RMIT University at TREC 2005: Terabyte and Robust Track

Yaniv Bernstein Bodo Billerbeck Steven Garcia
Nicholas Lester Falk Scholer Justin Zobel
School of Computer Science and Information Technology
RMIT University, GPO Box 2476V, Melbourne 3001, Australia

William Webber
Department of Computer Science and Software Engineering
The University of Melbourne, Melbourne 3010, Australia

1 Introduction

RMIT University participated in the terabyte and robust tracks in TREC 2005.

The terabyte track consists of the three tasks: adhoc retrieval, efficient retrieval, and named page finding. For the adhoc retrieval task we used a language modelling approach based on query likelihood, as well as a new technique aimed at reducing the amount of memory used for ranking documents. For the efficiency task, we submitted results from both a single-machine system and one that was distrubuted among a number of machines, with promising results. The named page task was organised by RMIT University and as a result we repeated last year's experiments, slightly modified, with this year's data.

The robust track has two subtasks: adhoc retrieval, and query difficulty prediction. For adhoc retrieval, we employed a standard local analysis query expansion method, sourcing expansion terms for different runs from the collection supplied, from a side corpus, or a combination of both. In one run, we also tested removing duplicate documents from the list of results; in order to predict topic difficulty, we evaluated different document priors of the documents in the result set, in the hope of supplying a more robust set of answers at the cost of returning a potentially smaller number of relevant documents. The second task was to predict query difficulty. To this end, we compared the order of the documents in the result sets to the ordering as determined by document priors. A high similarity in orderings indicated that the query was poor. Two different priors were used. The first was based on document access counts, where each document is given a score that is derived from how likely it is to be ranked by a randomly generated query. The second was directly related to the document size.

In this paper we outline our approaches and experiments in both tracks, and discuss our results.

2 The Zettair Search Engine

We used `zettair` for all experiments outlined in this paper. `Zettair` is a publicly available retrieval engine developed by the Search Engine Group at RMIT University. It is available under a BSD license from `http://www.seg.rmit.edu.au/zettair` and was described in more detail in our submission to TREC last year (Billerbeck et al., 2004).

3 Terabyte Track

We participated in all three tasks: adhoc, efficiency and named page finding. The description of the runs and the results are given in the following for each of the three tasks.

3.1 Adhoc Task

Unlike last year, where we used Okapi BM25, both adhoc runs we submitted this year are based on language modelling, using a query likelihood approach with Dirichlet smoothing.

Dirichlet-smoothed Language Modelling

The query likelihood can be formulated as follows:

$$P(q|d) \quad = \quad \prod_{t \in q} P(t|d)$$

where q is the user query, while document d is one of N documents in the collection, f_t of which contain term t. The Dirichlet-smoothed term probability (Zhai and Lafferty, 2004) is:

$$P(t|d) = \frac{|d|}{\mu + |d|} \times \frac{f_{t,d}}{|d|} + \frac{\mu}{\mu + |d|} \times \frac{f_t}{N}$$

where $|d|$ is the length of document d, $f_{t,d}$ is the number of occurrences of term t in document d, and μ is a smoothing parameter. Through a series of rank-equivalent transformations, the query likelihood can be efficiently computed by:

$$\log P(q|d) \stackrel{rank}{=} |q| \times \log \lambda_d + \sum_{t \in q \cap d} \log \left(\frac{N \times f_{t,d}}{\mu \times f_t} + 1 \right)$$

where $\lambda_d = \mu / (\mu + |d|)$. We trained μ on last year's terabyte topics, and set μ to a value of 1,500. No stopping was used for this task and – due to an oversight – neither was stemming. Using this approach to the query likelihood model we obtained the run ZETDIRHOC. This run was used as a baseline to compare against the following run.

Effective Accumulator Limiting

For the ZETDIRA run made use of a new accumulator scheme, which limits accumulators for greater memory efficiency during query evaluation.

This scheme is employed for query evaluation using document-ordered inverted lists processed in a term-wise fashion. It keeps accumulators in a linked list for ease of insertion and removal. Unlike previous schemes (Moffat and Zobel, 1996), once an accumulator is added for a particular document, it can be removed, depending on a partial similarity threshold that is varied in order to control the number of accumulators used to evaluate the query. For each term, the floating point partial similarity threshold corresponds to an $f_{d,t}$ occurrence threshold, which postings potentially creating new accumulators must exceed before being inserted into the accumulator list. However, existing accumulators are removed if they fall below the partial similarity threshold as they are examined. The partial similarity threshold is set to the minimum partial similarity value while there is no possibility of the accumulator limit being exceeded. Once a sufficiently frequent term is processed that limiting must be employed, an initial partial similarity threshold is estimated from the first few postings. This threshold is re-evaluated periodically, with the period between re-evaluations doubling each time. A simple statistical estimate of the number of expected accumulators at the end of the term is made during each re-evaluation. If this estimate is within a tolerance factor of the accumulator limit, no changes are made to the thresholds. Otherwise, the $f_{d,t}$ threshold is adjusted up or down by a step, with the partial similarity threshold adjusted correspondingly. The step quantity is initially set to half of the $f_{d,t}$ threshold, and halves after each re-evaluation. See Lester et al. (to appear) for a full description of this method.

For the ZETDIRA run, we used the new scheme with a limit of 100,000 accumulators, which is 0.4% of the collection size. We employed the same accumulator scheme for the baseline run, but since we used a large accumulator size – 240,000 accumulators for 24,000,000 documents, which has been shown to have virtually no impact on retrieval performance (Moffat and Zobel, 1996) – we effectively did not make use of the accumulator limiting scheme for the baseline.

Run	Technique employed	MAP	P@10	P@20	bpref	R-P
Median		0.282	–	–	0.303	–
ZETDIRHOC	Baseline	**0.279**	**0.556**	**0.516**	**0.300**	**0.341**
ZETDIRA	Adaptive pruning	0.276	0.548	0.508	0.295	0.336

Table 1: *Effectiveness of terabyte ad-hoc task runs based on mean average precision (MAP), precision at 10 and 20 documents returned (P@10, P@20), preference relation based on binary relevance judgements (bpref), and precision at the number of relevant documents for each query (R-P).*

Results

Results for the adhoc runs are shown in Table 1. Our results are reasonably close to the median results obtained over all submitted runs. As expected, the average effectiveness results of our adaptive pruning scheme are effectively the same as the baseline scheme. Note that this scheme has only a negligible effect on evaluation time, but it has a far lower memory consumption (see Lester et al. (to appear) for details).

3.2 Efficiency Task

We submitted two runs for the efficiency task. The first one is `zettair` as described in Section 3.1, using a query likelihood approach with Dirichlet smoothing, but here we used stemming (which was not used for the effectiveness task). There were also other minor differences, as explained below. For the second run, we used a modified system in order to evaluate queries in a distributed manner.

Monolithic

We made the following modifications to the search engine that was used for the baseline adhoc run. We used a stoplist of 477 terms to stop queries. However, the index was not stopped. Furthermore, in order to decrease the size of the index and speed up query evaluation (since less data has to be processed), we did not store word offsets in the index.

The machine we used for this experiment was a Intel Pentium IV 2.8 GHz with 2 GB of main memory running Fedora Core 2. More details are given in Table 2.

Distributed

For the distributed run (ZETDIST) we used a cluster of eight evaluator nodes and one controller. Each evaluator node was a single-processor 2.8 GHz Intel Pentium IV with 1 GB of RAM and a single 250 GB local SATA disk. The index was partitioned document-wise; that is to say, each of the nodes held a local index for one eighth of the documents in the collection. These local indexes were essentially independent of each other, except that global term weights were used during evaluation. Each query was forwarded to each of the nodes by the cluster controller. Each node then evaluated the query against its portion of the index, using the same approach as for the monolithic run, that is, no term offsets were stored in the index. However, no stopping was performed. Once evaluated, the top results and scores were sent back to the controller. Finally, the controller merged the scores and returned the results.

The total query evaluation time on the cluster was 2901.7 seconds. We note in passing that we observed the requirement, clarified on the task mailing list, that queries be executed serially. This means that the result for one query was returned before processing on the next query began. If this requirement was relaxed, and the parallel processing of queries was allowed (still requiring, though, that queries be started in arrival order), then a significant improvement in processing time could be achieved. For comparison, we conducted (but did not submit) a run allowing up to 24 queries to be processed in parallel. This run took 2283.6 seconds, a speedup of over 25%. Allowing the parallel processing of queries is particularly necessary to achieve maximal throughput for alternative architectures based around partitioning the index by vocabulary, rather than by corpus.

Characteristic	All runs submitted			Our runs submitted	
	Maximum	Median	Minimum	ZETDIR	ZETDIST
Percentage of doc collection indexed	100	100	66	100	100
Indexing time (minutes)	23,528	990	44	595	64
Av. time to return top 20 docs (secs)	4.4	0.47	0.024	0.23	0.06
Total proces. time for all topics (secs)	219,354	17,730	1,201	11,565	2,901
Total number of CPUs in system	23	2	1	1	8
Total amount of RAM in system (GB)	23	4	1	2	8
Size of on-disk file structures (GB)	600	63	5.99	17	19
Year of system purchase	2005	2004	2002	2004	2004
Estimated hardware cost (US$)	45,000	5,000	750	1,200	6,000

Table 2: *System details for runs submitted for the efficiency task.*

Run	Technique employed	MAP	P@10	P@20	bpref	R-P	Time (sec)
ZETDIR	Monolithic	**0.067**	0.570	**0.541**	**0.053**	**0.080**	0.23
ZETDIST	Distributed	0.061	**0.574**	0.530	0.052	0.075	**0.06**

Table 3: *Effectiveness and efficiency of terabyte efficiency runs. In addition to the measurements described in Table 1, average query times in seconds are shown. The median average query time of all systems was 0.47 seconds.*

Results

Table 2 gives a summary of the two systems that we used for the runs, as well as an overview of the range of systems that were used by other participants.

The results of our two runs are shown in Table 3. Since no more than 20 results per query were listed, only the precision at 10 and 20 documents returned is comparable with the adhoc results (other measurements such as MAP are not meaningful when comparing runs against those of other tasks, where up to 1000 answers were returned). Again, both runs produced quite comparable results; while the monolithic run is slightly ahead in most effectiveness measures, the distributed run by far outperforms the former in terms of efficiency, by a factor of four. However, this increase in efficiency is coupled with a similar increase in cost for the distributed system.

3.3 Named Page Finding Task

Our research group created the topics for this year's named page finding task. To avoid any possible advantage from our knowledge of the query creation process, we limited ourselves to producing runs using mostly the same configurations as submitted to last year's terabyte track.

We submitted a baseline run, one run that made use of anchor text, one that gave a higher weighting to pages that contained query terms in a small text window, and one where results were restricted to documents where all query terms appeared in the first 50 terms. These runs are briefly discussed in the following sections; for more detail, see our paper from TREC 2004 (Billerbeck et al., 2004).

Baseline

For the baseline run (ZETNP), we used the standard `zettair` system with Okapi BM25 (Sparck Jones et al., 2000):

$$BM25(q,d) = \sum_{t \in q} \log\left(\frac{N - f_t + 0.5}{f_t + 0.5}\right) \times \frac{(k_1+1) f_{d,t}}{K + f_{d,t}}$$

where $K = k_1 \times \left((1-b) + b \times \frac{|d| \times N}{\sum_{i \in N} |d_i|} \right)$, $k1 = 1.2$, and $b = 0.75$. Since we set $k3$ to 0, the query-term frequency component can be dropped from the typical formulation. The main reason for using Okapi is that it is easier to combine with the anchortext ranking scheme that we used, as described below.

Anchortext

For the ZETNPANC run, we combined results from the full text index with an index obtained from anchor text, using the metric devised by Hawking et al. (2004). For this scheme, the two scores are linearly interpolated, mixing the score of the full text with the anchor text score in a ratio of 4:1. This is the same setting as used last year (that is, we did not train on last year's data). This run was also used as a basis for the next run, described below.

The anchortext index was constructed from a collection of surrogate documents, where each document in the collection was replaced by the anchor text from inlinks pointing to the document.

Fuzzy Phrase

We followed our intuition (and earlier, unpublished results) that a document may be a good match if it contains phrases that are made up of query terms, even though these query terms were not explicitly specified by the user as a phrase. We extended the concept of phrases to include *fuzzy phrases*; here, query terms appear in a short text window, possibly interspersed with other terms. A necessary parameter to consider then is the *fuzziness* between an exact phrase and its fuzzy phrase counterpart. This parameter governs how large the text window may be. In our case, we used a window size of 20 terms. A fuzzy phrase is also one where terms appear in any order other than that specified in the query, as long as there are no other terms in-between.

To arrive at the ranking for the ZETNPANCFUZZ run, we mixed the baseline results (72%) in linear combination with scores resulting from anchor text (18%) and the fuzzy phrase scores (10%). The value of 10% was derived by training using the GOV1 collection and the topics and relevance judgements for the named page finding task last year.

Priority

The ZETNP50W run employs a *priority* scheme, for which those documents are ranked higher where all query terms appear within the first 50 terms of each document. Again, linear interpolation was used between scores derived conventionally and those based on the priority scheme, using a ratio of 9:1. Unlike last year, this year we did not make use of anchor text for this run.

Results

The results for the named page finding task are shown in Table 4. Our results are somewhat disappointing: even the best run is worse than the mean MRR achieved by all systems and runs. There are two possible reasons for this: we used (more or less) the same approaches as for last year, which were aimed at an adhoc task, rather than designed for a named page finding task. However, comparing our runs, the priority scheme (run ZETNP50W) was by far the best, which is surprising, since it did not make any use of anchor text, which is typically a good source of document descriptors for named page finding tasks.

4 Robust Track

We submitted four title-only runs and one mandatory description-only run to the Robust track. Our runs were all automatic; details of the runs are described in the following sections, and summarised in Table 5.

Run	Technique employed	MRR	Found in top 10	No answer found
Mean		0.379	–	–
ZETNP	Plain	0.067	46.8%	23.8%
ZETNPANC	Anchortext	0.258	44.0%	**23.4%**
ZETNPANCFUZZ	Anchortext, Fuzzy Phrase	0.277	46.0%	**23.4%**
ZETNP50W	Priority	**0.318**	**48.4%**	23.8%

Table 4: *Effectiveness of terabyte named page finding runs, showing mean reciprocal rank (MRR), the percentage of queries for which an answer was found in the top 10 documents, and the percentage of queries for which no answer was found. The mean MRR result is calculated over the median MRR of all queries submitted by all participants.*

Run ID	Run type	Expansion parameters		Source of expansion terms
		$\|R\|$	$\|E\|$	
RMIT5D1025	description-only	10	25	AQUAINT
RMIT5T0000	title-only	–	–	–
RMIT5T1025	title-only	10	25	AQUAINT
RMIT5T0530N	title-only	5	30	AQUAINT, NW, REUTERS
RMIT5T1545TD	title-only	15	45	NW

Table 5: *Summary of robust runs.*

4.1 Adhoc Task

We used the local analysis query expansion method as proposed by Robertson and Walker (1999). Using the setup as described in Section 2, the initial query is ranked against the collection using the Okapi BM25 similarity measure, and the top R documents are identified (see below for variations to this method). All terms from these documents are extracted and each of them is assigned a *Term Selection Value*:

$$TSV_t = \left(\frac{f_t}{N}\right)^{r_t} \times \binom{|R|}{r_t}$$

where f_t is the number of documents in the collection that contain term t, N is the number of documents in the collection, $|R|$ is the size of the local set and r_t is the the number of documents in R the contain term t. Using this formula, $|E|$ terms with the lowest selection value are selected and appended to the query. Their weighting is determined using the Robertson/Sparck Jones weight (Robertson and Sparck Jones, 1976, Robertson and Walker, 1999):

$$w_t = \frac{1}{3} \times \log \frac{(r_t + 0.5)/(f_t - r_t + 0.5)}{(|R| - r_t + 0.5)/(N - f_t - |R| + r_t + 0.5)}$$

The value of $1/3$ was recommended by Robertson in unpublished correspondence, and is used to dampen the impact of expansion terms on the document ranking, when compared to that of the original query terms. (We confirmed this value as suitable in unpublished experiments.) Finally, the query is rerun against the collection.

Collection Selection

In preparation for our runs, we used several collections. One is the target collection for the robust track this year, AQUAINT. It was also used for the novelty track last year and has been described in this context (Soboroff and Harman, 2003). Another is the newswire collection used for the adhoc tracks for TREC 7 and 8 (Voorhees and Harman, 1999). It consists of the data provided on TREC disks 4 and 5, not including the congressional record. We will refer to this collection as NW. Finally, we used the REUTERS collection, which contains the full set of newswire articles from the Reuters news agency for a one year period from August 1996 (`http://www.reuters.com/researchandstandards/corpus/`).

In some runs, in particular RMIT5T0530N and RMIT5T1545TD, instead of ranking initial queries against the AQUAINT collection in order to identify a local set of documents, alternative collections were used: for RMIT5T0530N, in addition to the AQUAINT collection, we also used the adhoc collection from NW; for RMIT5T1545TD we used only the latter.

The underlying idea for the use of enlarged collections or larger external collections is that retrieval effectiveness – particularly in the high ranks – is greater if a larger collection is used (Hawking and Robertson, 2003). Since the local set consists of documents in the top ranks, and local analysis is based on the assumption that sourcing terms from relevant documents is useful, the more relevant documents that are included in the local set, the higher the quality of expansion terms. This approach has been used since TREC 6, see for example Walker et al. (1997) or Allan et al. (1997).

A danger with using additional collections is that query terms might appear in different contexts than in those provided in the target collection (Kraaij, 2004, Chapter 3). We therefore only experimented with newswire collections, which we assume to have a similar use of vocabulary. Although different topics are bound to be covered in different newswire collections, we hope that the mix of those might be helpful for retrieval.

Training of Query Expansion Parameters and Collections

In earlier work, we found that varying the parameters $|R|$ (the number of documents in the local set) and $|E|$ (the number of expansion terms added to the query) can have a marked impact on retrieval effectiveness (Billerbeck and Zobel, 2004). We therefore trained the local analysis parameters by way of a simple combinatorial search through the following space of parameters, using last year's robust track collection and queries as well as relevance judgements. We varied the number of expansion terms ($|E|$) added to the initial query through 2, 5, 10, 15, 20, 25, 30, 60, 75, and 90. $|R|$, the size of the initially retrieved documents, was varied through 1, 2, 5, 10, 15, and 20. Each resulting set of parameters was tested by sourcing expansion terms from the AQUAINT, NW, or REUTERS collections, as well as any combination of these collections. The parameter settings and collection choice that showed most promise were used for the official runs submitted to the TREC robust track for 2005.

Co-derivative Document Removal

The robust track, and the gMAP measure, are designed to emphasise reasonable worst-case performance on poor queries over more fragile strategies that are capable of extremely good best-case performance but offer few guarantees in the worst case. As such, it is of greater importance to return some relevant documents consistently than to return many relevant documents occasionally.

Our intuition is that documents that are co-derived – that is, documents that share a common heritage – are more likely to either both be relevant or both be irrelevant. Given this, it is plausible to hypothesise that removing documents from the result list if they are co-derived with an earlier document will increase diversity, and hence increase the probability that at least some relevant documents are present amongst the top n documents in the result list.

We used the DECO document fingerprinting software (Bernstein and Zobel, 2004, 2005) to identify pairs of documents that shared some proportion of text. We set the threshold to 0.15, meaning that a pair of documents had to share about 15% of their text before they were considered co-derivative.

The removal of co-derivative documents was done as a postprocessing step. `Zettair` was used to create a run with 2,000 results for each query. The run was then postprocessed so that any document that was co-derived with any document higher in the result list was removed. Removal of documents resulted in promotion of all documents lower in the list. Once this process was complete, the top 1,000 remaining documents for each query were selected to form the official run submitted.

Due to the submission limit we were not able to submit official runs for the same set of parameters with duplicates removed and not removed. However, upon release of the official relevance judgements for the robust track we evaluated the same run in which co-derivative documents had not been removed. No other parameters were tuned or changed.

Run	MAP	P@10	gMAP
RMIT5T1545TD	0.146	0.410	0.098
without removing co-derived documents	0.186	0.416	0.119

Table 6: *Comparison of official run with co-derivative documents removed, and a run where all documents are left in the result set. See Table 7 for a description of columns.*

Run ID	Run type	MAP	P@10	gMAP	Area
Median	description-only	0.184	0.386	0.103	1.226
RMIT5D1025		0.160	0.368	0.066	1.447
Median	title-only	0.224	0.434	0.129	1.425
RMIT5T0000		0.157	0.398	0.089	1.633
RMIT5T1025		**0.224**	**0.432**	0.117	2.487
RMIT5T0530N		0.218	0.424	**0.129**	2.118
RMIT5T1545TD		0.146	0.410	0.098	**0.946**

Table 7: *Summary of robust results, showing mean average precision (MAP), precision at 10 retrieved documents (P@10) and the geometric MAP (gMAP). Also shown are the areas between the best possible ordering of predicted topic difficulty, and the difficulty we predicted. The median figures are for all runs submitted to the robust track by all participants.*

The results are shown in Table 6. The removal of co-derivative documents from the results list was not successful in improving the average performance of the `zettair` search engine for this dataset. This means that the promotion of documents in the run was insufficient to compensate for the removal of relevant documents by the co-derivative removal process. This occurs, presumably, because – in an effectively functioning search-engine – pairs of co-derived documents near the top of the run are more likely to be relevant than those promoted from the bottom of the run. Thus, the net effect is a loss of effectiveness as measured. We observed the same phenomenon for the TREC 2004 terabyte track (Bernstein and Zobel, 2005), but it is interesting to note that it is still the case even for the gMAP measure.

Results

Since we experimented with different parameters and collection selections for different runs, together with novel approaches such as the duplicate detection, it is difficult to conclude from the five submitted runs alone which approaches were useful and which were not.

Results are shown in Table 7. Unsurprisingly the run without expansion (RMIT5T0000) performed worst among our title-only runs for all three effectiveness measures. As discussed previously, the general TREC effectiveness evaluation framework (excluding the novelty tracks of previous years) rewards runs where multiple copies of the same (or at least co-derived) relevant document is ranked. The run which removed near duplicate documents (RMIT5T1545TD) therefore performed reasonably badly.

Our best runs were those that made use of expansion. While the run that sourced terms from the AQUAINT collection only (RMIT5T1025) achieved slightly higher mean average precision and precision at 10 figures, the run which sourced expansion terms from a larger pool of documents (RMIT5T0530N) achieved a marginally better geometric mean average precision. Both runs show comparable effectiveness.

Although we have shown in previous work that query expansion does not increase robustness for a query set on average (Billerbeck and Zobel, 2004), it does seem to increase the effectiveness of those queries that achieve only low performance.

4.2 Topic Difficulty Prediction

One task of the robust track is topic difficulty prediction. In this task participants are asked to predict the difficulty of a run by producing a ranking from best to worst performing topic. Using the average precision

of each topic, the predicted topic difficulty ranking is then compared to the actual performance over the run.

Past submissions to the robust track have typically either considered query term related statistics, or have relied on the similarity measure between the query and results (Voorhees, 2004). It has been shown that documents within a collection have a non-uniform likelihood of retrieval (Singhal et al., 1996, Garcia et al., 2004). Such information can be used to construct prior probabilities of document access. Other information retrieval tasks have used prior evidence such as PageRank (Page et al., 1998) and document length to improve retrieval effectiveness (Craswell et al., 2005, Kraaij et al., 2002). However, to the best of our knowledge no-one has explored the value of document priors for topic difficulty prediction. The use of such values for prediction is appealing because, for each topic, there is minimal computational effort required. The document priors are pre-calculated at indexing time, and query term specific data is not required.

Proposed Techniques

Our approach to query difficulty prediction attempts to take advantage of the non-uniform likelihood of document access by a retrieval system. For each document in the collection, a probability is generated that represents the likelihood that the document will be retrieved by the search system. That is, for any given query we generate a probability of document access. Using these probabilities, an absolute ordering can be achieved from most likely to be accessed to least likely to be accessed. In a sense, this ordering represents a form of default ranking for documents in the collection. We label this ranked set of probabilities the *retrieval-likelihood* ordering.

By comparing the absolute ordering of documents based on retrieval-likelihood to the order of the those documents returned in a topic result set, we can estimate how difficult a topic was to answer. The rationale is that those queries that produce ranked result sets that contain documents in much the same order as the global ordering do not have strong discriminating power. As such, these queries are considered to be difficult to answer. Conversely, queries that produce ranked result sets that are mostly ordered in a different manner to the global ordering are considered simpler to answer by the system.

Two aspects of importance to our approach are the generation of the retrieval-likelihood values, and the method used to compare the ordering of documents by retrieval-likelihood to that of the ranked result set.

Given a collection and a query log for that collection, access-counts can be used to measure the skew of document access by a search engine (Garcia et al., 2004). To generate our retrieval-likelihood probabilities we used an approach similar to that of access counting, but instead of processing a complete query log over a document collection, we processed a single query that was the amalgamation of every distinct term in the collection. Using the Okapi similarity metric (see Section 3.3), we generated a rank for every document in the collection based on its similarity to the current query. These ranks were then used as an indication of which documents in the collection are most likely to be retrieved. The advantage of this approach is that a result for every document in the collection can be generated with a single pass over the index. For comparison purposes, we also submitted a run using document length as the basis for our probabilities with larger documents being more likely to be retrieved. The length bias of search metrics was noted by Singhal et al. (1996).

To compare the order of the ranked results with the retrieval-likelihood ordering we applied three techniques. The first was to use Kendall's τ to measure the disagreement between the two orderings directly (Stuart, 1983). The second was to use the average rank of documents in the result sets, and the third was to use product of document priors. Due to the track guidelines we were only able to submit one prediction set with each submitted run. This limitation prevents the results from being directly comparable, as each submitted run uses a different query evaluation technique (see Section 4.1) and a different difficulty prediction technique.

To compare the different techniques outlined above, we submitted the following five runs:

- **as-k-1000**: This run uses Kendall's τ over the full 1,000 results per query, with the global ordering based on our access-count variant for the retrieval-likelihood of the documents. A low τ score marked disagreement between the ranked result set and the global ordering, therefore the query is considered to be simple to answer.

	AS-K-1000	AS-K-50	AS-P-1000	AS-A-1000	DL-K-1000
RMIT5D1025	1.447†	1.683	**1.374**	1.382	1.379
RMIT5T0000	**0.843**	1.633†	1.668	1.631	0.907
RMIT5T0530N	**1.210**	1.796	2.118†	2.149	1.307
RMIT5T1025	**1.566**	1.891	2.478	2.488†	1.732
RMIT5T1545TD	**0.917**	1.339	1.203	1.214	0.946†

Table 8: *Combinations of all runs and query difficulty prediction measures used. The five predictions that were submitted to TREC are marked with a* †.

- **as-k-50**: As in the previous run, but here we only consider the top 50 ranked results per query for the τ disagreement. This variant is based on the assumption that in a collection of this size it is unlikely that all 1,000 retrieved results will be highly relevant to the query, therefore they should not be considered in the difficulty prediction.

- **as-p-1000**: This run uses the product of the probabilities of documents in the result set. Our access-count variant is used for the retrieval-likelihood probabilities. High products suggest a larger range of regularly accessed documents, therefore the query is difficult to answer. Although simplistic in design, this scheme should indicate if the number of frequently retrieved documents per query suggests topic difficulty.

- **as-a-1000**: It is possible that in the product of probabilities approach, a few low probability documents can significantly skew the query difficulty estimate. An alternate way of considering the amount of commonality among the per-query results is to take the average of the global ranks of each document in the result set (that is, instead of considering actual prior probabilities, the prior rank position of each document is used). Similar to the the product technique, a higher average rank suggests a difficult query.

- **dl-k-1000**: The previous four techniques utilise priors based on access-counts. In this run we employ a technique that uses document length for priors. This approach uses Kendall's τ over the full 1,000 results per query in a similar manner to the as-k-1000 approach.

Results

Table 8 shows the area-under the curve results for variant runs. A lower area under the curve value signifies a stronger correlation between the prediction and the actual results. Values marked with a † denote the variants that were submitted with each of the five submitted runs. To enable a clearer comparison between the techniques, the remaining entries in the table have been post-calculated using the relevance judgements.

The AS-K-1000 method based on access-count document-likelihoods and Kendall's τ was the most accurate at predicting query difficulty over the variant runs with the exception of the description only run. The DL-K-1000 based on document length and Kendall's τ produced similar results to AS-K-1000, but was generally outperformed by the access-count priors.

The AS-K-50 technique, which is based only on the top 50 results per query, was unable to discriminate as effectively as using all results per query. This suggests that all documents, both likely and unlikely to be relevant, help determine the difficulty of the topic using our document-likelihood techniques.

The AS-P-1000 product of priors, and AS-A-1000 average rank techniques that were hoped to show that the fraction of frequently retrieved documents can indicate query performance, varied in effect producing only one relatively effective run of the five submissions.

While not as sophisticated as techniques such as query clarity (Cronen-Townsend et al., 2002), which have been successfully applied by Amati et al. (2004) in a similar context, the results of these predictions show that document priors are a factor that can be considered in predicting the performance of a retrieval system.

It is promising that these results do provide some indication of query difficulty given that only the query results are considered when making the prediction and no use is made of query term information. It is our

hope that, in combination with more advanced techniques, these approaches can help improve query difficulty prediction.

5 Conclusion

In this year's terabyte efficiency task, we confirmed the strong RMIT tradition of efficient systems while retaining good effectiveness. The adhoc task was disappointing, with the lack of stemming giving us poor effectiveness results, but still served to validate our new accumulator limiting scheme. The bulk of our effort in the named page task was invested in co-ordination of the task.

In the robust track, we confirmed what has become clear from last year's TREC: query expansion is essential in order to increase the robustness of badly performing queries. It also seems that using document priors based on document access counts is a promising technique for predicting query difficulty, particularly when used in conjunction with other methods, as we are planning to do in future.

Acknowledgements

This work is supported by the Australian Research Council. Hardware for some experiments was provided with the support of an RMIT University VRII grant.

References

Allan, J., Callan, J., Croft, W. B., Ballesteros, L., Byrd, D., Swan, R. and Xu, J. (1997), Okapi at TREC-6 automatic ad hoc, VLC, routing, filtering and QSDR, *in* E. M. Voorhees and D. K. Harman, eds, "Proc. Text Retrieval Conf. (TREC)", National Institute of Standards and Technology Special Publication 500-240, Gaithersburg, MD, pp. 169–206.

Amati, G., Carpineto, C. and Romano, G. (2004), Query difficulty, robustness, and selective application of query expansion, *in* S. McDonald and J. Tait, eds, "European Conf. on IR Research", Vol. 2997 of *Lecture Notes in Computer Science*, Springer, pp. 127–137.

Bernstein, Y. and Zobel, J. (2004), A scalable system for identifying co-derivative documents, *in* A. Apostolico and M. Melucci, eds, "Proc. String Processing and Information Retrieval Symp.", Springer-Verlag, Padova, Italy, pp. 55–67.

Bernstein, Y. and Zobel, J. (2005), Redundant documents and search effectiveness, *in* "Proc. Int. Conf. on Information and Knowledge Management". To appear.

Billerbeck, B., Cannane, A., Chatteraj, A., Lester, N., Webber, W., Williams, H. E., Yiannis, J. and Zobel, J. (2004), RMIT University at TREC 2004, *in* E. M. Voorhees and L. P. Buckland, eds, "Proc. Text Retrieval Conf. (TREC)", National Institute of Standards and Technology Special Publication 500-261, Gaithersburg, MD.

Billerbeck, B. and Zobel, J. (2004), Questioning query expansion: An examination of behaviour and parameters, *in* K.-D. Schewe and H. E. Williams, eds, "Proc. Australasian Database Conf.", Vol. 27, Australian Computer Society, Dunedin, New Zealand, pp. 69–76.

Craswell, N., Robertson, S., Zaragoza, H. and Taylor, M. (2005), Relevance weighting for query independent evidence, *in* A. Moffat, G. Marchionini, J. Tate, R. Baeza-Yates and N. Ziviani, eds, "Proc. ACM-SIGIR Int. Conf. on Research and Development in Information Retrieval", ACM Press, New York, Salvador, Brazil, pp. 416–423.

Cronen-Townsend, S., Zhou, Y. and Croft, W. B. (2002), Predicting query performance, *in* M. Beaulieu, R. Baeza-Yates, S. H. Myaeng and K. Järvelin, eds, "Proc. ACM-SIGIR Int. Conf. on Research and Development in Information Retrieval", ACM Press, New York, Tampere, Finland, pp. 299–306.

Garcia, S., Williams, H. E. and Cannane, A. (2004), Access-ordered indexes, *in* V. Estivill-Castro, ed., "Proceedings of the 27th Conference on Australasian Computer Science", Vol. 26, Australian Computer Society, Dunedin, New Zealand, pp. 7–14.

Hawking, D. and Robertson, S. E. (2003), "On collection size and retrieval effectiveness", *Kluwer International Journal of Information Retrieval* **6**(1), 99–150.

Hawking, D., Upstill, T. and Craswell, N. (2004), Toward better weighting of anchors, *in* M. Sanderson, K. Järveln, J. Allan and P. Bruza, eds, "Proc. ACM-SIGIR Int. Conf. on Research and Development in Information Retrieval", ACM Press, New York, Sheffield, UK, pp. 512–513.

Kraaij, W. (2004), Variations on Language Modeling for Information Retrieval, PhD thesis, University of Twente.

Kraaij, W., Westerveld, T. and Hiemstra, D. (2002), The importance of prior probabilities for entry page search, *in* M. Beaulieu, R. Baeza-Yates, S. H. Myaeng and K. Järvelin, eds, "Proc. ACM-SIGIR Int. Conf. on Research and Development in Information Retrieval", ACM Press, New York, Tampere, Finland, pp. 27–34.

Lester, N., Moffat, A., Webber, W. and Zobel, J. (to appear), Space-limited ranked query evaluation using adaptive pruning, *in* "Proc. of the Int. Conf. on Web Information Systems Engineering".

Moffat, A. and Zobel, J. (1996), "Self-indexing inverted files for fast text retrieval", *ACM Transactions on Information Systems* **14**(4), 349–379.

Page, L., Brin, S., Motwani, R. and Winograd, T. (1998), The PageRank citation ranking: Bringing order to the web, Technical report, Stanford Digital Library Technologies Project.

Robertson, S. E. and Sparck Jones, K. (1976), "Relevance weighting of search terms", *Jour. of the American Society for Information Science* **27**(3), 129–146.

Robertson, S. E. and Walker, S. (1999), Okapi/Keenbow at TREC-8, *in* E. M. Voorhees and D. K. Harman, eds, "Proc. Text Retrieval Conf. (TREC)", National Institute of Standards and Technology Special Publication 500-246, Gaithersburg, MD, pp. 151–161.

Singhal, A., Salton, G., Mitra, M. and Buckley, C. (1996), "Document length normalization", *Information Processing & Management* **32**(5), 619–633.

Soboroff, I. and Harman, D. K. (2003), Overview of the TREC 2003 novelty track, *in* E. M. Voorhees and L. P. Buckland, eds, "Proc. Text Retrieval Conf. (TREC)", National Institute of Standards and Technology Special Publication 500-255, Gaithersburg, MD, pp. 38–53.

Sparck Jones, K., Walker, S. and Robertson, S. E. (2000), "A probabilistic model of information retrieval: Development and comparative experiments. Parts 1&2", *Information Processing & Management* **36**(6), 779–840.

Stuart, A. (1983), "Kendall's tau", *Encyclopedia of Statistical Sciences* **4**, 367–369.

Voorhees, E. M. (2004), Overview of the TREC 2004 robust track, *in* E. M. Voorhees and L. P. Buckland, eds, "Proc. Text Retrieval Conf. (TREC)", National Institute of Standards and Technology Special Publication 500-261, Gaithersburg, MD.

Voorhees, E. M. and Harman, D. K. (1999), Overview of the eighth Text REtrieval Conference (TREC-8), *in* E. M. Voorhees and D. K. Harman, eds, "Proc. Text Retrieval Conf. (TREC)", National Institute of Standards and Technology Special Publication 500-246, Gaithersburg, MD, pp. 1–23.

Walker, S., Robertson, S. E., Boughanem, M., Jones, G. J. F. and Sparck Jones, K. (1997), Okapi at TREC-6 automatic ad hoc, VLC, routing, filtering and QSDR, *in* E. M. Voorhees and D. K. Harman, eds, "Proc. Text Retrieval Conf. (TREC)", National Institute of Standards and Technology Special Publication 500-240, Gaithersburg, MD, pp. 125–136.

Zhai, C. and Lafferty, J. (2004), "A study of smoothing methods for language models applied to information retrieval", *ACM Transactions on Information Systems* **22**(2), 179–214.