Okapi/Keenbow at TREC–8

S E Robertson[*] S Walker[†]

1 Summary

Automatic ad hoc and web track

Three ad hoc runs were submitted: long (title, description and narrative), medium (title and description) and short (title only). "Blind" expansion was used for all runs. The queries from the medium ad hoc run were reused for the small web track submission. Most of the negative expressions were removed from the narrative field of the topic statements, and a new expansion term selection procedure was tried.

Adaptive filtering

Methods were similar to those we used in TREC–7. Six runs were submitted.

VLC track

Two unexpanded ad hoc runs were submitted.

2 Okapi at TRECs 1–7

The Okapi search systems used for TREC are descendants of the Okapi systems developed at the Polytechnic of Central London[1] between 1982 and 1988 under a number of grants from the British Library Research & Development Department and elsewhere. These early Okapi systems were experimental highly-interactive reference retrieval systems of a probabilistic type, some of which featured automatic query expansion [1, 2, 3].

All the Okapi work in connection with TRECs 1–6 was done at the Department of Information Science, City University, London. Most of the Okapi TREC–7 entries were done from Microsoft Research, Cambridge (UK).

For TREC–1 [4], the low-level search functions were generalized and split off into a separate library — the Okapi Basic Search System (BSS). User interfaces or batch processing scripts access the BSS using a simple command language-like protocol. However, our TREC–1 results were very poor [4], because the classical Robertson/Sparck Jones weighting model [5] which Okapi systems had always used took no account of document length or within-document term frequency.

During TREC–2 and TREC–3 a considerable number of new term weighting and combination functions were tried; a runtime passage determination and searching package was added to the BSS; and methods of selecting good terms for routing queries were developed [6, 7]. During the TREC–2 work "blind" query expansion (feedback using terms from the top few documents retrieved in a pilot search) was tried for the first time in automatic ad hoc experiments, although we didn't use it in the official runs until TREC–3. Our TREC–3 automatic routing and ad hoc results were good.

TREC–4 [8] did not see any major developments. Routing term selection methods were further improved.

By TREC–5 many participants were using blind expansion in ad hoc, several of them more successfully than Okapi [9, 10]. In the routing, we tried to optimize term weights after selecting good terms (as did at least one other participant); our routing results were again among the best, as were batch filtering runs.

In TREC–6 [11] we continued to investigate blind expansion, with mixed results. We also introduced a new weighting function designed to make use of documents known or assumed to be non-relevant. In routing and filtering

[*] Microsoft Research Ltd, 1 Guildhall Street, Cambridge CB2 3NH, UK, and City University, London, UK. email ser@microsoft.com

[†] Microsoft Research Ltd, 1 Guildhall Street, Cambridge CB2 3NH, UK. email sw@microsoft.com

[1] Now the University of Westminster.

we continued to extend the optimization procedure, including a version of simulated annealing. Again our routing and filtering results were among the best. The Okapi BSS was modified to handle large databases for the VLC track.

We entered the adaptive filtering track, with fairly good results, for the first time in TREC–7 ([12]). Routing and batch filtering were dropped.

3 The system

At the Microsoft Research laboratory in Cambridge, we are developing an evaluation environment for a wide range of information retrieval experiments. This environment is called Keenbow. The Okapi BSS is now seen as a component of Keenbow.

3.1 The Okapi Basic Search System (BSS)

The BSS, which has been used in all Okapi TREC experiments, is a set-oriented ranked output system designed primarily for probabilistic–type retrieval of textual material using inverted indexes. There is a family of built-in weighting functions as defined below (equation 1) and described more fully in [7, Section 3]. In addition to weighting and ranking facilities it has the usual boolean and quasi-boolean (positional) operations and a number of non-standard set operations. Indexes are of a fairly conventional inverted type. There have been no major changes to the BSS during TREC–8.

Weighting functions

All TREC–8 searches used varieties of the Okapi **BM25** function first used in TREC–3 (equation 1).

$$\sum_{T \in Q} w^{(1)} \frac{(k_1+1)tf}{K+tf} \frac{(k_3+1)qtf}{k_3+qtf} \tag{1}$$

where

Q is a query, containing terms T
$w^{(1)}$ is the Robertson/Sparck Jones weight [5] of T in Q

$$\log \frac{(r+0.5)/(R-r+0.5)}{(n-r+0.5)/(N-n-R+r+0.5)} \tag{2}$$

N is the number of items (documents) in the collection
n is the number of documents containing the term
R is the number of documents known to be relevant to a specific topic
r is the number of relevant documents containing the term
K is $k_1((1-b)+b.dl/avdl)$
k_1, b and k_3 are parameters which depend on the on the nature of the queries and possibly on the database; k_1 and b default to 1.2 and 0.75 respectively, but smaller values of b are sometimes advantageous; in long queries k_3 is often set to 7 or 1000 (effectively infinite)
tf is the frequency of occurrence of the term within a specific document
qtf is the frequency of the term within the topic from which Q was derived
dl and $avdl$ are respectively the document length and average document length measured in some suitable unit.

Term ranking for selection

Prior to TREC–8 the method used was that proposed in [13] by which terms are ranked in decreasing order of

$$TSV = r.w^{(1)} \tag{3}$$

This time a new method was tried; this is discussed in section 4.

Passage determination and searching

Since TREC–3 the BSS has had facilities for search-time identification and weighting of any sub-document consisting of an integral number of consecutive paragraphs. It was described, and some results reported, in [7]. Passage searching almost always increases average precision, by anything from 2%–10%, as well as recall and precision at the higher cutoffs. It often, perhaps surprisingly, reduces precision at small cutoffs, so is not used in pilot searches for expansion runs.

3.2 Hardware

All the TREC–8 processing was done at Microsoft Research, Cambridge. Most of the work was done on a 300 MHz Sun Ultra 10 with 256 MB and a Dell with two 400 MHz Pentium processors and 512 MB. Both machines were running Solaris 2.6. Mainly to cater for the VLC track there was about 170GB of disk storage, most of which was attached to the Sun. The network was 100MHz ethernet.

3.3 Database and topic processing

Text processing

For interactive purposes it is necessary to provide for the readable display of documents. Since we have not (yet) implemented a runtime display routine, nor adequate parsing and indexing facilities, for SGML data, all the TREC input text is subjected to batch conversion into a uniform displayable format before further processing. This is done by means of hacked up shell scripts specific to the input dataset. For most of the TREC data, output records have three fields: document number, any content unsuitable for indexing (or not to be searched—such as controlled descriptors in some datasets), and the searchable "TEXT" and similar portions. However, only two fields were used for the VLC98 collection.

Indexing

All the TREC text indexing was of the keyword type. A few multiword phrases such as "New York", "friendly fire", "vitamin E" were predefined and there was a pre-indexing facility for the conflation of groups of closely related or synonymous terms like "operations research" and "operational research" or "CIA" and "Central Intelligence Agency". A stemming procedure was applied, modified from [14] and with additional British/American spelling conflation. The stoplist contained about 220 words.

Topic processing

Apart from the narrative field, which is only used in "long" topic queries, topic text is processed in the same way as text to be indexed. In the narrative field terms such as "document", "describe(s)", "relevan…", "cite…" are deleted (in addition to the normal stop terms). There is a facility, new for TREC–8, for removing most negative expressions. This was tried on most of the past TRECs' ad hoc data, and often made little difference. However, it was rarely detrimental, and appears beneficial in TREC–8 (see Table 2).

4 Term selection for query expansion

Readers who took part in early TRECs will recall discussions on the issue of "selective versus massive" query expansion. Many participants did some form of query expansion, particularly by extracting terms from previously known relevant documents in the routing task. The point at issue was whether one should include all candidate terms (possibly with small weights), or be selective and add only a small number. The Okapi team had a bias towards being very selective in expansion.

Since then, of course, many teams, including us, have applied and adapted the relevance feedback methods to blind expansion (pseudo-relevance feedback). The same problem arises; in fact in this case, there is an earlier stage of selection, namely the selection of a number of *documents* from the top of the initial retrieval ranking.

Our general method for query expansion following relevance feedback is to rank terms according to some measure of their likely contribution to the effectiveness of the search, and then to take a certain (fixed) number from the top of this ranking. We have used various measures for this ranking purpose, but a common feature of all such measures is that they are intended *only* for ranking, that is for measuring *relative* contribution; none is susceptible to the use of an absolute threshold or criterion for inclusion. Last year's work on thresholding for filtering inspired us to consider the possibility of devising an absolute measure which could be compared to an absolute threshold. The potential advantages of such a measure are:

- If there are more good terms, more can be included, and vice-versa.
- The measure would take account of the number of relevant documents; one might expect to get stronger evidence for more good terms from more relevant documents.

The method discussed below is a first attempt at such a measure. We have applied it to the blind expansion problem, although it is in principle applicable to real relevance feedback as well. We have not yet tackled the additional selection problem for blind feedback (document selection), and therefore the second reason above does not really apply; however, there may be a secondary effect here, that if the initial retrieval generates a coherent set of documents, there is more chance of more good terms emerging from the analysis than if the initial search is poor and generates a more random set.

It is also worth pointing out that although the idea was inspired by the filtering-threshold problem, the method itself is very different.

4.1 Statistical significance of new terms

We are looking for new terms which are sufficiently strongly associated with relevance to contribute to performance. A measure which might satisfy the above requirement would be a measure of the statistical significance of any given term's association with relevance. Significance measures are typically on a scale which allows a comparison with an absolute likelihood of the null hypothesis (that is, an absolute probability of the observation given a hypothesis of no association). This absolute value might be (say) 5%, 1% or 0.1%; although the choice of level is largely arbitrary, any fixed single choice would satisfy the requirements given above, when applied to different topics or different numbers of relevant documents. For reasons given below, these specific values are not themselves appropriate to this case, but the principle remains.

As in most of our various term selection measures, we look primarily at term presence/absence (we have had very little success with methods which take account of term frequency). Thus the relation between the term and relevance (known or assumed) is defined by the usual 2 × 2 table:

	Relevant	Not relevant	
Term t present	r_t	$n_t - r_t$	n_t
Term not present	$R - r_t$	$N - R - n_t + r_t$	$N - n_t$
	R	$N - R$	N

(as usual, all documents not known or assumed to be relevant are assumed to be not relevant).

The null hypothesis is that the term is not associated with relevance. The likelihood of the null hypothesis, given the above data, may be approximated as follows. The probability of the term occurring in a document taken at random is n_t/N. We assume that the term occurs in a small proportion of the collection as a whole (for all terms of interest, this will be the case). Then the probability that it occurs in exactly r_t out of the R relevants is approximately

$$\left(\frac{n_t}{N}\right)^{r_t} \binom{R}{r_t} \tag{4}$$

The second factor is the number of ways we can choose r_t from R, and can be calculated as $\frac{R!}{r_t!(R-r_t)!}$.

4.2 The criterion

As indicated, a usual criterion for rejection of a null hypothesis would be a likelihood of less than (say) 1%. However, if we were to apply such a criterion in this case, we would be likely to get a great deal of noise. The reason is that there are so many terms in a text retrieval system to start with. Suppose that the total vocabulary in the system (indexed terms) is 100,000. Then we might expect 1000 of these terms to exceed this criterion, *even if the null hypothesis is true of all of them.*

This then suggests that we should set the criterion in relation to the size of the vocabulary, V. A threshold of $1/V$ would imply that we might expect (assuming the null hypothesis applies to all terms) a single noise term. Safety might suggest setting an even stricter threshold; however, since there will be many terms in the vocabulary that have a negative association with relevance, and many others that occur in one document only (which would give them no chance of being selected on any such criterion), it is probably reasonable to relax the threshold somewhat. In the experiments, we have used a threshold $1/Ve^c$, for some constant c. Thus we may express the criterion as:

$$\left(\frac{n_t}{N}\right)^{r_t} \begin{pmatrix} R \\ r_t \end{pmatrix} < \frac{1}{Ve^c}$$

$$\text{or} \qquad r_t \log \frac{N}{n_t} - \log \begin{pmatrix} R \\ r_t \end{pmatrix} - \log V > c \tag{5}$$

If the threshold c is set at 0, this is equivalent to taking a likelihood threshold of $1/V$, that is to accepting about one noise term under the simplest model as discussed above. A positive c is a stricter threshold; a value of 4.6 corresponds to having a less than 1% chance of accepting any noise terms. Experiments suggested that we could afford to relax the threshold; we used some negative c values.

5 Automatic adhoc and web tracks

The procedure for the official runs (Table 1) was as follows. Pools of potential expansion terms were generated by running pilot searches on terms extracted from the appropriate topic fields against the TREC disks 1–5 database, and outputting the top R documents. Terms extracted from these documents were weighted with the usual $w^{(1)}$ formula. In some cases an additional weighting was applied to topic terms. Terms were selected from the pools by means of the new selection procedure equation 5; two runs were done, using different term selection thresholds, for each topic source type (long, medium and short), and these were merged in pairs to produce the submitted runs. The thresholds varied between −4 and 4.6; there were other minor variations between runs, including for example the exact treatment of query terms.

Table 1: Automatic ad hoc and web track, official runs

			AveP					
Run	source	R	score	$\geq$ med	P10	P30	RPrec	Rcl
ok8alx	TND	20	324	46	570	443	354	694
ok8amxc	TD	14	317	45	550	425	347	679
ok8wmx	web, as ok8amxc		383	47	516	399	518	900
ok8asxc	T	10	279	32	488	380	309	637

Table 2 summarizes some diagnostic ad hoc runs, this time with single thresholds. It is not clear that the new term selection procedure offers any advantage over the old one; more investigation is needed. Removal of negative expressions was beneficial overall, increasing average precision in 17 topics and decreasing it in 10. However, trials on data from previous TRECs have shown that the effect averaged over 50 topics is not always beneficial.

6 Adaptive filtering

The system for adaptive filtering is very similar to the one used for TREC–7. It is described in [12] and more fully in a paper to appear next year [15]. A brief summary only will be given here. Please note that the equation for adapting the β value given in the TREC–7 proceedings contains a mistake; the correct version is given below and in the Journal of Documentation paper.

We assume the usual filtering situation, as constrained by the TREC filtering track rules. In particular, we assume an incoming stream of documents, a set of persistent user profiles (initially based on text topics), an accumulating collection of all the documents that have arrived so far, and a history for each profile, including relevance judgments for documents previously returned to the user. The entire process is switched on at time $t = 0$.

6.1 Calibration

The system used the usual BM25 match function, but the filtering task requires a threshold to be applied to the match values, to drive a retrieval decision. Thresholding in this system is tied to a calibration of the match function, to give values which can be regarded as actual probabilities of relevance. An initial calibration of the score is modified

Table 2: Ad hoc diagnostic runs

“Long topic” runs	Conditions	AveP	% gain	P30	% gain
ok8alpl1.neg	baseline (no expansion, no passages)	271	0.0	380	0.0
ok8alpl1	baseline + neg exprns removed + minor mods to parsing rules	277	2.2	386	1.6
ok8al2np.tns	+ exp 20 docs, eqn 5 term selection, threshold $c = 0.0$	310	14.4	427	12.4
ok8al2.tns	+ passages	327	20.6	437	15.0
ok8al2np.tns.rsv	Same # trms/query as ok8al2np.tns but trm sel by eqn 3	310	14.4	432	13.7
ok8al2np.tns.rsv.f51	as previous but fixed 51 terms per query	314	15.9	436	14.7
ok8al2np.tns.rsv.f30	as previous but fixed 30 terms per query	310	14.4	432	13.7
“Medium topic” runs	Conditions	AveP	% gain	P30	% gain
ok8ampl1	baseline (no expansion, no passages)	261	0.0	361	0.0
ok8am1np.tns	+ exp 14 docs, eqn 5 term selection, threshold $c = -4.0$	304	16.5	415	15.0
ok8am1.tns	+ passages	318	21.8	422	16.9
ok8am1np.tns.rsv	Same # trms/query as ok8am1np.tns but trm sel by eqn 3	298	14.2	411	13.9
ok8am1np.tns.rsv.f55	as previous but fixed 55 terms per query	308	18.0	417	15.5
“Short topic” runs	Conditions	AveP	% gain	P30	% gain
ok8aspl1	baseline (no expansion, no passages)	239	0.0	343	0.0
ok8as1np.tns	+ exp 10 docs, eqn 5 term selection, threshold $c = 3.0$	270	13.0	371	8.2
ok8as1.tns	+ passages	279	16.7	377	9.9
ok8as1np.tns.rsv	Same # trms/query as ok8as1np.tns but trm sel by eqn 3	267	11.7	363	5.8
ok8as1np.tns.rsv.f17	as previous but fixed 17 terms per query	288	20.5	383	11.7

on a per-topic basis, as feedback is obtained. The initial calibration is based on a logistic regression analysis of some training data. This gives values for β and γ in the equation:

$$\log O(R|D) = \beta + \gamma\, f(\text{score}, \mathit{ast1}, \mathit{maxscore}, \mathit{ql}) \tag{6}$$

$f(\text{score}, \ldots)$ is basically a normalization function for the score, taking into account some variables which might be expected to affect it; two forms of $f()$ were used (see below). *maxscore* is the theoretical maximum score; *ql* is the query length in terms.

ast1 is the average score of the top 1% of retrieved documents; under TREC filtering rules (no access to any part of the document stream before $t = 0$), we do not initially have a direct method of estimating this, so another linear regression is performed on the same training data to estimate it for the first simulated week:

$$\mathit{ast1} = \alpha_1 + \alpha_2\, \mathit{maxscore}$$

After the first week, *ast1* is estimated directly from the accumulated collection.

6.2 Score normalization

According to the arguments presented in the last TREC, it would be reasonable to assume that $f(\text{score}, \ldots)$ should be linear in the score – that is, the score is assumed to be a linear function of the log-odds of relevance. However, this assumption depends on the basic independence assumptions of the probabilistic model, which may be doubted. A close look at the regression data suggested strongly that the relationship was non-linear, in particular that very high scores did not imply correspondingly high probabilities of relevance. We therefore tried a non-linear function as well as a linear one. Choice of functions was determined by the best-fitting logistic regression; we tried various linear and non-linear functions and chose the best of each after completing the regression. We also did the same exercise on three different training sets and took a guess at a compromise solution (this last step was relatively easy, as the solutions from the three training sets were generally very close).

The first step normalization was a linear one:

$$\mathit{normscore} = \frac{\text{score}}{\mathit{ast1} + 0.22\mathit{maxscore} - 1.3\mathit{ql}} \tag{7}$$

For the linear normalization we used exactly this, that is:

$$f(\text{score}, \ldots) = \mathit{normscore}$$

For the non-linear normalization, we made a further transformation:

$$f(\text{score}, \ldots) = \frac{\mathit{normscore}}{0.4 + \mathit{normscore}} \tag{8}$$

6.3 Adaptation

Given a document score and an estimated *ast1*, equation 6 can be used to estimate the log-odds of relevance of any specific document, to be compared to an absolute threshold determined by the desired utility measure. The result of applying the equation to the score for document D_i will be denoted c_i (for calibrated score). c_i is on a log-odds scale, but can be converted back to a probability p_i:

$$\begin{aligned} c_i &= \beta + \gamma\, f(\text{score}, \mathit{ast1}, \mathit{maxscore}, \mathit{ql}) \qquad (9) \\ p_i &= \frac{\exp c_i}{1 + \exp c_i} \end{aligned}$$

for some estimated β, γ and *ast1*.

As we obtain feedback on the model, as well as re-estimating *ast1*, we adjust the calibration by correcting β (γ is left unchanged). β estimation is based on a Bayesian argument, in order to prevent it going wild with small amounts of data. The Bayesian prior is represented by m mythical documents whose estimated probabilities of relevance are assumed to be correct at 0.5. We suppose an iterative sequence of estimates $\beta^{(n)}$ and corresponding values $c_i^{(n)}$ and $p_i^{(n)}$ for each document. Then the gradient descent formula is:

$$\beta^{(n+1)} = \beta^{(n)} + \frac{r - \sum_{i=1}^{j} p_i^{(n)} + m\, \frac{1-\exp(\beta^{(n)}-\beta^{(0)})}{2(1+\exp(\beta^{(n)}-\beta^{(0)}))}}{\sum_{i=1}^{j} p_i^{(n)}(1-p_i^{(n)}) + m\, \frac{\exp(\beta^{(n)}-\beta^{(0)})}{(1+\exp(\beta^{(n)}-\beta^{(0)}))^2}} \tag{10}$$

$\beta^{(0)}$ is the initial estimate provided by the original regression equation 6.

(Please note that the first component of the denominator of this equation was incorrect in our TREC–7 report The form we actually used was however correct.)

In the experiments, as last year, one iteration only was performed at each stage (simulated week), and only when new documents have been assessed; however, successive stages are cumulative, and at each stage all previously assessed documents are included in the estimation process. This procedure represents a very simple-minded approach to the normal iterative estimation suggested by the above argument.

Again as last year, we use a 'ladder' of thresholds, starting lower down in order to get some documents to the user for feedback purposes, even if this lowers performance in the early stages of the profile. The current ladder is given in table 3.

Table 3: The Ladder: selection thresholds

(Slightly modified from the TREC–7 version)
Initial points are explained in the text.

$P(R\|D)$	$\log O(R\|D)$	
0.5	0	
0.4	-0.4	Final (LF1 runs)
0.25	-1.1	Final (LF2 runs)
0.18	-1.5	First week
0.13	-1.9	
0.1	-2.2	High start
0.07	-2.6	
0.05	-2.9	
0.04	-3.2	Low start

In the circumstances of the TREC filtering task, an additional constraint applies: because the initial estimate (based on *maxscore* rather than on *ast1*) may be unreliable, and may in particular lead to the retrieval of many too many documents, the threshold is kept high for the first week. When a direct estimate of *ast1* becomes available, the ladder is brought into effect, and the threshold is moved down to the appropriate place (possibly above the bottom if we found some documents in the first week).

Various other feedback methods may be brought into effect at various stages in the history of the profile. These include:

- Reweighting query terms
- Query expansion based on term selection value
- Query optimization (weights and/or selection of terms)
- Threshold optimization.

In general, any query adjustment has to be undertaken before any threshold setting, as it affects both *ast1* and the scores of the judged documents, all of which are used in threshold setting.

As last year, on this occasion we have tried only the threshold optimization. No term reweighting or query expansion methods were tried.

6.4 Experiments

Training

The training databases were: LA Times data with TREC topics 301–350; AP newswire data with topics 51–150; Wall Street Journal data with topics 51–150. All fields of the topics (Title, Description, Narrative) were used.

Each topic was searched on this collection, and the top ranked documents were retrieved, the number being specified as 1% of the collection. A series of logistic regression analyses were performed, to estimate β and γ in equation 6, with different functions for $f(\text{score}, \ldots)$. The final choices for this function were those specified above (equations 7 and 8); the final choices for β and γ were -7.5 and 6.6 respectively for the linear score normalization, and -18.3 and 24.4 for the non-linear normalization.

The test topics themselves were also used in their entirety (title, description, narrative). They were initially searched on a database consisting of the three training databases merged, to establish initial terms and weights.

Adaptive procedure

Documents were processed in weekly batches. For the first week, the threshold was set at the point labelled 'First week' (because of the uncertainty of the *ast1* value). From the following week, a direct estimate of *ast1* is available, and two different initial points were tried (labelled 'High start' and 'Low start' in the table). Thereafter, the usual ladder rule applied: each profile was moved up one notch for each relevant document found. (As some profiles will have found relevant documents in the first week, these ones will never actually be at their theoretical starting point.) *ast1* is re-estimated from the accumulated collection for the first six weeks.

After each week in which some documents have been found for a profile (irrespective of relevance), the adaptive calibration of β is invoked. That is, for each previously seen document, a value of c_i is calculated according to the current profile and the current value of *ast1*, and one iteration only of the iterative formula 10 is then applied. The value of m in equation 10 was 5. The new value of β remains in force for this profile until the next invocation of the adaptive calibration.

Runs

Twelve candidate runs were completed, defined by the following parameters:
Treatment: 1 Base (Low start, no threshold optimization); 2 High start; 3 High start + Threshold optimization.
Utility function: 1 LF1; 2 LF2.
Score normalization; 1 Linear; 2 Non-linear.

Runs are numbered in the form ok8fxyz, where x is the treatment, y is the utility function, and z is the score normalization. Because of the limitation on the number of submissions, a random choice of six of these runs was submitted.

Results

Official results for the submitted runs, plus unofficial results for the others, are shown in Table 4. The table makes a topic-by-topic comparison with the other submissions to TREC, on utility scores. Some recall and precision results for all 12 runs (macro average), using the relevance judgments made for TREC–8, are shown in Table 5.

Table 4: Adaptive filtering: comparison with other TREC–8 official runs

Columns contain number of topics (out of 50)
* unofficial results

	All years					1994 only				
	Worst	< med	Median	> med	Best	Worst	< med	Median	> med	Best
	Utility function LF1 (out of 19 systems)									
ok8f111*	0	42	1	7	0					
ok8f112	0	43	3	4	0	0	25	5	20	6
ok8f211*	0	26	5	19	2					
ok8f212*	0	31	2	17	1					
ok8f311	0	23	6	21	2	0	13	14	23	9
ok8f312	0	27	5	18	1	0	15	10	25	9
	Utility function LF2 (out of 13 systems)									
ok8f121*	0	38	1	11	0					
ok8f122	23	39	4	7	0	16	26	7	17	4
ok8f221*	0	24	3	23	1					
ok8f222	1	28	3	19	2	4	24	10	16	3
ok8f321	0	25	4	21	2	2	17	14	19	5
ok8f322*	1	29	1	20	3					

Discussion of results

Overall the results are disappointing. This is partly the result of continuing to concentrate on the thresholding problem, at the expense of any query expansion/modification techniques. It is encouraging that our relative performance improves over the three simulated years of the task, even despite the lack of any query modification; the threshold adaptation methods are certainly valuable.

Starting higher up the ladder than the bottom (ok8f2xx) is clearly advantageous. As indicated at TREC–7, the ladder is very arbitrary, and it is entirely possible that we have overemphasized the argument about getting additional feedback material early. The non-linear normalization of score (ok8fxx2) had a very slight negative effect. Although it gives without doubt a better approximation to the probability of relevance, its only real effect is outside the range in which we are interested (in particular, the very top scoring documents), and it appears that the linear approximation is a quite good enough substitute for this purpose, and may even gain by being simpler. The threshold optimization seems to help a little.

7 VLC

Database processing

Before indexing, the source text was reduced by removing lines starting with “Server:”, “Content-type:”, “Last modified:”, etc, document numbers were then identified, followed by the removal of all text inside ‘< ... >’. Dates and URLs were retained, but not indexed. This reduced the indexable text by almost 50% to a little over 50 GB. (Source text was unchanged from TREC–7.)

Examination of a few of the documents suggested that there was quite a lot of non-text material (compressed data etc). It was decided that it would not be practicable to remove (or avoid indexing) this material. This resulted in an index with a very large dictionary file of some 70 million terms most of which are nonsensical nonce-words, a typical sequence being “qetura”, “qetutmz7”, “qetuwuqgrslk79”, “qetv”, “qetv9pif0yk9”, ...

Table 5: Recall and precision results

Average of ratios precision and recall results All years based on 49 topics (excluding one with no relevant documents) (none of these retrieved sets was empty)		
	Precision	Recall
ok8f111	0.174	0.338
ok8f112	0.160	0.343
ok8f121	0.168	0.362
ok8f122	0.151	0.375
ok8f211	0.239	0.257
ok8f212	0.227	0.274
ok8f221	0.220	0.295
ok8f222	0.203	0.326
ok8f311	0.248	0.252
ok8f312	0.238	0.265
ok8f321	0.236	0.271
ok8f322	0.226	0.289

The database was reindexed without positional information for TREC–8. The resulting index size was 14 GB (compared with 34 GB for the full index used in TREC–7). The total number of indexed tokens from the 18.6 million documents was about 5800 million (mean 312 per document), and the corresponding figure for types was 2600 million (140 per document).

Results

Table 6 summarizes the results. The official runs are ok8v1 and ok8v2. All runs used plain unexpanded ad hoc searches. The standard stop list used for ok8v1 contains 222 words. The extended stoplist used for all other runs contained also "i", "inform..", "does", "me", "find"; these were added with the intention of speeding searches, which they did. With normal "TREC-sized" databases and memory in the 256–512 MB range, BSS searches usually go more quickly if output sets are formed in (virtual) memory rather than explicitly on disk; but for the VLC much more physical memory would be required for this to hold. Finally, the BSS facility for heuristic limitation of output set size (which has been in use at least since TREC–3) has a marked effect when the required output is few documents from a large database.

Table 6: VLC results

Run	Conditions	Secs/query	Mod. AveP	P10	P30
ok8v1	no expansion, no passages	5.88	431	568	528
ok8v2	as v1 but slightly larger stoplist	4.30	445	560	538
ok8v23	as v2 but temp files instead of memory	3.82	445	560	538
ok8v22	as v23 but no output set size limitation	7.90	445	560	538

8 Discussion

The methods used in Okapi in general continue to give good results in several tracks. However, it is noticeable that of the three main modifications introduced this year (absolute expansion term selection, removal of negations, non-linear model for score calibration), only the one with a linguistic motivation seemed to help us.

It is clear that in order to achieve reasonable results in the filtering track, we need to move on to query expansion. However, the concentration on threshold setting has been useful, and the improvement in relative performance in the third year of the simulation probably indicates that it is doing something which other methods are failing to achieve.

References

[1] Mitev, N.N., Venner, G.M. and Walker, S. *Designing an online public access catalogue: Okapi, a catalogue on a local area network*. British Library, 1985. (Library and Information Research Report 39.)

[2] Walker, S. and Jones, R.M. *Improving subject retrieval in online catalogues: 1. Stemming, automatic spelling correction and cross-reference tables*. British Library, 1987. (British Library Research Paper 24.)

[3] Walker, S. and De Vere, R. *Improving subject retrieval in online catalogues: 2. Relevance feedback and query expansion*. British Library, 1990. (British Library Research Paper 72.) ISBN 0-7123-3219-7

[4] Robertson, S.E. *et al.* Okapi at TREC. In: [16], p21–30.

[5] Robertson, S.E. and Sparck Jones K. Relevance weighting of search terms. *Journal of the American Society for Information Science* 27, May–June 1976, p129–146.

[6] Robertson, S.E. *et al.* Okapi at TREC–2. In: [17], p21–34.

[7] Robertson, S.E. *et al.* Okapi at TREC–3. In: [18], p109–126.

[8] Robertson, S.E. *et al.* Okapi at TREC–4. In: [19], p73–96.

[9] Beaulieu, M.M. *et al.* Okapi at TREC–5. In: [10], p143–165.

[10] *The Fifth Text REtrieval Conference (TREC–5)*. Edited by E.M. Voorhees and D.K. Harman. Gaithersburg, MD: NIST, 1997.

[11] Walker S. *et al.* Okapi at TREC–6: Automatic ad hoc, VLC, routing, filtering and QSDR. In: [20], p125–136.

[12] Robertson, S.E., Walker, S. and Beaulieu, M. Okapi at TREC–7: automatic ad hoc, filtering, VLC and interactive track. In: [21], p253–264.

[13] Robertson, S.E. On term selection for query expansion. *Journal of Documentation* 46, Dec 1990, p359–364.

[14] Porter, M.F. An algorithm for suffix stripping. *Program* 14 (3), Jul 1980, p130-137.

[15] Robertson, S.E. and Walker, S. Threshold setting in adaptive filtering. *Journal of Documentation*, 56, 2000 (to appear).

[16] *The First Text REtrieval Conference (TREC–1)*. Edited by D.K. Harman. Gaithersburg, MD: NIST, 1993.

[17] *The Second Text REtrieval Conference (TREC–2)*. Edited by D.K. Harman. Gaithersburg, MD: NIST, 1994.

[18] *Overview of the Third Text REtrieval Conference (TREC–3)*. Edited by D.K. Harman. Gaithersburg, MD: NIST, 1995.

[19] *The Fourth Text REtrieval Conference (TREC–4)*. Edited by D.K. Harman. Gaithersburg, MD: NIST, 1996.

[20] *The Sixth Text REtrieval Conference (TREC–6)*. Edited by E.M. Voorhees and D.K. Harman. Gaithersburg, MD: NIST, 1998.

[21] *The Seventh Text REtrieval Conference (TREC–7)*. Edited by E.M. Voorhees and D.K. Harman. Gaithersburg, MD: NIST, 1999.