Techniques for Efficient Query Expansion

Bodo Billerbeck and Justin Zobel

School of Computer Science and Information Technology,
RMIT University, Melbourne, Australia
{bodob,jz}@cs.rmit.edu.au

Abstract. Query expansion is a well-known method for improving average effectiveness in information retrieval. However, the most effective query expansion methods rely on costly retrieval and processing of feedback documents. We explore alternative methods for reducing query-evaluation costs, and propose a new method based on keeping a brief summary of each document in memory. This method allows query expansion to proceed three times faster than previously, while approximating the effectiveness of standard expansion.

1 Introduction

Standard ranking techniques in information retrieval return documents that contain the same terms as the query. While the insistence on exact vocabulary matching is often effective, identification of some relevant documents involves finding alternative query terms. Previous work has shown that through query expansion (QE) effectiveness is often significantly improved (Rocchio, 1971, Robertson and Walker, 1999, Carpineto et al., 2001).

Local analysis has been found to be one of the most effective methods for expanding queries (Xu and Croft, 2000). For those methods the original query is used to determine top-ranked documents from which expansion terms are subsequently extracted. A major drawback of such methods is the need to retrieve those documents during query evaluation, greatly increasing costs. In other work (Billerbeck et al., 2003), we explored the use of surrogates built from past queries as a cheap source of expansion terms, but such surrogates require large query logs to be usable.

In this paper, we identify the factors that contribute to the cost of query expansion, and explore in principle the alternatives for reducing these costs. Many of these approaches compromise effectiveness so severely that they are not of practical benefit. However, one approach is consistently effective: use of brief summaries – a pool of the most important terms – of each document. These *surrogates* are much smaller than the source documents, and can be rapidly processed during expansion. In experiments with several test sets, we show that our approach reduces the time needed to expand and evaluate a query by a factor of three, while approximately maintaining effectiveness compared to standard QE.

2 Background

Relevance feedback is used to refine a query using knowledge of whether documents retrieved by this query are relevant. Weighted terms from judged documents are added to the original query, where they act as positive and negative examples of the terms that should occur in relevant and non-relevant documents. The modified query is then reissued, in the hope of ranking the remaining relevant documents more highly (Rocchio, 1971, Ruthven and Lalmas, 2003). Interactive QE can significantly increase effectiveness (Magennis and van Rijsbergen, 1997), although on average – for non expert users – automatic expansion is more likely to lead to better performance (Ruthven, 2003).

In automatic QE, also called pseudo relevance feedback, the query is augmented with expansion terms from highly-ranked documents (Robertson and Walker, 1999). An alternative (Qiu and Frei, 1993, Gauch and Wang, 1997) is to examine the document collection ahead of time and construct similarity thesauri to be accessed at query time. The use of thesauri in general has been shown to be less successful than automatic QE (Mandala et al., 1999), though the two approaches can be successfully combined (Xu and Croft, 2000).

An effective method for QE, used throughout this paper, is based on the Okapi BM25 measure (Robertson and Walker, 1999, Robertson et al., 1992). Slightly modified, this measure is as follows:

$$bm25(q,d) = \sum_{t \in q} \log\left(\frac{N - f_t + 0.5}{f_t + 0.5}\right) \times \frac{(k_1+1) f_{d,t}}{K + f_{d,t}}$$

where terms t appear in query q; the collection contains N documents d; f_t documents contain a particular term and a particular document contains a particular term $f_{d,t}$ times; K is $k_1((1-b)+b\times L_d/AL)$; constants k_1 and b respectively are set to 1.2 and 0.75; and L_d and AL are measurements in a suitable unit for the document length and average document length respectively. The modifications to the original formulation (see Sparck-Jones et al. (2000) for a detailed explanation) is the omission of a component that deals with repeated query terms. In the queries we use, term repetitions are rare.

In this paper we use the expansion method proposed by Robertson and Walker (1999) where E terms with the lowest *term selection value* are chosen from the top R ranked documents:

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

where a term t is contained in r_t of the top ranked R documents. The expansion terms get added to the original query, but instead of using their Okapi value, their weight (Robertson and Walker, 1999) is chosen by the formula:[1]

$$\frac{1}{3} \times \log\left(\frac{(r_t + 0.5)/(R - r_t + 0.5)}{(f_t - r_t + 0.5)/(N - f_t - R + r_t + 0.5)}\right)$$

[1] The factor of $\frac{1}{3}$ was recommended by unpublished correspondence with the authors. It de-emphasises expansion terms and prevents query drift, that is, "alteration of the focus of a search topic caused by improper expansion" (Mitra et al., 1998). We confirmed in unpublished experiments that the value of the factor is suitable.

We have shown previously that best choices of R and E depend on the collection used and should in principle be carefully optimised (Billerbeck and Zobel, 2004); to reduce the complexity of the experiments, in this paper we use the standard values of $R = 10$ and $E = 25$.

Although there has been a great deal of research on efficient evaluation of ranked queries (Witten et al., 1999, pages 207–210), there is no prior work on efficient QE for text retrieval, the focus of this paper.

3 Query expansion practicalities

In most expansion methods making use of local analysis, there are five key stages. First, the original query is used to rank an initial set of documents. This set is then retrieved from disk and all terms are extracted from those documents. Terms are evaluated and ranked in order of their potential contribution to the query. The top ranked terms are appended to the query, and finally the reformulated query is reissued and a final set of documents is ranked.

Each phase of the ranking process has scope for efficiency gains, but some of the gains involve heuristics that can compromise effectiveness. In this section we explore these options; this exploration provides a focus for the experiments reported later in this paper. Some of the concepts introduced here – in particular, associations and surrogates – are described in more detail in the next section.

Initial ranking. During the first stage, documents are ranked according to the original query. For each query term the inverted list is retrieved, if it hasn't been cached, and processed. For each document referenced in the list, a score is calculated and added to a list of scores that is kept for (say) 20,000 documents (Moffat and Zobel, 1996). Once all query terms have been processed, the top R documents are used for the next stage.

The cost of accessing an inverted list depends on the disk access time. For a long list, the costs are directly proportional to list size. If the list is organised by document identifier, the whole list must be fetched for each query term.

A way of reducing the cost of retrieving and processing the inverted lists is to cut down the volume of list information that has to be retrieved. This has been achieved by, for example, Anh and Moffat (2002), where documents are not stored in the order they are encountered during indexing, but in order of the *impact* a term has in a particular document. For instance, a term has more impact in a document in which it occurs twice, than another of the same length in which it occurs once. Using this ordering means that either the processing of lists can be stopped once a threshold is reached, or that the lists are capped to begin with, leading to lower storage requirements, reduced seek times, and allowing more lists to be cached in memory. We have not used impacts in our experiments, but the gains that they provide are likely to be in addition to the gains that we achieve with our methods.

Another way to reduce list length, discussed in more detail later, is to index only a fraction of the document collection for the initial ranking. Initial ranking is traditionally on the document collection, but there is no particular reason why other collections should not be used. Another option, also explored later, of this

kind is to use document surrogates. A drawback of these approaches is that the full index still needs to be available for the final ranking and thus is loaded at the same time as auxiliary indexes. This means that some of the advantage of using shorter lists is negated by having less space available to cache them.

Fetching documents. Having identified the highly ranked documents, these need to be fetched. In the vast majority of cases these documents are not cached from a previous expansion or retrieval process (assuming a typical memory size), and therefore have to be fetched from disk, at a delay of a few milliseconds each.

Traditionally, full-text documents are fetched. This is the most expensive stage of expansion and therefore the area where the greatest gains are available. We have shown previously that surrogates – which are a fraction of the size of the documents – can be more effective than full-text documents (Billerbeck et al., 2003). Using surrogates such as query associations is more efficient, provided that those surrogates can be precomputed, as discussed later.

Another approach is limiting the number of documents available for extraction of terms, which should result in higher efficiency, due to reduced cache misses when retrieving the remaining documents and otherwise smaller seek times as it can be expected that the limited number of documents are clustered on disk. Documents could be chosen by, for example, discarding those that are the least often accessed over a large number of queries (Garcia et al., 2004).

A more radical measure is to use in-memory document surrogates that provide a sufficiently large pool of expansion terms, as described in the following section. If such a collection can be made sufficiently small, the total cost of expansion can be greatly reduced. Typically full text document collections don't fit into main memory, but well-constructed surrogates may be only a small fraction of the size of the original collection. Our surrogates are designed to be as small as possible while maintaining effectiveness.

Extracting candidate terms. Next, *candidate terms* (that is, potential expansion terms) are extracted from the fetched documents. These documents need to be parsed, and terms need to be stopped. (We do not use stemming, since in unpublished experiments we have found that stemming does not make a significant difference to effectiveness.)

This phase largely depends on the previous phase; if full text documents have been fetched, these need to be parsed and terms need to be stopped. In the case of query associations, the surrogates are pre-parsed and pre-stopped and extraction is therefore much more efficient.

The in-memory surrogates we propose can be based on pointers rather than the full terms in memory. The pointers reference terms in the dictionary used for finding and identifying statistics and inverted lists. They have a constant size (4 bytes) and are typically smaller than a vocabulary term. This approach also eliminates the lookups needed in the next stage.

Selecting expansion terms. The information (such as the inverse document frequency) necessary for calculation of a term's TSV is held in the vocabulary, which may be held on disk or (as in our implementation) in memory; even when held on disk, the frequency of access to the vocabulary means that typically

much of it is cached. As a result, this phase is the fastest and can only be sped up by providing fewer candidate terms for selection.

Query associations typically consist of 20–50 terms, as opposed to the average of 200 or more for web documents. Use of surrogates could make this stage several times more efficient than the standard approach. Surrogates are a strict subset of full text documents, and usually are a tiny fraction thereof, ensuring that selection is efficient.

Final ranking. Finally the document collection is ranked against the reformulated query. Similar considerations as in the first phase are applicable here. We have shown previously (Billerbeck et al., 2003) that final ranking against surrogates is, unsurprisingly, ineffective. The only option for efficiency gains at this stage is to use an approach such as impact-ordering, as discussed earlier.

4 Methods of increasing efficiency for QE

In the previous section we identified costs and plausible approaches for reducing them. In this section, we consider the most promising methods in more detail, setting a framework for experiments. In particular, we propose the novel strategy of using bag-of-word summaries as a source of expansion terms.

Query associations. Query associations (Scholer and Williams, 2002) capture the topic of a document by associating past user queries with the documents that have been highly ranked by that query. We have previously shown (Billerbeck et al., 2003) that associations are effective when useful query logs are available. A disadvantage of using associations is that an extra index needs to be loaded and referenced during query evaluation. However, this penalty is small, as associations are likely to be a small fraction of collection size. The advantages are that associations are usually pre-stemmed and stopped, stored in a parsed form, and cheap to retrieve.

Rather than indexing the associations, it would be possible in principle to rank using the standard index, then fetch and expand from the associations, but in our earlier work (Billerbeck et al., 2003) we found that it was necessary to rank against the associations themselves.

Reducing collection size for sourcing expansion terms. The intuition underlying expansion is that, in a large collection, there should be multiple documents on the same topic as the query, and that these should have other pertinent terms. However, there is no logical reason why the whole collection should have to be accessed to identify such documents. Plausibly, documents sampled at random from the collection should represent the overall collection in respect of the terminology used. In our experiments, we sampled the collection by choosing every nth document, for n of 2 and 4. Other options would be to use centroid clusters or other forms of representative chosen on the basis of semantics. Documents could also be stored in a pre-parsed format (such as a forward index), which we have not tested.

In-memory document summaries. The major bottleneck of local analysis is the reliance on the highly ranked documents for useful expansion terms. These documents typically need to be retrieved from disk. We propose that summaries of all documents be kept in memory, or in a small auxiliary database that is likely to remain cached. A wide range of document summarisation techniques have been investigated (Goldstein et al., 1999), and in particular Lam-Adesina and Jones (2001) have used summarisation for QE. In this work, representative sentences are selected, giving an abbreviated human-readable document.

However, summaries to be used for QE are not for human consumption. We propose instead that the summaries consist of the terms with the highest $tf.idf$ values, that is, the terms that the expansion process should rank highest as candidates if given the whole document. To choose terms, we use the function:

$$tf.idf = \log\left(\frac{N}{f_t}\right) \times \log\left(1 + f_{d,t}\right)$$

where N is the number of documents in the collection, f_t of which contain term t, and $f_{d,t}$ is the number of occurrences of t in document d.

Given these values, we can then build summaries in two ways. One is to have a fixed number S of highly-ranked terms per document. The other is to choose a global threshold C, in which case each summary consists of all the document terms whose $tf.idf$ value exceeds C. Instead of representing summaries as sequences of terms, it is straightforward to instead use lists of pointers to the vocabulary representation of the term, reducing storage costs and providing rapid access to any statistics needed for the TSV. During querying, all terms in the surrogates that have been ranked against the original query are then used for selection. This not only avoids long disk I/Os, but also the original documents – typically stored only in their raw form – do not need to be parsed. S or C can be chosen depending on collection size or available memory.

Although it is likely that query-biased summaries (Tombros and Sanderson, 1998) – as provided in most contemporary web search engines – would be more effective (Lam-Adesina and Jones, 2001), such a method cannot be applied in the context of efficient QE, as query-biased summaries cannot be precomputed.

Other approaches. Since the original query terms effectively get processed twice during the ranking process, it seems logical to only process the original query terms during the initial ranking, and then, later, process the expansion terms without clearing the accumulator table that was used for the initial ranking.

However, as explored previously (Moffat and Zobel, 1996), limiting the number of accumulators aids efficiency and effectiveness. To support this strategy, query terms must be sorted by their inverse document frequency before the query is processed. Because most expansion terms have a high inverse document frequency – that is, they appear in few documents and are relatively rare – it is important that they be processed before most of the original query terms, which typically have lower values. (The effect is similar – albeit weaker – to that of impact ordered indexes as discussed previously.) This means that the original query must be processed again with the expansion terms for final ranking. Intuition suggests that this argument is incorrect, and the original query terms should be allowed to choose the documents; however, in preliminary experiments we found

that it was essential to process the original terms a second time. Processing only expansion terms in the second phase reduced costs, but led to poor effectiveness.

Other strategies could also lead to reduced costs. Only some documents, perhaps chosen by frequency of access (Garcia et al., 2004) or sampling, might be included in the set of surrogates. A second tier of surrogates could be stored on disk, for retrieval in cases where the highly-ranked documents are not amongst those selected by sampling. Any strategy could be further improved by compressing the in-memory surrogates, for example with d-gapping (Witten et al., 1999, page 115) and a variable-byte compression scheme (Scholer et al., 2002).

Note that our summaries have no contextual or structural information, and therefore cannot be used – without major modifications – in conjunction with methods using such information, such as the local context analysis method of Xu and Croft (2000) or the summarisation method of Goldstein et al. (1999).

5 Experiments

Evaluating these approaches to QE requires that we test whether the heuristics degrade effectiveness, and whether they lead to reduced query evaluation time. To ensure that the time measurements were realistic, we used Lucy[2] as the underlying search engine.

The test data is drawn from the TREC conferences (Harman, 1995). We used two collections. The first was of newswire data, from TREC 7 and 8. The second was the WT10g collection, consisting of 10 gigabytes of web data crawled in 1997 (Bailey et al., 2003) for TREC 9 and 10. Each of these collections has two sets of 50 topics and accompanying relevance judgements. As queries, we used the title field from each TREC topic. We use the Wilcoxon signed rank test to evaluate the significance of the effectiveness results (Zobel, 1998).

For timings, we used 10,000 stopped queries taken from two query logs collected for the Excite search engine (Spink et al., 2002); these are web queries and thus are suitable for the WT10g runs. Since we were not able to obtain appropriate query logs for the newswire data, we used the same 10,000 queries for this collection. The machine used for our timings is a dual Intel Pentium III 866 MHz with 768 MB of main memory running Fedora Core 1.

Results

We used the TREC 8 and TREC 10 query sets to explore the methods. Results for this exploration are shown in Table 1. We applied the best methods found in Table 1 to the TREC 7 and TREC 9 query sets, as shown in Table 2. The tables detail the collection, the method of expansion, average precision, precision at 10, and r-precision values, as well as auxiliary memory required. A second index is needed for the runs where associations or fractional collections are used for initial ranking and candidate term extraction.

[2] Lucy/Zettair is an open source search engine being developed at RMIT by the Search Engine Group. The primary aim in developing Lucy is to test techniques for efficient information retrieval. Lucy is available from `http://www.seg.rmit.edu.au/`

Table 1. *Performance of expansion techniques of TREC queries on the TREC newswire and WT10g collections, for TREC 8 and TREC 10 queries. Effectiveness results shown are average precision (AvP), precision at 10 (P@10), and R-Precision (R-P). Also shown is the average query time over 10,000 queries and the amount of overhead memory required for each method; "index" marks the need to refer to an auxiliary index during expansion. A † marks results that are significantly different to the baseline of no expansion at the 0.10 level, and ‡ at the level of 0.05.* S *is the number of summary terms used, and* C *specifies the cutoff threshold for the selection value.*

TREC	Expansion Method	Time (ms)	AvP	P@10	R-P	Mem (MB)
8	None	23	0.221	0.442	0.260	n/a
8	Standard	211	0.247‡	0.466	0.288‡	n/a
8	Assoc.	179	0.219	0.400	0.263	index
8	Half1	201	0.241‡	0.436	0.283‡	index
8	Half2	185	0.235†	0.430	0.275‡	index
8	Quarter1	167	0.221	0.382‡	0.255	index
8	Quarter2	183	0.237	0.430	0.278	index
8	Quarter3	175	0.220	0.434	0.268	index
8	Quarter4	174	0.218	0.390‡	0.273	index
8	$S = 1$	46	0.231‡	0.446	0.267‡	6
8	$S = 10$	54	0.238‡	0.438	0.271‡	24
8	$S = 25$	59	0.244‡	0.456	0.277‡	54
8	$S = 40$	61	0.245‡	0.452	0.275‡	83
8	$S = 50$	64	0.243‡	0.454	0.281‡	102
8	$S = 100$	72	0.240‡	0.450	0.282‡	183
8	$C = 1.0$	58	0.243‡	0.448	0.280‡	56
10	None	62	0.163	0.290	0.190	n/a
10	Standard	615	0.180	0.288	0.202	n/a
10	Assoc.	835	0.180	0.272†	0.209	index
10	$S = 1$	139	0.138	0.218	0.150	19
10	$S = 10$	177	0.153†	0.227	0.169	76
10	$S = 25$	202	0.156	0.224	0.170	166
10	$S = 28$	204	0.185	0.308	0.217†	183
10	$S = 50$	221	0.156	0.224	0.170	296
10	$S = 100$	245	0.156	0.224	0.170	296
10	$C = 1.0$	217	0.185‡	0.312†	0.213†	190

For TREC 8 and to a lesser extent TREC 10, standard QE improves over the baseline, but in both cases query evaluation takes around nine times as long. Several of the methods proposed do not succeed in our aims. Associations take as long as standard QE, and effectiveness is reduced. For TREC 8 the surrogates are arguably inappropriate, as the web queries may not be pertinent to the newswire data; however, this issue highlights the fact that without a query log associations cannot be used.

Using halves ($n = 2$) or quarters ($n = 4$) of the collection also reduces effectiveness, and has little impact on expansion time; this is due to the need to load and access a second index. Larger n led to smaller improvements in QE; in experiments with $n = 8$, not reported here, QE gave no improvements. Reducing R to roughly a quarter of its original size in order to cater for a smaller number of

Table 2. *As in Table 1, but showing results only for the methods that worked best on TREC 8 and TREC 10.*

TREC	Expansion Method	Time (ms)	AvP	P@10	R-P	Mem (MB)
7	None	23	0.191	0.456	0.248	n/a
7	Standard	211	0.232‡	0.452	0.286‡	n/a
7	$S = 40$	61	0.220‡	0.426†	0.279‡	83
7	$C = 1.0$	58	0.215‡	0.426†	0.272‡	56
9	None	62	0.193	0.267	0.223	n/a
9	Standard	615	0.177	0.260	0.200	n/a
9	$S = 28$	204	0.161	0.269	0.176	183
9	$C = 1.0$	217	0.162	0.256	0.169‡	190

relevant documents – as intuition might suggest – only further degrades results. This is consistent with previous work which shows that retrievel effectiveness especially in the top ranked documents is greater for larger collections than sub-collections (Hawking and Robertson, 2003) which means that there is a higher likelyhood of sourcing expansion terms from relevant documents when using local analysis QE. It was also found that QE works best when expansion terms are sourced from collections that are a superset of documents of the one targeted (Kwok and Chan, 1998).

However, our simple $tf.idf$ summaries work well. Even one-word ($S = 1$) summaries yield significantly improved average precision on TREC 8, for a memory overhead of a few megabytes. The best cases were $S = 40$ on TREC 8 and $S = 28$ on TREC 10, where processing costs were only a third those of standard QE. These gains are similar to those achieved by (Lam-Adesina and Jones, 2001) with summaries of 6–9 sentences each, but our summaries are considerably more compact, showing the advantage of a form of summary intended only for QE. While the memory overheads are non-trivial – over 180 megabytes for TREC 10 – they are well within the capacity of a small desktop machine.

Results on TREC 7 for the summaries are equally satisfactory, with good effectiveness and low overheads. Results on TREC 9 are, however, disappointing. We had already discovered that expansion on TREC 9 does not improve effectiveness (Billerbeck and Zobel, 2004); our results here are, in that light, unsurprising. The principal observation is that QE based on summaries is still of similar effectiveness to that based on full documents.

We show only one value for the cutoff threshold, $C = 1.0$. This leads to the same effectiveness for similar memory overhead. Summaries and choice of S and C are further examined in Figures 1 and 2 for newswire and web data respectively. These show that a wide range of S values (left figure) and C values (right figure) lead to improved effectiveness, in some cases exceeding that of standard QE.

6 Conclusions

We have identified the main costs of query expansion and, for each stage of the query evaluation process, considered options for reducing costs. Guided by pre-

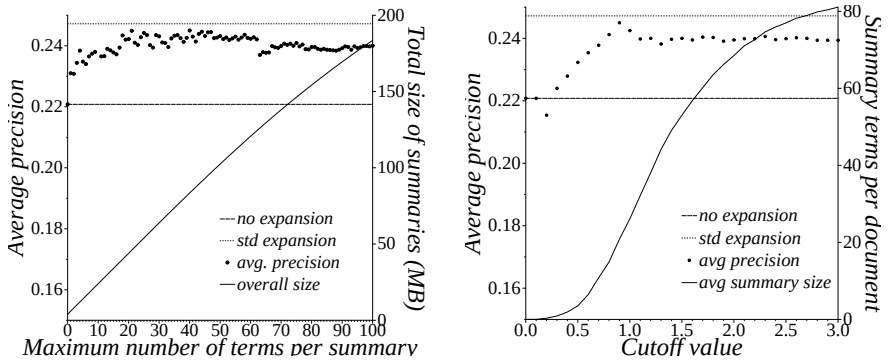

Fig. 1. *Varying average precision and associated memory cost with the number and cutoff value of summary terms respectively. Using the TREC 8 collection and queries.*

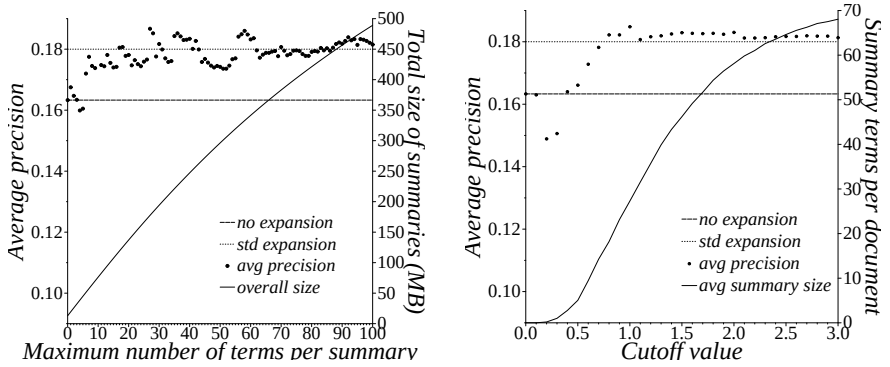

Fig. 2. *As in previous figure, but using the TREC 10 collection and queries.*

liminary experiments, we explored two options in detail: expansion via reduced-size collections and expansion via document surrogates. Two forms of surrogates were considered: query associations, consisting of queries for which each document was highly ranked, and $tf.idf$ summaries.

The most successful method was the $tf.idf$ summaries. These are much smaller than the original collections, yet are able to provide effectiveness close to that of standard QE. The size reduction and simple representation means that they can be rapidly processed. Of the two methods for building summaries, slightly better performance was obtained with those consisting of terms whose selection value exceeded a global threshold. The key to the success of this method is that it eliminates several costs: there is no need to fetch documents after the initial phase of list processing, and selection and extraction of candidate terms is trivial.

Many of the methods we explored were unsuccessful. Associations can yield good effectiveness if a log is available, but are expensive to process. Reduced-size collections yielded no benefits; it is possible that choosing documents on a more principled basis would lead to different effectiveness outcomes, but the costs are unlikely to be reduced. Streamlining list processing by carrying accumulator

information from one stage to the next led to a collapse in effectiveness. Our $tf.idf$ summaries, in contrast, maintain the effectiveness of QE while reducing time by a factor of three.

Acknowledgements

This research is supported by the Australian Research Council and by the State Government of Victoria. Thanks to Nick Lester and William Webber from the SEG group for their help with Lucy. Thanks also to Falk Scholer for letting us use his pre-built associated queries.

References

V. N. Anh and A. Moffat. Impact transformation: effective and efficient web retrieval. In *Proc. ACM-SIGIR Int. Conf. on Research and Development in Information Retrieval*, pages 3–10. ACM Press, New York, 2002.

P. Bailey, N. Craswell, and D. Hawking. Engineering a multi-purpose test collection for web retrieval experiments. *Information Processing & Management*, 39(6):853–871, 2003.

B. Billerbeck, F. Scholer, H. E. Williams, and J. Zobel. Query expansion using associated queries. In *Proc. Int. Conf. on Information and Knowledge Management*, pages 2–9. ACM Press, New York, 2003.

B. Billerbeck and J. Zobel. Questioning query expansion: An examination of behaviour and parameters. In K.-D. Schewe and H. E. Williams, editors, *Proc. Australasian Database Conf.*, volume 27, pages 69–76. CRPIT, 2004.

C. Carpineto, R. de Mori, G. Romano, and B. Bigi. An information-theoretic approach to automatic query expansion. *ACM Transactions on Information Systems*, 19(1): 1–27, 2001.

S. Garcia, H. E. Williams, and A. Cannane. Access-ordered indexes. In V. Estivill-Castro, editor, *Proceedings of the 27th Australasian Computer Science Conference*, volume 26, pages 7–14, Dunedin, New Zealand, January 2004.

S. Gauch and J. Wang. A corpus analysis approach for automatic query expansion. In *Proc. Int. Conf. on Information and Knowledge Management*, pages 278–284. ACM Press, New York, 1997.

J. Goldstein, M. Kantrowitz, V. Mittal, and J. Carbonell. Summarizing text documents: sentence selection and evaluation metrics. In *Proc. ACM-SIGIR Int. Conf. on Research and Development in Information Retrieval*, pages 121–128. ACM Press, New York, 1999.

D. Harman. Overview of the second Text REtrieval Conference (TREC-2). *Information Processing & Management*, 31(3):271–289, 1995.

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

K. L. Kwok and M. Chan. Improving two-stage ad-hoc retrieval for short queries. In *Proc. ACM-SIGIR Int. Conf. on Research and Development in Information Retrieval*, pages 250–256. ACM Press, 1998.

A. M. Lam-Adesina and G. J. F. Jones. Applying summarization techniques for term selection in relevance feedback. In *Proc. ACM-SIGIR Int. Conf. on Research and Development in Information Retrieval*, pages 1–9, New Orleans, Louisiana, United States, 2001. ACM Press, New York.

M. Magennis and C. J. van Rijsbergen. The potential and actual effectiveness of interactive query expansion. In *Proc. ACM-SIGIR Int. Conf. on Research and Development in Information Retrieval*, pages 324–332. ACM Press, New York, 1997.

R. Mandala, T. Tokunaga, and H. Tanaka. Combining multiple evidence from different types of thesaurus for query expansion. In *Proc. ACM-SIGIR Int. Conf. on Research and Development in Information Retrieval*, pages 191–197, Berkeley, California, United States, 1999. ACM Press, New York.

M. Mitra, A. Singhal, and C. Buckley. Improving automatic query expansion. In W. B. Croft, A. Moffat, C. J. van Rijsbergen, R. Wilkinson, and J. Zobel, editors, *Proc. ACM-SIGIR Int. Conf. on Research and Development in Information Retrieval*, pages 206–214, Melbourne, Australia, August 1998. ACM Press, New York.

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

Y. Qiu and H.-P. Frei. Concept based query expansion. In *Proc. ACM-SIGIR Int. Conf. on Research and Development in Information Retrieval*, pages 160–169. ACM Press, New York, 1993.

S. E. Robertson and S. Walker. Okapi/Keenbow at TREC-8. In *Proc. Text Retrieval Conf. (TREC)*, pages 151–161, Gaithersburg, Maryland, 1999. NIST Special Publication 500-264.

S. E. Robertson, S. Walker, M. Hancock-Beaulieu, A. Gull, and M. Lau. Okapi at TREC. In *Proc. Text Retrieval Conf. (TREC)*, pages 21–30, 1992.

J. J. Rocchio. Relevance feedback in information retrieval. In E. Ide and G. Salton, editors, *The Smart Retrieval System — Experiments in Automatic Document Processing*, pages 313–323. Prentice-Hall, Englewood Cliffs, New Jersey, 1971.

I. Ruthven. Re-examining the potential effectiveness of interactive query expansion. In *Proc. ACM-SIGIR Int. Conf. on Research and Development in Information Retrieval*, pages 213–220. ACM Press, New York, 2003.

I. Ruthven and M. Lalmas. A survey on the use of relevance feedback for information access systems. *Knowledge Engineering Review*, 18(2):95–145, 2003.

F. Scholer and H. E. Williams. Query association for effective retrieval. In C. Nicholas, D. Grossman, K. Kalpakis, S. Qureshi, H. van Dissel, and L. Seligman, editors, *Proc. Int. Conf. on Information and Knowledge Management*, pages 324–331, McLean, Virginia, 2002.

F. Scholer, H. E. Williams, J. Yiannis, and J. Zobel. Compression of inverted indexes for fast query evaluation. In *Proc. ACM-SIGIR Int. Conf. on Research and Development in Information Retrieval*, pages 222–229. ACM Press, New York, 2002.

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

A. Spink, D. Wolfram, Major B. J. Jansen, and T. Saracevic. From e-sex to e-commerce: Web search changes. *IEEE Computer*, 35(3):107–109, March 2002.

A. Tombros and M. Sanderson. Advantages of query biased summaries in information retrieval. In *Proc. ACM-SIGIR Int. Conf. on Research and Development in Information Retrieval*, pages 2–10. ACM Press, New York, 1998.

I. H. Witten, A. Moffat, and T. C. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images.* Morgan Kaufman, San Francisco, California, United States, 2nd edition, 1999.

J. Xu and W. B. Croft. Improving the effectiveness of information retrieval with local context analysis. *ACM Transactions on Information Systems*, 18(1):79–112, 2000.

J. Zobel. How reliable are the results of large-scale information retrieval experiments? In W. B. Croft, A. Moffat, C. J. van Rijsbergen, R. Wilkinson, and J. Zobel, editors, *Proc. ACM-SIGIR Int. Conf. on Research and Development in Information Retrieval*, pages 307–314, Melbourne, Australia, August 1998. ACM Press, New York.