Okapi at TREC–7: automatic ad hoc, filtering, VLC and interactive track

S E Robertson* S Walker† M Beaulieu‡

Advisers: E Michael Keen (University of Wales, Aberystwyth), Karen Sparck Jones (Cambridge University), Peter Willett (University of Sheffield)

1 Summary

Automatic ad hoc

Three runs were submitted: medium (title and description), short (title only) and a run which was a combination of a long run (title, description and narrative) with the medium and short runs. The average precision of the last mentioned run was higher by several percent than any other submitted run, but another participant[1] recently noticed an impossibly high score for one topic in the short run. This led to the discovery that due to a mistake in the indexing procedures part of the SUBJECT field of the LA Times documents had been indexed. Use of this field was explicitly forbidden in the guidelines [1] for the ad hoc track. The official runs were repeated against a corrected index, and the corrected results are reported below, average precisions being reduced by about 2–4%.

Adaptive filtering

This year saw a major departure from the previous Okapi approach to routing and filtering. We concentrated our efforts on the twin problems of (a) starting from scratch, with no assumed history of relevance judgments for each topic, and (b) having to define a threshold for retrieval. The thresholding problem is interesting and difficult; we associate it with the problem of assigning an explicit estimate of the probability of relevance to each document. We describe and test a set of methods for initializing the profile for each threshold and for modifying it as time passes. Two pairs of runs were submitted: in one pair queries remained constant, but in the other query terms were reweighted when fresh relevance information became available.

VLC track

Four runs on the full database were submitted, together with one each on the 10% and 1% collections. Unexpectedly, unexpanded queries did better than expanded ones; the best run used all topic fields and some adjacent term pairs from the topics. The best expanded run used one of the TREC–7 ad hoc query sets (expanded on disks 1–5).

Interactive track

Two pairwise comparisons were made: Okapi with relevance feedback against Okapi without, and Okapi without against ZPrise without. Okapi without performed somewhat worse than ZPrise, and Okapi with only partially recovered the deficit.

*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

‡Department of Information Studies, University of Sheffield, Western Bank, Sheffield S10 2TN, UK. email m.beaulieu@sheffield.ac.uk. *All three previously at:* Centre for Interactive Systems Research, Department of Information Science, City University, Northampton Square, London EC1V 0HB, UK.

[1]David Hawking of CSIRO, Canberra

2 Okapi at TRECs 1–6

The Okapi search systems used for TREC are descendants of the Okapi systems developed at the Polytechnic of Central London[2] 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 [2, 3, 4].

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

For TREC–1 [5], 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.

Our TREC–1 results were very poor [5], because the classical Robertson/Sparck Jones weighting model [6] 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 [7, 8]. 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 both relatively good.

TREC–4 [9] 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 with more success than Okapi [10, 11]. 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 the filtering track runs.

In TREC–6 [12] we continued to investigate blind expansion, with continuing 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 we continued to extend the optimization procedure, including a version of simulated annealing. Again our routing and filtering results were among the best.

3 The system

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 [8, 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 were again no major changes to the BSS during TREC–7.

Weighting functions

All TREC–7 searches used varieties of the Okapi **BM25** function first used in TREC–3 (equation 1).

$$\sum_{T \in \mathcal{Q}} w^{(1)} \frac{(k_1+1)tf}{K+tf} \frac{(k_3+1)qtf}{k_3+qtf} + k_2 . |\mathcal{Q}| . \frac{avdl-dl}{avdl+dl} \tag{1}$$

where

$\mathcal{Q}$ is a query, containing terms T
$w^{(1)}$ is either the Robertson/Sparck Jones weight [6] of T in $\mathcal{Q}$

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

[2]Now the University of Westminster.

or else a slightly modified and more general version which takes account of non-relevance as well as relevance information [14]

$$\frac{k_5}{k_5+\sqrt{R}}(k_4+\log\frac{N}{N-n})+\frac{\sqrt{R}}{k_5+\sqrt{R}}\log\frac{r+0.5}{R-r+0.5}-\frac{k_6}{k_6+\sqrt{S}}\log\frac{n}{N-n}-\frac{\sqrt{S}}{k_6+\sqrt{S}}\log\frac{s+0.5}{S-s+0.5} \quad (3)$$

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
S is the number of documents known to be nonrelevant to a specific topic
s is the number of nonrelevant documents containing the term
K is $k_1((1-b)+b.dl/avdl)$
k_1, b, k_2, k_3 and k_4 are parameters which depend on the on the nature of the queries and possibly on the database. For the TREC–7 experiments, k_1 was 1.2 and b was 0.75 except where stated otherwise; k_2 was always zero and k_3 anything from 0 to 1000; when there was not much relevance information -0.7 was a good value for k_4, otherwise zero.
k_5 and k_6 determine, in equation 3, how much weight is given to relevance and non-relevance information respectively. Typical ranges are 0–4 for k_5 and 4–∞ for k_6
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 the document length and average document length (arbitrary units) resp.

When k_2 is zero, as it was for all the results reported here, equation 1 may be written in the simpler form

$$\sum_{T\in Q} w^{(1)}\frac{(k_1+1)tf}{K+tf}\frac{(k_3+1)qtf}{k_3+qtf} \quad (4)$$

Nonrelevance information

The extension to the basic weighting formula given by equation 3 is motivated mainly by the desire to make use of explicit judgment of nonrelevance, rather than relying entirely on the "complement" method, by which all documents not known to be relevant are assumed to be nonrelevant. There is a fuller discussion in [14]. This formula might be used in various environments; the particular use reported here has to do with "blind" expansion, where there are no explicit judgments of relevance. Further detail is given below.

Term ranking for selection

In [8] there is a brief discussion of some alternative ways of ranking potential expansion terms. It appeared that no method was superior to the method proposed in [15] by which terms are ranked in decreasing order of $TSV = r.w^{(1)}$. In line with the "nonrelevance" version of $w^{(1)}$ (equation 3) Boughanem has proposed the more general function

$$TSV = (r/R - \alpha s/S).w^{(1)} \quad (5)$$

where $\alpha \in [0,1]$ and r, R, s and S are as above.

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 [8]. Passage searching almost always increases average precision, by about 2–10 percent, as well as recall and precision at the higher cutoffs. It often, surprisingly, reduces precision at small cutoffs, so is not used in pilot searches for expansion runs. Recently a mistake in the passage searching code, undetected since 1993, was found and corrected; some past TREC runs were repeated in the hope that results might be improved, but unfortunately the correction seems to make very little difference in practice.

3.2 Hardware

Apart from the interactive track almost all the TREC–7 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. Mainly to cater for the VLC track there was about 170GB of disks, most of which were attached to the Sun. Both machines were running Solaris 2.6. Some use was also made of a PC running Linux. The network was 100MHz ethernet.

3.3 Database and topic 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 for SGML data, nor adequate parsing and indexing facilities, all the TREC input text is subjected to batch conversion into a uniform displayable format before further processing. This is done by means of shell scripts which are hacked up to suit the input dataset. For most of the TREC data, the processed records have three fields: document number, any content which may be useful for display but not for searching, and the searchable "TEXT" and similar portions. However, only two fields were used for the VLC98 collection.

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 [16] and with additional British/American spelling conflation. The stoplist contained about 220 words.

Initial topic processing deleted terms such as "document", "describe(s)", "relevan...", "cite..." from any description, narrative or summary fields. What is left was then processed in the same way as text to be indexed.

4 Automatic adhoc

Again, there was no significant change in methods for TREC–7. Further experiments were done using adjacent term pairs from topic statements, but any gain was very slight. To try the effect of emphasizing the difference in weights between rare and common terms some runs were done with the w^1 termweights raised to a power $1 + \alpha$ for some $0 < \alpha \leq 0.5$. Where there was a fairly large number of terms in the query, values of α in the range 0.1–0.3 sometimes improved average precision and some other statistics by around 2% over 50 topics. However, any improvement in our results (Table 1) seems to have come from merging the output from a number of runs. In the past we have not obtained any consistently good results from merging ad hoc runs (although the technique has been extensively used in routing to compensate for the overfitting which results from query optimization), but some experiments on the TREC–6 data showed that a considerable gain in most of the TREC statistics could have been obtained. Thus, the best Okapi TREC–7 run, ok7ax, is a linear combination in the proportions 2:2:1 of runs derived from "long", "medium" and "short" queries, document weights of the source runs having first been normalized by dividing by the mean weight. Merging *unexpanded* runs however did not improve average precision.

Table 1 gives the results of the official runs, and some others, after they were repeated using corrected indexes.[3] The average precisions for the original uncorrected runs are given for comparison. The best run, ok7ax, is now equal on average precision to the best non-Okapi TREC–7 run. In more detail, the effect of the correction on average precision for this run was as follows: for 19 topics the score was unchanged, 18 topics were worse (in one case, topic 397, the score went from 321 to 081), and the remaining 13 gained, but not by enough to compensate for the losses.

Comparing the expanded and non-expanded runs in Table 1 suggests that blind expansion worked rather well on the TREC–7 topics, giving gains of 23%–25% on runs derived from short and medium queries. For all the expanded runs the pilot search was on the full disks 1–5 database.

5 Adaptive filtering

5.1 General concepts

We assume for the filtering task the following scenario. There is a constant stream of incoming documents, and a number of requests represented by profiles. Profiles my be expected to remain in existence for some time, but to be modified, possibly at every incoming document. For each profile there will be a history, including information about

[3] The database used for generating expansion terms was also reindexed.

Table 1: Automatic ad hoc results. The first three are corrected versions of the official submitted runs; the *uncorrected* average precision is also given.

	AveP						
Run	corr	uncorr	$\geq$ med	P10	P30	RPrec	Rcl
ok7ax (combination of al, am and as)	296	(303)	47	552	419	323	717
ok7am: expanded from title + desc	281	(288)	47	530	403	309	688
ok7as: expanded from title only	253	(261)	37	472	375	286	605
ok7al: expanded from title + desc + narr	284	(290)	43	548	418	310	700
as ok7al but no expansion	248	(254)	41	518	378	288	647
as ok7am but no expansion	226	(233)	37	474	367	269	607
as ok7as but no expansion	200	(208)	24	438	333	246	543
as ok7ax but no expansion	247		41	510	395	279	655

Table 2: Automatic ad hoc run parameter details

All pilot searches used the disks 1–5 database. The number (S in equation 3) of assumed nonrelevant documents was 500 throughout, and there was a gap G of 500 documents between R and S. All final, but not pilot, runs used passage retrieval.

	Pilot				Final						
Run	R	k_1	b	k_3	k_1	b	k_3	k_5	k_6	# terms	topic term bonus
ok7al	15	1.6	0.8	1000	0.8	0.8	1000	1	64	30	×2.5
ok7am	10	1.4	0.6	1000	1.4	0.6	1000	1	64	30	×2.5
ok7as	6	1.0	0.5	0	1.5	0.6	0	1	128	20 + title terms	×3.5

documents retrieved, and about user judgments on documents. More specifically, we assume that we are operating within the parameters laid down for the TREC–7 filtering track. That is: at time $t = 0$ the system is switched on, with both a set of topics and the start of a stream of documents. There are no previous relevance judgments for these topics, and no previous documents in the stream, although topic representations (queries) may be tested for e.g. indicative term frequencies or document scores against a different database.

The document collection

We will further assume that the document collection is cumulated: at any stage of the process, we may conduct a new search on the collection as it has accumulated up to now, or discover from this collection any information we need (such as, for example, information about term occurrences). This historical information is taken as a more-or-less reliable guide (at least after we have accumulated a reasonable number of documents) to the future behaviour of documents and terms. This assumption clearly ignores the possibility of substantial changes in usage over time, e.g. as might result from a sudden surge of interest in a particular topic.

Thresholding and probability of relevance

The filtering task requires the system to make a binary decision about each incoming document, as to whether or not to send it to the user. In systems based on ranking by means of a matching function, this binary decision can be cast in the form of a threshold on the matching score. Systems using such methods often produce scores on an arbitrary scale, since only the order matters; thresholding could then be treated as an empirical problem on this arbitrary scale. An alternative would be to give the score an absolute value, perhaps in terms of probability of relevance.

The evaluation criteria defined for the TREC–7 filtering task make the connection explicit, by being expressed in terms of an absolute marginal probability of relevance. In what follows, the two problems (of setting thresholds and of obtaining absolute probability scores) will be regarded as strongly linked.

Adapting from sparse data

In the TREC–7 adaptive filtering task, the system needs to be able to adapt to small amounts of information. This adaptation should make use of non-evaluative information (essentially size of output and collection characteristics) as well as relevance judgments. While the old routing experiments allowed us to take a very empirical approach to optimizing queries on the large training set, without strong dependence on models, the adaptive case will require a very strong model base, to compensate for the sparsity of the information.

Probability of relevance in the probabilistic model

The probability of relevance will be denoted $P(R|D)$. Here D is taken to be all the information we have about a document – its description (a specific query is assumed).

The probabilistic model of retrieval is of course based on the probability of relevance, so one might expect that it would be easy to obtain an estimate of that probability within a probabilistic system. However, this is not the case. The traditional argument of the probabilistic model relies on the idea of ranking: the function of the scoring method is not to provide an actual estimate of $P(R|D)$, but to rank in $P(R|D)$ order.

In practice the document score calculated in Okapi (probably as in many other probabilistic systems) might be assumed to have some connection to the probability of relevance, but the exact functional relation is not derivable from the model. At best, we might assume that the score is linear in the log-odds of relevance, with the required two parameters unknown.

In the next two sections, we discuss the details of a method of defining and modifying thresholds in a filtering process. One objective is that the thresholds should be interpretable in terms of probability of relevance, so that the TREC utility functions can be used directly; hence the section on calibration. A second is that the method should minimise the twin dangers of getting nothing at all or far too much (the "selectivity trap"). This motivates some aspects of the calibration method and some of adaptation. Finally, the method must allow the threshold to be adaptive; indeed, it must adapt sensibly to changes in the query (e.g. reweighting or adding new terms) as well as to feedback from the user.

5.2 Calibration – basic ideas

Logistic regression

The one method that has been tried to calibrate retrieval scores, so that they might be interpreted as probabilities, is logistic regression on a training set [17]. We have used exactly this method, which fits well with the probabilistic model. The object is solely to calibrate a predefined scoring function, not (as in the Berkeley work) to improve the scoring function itself.

A reference point for the score

Scores in the Okapi system tend to be rather unpredictable, in the sense that different queries will have top-scoring documents in very different ranges of scores. In order to avoid the selectivity trap, we need to relate the score for a given query to some identifiable reference point. The two parameters we have considered for the purpose of defining this reference point are (a) the theoretical maximum score, and (b) the average score of the top 1% of the ranking (*ast1*). The first has the advantage of being definable at time $t = 0$, before we have any results; the second has the advantage of being strongly related to the actual behaviour of the particular query, in the part of the scale that we are interested in. We should achieve a reliable estimate of *ast1* after we have seen one or two thousand documents; it is a distributional parameter which does not depend on the size of the sample, so that when we have enough data to estimate it, the estimate should be reasonably stable.

Some preliminary experiments suggest that *ast1* is a slightly better reference point to use than the theoretical maximum score. Under TREC conditions, we cannot estimate it until a couple of weeks into the simulated time period, though under realistic conditions we might expect to be able to estimate it from the beginning. For the TREC experiment we have used the theoretical maximum score for the first week, and then switched to *ast1*.

Initial calibration

The method therefore involves using a model of the form

$$\log O(R|D) = \beta + \gamma \frac{score}{ast1} \tag{6}$$

where β and γ are initially estimated by logistic regression on a training set. Under TREC rules this has to be different sets of both documents and queries from the filtering test; we have used a collection consisting of disks 1–3 without AP and topics 51-150 (further details are given below). A second regression on the same set gives a model of the form

$$ast1 = \alpha_1 + \alpha_2 \, maxscore$$

where *maxscore* is the theoretical maximum score. This is used in the first week only to substitute for *ast1* in equation 6.

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:

$$\begin{aligned} c_i &= \beta + \gamma \frac{score_i}{ast1} \qquad (7) \\ p_i &= \frac{\exp c_i}{1 + \exp c_i} \end{aligned}$$

for some estimated β, γ and *ast1*.

5.3 Calibration – adaptation

Introduction

The initial calibration defined above will need adapting according to the output from the profile (in a profile-specific fashion). It is assumed that as time passes and documents are matched against the profile, the estimate of *ast1* will be adjusted. Any change to the profile will also require re-estimation of *ast1*, *via* a recalculation of the scores of previously-considered documents (in other words, the new profile will be run as a query against the accumulated collection for this purpose).

In addition, the adaptation mechanism must be capable of responding to either quantitative or qualitative evidence concerning the performance of the threshold setting. The qualitative method suggested here is intended to deal with both qualitative evidence and one direction of quantitative evidence. It assumes that relevance feedback information (positive or negative) is quickly obtained on all items submitted to the user, and that the submission of too many documents will be reflected in the relevance judgments. Quantitative adaptation in the other direction (nothing retrieved) is combined with another aspect – see section 5.4 below.

Qualitative calibration

Given relevance feedback data for a profile, probably a relatively small amount, it may be hard to get any reliable new estimate of γ in equation 6. However, an adaptive approach to the intercept β only may be appropriate. Since equation 7 predicts an explicit probability of relevance for every submitted document, we may compare these predictions against the outcome and use any systematic error to adjust β.

Suppose the user has seen and judged j items, of which r are relevant. Then a straight maximum-likelihood estimate for β (assuming γ given) would be obtained by solving the equation

$$r - \sum_{i=1}^{j} p_i = 0$$

Substituting for p_i via equation 7, and treating β as the single variable, we may apply a Newton-Raphson (gradient descent) method to improve the initial estimate of β.

A problem with this approach (a general characteristic of such maximum likelihood estimators) is that it goes haywire if either $r = 0$ or $r = j$; these events are very likely when we have only a small amount of relevance information. In order to deal with this problem, we may seek a Bayesian estimator which constrains the estimate by means of some prior distribution, so that it remains within sensible bounds, and allows the evidence to dominate only when there is enough of it. One way to achieve this is to assume that we have some (m) mythical documents whose estimated probabilities of relevance are correct; the value of m will determine the strength of the prior constraint. These mythical documents will have most effect on the estimation process if they are near the centre of the probability scale, and in what follows, they are assumed to have the score that would make their estimated probabilities of relevance equal to 0.5. Thus instead of j assessed and r relevant documents, we have $(j+m)$ assessed and $(r + \frac{1}{2}m)$ relevant documents, and $p_{j+1} = p_{j+2} = \ldots = p_{j+m} = 0.5$.

Including these mythical documents and spelling out the gradient descent method in full, 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:

$$\beta^{(n+1)} = \beta^{(n)} + \frac{r - \sum_{i=1}^{j} p_i^{(n)} + m \frac{1-\exp(\beta^{(n)}-\beta^{(0)})}{2(1+\exp(\beta^{(n)}-\beta^{(0)}))}}{\sum_{i=1}^{j} \frac{p_i^{(n)}}{1-p_i^{(n)}} + m \frac{\exp(\beta^{(n)}-\beta^{(0)})}{(1+\exp(\beta^{(n)}-\beta^{(0)}))^2}} \tag{8}$$

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

This method for dealing with the qualitative calibration problem also addresses one side of the quantitative problem: if the profile retrieves far too much, many of them are likely to be irrelevant, and the method will raise the threshold.

In the experiments, 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.

5.4 The value of feedback

Given that we have an explicit estimate of probability of relevance for any document, and a user-defined utility criterion interpreted as a threshold probability, the obvious strategy is simply to set this threshold from the word go. This, however, ignores a significant factor: every document assessed as relevant by the user provides additional information, from which we may make better estimates of probability of relevance for future documents. Therefore there may be long-term gains from lowering the threshold initially. We have at present no way of quantifying these possible gains; our solution to this problem, highly adhoc and pragmatic, is to define a "ladder" of thresholds. A profile starts at the bottom of the ladder, and climbs one rung for every relevant document found.

The ladder can be defined in terms of probability or log-odds. Two of the points correspond to the thresholds defined for the TREC filtering track as F1 and F3. Thus the target position on the ladder depends on this user-specific utility function. The ladder used is given in Table 3; it must be stressed that the points on it were pulled out of a hat, and do not in any way reflect any quantitative argument.

Table 3: The Ladder: selection thresholds

A profile starts at the bottom, and climbs one rung of the ladder for each relevant document found, up to the target utility point.

$P(R\|D)$	$\log O(R\|D)$	Utility point
0.5	0	
0.4	-0.4	F1
0.3	-0.9	
0.2	-1.4	F3
0.13	-1.9	
0.1	-2.2	
0.07	-2.6	
0.05	-2.9	
0.04	-3.2	

The top point is provided for fine tuning purposes as part of the experiment (to be described). 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 (at the appropriate utility point) 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).

The use of the ladder should have another beneficial side-effect. If we are in danger of retrieving nothing, setting the threshold low initially is likely to avoid the problem. We would not then be raising the threshold until our calibration has improved.

5.5 Model-free threshold discovery

The above arguments are intended to be used in the early life of a profile, when little relevance information is available. In past TRECs, in the routing experiment (where each profile is deemed to be already mature), we achieved good results by iterative optimisation of the queries. This technique was then extended to threshold setting for filtering [12], by running the optimised query retrospectively against the training set, and observing which threshold maximised the utility function concerned (the earliest or highest such threshold in the case of ties). Neither of these processes (iterative optimisation of query or threshold discovery) makes any appeal to the probabilistic model; instead, it treats the utility function as an arbitrary performance criterion to be optimised.

In the experiments to be described, we have attempted no iterative optimisation, although it would probably be desirable to do so at some point. However, in one of the runs below, where we have a reasonable amount of feedback for a particular profile, we initiated a similar threshold-discovery process on the current query.

5.6 Experiments

Initial queries (profiles)

Profiles were derived from the title, concepts, description and narrative fields of topics 51–150, using a database consisting of disks 1–3 without AP (835,248 documents) to derive term weights. The matching function used was BM25 with $k_1 = 2.0$, $b = 0.8$ and $k_3 = 1000$.

Initial calibration

Each topic was searched on this collection, and the top ranked documents were retrieved: the number was specified as 1% of the collection, namely 8352 documents. These were assigned their relevance values as defined for TREC; unjudged documents were assumed irrelevant. The entire set of 835,200 observations was put through the SPSS logistic regression program, with relevance as the dependent variable and $\frac{score}{ast1}$ as the single independent variable. The result was $\beta = -6.77$ and $\gamma = 2.68$. These results are in fact very typical – in a number of different experiments with different topics and documents, we usually obtained very similar results. However, one factor which made a significant difference was the chosen cutoff; if we stopped much earlier, say 1000 documents, we would get a smaller γ and a smaller absolute value of β. Further investigation of the statistical relationships here is desirable.

A regular (not logistic) regression of *ast1* (dependent) against *maxscore* (independent) gave $\alpha_1 = 55$ and $\alpha_2 = 0.036$. These four values were used in the experiment: α_1 and α_2 for the first week, γ throughout the experiment, and $\beta = \beta^{(0)}$ as initial value and seed for the adaptive method.

Adaptive procedure

Documents were processed initially in weekly batches. For speed, after the first six weeks, batches became four-weekly until the end of 1989; as there was no relevance information for 1990 the whole of this year was run as a single batch.

For the first week, the threshold was defined by the appropriate position on the ladder for the utility function being used, and the maximum score was used to provide an estimate for *ast1* and thereby for c_i for each document. From the following week, a direct estimate of *ast1* is available, and the usual ladder rule applied: starting at the bottom, 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 the bottom of the ladder.) *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 8 is then applied. The value of m in equation 8 was 5. The new value of β remains in force for this profile until the next invocation of the adaptive calibration.

Results

Two pairs of runs were submitted, ok7ff{13}2 and ok7ff{13}3; the first '1' or '3' refers to the utility function number. In the first pair the queries were retained unchanged throughout, so only the threshold was varied. In the second pair query terms were reweighted using equation 2 after each batch which contained one or more relevant documents.

Another difference was that in the first pair, the model-free threshold discovery process replaced β-adjustment and the Ladder threshold after at least 12 documents had been retrieved, of which at least 2 were relevant. This rather early invocation of the model-free process is likely to have resulted in the Ladder method being disabled for many topics. The second pair of runs did not make use of the model-free process.

Table 4 lists the number of topics obtaining at least the median score. For comparison, it also shows the results (relative to median scores) if no documents at all had been retrieved.

Table 4: Adaptive filtering results, number $\geq$ median

Condition	1988	1989	1990
Function F1, constant queries	28	40	40
F1, queries reweighted	28	35	31
F1, no output	42	25	42
F3, constant queries	35	38	42
F3, queries reweighted	35	39	39
F3, no output	24	24	35

With constant queries it appears that the threshold adaptation worked fairly well. The runs in which query terms were reweighted did not do so well. This requires further investigation. It is also worth noting that it is quite hard to do better than a simple "retrieve nothing" rule on a given utility function, particularly F1.

6 VLC

Source 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.

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", ... 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). The main (keyword) index had a total size of about 34 GB, split into six portions. It contained full within-document positional information.

Results

Table 5 summarizes the results. It seems strange that expansion on this very large collection should produce such poor results. More investigation is needed. The use of pairs of adjacent terms from the topic statements seems to have been slightly beneficial. An adjacent pair from a single topic subfield qualified for inclusion if it occurred in the (full) database at least five times as often as the expected number of times; pair weight for a word-pair AB was $log(n(A \text{ AND } B)/n(A \text{ ADJ } B))$ [18].

7 Interactive Okapi experiment

The system used for the interactive experiment for TREC-7 was exactly the same as that used for TREC-6. Essentially the system offers the user an incremental query expansion facility based on the idea of a working query. The working query consists of a combination of user-entered terms and system-generated terms extracted during relevance feedback. Extracted terms are displayed incrementally whenever the user makes a positive relevance judgement, provided the term occurs in at least three relevant documents and has a selection value (TSV) equal to two-thirds the average TSV of all the terms which have appeared in the working query to date. The ranked working query appears in a separate window and the maximum number of terms is defaulted to twenty.

The aim in TREC-7 was to build on the experimental conditions of the previous round, where ZPrise without relevance feedback achieved higher precision than the Okapi system with relevance feedback but recall was better

Table 5: VLC results

Condition	Collection	P20
No expansion, pairs, all topic fields	full	598
As previous	10%	369
As previous	1%	188
No expansion, no pairs, all topic fields	full	588
No expansion, pairs, title and description	full	541
As previous	10%	376
As previous	1%	180
No expansion, no pairs, title and description	full	525
Expansion 30 docs, all topic fields	full	545
As previous	10%	357
As previous	1%	192
Expansion 15 docs, all topic fields	full	538
Queries as ad hoc run ok7al1, all topic fields	full	562
Expansion, title and description	full	509
As previous	10%	357
As previous	1%	202

in Okapi. The focus here was thus on a three way comparison between two versions of Okapi, one with relevance feedback and one without and with ZPrise as a control. In order to optimise on the number of searcher participants, the experiment was conducted on two sites, City and Sheffield. Eight subjects at each site conducted a total of eight searches each, four on each of two systems. Subjects at City carried out searches on the version of Okapi without feedback and on ZPrise, whilst at Sheffield they searched on both versions of Okapi, with and without feedback. Hence across the sites a total of sixty-four searches were carried out on Okapi without feedback and thirty-two on Okapi with feedback as well as thirty-two on ZPrise. Unfortunately due to the constraint within the experimental design of having to limit the number of systems subjects could search, only half as many searches were undertaken on the feedback version of Okapi than on the one without feedback.

All sixteen subjects on both sites were new to the systems concerned. They included eight masters and five doctoral students, as well as three research assistants all of whom were recruited within the respective information science departments.

The overall results for instance recall and precision for the three systems are given in Table 6.

Table 6: Interactive task results

System	Relevance feedback	Mean Recall	Mean Precision	Number of searches
Okapi	No	.383	.653	64
Okapi	Yes	.397	.692	32
ZPrise	No	.399	.714	32

ZPrise clearly outperformed the Okapi system without relevance feedback on both measures. Although it also achieved better results than the version of Okapi with relevance feedback, recall was only very marginally better and the main difference is in the precision. We conclude that although Okapi with relevance feedback is better than with no relevance feedback, in an interactive environment the query expansion facility does not appear to be achieving the desired level of performance.

References

[1] http://trec.nist.gov/act_part/guidelines/7guides_adhoc.html

[2] 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.)

[3] 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.)

[4] 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

[5] Robertson, S.E. *et al.* Okapi at TREC. In: [19], p21–30.

[6] 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.

[7] Robertson, S.E. *et al.* Okapi at TREC–2. In: [20], p21–34.

[8] Robertson, S.E. *et al.* Okapi at TREC–3. In: [21], p109–126.

[9] Robertson, S.E. *et al.* Okapi at TREC–4. In: [22], p73–96.

[10] Beaulieu, M.M. *et al.* Okapi at TREC–5. In: [11], p143–165.

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

[12] Walker S. *et al.* Okapi at TREC–6: Automatic ad hoc, VLC, routing, filtering and QSDR. In: [13], p125–136.

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

[14] Robertson, S.E. and Walker S. On relevance weights with little relevance information. In *Proceedings of the 20th annual international ACM SIGIR conference on research and development in information retrieval.* Edited by Nicholas J Belkin, A Desai Narasimhalu and Peter Willett. ACM Press, 1997. p16–24.

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

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

[17] Cooper, W.S. and Gey, F.C. Experiments in the Probabilistic Retrieval of Full Text Documents. In: [21], p127–134.

[18] Robertson, S.E. Unpublished note on phrase weighting. 1996.

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

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

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

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