics, more aspects were saved using the classification-based interface than the ranked list interface. That the opposite occurred for topics 438 and 446 was unexpected.

Examination of the data and session log files revealed that the categories for topic 438 were not ranked correctly due to a bug in the code (fortunately the other 5 topics were not affected), as a result, the most relevant categories were not ranked on the top. Therefore, we have excluded this topic from any further evaluation.

We also noticed that five of the twelve subjects did not choose "country" as the basis for classification for the topic 446. It may be that some subjects had difficulty understanding the information need represented by the topic.

There was no correlation between the results of FA-1 and VZ-1 tests and the number of aspects saved by subjects.

Assessors' judgement. The distribution of the assessed aspects for each topic in the pool of candidate documents is shown in Table 11 and Table 12.

Two sets of lists were extracted from the experiment logs: the ordered lists of documents whose full text was viewed during each search, or the "read" lists; and the ordered lists of documents from which subjects saved at least one aspect, or the "saved" lists. Table 13 presents the average number of documents read/saved from each system, and aspectual precision and recall for each list under each system.

Table 13 shows that, on the average, subjects read more documents using the classification-based interface, but the documents comprising the classification-based interface read list on average contained fewer aspects than those from the list interface. Subjects saved approximately the same number of documents from the classification-based interface, but again, these sets of saved documents do not cover as many aspects as those from the list interface, although the difference on aspectual recall between these two interfaces is not significant.

We had expected that the classification-based information organization of the retrieved documents would do better because we believe this organisation is closer to users' expectation of the answer. We think the following issues could explain the reason why the classification-based interface didn't perform better than the list interface:

- For the country topics 414, 428 and 446, the average aspectual recall for the top 20 documents of the ranked list is 0.516. To outperform the basic ranked list, the categorization approach would need to offer the user a reasonable viewing sequence with a higher recall. However, the category ranking, as implemented, failed to provide this. For example, consider the following strategy: if a user were to view the highest ranked, previously unread document from each of the first 20 ordered categories in turn, the average aspectual recall for this list of 20 documents is only 0.509. If the user modified this strategy by skipping categories for which they had already found relevant aspects (in a previously read document), the average aspectual recall drops further to 0.458.
- Many documents appear in more than one category. For topics 414, 428 and 446, each subject expanded 14.3 categories on average, whereas each subject read 18.7 documents on average, indicating that typically more than one document was selected from each category. But for many of the first 20 categories from each topic (414: 6/20; 428: 9/20; 446: 0/20), the second ranked documents in each category contributed no additional aspects. This may indicate that while the classification-based interface is suitable for organizing answers and for finding specific aspects, it is not very effective for the TREC-8 task of finding non-repeated aspects.

Evaluation of the experiment based on the subjects' judgement presents a clearly different picture to evaluation based on the assessors' judgement. This suggests that the subjects' understanding of the sought-after aspects is slightly different from the assessors' view, or that the subjects did not fully understand what was required for some topics. For example, for Topic 428 ("What countries other than the US and China have or have had a declining birth rate"), there is confusion over what is a region of a country what is a country. Some subjects saved documents that discussed regions with declining birth rates within countries, but that did not necessarily imply that the country as a whole had a declining birth rate. Some subjects also saved documents that predicted that particular countries would in the future have a declining birth rate. Topic 446 ("In what countries have tourists been subjects to acts of violence causing bodily harm or death") is another example. Some subjects saved documents that mentioned countries in which attacks on tourists had occurred but which were judged not relevant (no aspects present) by the assessors because the documents did not explicitly indicate that bodily harm or death had in fact occured. To reduce the judgement mismatch between subjects and assessors, more detailed or specific description of instances may be needed.

Subject satisfaction

User satisfaction is also an important measure of human-computer interfaces. A user-satisfaction questionnaire, adapted from [4], was used to assess each user's satisfaction with the

Topic	Number of documents								
	5	10	15	20	50	100	150	200	300
408	0.250	0.292	0.333	0.375	0.542	0.750	0.750	0.750	0.792
414	0.667	0.833	0.833	1.000	1.000	1.000	1.000	1.000	1.000
428	0.192	0.308	0.346	0.423	0.538	0.731	0.769	0.808	0.846
431	0.325	0.450	0.500	0.500	0.750	0.975	1.000	1.000	1.000
446	0.125	0.125	0.125	0.125	0.250	0.438	0.438	0.438	0.438
AV:	0.312	0.402	0.428	0.485	0.616	0.779	0.791	0.799	0.815

Table 11: Aspectual recall for the candidate document pool of the 300 highest-ranked documents

	408	414	428	431	446
Total aspects in all documents	24	12	26	40	16
Total documents containing aspects	71	16	40	67	58
Aspects in candidate documents	19	12	22	40	7
Candidate documents containing aspects	56	16	30	66	21

Table 12: Aspect coverage for each topic, as judged by the NIST assessors.

two interfaces. The questionnaire focused on subjects' satisfaction with the presentation format, the delivered data, an interface's ease-of-use, and the time available for the topics. Members of one group of 12 subjects were required to fill in this questionnaire after completing each session of three topics with an interface. Subjects were asked to respond to the questions using a five point Likert-type scale, where 1 = almost never; 2 = some of the time; 3 = about half of the time; 4 = most of the time; and 5 = almost always.

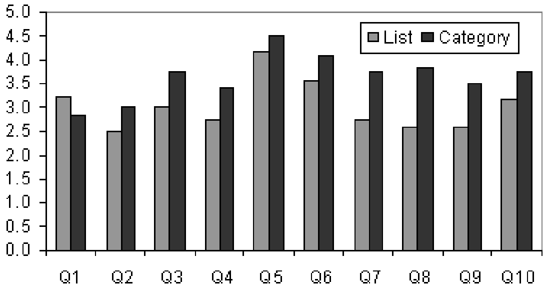

Figure 5: The results from subjects' satisfaction questionnaire (Q1: Too much information, Q2: Precise instances, Q3: Sufficient instances, Q4: Enough search time, Q5: Easy to use, Q6: User friendly, Q7: Clear organization, Q8: Useful format, Q9: Organization what is needed, Q10: Satisfaction with the interface.)

The results from the questionnaires are shown in Figure 5. The X-axis shows the questions asked, and the Y-axis the mean score for each question. Q1- Q3 shows subjects' satisfaction with the displayed contents; Q3 the time available; Q5-Q6 the ease of use; Q7-Q9 the way the data was organized; and Q10 overall satisfaction with the interface. Note that for Q1, a lower score indicates greater satisfaction. From Figure 5 we can see that, for all questions, the satisfaction scores for the classification-based interface are higher than the ranked list interface. This difference is statistically significant ($p < 0.001$, paired, one tail t- test).

Figure 5 also suggests that the organization of retrieved data may influence the subjects' perception of it. Although both interfaces offered the same amount of information, albeit differently organized, subjects nonetheless felt that the ranked list interface showed too much information, and felt able to find neither enough information nor sufficiently precise information to answer the topics. Given that subjects saved approximately the same number of facts with each interface, this indicates a strong discrepancy between subjects' preferences and their performance.

Although most comments from the post-experiment questionnaire regarding the classification-based interface were positive, offering suggestions for further improvement, some negative ones were made. Subjects wanted the set of categories to be derived from more than one keyword for some topics; for example, topic 408 includes the phrase "tropical storm" which is a more logical basis for categorization than either "tropical" or "storm" in the context of the topic. Subjects also disliked only having one chance to select the classification axis; when subjects later decided they had made the wrong choice, there was no mechanism for them to undo that decision. We had recognized that such a mechanism would be a desirable feature beforehand, but chose not to include it in this preliminary experiment in order to focus on evaluating dynamic classification. Our main object has been to determine whether a classification-based interface can help the subject to find more facts, under the assumption that a subject can select a right set of categories. This assumption may not have been a valid one, as previously noted.

Another problem observed was that some classifications resulted in too many categories being created (over 500 in one case). This could be addressed by imposing a maximum for the number of categories that may be formed; an appropriate value for this threshold would need to be determined. An alternative is to use some form of hierarchical or grouped super-categorization.

A general problem we experienced is the shortcomings of using WordNet as the source for hyponyms. Not surprisingly, WordNet was not always able to supply hyponyms appropriate to the context of the topics. Additionally, the issue of sense-selection applies here, as in all natural-language dependent systems.

	Read List		Saved List	
	category Mean (SD)	list Mean (SD)	category Mean(SD)	list Mean (SD)
Documents read/saved	18.7(10.5)	16.0(7.2)	6.35(4.06)	6.30(2.6)
Aspectual Precision	0.44(0.26)	0.52(0.26)	0.72(0.26)	0.75(0.24)
t-test	< 0.04 (*sig.*)		< 0.26 (*not sig.*)	
Aspectual Recall	0.36(0.23)	0.41(0.20)	0.29(0.21)	0.32(0.19)
t-test	< 0.08 (*not sig.*)		< 0.19 (*not sig.*)	

Table 13: Comparison of performance between category and list - for five topics

4.5 Conclusion

We evaluated our preliminary hyperthesis of organizing retrieved documents according to the intentions behind a question. We are glad to see that subjects are significantly more satisfied with the classification based interface. Although our current experiment result did not show a significant benefit in terms of aspectual recall by using classification based interface, we still believe it has a potential and could be improved. The further improvements may include:

- Utilizing both the phrases and the single words as the basis for classification.
- Allowing users to select more than one term or phrase as the basis for classification.
- Allowing users to remove or modify individual categories; this may provide a user-driven way around problems of sense-selection.
- Providing a better description of each potential classification and how it was derived.
- Developing a better ranking for categories.

5 Spoken Document Retrieval Track

Two runs were submitted for this year's Quasi-SDR runs. The word-based documents were first translated to phonemes using a text-to-phoneme algorithm. We assumed that there is a certain level of word recognition error for each type of transcription. Given this, we utilised a passage retrieval technique to perform the retrieval.

5.1 Text-to-phoneme Algorithm

The words were translated to phoneme using a modified version of the text-to-speech translation algorithm given by Carney [2]. The original algorithm was modified to produce American pronounciation as close as possible to those given by CMU [3]. The advantage of this approach is the implicit handling of unknown words. This will not be a problem if there is a translation dictionary containing all the words in the collection. As a test, the reference and baseline transcriptions were translated to phonemes using the CMU pronounciation dictionary [3]. There were approximately 18,000 and 300 unknown words in the reference and baseline transcriptions respectively. The reference collection has approximately 34,500 unique words while the baseline collection has about 19,000 unique words. This may mean that a lot of unknown words were not being recognised by the automatic recognition system. A problem of this algorithmic approach is the inconsistencies in the generation of some of the pronounciations.

Len	1	2	3	4	5	6
# of Words	11	32	45	51	55	46
Len	7	8	9	10	11	12
# of Words	5	28	16	12	4	4

Table 14: *Distribution of unique query words for different phoneme lengths.*

5.2 Matching Algorithm

The algorithm described here is based around the notion of a passage for each indexing feature, whether that feature is a word or a sequence of phonemes. In this instance, a passage is defined for a query term which is a sequence of phonemes translated from a query word. Potential passages are found within each document for each query term by way of approximate matching. Later, the retrieved passages for each query term is weighted according to $tf.idf$ scores and combined for each document before similarity scores are calculated.

The following subsections describe each individual components.

Window Size

The window size determines the error margin allowed for the transcriptions. If the word error rate (or WER) of a recogniser is small, then it is possible to have a small window and yet find all the phoneme sequences with a high degree of match. For higher WER, a larger window may be required to find all the phoneme sequences describing the required word, although this may lead to higher false matches.

From the translation of the query words to phoneme sequences, we found that phoneme sequences range from 1 phoneme to 12 phonemes (See Table 14 for further details). Most words were between 4 to 6 phonemes long. Here, the assumption made is that query terms of medium length required a larger error margin than longer query terms.

For each term in the query, we have to determine a suitable window size (or passage length) for matching documents. If the number of phonemes in the query term is less than 8, length of the matching window is set to number of phonemes times 1.5. For query terms with more than 8 phonemes, window size is set to number of phonemes plus 4.

Given that short query terms are likely to have many matches, both false and correct ones, those that are shorter than three phonemes are discarded and not used in the matching and ranking processes. This acts as a form of stopping.

Phoneme-based Matching

The scores for matching sequences for each query term is calculated. First, the size of a passage is determined by the window size described previously. Therefore for each passage, its size is a constant for that particular query term or phoneme sequences. At this stage, we need to maximise the score for the longer sequences matched. Then the score is penalised if the sequences matched in the passages are not in-order.

Passages in documents are scored as follows. Locate all possible *exact* phoneme matches between a query term t and passage p. Sort them according to the longest sequence matched in decreasing order. Using the longest sequence as a starting point, determine as many phoneme sequence matches for each passage as possible. The passage score is the sum of square of non-overlapping phoneme matching sequences:

$$Score_1(p,t) = \sum_{m \in M} [len(m)]^2 \tag{11}$$

where M is a set of phoneme sequence matches found between t and p. $len(m)$ is the length of m expressed in the number of phonemes. By taking the $len(m)^2$, we will always favour the longest sequence matched in any passage.

Equation 11 does not make a distinction between passages that have phoneme sequences in the same order as in the query term from those passages that are not. For example, the query HAIR has the phoneme sequence "HH EY R", while GREYHOUND is translated as "G R EY HH AW N D ". The false matching of "HH" and "EY" in GREYHOUND will have the same score as other words containing the same sequence in the right order. As a result, we attempt to "penalise" those passages where disjoint phoneme sequence matches do not have the same order as in the query term. We devised a simple heuristic that attempts to find all matching sequences in passages that are not in order and use their sum as the factor to penalise the passage score:

$$Penalty(p,t) = \sum_{u_m \in U_M} [len(t) - len(u_m)]$$

where U_M is a set of phoneme sequence matches between t and p that are found to be out of order with respect to t. Therefore the refined passage score becomes:

$$Score_2(p,t) = Score_1(p,t) - Penalty(p,t) \tag{12}$$

We note that other, possibly better techniques, could be used to distinguish such passages. This is an ongoing work for this approach.

A final modification to $Score_2(p,t)$ is required so passages selected for individual query terms can be combined in order to arrive at the overall score of documents. Passage scores are normalised with respect to the number of phonemes in the query term:

$$Score_3(p,t) = Score_2(p,t)/len(t) \tag{13}$$

This also ensures that short query sequences aren't penalised too heavily although longer sequences are usually found with higher accuracy. At the end of the matching, we output 1000 best passages with the highest score (that is, $Score_3(p,t)$). Note that this scoring mechanism permits multiple passages to be matched within a document.

Individual Term-passages Weighing

Selecting passages is not much use if documents need to be ranked. The process described in the previous section could be thought of as a means of word-spotting. The aim here however, is to rank documents. Therefore, taking the approximate word matches as produced in the previous section, we attempt to weigh them in documents.

The idea is to assume that passages with high scores are likely to correspond to the "true" word matches in the documents. By "true" we mean that the actual word appears in the document but was possibly misrecognised.

Passage scores, as computed above, can be considered as an indicator of approximate word matches in documents. Therefore, passage scores are used to accumulate in-document frequencies for query terms as follows:

$$f_{d,t} = \sum_{p \in P} \frac{Score_3(p,t)}{Score_{max}(t)}$$

where P is the set of passages matching the query term t in document d. Passage scores are normalised with respect to the maximum score for the query term t ($Score_{max}(t)$). Passages with low scores are discarded because they are likely to be false matches.

Similarly, while computing $f_{d,t}$, we estimate document frequency for query term t (f_t) as the number of documents in which $f_{d,t}$ is higher than zero. (Note: this may be undesirable because the query terms will not be discriminated well because most of them will have high f_t values.)

Given the weights of query terms in documents, a final step is to compute the similarity score of query q to document d. This can be done by using standard tf.idf weighting [10] or more modern weighting approach such as Okapi that we have described in Section 1.2 (see Equation 1). (Note that in our case the matches of query terms in documents are not exact due to our approximate matching algorithm.)

Results

The retrieval effectiveness of our official run (mds-base) is low. This is due to mainly two reasons. First, term weights in documents were computed using the standard tf.idf formulation [10], and no document length normalisation was used. In our post-TREC runs we added Okapi measure with document length normalisation as an alternative to tf.idf weighting. The official run that used the base transcription (BASE) is marked as mds-base in Table 15. In addition there is mds-ref run which used identical approach to mds-base except the collection was a manual transcription (REF).

Besides standard tf.idf weighting there is room to improve in the way document frequencies are estimated and the document length component be used in computing similarities for documents. Our basic approach estimates document frequencies for terms (or f_t) by assuming that if there is at least one "partial" match of the query term in document, this document is counted (that is when $Score_3(p,t)$ is greater than zero for passage p in a document). It turns out that most query terms end up with high f_t values because of the approximate matches on documents. The skewed distribution of f_t towards high values leads to poor discrimination of terms when similarities between documents and queries are computed [10].

There is no obvious way of defining document frequencies in the context of phoneme based retrieval because there is no notion of words in documents. However, we define an upper bound for retrieval using the basic window-based ranking as described earlier by extracting document frequencies from REF collection. (Note that in real situation this information would not be available.) The retrieval effectiveness for this run is marked as ft_{REF} in Table 15.

Run	p@5	p@30	AvgP	AvgP(Revised)	Recall
Text-based runs (REF collection)					
q-unstopped	0.5320	0.2947	0.3857	-	1518
q-stopped	0.5200	0.2953	0.3781	-	1558
Text-based runs (BASE collection)					
q-unstopped	0.5000	0.2740	0.3197	-	1381
q-stopped	0.4880	0.2747	0.3169	-	1464
REF (manual) converted to phonemes					
mds-ref	0.2920	0.1647	0.1740	0.1775	1398
Base (word-recogniser) converted to phonemes					
mds-base	0.2320	0.1293	0.1323	0.1350	1251
ft_{REF}	0.3680	0.2107	0.2119	-	1305
ft_{REF},Wd_{REF}	0.4040	0.2173	0.2445	-	1275
ft_{WSJ},Wd_{REF}	0.3960	0.2280	0.2752	-	1347
ft_{TREC},Wd_{REF}	0.3880	0.2340	0.2811	-	1349

Table 15: *Retrieval results for reference and baseline transcripts using Cosine-based weighting.*

An alternative for approximating document frequencies is to use an auxiliary collection which is not related to the speech collection. In this experiments we used WSJ data from disk 1 and 2 of TREC and TREC-7 (also used in TREC-8). The retrieval effectiveness for this run is marked as ft_{WSJ} and ft_{TREC} in Table 15 respectively.

Another shortcoming of our official run is that there was no document length normalisation; longer documents are expected to be favoured because they are more likely to have more matches of query terms than shorter documents. However, there is no clear way of incorporating document length (or Wd for document d) that is expressed in terms of phones into the standard vector space model. Under superficial circumstances we assume that we have access to document lengths as produced by REF collection. These are used in conjunction with ft_{REF} and is shown as Wd_{REF} in Table 15.

As it can be seen, both document frequencies and document lengths improve effectiveness.

Table 16 illustrates our results using Okapi-based weightings. Here, the number of phonemes were used to determine the the document length Wd_{phn} instead of the estimated document lengths in terms of words as used in the Cosine-based weighting. Table 17 gives use the results for the base transcriptions in phonemes where the queries were stopped prior to being translated to phoneme sequences. At this moment, we are unable to draw any conclusions from these results.

We also try to eliminate passages that had at least one matching sequence that is not in order of the query term. This is marked as "ft_{REF},Wd_{phn},$force\_order$" and had no impact on retrieval. However, this might not be right if a good sequence happens to align with another sequence matching by chance in the right order. Such possible good passages would not be discounted in this case.

5.3 Conclusion for Quasi-SDR

This year we attempted a passage-based technique to perform retrieval using phoneme sequences. Documents and queries were first translated to phonemes using a rule-based text-to-speech algorithm. A passage was created for each query term and approximate matches were computed within each document. These passages were combined using either cosine or Okapi-based weighting scores for each document before similarity was computed for each query.

Table 15 illustrates an important point. Although the phoneme based retrieval of speech documents using word-based transcripts is not as effective as when words are used, the approach is more general than text-based retrieval. Retrieval of word-based transcripts is effective as long as two conditions are met: word recognition accuracy is high and a single language is used. For noisy word-based speech recognition, accuracy can be as low as 40%. On the other hand, phoneme based retrieval using ngrams and approximate matching works well under good conditions in comparison to word-based approach but will be superior in cases when poor recognition in which our data is not a good representation. Approximate matching methods like Wechsler [14] and Ng [8] are similar to this technique but different because recognition information like confusion matrices are required to determine the approximate matches. Note however, that recent attempts to recover from poor transcripts from a word-based recogniser was shown to be effective [12].

More analysis and experimentations are required to further test the assumptions made and to improve on our approximate matching strategies. Variables to investigate include determining an optimal window size, calculating the scores in the approximate matching process and the weighting algorithms for similarity computations.

Acknowledgements

The speech experiments were conducted while on leave at Sharp Laboratories of Europe Ltd. Thanks to Dr. Victor Poznański for discussions and encouragement. Also to the IT group for providing the resources.

References

[1] N. J. Belkin, P. G. Marchetti, and C. Cool. Braque: Design of an interface to support user interaction in information retrieval. *Information Processing and Management*, 29(3):325–344, 1993.

[2] E. Carney. *A Survey of English Spelling.* Routledge, London, 1994.

[3] Carnegie Mellon University Pronouncing Dictionary. Cmudict.0.4. Available from http://www.speech.cs.cmu.edu/cgi-bin/cmudict, 1995.

[4] William J. Doll and Gholamreza Torkzadeh. The measurement of end-user computing satisfaction. *MIS Quarterly*, pages 259–274, June 1988.

Run	p@5	p@30	AvgP	Recall
Text-based runs (REF collection)				
q-unstopped	0.6240	0.3307	0.4399	1537
q-stopped	0.6160	0.3300	0.4389	1568
Text-based runs (BASE collection)				
q-unstopped	0.6000	0.3073	0.3888	1451
q-stopped	0.5840	0.3047	0.3877	1485
Base (word-recogniser) converted to phonemes				
base	0.2640	0.1493	0.1553	1248
ft_{REF}	0.4240	0.2320	0.2438	1329
ft_{REF},Wd_{phn}	0.5000	0.2567	0.3045	1316
ft_{WSJ},Wd_{phn}	0.4160	0.2187	0.2780	1299
ft_{TREC},Wd_{phn}	0.3560	0.2033	0.2482	1185
ft_{REF},Wd_{phn},$force\_order$	0.5000	0.2553	0.3041	1317

Table 16: *Retrieval results for reference and baseline transcripts using Okapi-based weighting.*

Run	p@5	p@30	AvgP	Recall
Base (word-recogniser) converted to phonemes				
base-stopped queries	0.4160	0.2307	0.2376	1337
ft_{REF}	0.4200	0.2380	0.2544	1329
ft_{REF},Wd_{phn}	0.5040	0.2600	0.3075	1317
ft_{WSJ},Wd_{phn}	0.4160	0.2187	0.2781	1308
ft_{TREC},Wd_{phn}	0.3640	0.2067	0.2543	1250
ft_{REF},Wd_{phn},$force\_order$	0.5040	0.2587	0.3070	1318

Table 17: *Same as Table 16 but for stopped queries.*

[5] Christiane Fellbaum, editor. *WordNet: An Electronic Lexicial Database.* The MIT Press, Cambridge Masschusetts, 1998.

[6] M. Fuller, M. Kaszkiel, D. Kim, C. Ng, J. Robertson, R. Willinson, M. Wu, and J. Zobel. TREC7 ad hoc, speech, and interactive tracks at MDS/CSIRO. In *Proceeding of Seventh Text Retrieval Conference (TREC-7)*, November 1998.

[7] J.B. Lovins. Development of a stemming algorithm. *Mechanical Translation and Computation*, 11(1-2):22–31, 1968.

[8] K. Ng. Towards robust methods for spoken document retrieval. In *Proceedings of International Conference on Spoken Language Processing*, volume 3, pages 939 – 942, Dec 1998.

[9] J.J. Rocchio. Relevance feedback in information retrieval. In *The SMART Retrieval System – Experiments in Automatic Document Processing.* Prentice Hall, Inc., 1971.

[10] G. Salton and M.J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, New York, USA, 1983.

[11] A. Singhal, C. Buckley, and M. Mitra. Pivoted Document Length Normalisation. In H.-P. Frei, D. Harman, P. Schäuble, and R. Wilkinson, editors, *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1996.

[12] A. Singhal and F. Pereira. Document expansion for speech retrieval. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 34–41, 1999.

[13] S. Walker, S. Robertson, M Boughanem, G. Jones, and K. S. Jones. Okapi at TREC-6 Automatic ad hoc, VLC, routing, filtering and QSDR. In *Proceedings of the Sixth Text REtrieval Conference. (TREC-6)*, 1997.

[14] M. Wechsler, E. Munteanu, and P. Schauble. New techniques for open-vocabulary spoken document retrieval. In *Proceedings of the 21st ACM-SIGIR International Conference on Research and Development in Information Retrieval*, pages 20 – 27, Melbourne, Australia, Aug 1998.

[15] Ross Wilkinson and Michael Fuller. Integrated information access via structure. In Maristella Agosti and Alan Smeaton, editors, *Information Retrieval and Hypertext*, pages 257–271. Kluwer Academic Publishers, 1996.

[16] I. Witten, A. Moffat, and T. Bell. *Managing Gigabytes: Compressing and indexing documents and images.* Van Nostrand Reinhold, 1994.