Okapi at TREC–5

M M Beaulieu* M Gatford* Xiangji Huang* S E Robertson* S Walker*
P Williams*

Jan 31 1997

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

1 Introduction: summary

General

City submitted two runs each for the automatic ad hoc, very large collection track, automatic routing and Chinese track; and took part in the interactive and filtering tracks. There were no very significant new developments; the same Okapi-style weighting as in TREC–3 and TREC–4 was used this time round, although there were attempts, in the ad hoc and more notably in the Chinese experiments, to extend the weighting to cover searches containing both words and phrases. All submitted runs except for the Chinese incorporated run-time passage determination and searching. The Okapi back-end search engine has been considerably speeded, and a few new functions incorporated. See Section 3.

Automatic ad hoc (and VLC track)

After a good deal of rather fruitless exploration, including further work with two-term phrases, the method finally used was much the same as in TRECs 3 and 4: expansion using terms from the top documents retrieved by a pilot search on topic terms. City96a1 used all topic fields and city96a2 the ⟨desc⟩ field only. The results appear disappointing. Additional runs made since the relevance information became available seem to show that we would have done better without expansion. Two runs using the method of city96a1 were also submitted for the Very Large Collection track. See Section 4.

Automatic routing (and filtering track)

These runs used elaborated versions of City's TREC–3 and 4 methods. The training database and its relevant documents were partitioned into three parts. Working on a pool of terms extracted from the relevant documents for one partition, an iterative procedure added or removed terms and/or varied their weights. After each change in query content or term weights a score was calculated by using the current query to search a second portion of the training database and evaluating the results against the corresponding set of relevant documents. Methods were compared by evaluating queries predictively against the third training partition. Queries from different methods were then merged, and the results evaluated in the same way. Two official runs were submitted, one (city96r2) using the best *method* to produce a set of queries from two halves of the training database, the other (city96r1) using the best query for each topic from those produced during training. These runs appear to have been fairly successful. Three filter track runs and thresholds were also produced using the queries from city96r1 with thresholds derived from the "predictive" portion of the training database. See Section 5.

*Centre for Interactive Systems Research, Department of Information Science, City University, Northampton Square, London EC1V 0HB, UK. email {mmb,mg,xjh,ser,sw,pw}@is.city.ac.uk

Chinese track

City took part in the Chinese track for the first time. Two runs were submitted, one based on character searching and the other on words or phrases (words being identified by lookup as there are no word-delimiters in Chinese). Much of the work involved investigating plausible methods of applying Okapi-style weighting to phrases. (Section 6.)

Interactive

The task definition and the type of users conducting the searches were the main differences for this round of interactive TREC. Past performance in interactive searching had shown that the query expansion facility in the Okapi system was effective in retrieving additional similar items. However, the task of finding different aspects proved to be more taxing. To cater for the different type of users, the GUI used had added functionality to (a) allow users to reverse certain actions taken, and (b) to allow the addition of terms extracted from relevant documents to the query to be handled incrementally rather than requested explicitly by the user. The results were rather disappointing and clearly indicate that we need to carry out further experiments to establish an effective method of conducting such a task. (Section 7.)

2 Background

The search systems City have used for TREC are descendants of the Okapi systems which were 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 query expansion [1, 2, 3].

For TREC–1 [4], the low-level search functions were generalized and split off into a separate library — the Okapi Basic Search System (BSS). Interfaces or scripts for batch processing access the BSS using a simple command language-like protocol.

City's 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 new methods of routing query enhancement were developed [6, 7]. Our TREC–3 automatic routing and ad hoc results were relatively good.

TREC–4 [8] did not see any major developments. Routing term selection methods were further improved. In view of the very concise topic statements for the ad hoc, a try was made at manual tweaking of the automatically derived queries; this was a sad failure.

City have always taken part in the Interactive TREC tracks. The TREC–1 system was a simple command-language version of the old pre-TREC Okapi systems. Different interfaces have been written each year, partly in conformance with whatever was the current definition of the interactive tasks. But the interfaces have also become steadily more colourful and GUI–oriented.

3 The system

3.1 The Okapi Basic Search System (BSS)

The BSS, which has been used in all City's 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 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.

The BSS undergoes continual small changes (not always for the better). This time a lot of work was done on trying to speed it up, particularly with regard to set combination, with the result that under reasonable conditions routing term selection runs at about four times the speed it did during TREC–4.

[1] Now the University of Westminster.

Weighting functions

The functions available were identical to those used in TRECs 3 and 4. The only ones used during TREC–5 were varieties of the **BM25** function [7, Section 3.2]

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

where

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

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

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, k_2 and k_3 are parameters which depend on the database and possibly on the nature of the topics. For the TREC–5 experiments, typical values of k_1, k_3 and b were 1.0–2.0, 8 and 0.6–0.75 resp., and k_2 was zero throughout
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 is the document length (arbitrary units)
$avdl$ is the average document length.

When k_2 is zero, as it was for all the TREC–5 experiments except the Chinese, equation 1 may be written in the simpler form

$$\sum_{T \in \mathcal{Q}} w^{(1)} \frac{(k_1+1)tf}{K+tf} \frac{(k_3+1)qtf}{k_3+qtf} \tag{3}$$

3.2 Hardware

Most of the TREC processing was done on two Suns: an SS20 with 160 MB of memory and an SS10 with 320 MB, both single-processor. Most of the 25 GB of disk was attached to the SS10.

3.3 Databases

Disks 1 and 2 were used for ad hoc "training"; disks 1–3 + the TREC–4 routing data for the routing training. Apart from the official TREC–4 ad hoc and routing test databases, a new database of disks 1–4 was made for the VLC task. The usual three–field structure was used, common to all source datasets, as for all TRECs after the first. The first field was always the DOCNO and the third field contained all the searchable text, mainly the TEXT portions but also headline or title-like material for some datasets and documents. The second field was unindexed (and unsearchable) and so only (possibly) useful for display to users of the interactive system. It was empty except in the case of SJM, when it contained the DESCRIPT field; and the Ziff JOURNAL, AUTHOR and DESCRIPTORS fields. All the databases were set up in such a way that search-time passage determination and searching could be done.

4 Automatic ad hoc

4.1 Outline

Most of the experiments were done using the TREC–3 ad hoc topics (151–200) and database (disks 1 & 2). In TRECs 3 and 4 the better automatic ad hoc runs involved "blind" feedback from the top few documents retrieved

Table 1: Automatic ad hoc: some training results, topics 151-200 on disks 1 and 2

Method	AveP	P5	P20	P30	P100	R-Prec	Rcl
"Long" topics							
Baseline: unexpanded topic term run	0.349	0.740	0.645	0.603	0.445	0.391	0.688
As above + passages	0.350	0.720	0.640	0.591	0.444	0.401	0.696
As row 1 but with pairs, plausibility threshold 10	0.348	0.744	0.647	0.597	0.443	0.392	0.690
As above but with passages	0.350	0.708	0.638	0.587	0.442	0.400	0.695
Expansion: 60 terms from 30 docs	0.408	0.760	0.686	0.647	0.483	0.429	0.744
As above + passages	0.415	0.760	0.687	0.651	0.482	0.432	0.752
As above but 50 terms	0.414	0.772	0.687	0.653	0.482	0.435	0.756
As above but 55 terms	0.416	0.768	0.690	0.649	0.485	0.437	0.755
TREC–3 citya1 official result for comparison	0.401	0.740		0.625	0.476	0.422	0.739
"Short" topics							
Unexpanded topic term run	0.225	0.612	0.492	0.443	0.322	0.300	0.514
As above but with pairs, plausibility threshold 10	0.221	0.612	0.477	0.441	0.313	0.293	0.513
Expansion: 30 terms from 15 docs + passages	0.311	0.680	0.553	0.517	0.388	0.351	0.607

Table 2: Automatic ad hoc: some test results, topics 251-300 on disks 2 and 4

Method	AveP	P5	P20	P30	P100	R-Prec	Rcl
"Long" topics							
Unexpanded topic term run	0.231	0.508	0.360	0.322	0.206	0.273	0.473
As above but with passages	0.241	0.484	0.371	0.323	0.207	0.288	0.482
As row 1 but with pairs, plausibility threshold 10	0.233	0.504	0.366	0.326	0.210	0.277	0.481
As above but with passages	0.240	0.508	0.378	0.330	0.208	0.280	0.486
Expansion: 55 terms from 30 docs + passages (city96a1)	0.216	0.428	0.358	0.324	0.216	0.248	0.536
As city96a1 but terms from "best" passage of each doc	0.223	0.456	0.373	0.342	0.218	0.253	0.531
city96a1 evaluated on 47 topics	0.229	0.455	0.380	0.344	0.229	0.264	0.536
"Short" topics							
Unexpanded topic term run	0.152	0.368	0.283	0.248	0.157	0.193	0.383
As above but with passages	0.171	0.376	0.274	0.244	0.160	0.214	0.390
Expansion: 30 terms from 15 docs + passages (city96a2)	0.175	0.336	0.284	0.260	0.178	0.213	0.469
city96a2 evaluated on 47 topics	0.185	0.357	0.301	0.276	0.190	0.227	0.469

Table 3: Very large collection track: official results on test database

Method	P20
city96vlc1: 55 terms from 30 expansion docs, disks 2 & 4	0.262
city96vlc2: as city96vlc1 but disks 1–4	0.467

by a topic-term pilot search. In TREC–4 we tried manual "tweaking" of the queries thus obtained; this was not a success. Casting around for something else to try (other than procedures which involve non-statistical features of terms), it was decided to have another go at adding adjacent topic term pairs to the queries. We tried this in TREC–2 with neutral results, but this time pairs were restricted to those which occurred (in the databases) with frequency much higher than the expected value and a new pair-weighting formula was tried. However, results were still no better than those obtained using single terms only. In desperation, two long shots were tried: a form of term selection analogous to that used in the automatic routing, but without relevance information; and merging results obtained by different methods. Neither of these gave any improvement. However, repeats of TREC–3 runs gave results which were slightly better than our official TREC–3 run.

Training and official results are given in Tables 1 and 2 respectively, and City's position relative to the median scores in Table 6. VLC results are in Table 3.

4.2 Adjacent pairs

If s and t are two terms with frequencies $n(s)$ and $n(t)$ respectively the *plausibility* of the adjacent pair st is $n(st) \times C/(n(s)n(t))$, where C is the total number of tokens in the collection.

Example

The Narrative field of topic 2:

> To be relevant, a document must discuss a currently proposed acquisition (which may or may not be identified by type, e.g., merger, buyout, leveraged buyout, hostile takeover, friendly acquisition). The suitor and target must be identified by name; the nationality of one of the companies must be identified as U.S. and the nationality of the other company must be identified as NOT U.S.

gives, at plausibility threshold 10, “friendli acquisit” with plausibility 44, “hostil takeover” with 939 and “leverag buyout” with 4472 (and of course single terms).

4.2.1 Weights for term pairs

If a query contains a term pair as well as the individual terms, and all three are given “natural” (e.g. inverse collection frequency) weights, some allowance ought to be made for the mutual dependency between each term and the pair. In TREC–2 we tried several ways of adjusting pair or term weights [6], but these had little effect on the results. Steve Robertson suggested that a plausible pair weight would be $\log(n_{s\mathrm{AND}t}/n_{s\mathrm{ADJ}t})$ (where the ns this time are the numbers of posted documents), and this appeared to work a little better than giving both pairs and single terms their $w^{(1)}$ weights. However, the results were about the same as those from giving pairs just a “small” bonus weight.

4.3 Other ad hoc query generation methods

Query expansion

The top R documents from the pilot search were output and all terms other than stop and semi-stop terms extracted. These were $w^{(1)}$–weighted in accordance with their occurrence in the top documents, with terms occurring more than once in the topic statement loaded with a k_3 value (usually either 8 or 1000 — varying within this range made little difference). The terms were then arranged in descending RSV value, and the required number taken from the top of the list, any term with RSV less than 3 being discarded.

Term selection after expansion

Some selection runs were done as follows. The top documents (typically 50 of them) retrieved by an assumed good set of queries from an expansion run, were treated as relevant, and a routing-type selection procedure done on terms from one of the expanded term sets. Scoring was usually on the top 100 documents retrieved at each stage in the selection procedure. Note that only one database was used instead of at least two as in the routing. The best runs were about as good as the best runs obtained by other methods.

Iterated expansion

A few runs were done using a second iteration of the expansion procedure. They were better than expected but not quite as good as the best singly-expanded runs.

Merging runs or queries

Several pairs of the better runs were merged, with results which were not very encouraging. However, the union of the best selection run and the best plain expansion run did give marginally the best results on most measures.

4.4 Very large collection track

Two runs were submitted: the baseline run on disks 2 and 4; and a “full” run on disks 1, 2, 3 and 4. The city96a1 run was re-used for the baseline run city96vlc1 (see Section 4.5), and the same method against the four-disk database for city96vlc2.

4.5 Ad hoc results

The new topics consist, like the TREC–3 ones, of title, narrative and description only. One of the runs submitted had to be based on the description field only.

City96a1 (and city96vlc1 and 2), using all topic fields, was based on 30 feedback documents, with a bonus to all topic terms (all that were in the expanded termset), any terms which did not occur in at least 3 of the 30 feedback documents were rejected. Each query consisted of the top 55 terms, and passage searching was used. City96a2 (⟨desc⟩ only) used the same method as city96a1 but with 15 feedback documents and 30 terms per query.

Trial results (Table 1), done on the TREC–3 ad hoc topics and test database, showed a small improvement over our best TREC–3 result. Query expansion was markedly beneficial, passage searching slightly beneficial and the use of topic term pairs neutral. The methods which gave the best trial results were used on the test database and topics for the official runs (Table 2).

The official results showed that query expansion gave higher recall but poorer low precision, while the effect of passages and pairs was much as in the trials. Expansion appeared to work a little better when terms were taken only from the best passage of each expansion document.

The VLC runs (Table 3) used the same method as city96a1. Note that the precision at 20 documents for the city96vlc1 run is not the same as that for city96a1 although the queries were the same. There appear to be two reasons for this:

1. the VLC assessments covered only documents retrieved in participants' "full" VLC run: thus, some documents retrieved in "baseline" runs were not assessed;
2. there are nearly 300 disagreements between the VLC relevance judgments and the official TREC–5 disks 2 and 4 qrels.

5 Automatic routing

5.1 Training database

The training database was constructed from disks 1–3 together with the TREC–4 routing set (without its Ziff, which is the same as disk 3's). This contains about $1\frac{1}{4}$ million documents. It was systematically partitioned into sub-databases in two ways: into halves, by selecting odd or even record numbers; and into thirds analogously.

5.2 Outline of procedure

Some control runs were done first, using topic terms and no relevance information (including some runs incorporating pairs of adjacent terms), and using topic terms reweighted in the light of relevance information.

The general procedure for finding a good query term selection method was similar to that of TREC–4, except that the partition of the training database into three parts was used.

term pool: a pool of terms was extracted from the known relevant documents in one third of the training database (the topic statements were not used[2]);

term selection: queries were generated by selection from the term pool using the second partition of the database; a number of different selection methods were tried;

evaluation: the queries resulting from each selection method were evaluated predictively against the third partition.

Having found a good selection method using the partition of the database into thirds, the partition into halves may be used to produce a final set of queries, thus making use of more of the available relevance information. Relevant documents from one half were used to produce the term pool and term selection was done on the other half. A set of queries generated by this method was submitted as City96r2. The filter runs and city96r1 consisted of queries produced using the partition into thirds.

Some training results are given in Table 4, official results in Table 5, with comparative results in Table 6.

5.3 Term selection measure and algorithms

In TREC–4 a number of different scoring measures were tried. None performed better than the TREC average precision, and this was the only measure used for TREC–5. However, a number of new selection algorithms were tried.

For all selection algorithms, the available terms were first arranged in descending RSV[3] [9] order to form a term pool of a given size. An initial query (sometimes empty) was formed from terms at the top of the pool, then terms

[2]During term *selection* a bonus was generally given to topic terms by using a non-zero value of k_3 in equation 3

[3]Note that RSV stands for Robertson Selection Value not Retrieval Status Value.

Table 4: Automatic routing: predictive results on part of training database

Method	AveP	P5	P30	P100	R-Prec	Rcl
Topic term runs						
Topic single terms, $k_3 = 1000$	0.309	0.616	0.507	0.387	0.365	0.717
Topic, single terms + pairs with Robertson pair wts (see 4.2.1), $k_3 = 1000$	0.315	0.616	0.511	0.391	0.370	0.723
Topic single terms $F4$-reweighted, $k_3 = 0$	0.345	0.688	0.563	0.426	0.392	0.756
Runs with term selection						
Best fixed-weight run, pool 298, $k_3 = 3$	0.493	0.852	0.731	0.535	0.503	0.842
All weights varied on each iteration + 10 final passes, pool 100, $k_3 = 0$	0.496	0.852	0.724	0.535	0.508	0.848
Weights varied every 20-change, term pool 298, $k_3 = 3$	0.497	0.848	0.735	0.538	0.507	0.849
The above run merged with its inverse	0.504	0.852	0.737	0.543	0.507	0.854
3 runs and their inverses all merged	0.529	0.884	0.745	0.562	0.521	0.869
(The official city96r2 used the above method using half-databases)						
Individually "best" query for each topic (individualized k_1 and b)	0.547	0.924	0.768	0.567	0.537	0.872
(The official city96r1 used the above method)						

Table 5: Automatic routing: official results on test database

Method	AveP	P5	P30	P100	R-Prec	Rcl
city96r2 on 39 topics	0.386	0.661	0.532	0.380	0.415	0.736
city96r1 on 39 topics	0.372	0.692	0.508	0.371	0.405	0.719
city96r2 on 45 topics	0.347	0.582	0.465	0.331	0.360	0.736
city96r1 on 45 topics	0.329	0.600	0.444	0.322	0.351	0.719

Table 6: Automatic routing and ad hoc: comparisons with other participants' results

Routing (45 topics)												
	Precision at 100			Precision at 1000			Average precision			Precision at R docs		
	$\geq$ median	=best	=worst	$\geq$ med	=best	=worst	$\geq$ med	=best	=worst	$\geq$ med	=best	=worst
city96r1	43	5	0	44	12	1	41	3	1	45	9	0
city96r2	45	6	0	45	17	0	43	3	0	45	10	0

Ad hoc (50 topics) The medians for city96a1 are from the "long" topic runs, those for city96a2 from the "short".									
	Precision at 100			Precision at 1000			Average precision		
	$\geq$ median	=best	=worst	$\geq$ med	=best	=worst	$\geq$ med	=best	=worst
city96a1	41	12	0	39	18	1	35	8	0
city96a2	39	4	1	42	10	1	34	4	1

were added or removed in accordance with the algorithm being used. After each addition or removal a score was calculated (standard TREC average precision at cutoff 1000 on the set of known relevant documents for whichever sub-database was being used for selection).

In TREC–4, three types of iterative selection procedure were tried. These were, briefly

Find best. In each iteration every undropped term was examined. At the end of the iteration the highest scoring term was added to or removed from the query;

Choose first positive. In each iteration the first term which increased the score by an amount greater than a predetermined threshold percentage was added to or removed from the query;

Choose all positive (CAP). Every term was considered and immediately added to or removed from the query if it increased the score by more than the threshold. An iteration ended when all "live" terms had been considered.

There were a number of stopping rules, but, in practice, after a smallish number of iterations there would be no single term whose addition or removal increased the score.

Surprisingly, the last method (CAP) appeared from the TREC–4 work to be the best, and this has been borne out in TREC–5. So CAP was used for all runs after some preliminary trials. The new features in term selection were

- varying term weights during selection
- merging queries produced by different selection methods and database combinations.

Varying term weights during selection

Even with our increased computing power since TREC–4, anything at all comprehensive was out of the question. Various combinations of the following were tried (all during CAP selection):

- the weight of the term under consideration could be decreased or increased by given proportions; in some cases these proportions were varied (uniformly over all terms) during a selection run; typical proportions were $w \to .67w$ and $w \to 1.5w$
- a term not already included was tried at its original weight, then the lower weight and finally the higher weight; the term's weight was set to the first one (if any) which gave a score increase
- every non-included term was tried at up to three weights
- all included terms were retried at a lower then a higher weight every time the current query had changed by a certain amount
- after the selection procedure had stopped a number of weight variation passes were done, with the variation proportions changed after any pass which gave increase in score.

Watching the progress of a selection run it often appeared that weight variation was going to be highly beneficial. However, term selection/rejection and weight variation tended to cancel one another out, and the net benefit from weight variation appears to be small. It seems that it is better not to be too radical: the best results are probably obtained by varying weights after the termset has changed by 20 or so, together with some passes at the end.

Merging runs

Merging a number of runs, usually in pairs – a run with its inverse – made a very significant improvement in the predictive results. The best single query set came from merging three pairs of runs. It seems likely that this is due to a reduction in the over-fitting effect which must result from such extensive use of the relevance information. When there are a large number of known relevant documents it may be advantageous to partition the training database further, using more than one partition during the term selection process.

5.4 Filtering

The city96r1 queries were run against the "predictive" third of the training database (the portion of the database not used as term source or for term selection). Each of the three utility functions ($R-3N$, $R-N$ and $3R-N$, where R is the number of relevant documents retrieved and N the number of nonrelevant) was evaluated after each document was retrieved, and the lowest weight which corresponded to a maximum of each of the functions was submitted as threshold. This simple procedure produced fairly satisfactory results.

5.5 Results

Table 4 summarizes the results of a few of the training runs. It appears that weight variation as we tried it does not give much improvement over straight selection/rejection of terms. Runs using a considerable amount of weight variation on relatively small term pools (e.g. row 5 of the table) seem to give as good results as straight selection on large term pools (and give shorter queries).

The break-through (if that is the right way to describe it) came when we tried merging queries with their inverses. Although different and stochastically independent databases were used as source of terms and for selection, it is probably the case that the selected termset is closely "fitted" to the database used for selection. Merging queries from different selection databases must compensate for this effect. More investigation is needed. It may be that results could be improved by partitioning the training database still further, provided there are enough known relevant documents.

The official results (Table 5) look poor compared with previous years. However, the test database was very small, rather uniform and of limited scope. There were no relevant documents for at least four of the topics, and very few for six others. (Another topic missed assessment.) NIST sensibly evaluated results both on the 45 "non-empty" topics and on the 39 with at least eight relevant documents. In comparison with median scores from all full participants, city96r2 was as good as or higher than all medians on 43 of 45 topics and lower on average precision only on 2 of the low-posted topics; city96r1 slightly less good (Table 6).

6 Chinese track experiment

In this section, we describe the experiments with our Chinese text retrieval system modelled on Okapi. The experiments focus on applying the probabilistic model to Chinese text retrieval and comparing between two document-indexing approaches: word and single character approaches. About 175 megabytes of Chinese text and 28 Chinese topics were made available for runs in the ad hoc environment. Two City Chinese runs are submitted for evaluation, which are city96c1 for the word approach and city96c2 for the character approach. For evaluating the BM25 function, two new runs city96c12.BM25 and city96c24.BM25 are also generated for word and character approach respectively.

6.1 Automatic indexing

There are two kinds of methods for indexing Chinese text. One indexing method is to use words and phrases in texts as indexing terms to represent the texts. We refer to this method as the word approach. Since the Chinese words in a text are not separated by spaces, text segmentation, which divides text into linguistic units (commonly words and phrases), is regarded as a necessary precursor of word approach system [10]. To segment the Chinese text, a dictionary with about 70,000 Chinese words and phrases was built and a longest match algorithm for text segmentation was implemented. This algorithm is to extract a string of a certain number k (usually $k = 7$) of characters and to search for it in the dictionary. If it exists in the dictionary, the process continues with the next k-character text string. If it doesn't, the last character in the string is removed and the new string is then searched for in the dictionary. Figure 6.1 shows the longest match algorithm. According to the word segmentation produced by this algorithm, the Chinese TREC texts consist of approximately 43.6 million words and phrases.

The other method for indexing texts is based on single characters, in which texts are indexed by the characters appearing in the texts [11]. Usually each Chinese character is given a unique two-byte machine code. Therefore, it is a purely mechanical procedure to segment TREC Chinese text into single characters.

6.2 Probabilistic weighting functions

Basic weighting function

The basic weighting functions are based on the Robertson-Sparck Jones weight [5], which approximates to inverse collection frequency (ICF) shown in equation 4 when there is no relevance information.

$$ICF = \log \frac{N - n + 0.5}{n + 0.5} \tag{4}$$

The function above can be extended to include document length and within-document and within-query term frequency as providing further evidence, which have been shown to be highly beneficial in English text retrieval [12].

Figure 1: The longest match algorithm.

ALGORITHM: The Longest Match
INPUT: A string of Chinese characters $a_1 a_2 \cdots a_k$;
OUTPUT: A set of Chinese words and phrases

```
begin
    let i be 1 and j be k
    while (i <= k)
    begin
        if a_i is a digital character or English character
        then call special_string_processing(a_i ··· a_j);
        else if there is no character a_i in the index file of word segmentation
                dictionary or i equals to j
            then put a_i into segmentation result file;
                increase i by 1;
            else if there is a string a_i ··· a_j in the lexicon file of word
                    segmentation dictionary
                then put a_i ··· a_j into segmentation result file;
                    let i be j;
                else decrease j by 1;
    endwhile ;
end
```

One of these extended functions that has been implemented in our Chinese text retrieval system is:

$$w = \frac{tf}{(k_1.dl/avdl + tf)} \times \log \frac{N - n + 0.5}{n + 0.5} \times \frac{qtf}{(k_3 + qtf)} \quad \oplus \quad k_2 * nq * \frac{(avdl - dl)}{(avdl + dl)} \tag{5}$$

where

N is the number of indexed documents in the collection,
n is the number of documents containing a specific term,
tf is within-document term frequency,
qtf is within-query term frequency,
dl is the length of the document,
$avdl$ is the average document length,
nq is the number of query terms,
the k_i are tuning constants, empirically determined, and
the $\oplus$ in the formula indicates that the following component is added only once per document, rather than for each term.

This formula now defines the weight of a term (that is, the contribution of that term to the total score for the document) in the context of an individual document. This is the formula referred to as BM11 in our TREC–3 work [7]. Some results are shown below with the later BM25 function

Phrase weighting

Okapi's usual method for dealing with phrases in English has been as follows. If a phrase is identified at indexing time, it is indexed as a single unit, and the component words of the phrase are not normally indexed. If it is identified only at search time, it may be searched as a single unit by means of an adjacency operator. Again the words may not be normally searched on their own, although some variations have been tried.

It would be reasonable to seek a method that allowed a match on a phrase or on either or all of its component parts. It might then be reasonable to assume that, for a phrase consisting of two terms $t_1 t_2$,

$$w(t_1), w(t_2) \leq w(t_1 \wedge t_2) = w(t_1) + w(t_2) \leq w(t_1 \ adj \ t_2).$$

Here *adj* is the adjacency operator. The equation in the middle represents the usual scoring method for the $\wedge$ (and) operator: the score assigned to a document is the sum of the weights of the matching terms. The assumption is that the phrase carries a higher score than the two terms separately.

The problem of devising such a method consistent with the probabilistic model has not generally been tackled in text retrieval in English. But for text retrieval in Chinese, the problem is likely to be more serious than it is in English. This would be so in a word-based system, since there are likely to be considerable differences between Chinese speakers as to whether a given combination of characters is actually a single word or a phrase. But it is even more serious in a character-based system, where one would want a match on a complete word or phrase to carry a higher score than matches on the component characters.

Weighting functions for Chinese TREC experiment

We need, therefore, a weighting function which will enable us to cope with phrases in a word-based system, and with words or phrases in a character-based system. Suppose, then, that we have a sequence of j adjacent units $t_1t_2...t_j$ (characters or words) constituting a single larger unit (word or phrase). In the Robertson/Sparck Jones model, each unit (large or small) has a "natural" weight, given by the formula; let these be w_{t_i} and $w_{t_1t_2...t_j}$ respectively. Then we can suggest a number of weighting functions which satisfy (or will probably satisfy) the above condition or something like it. Table 7 gives a few such functions which have been implemented.

Table 7: Weight methods

Weight methods	$w(t_1\ adj\ t_2\ adj \cdots adj\ t_j)$	$w(t_1 \wedge t_2 \wedge \cdots \wedge t_j)$
$Weight_0$	$\sum_{i=1}^{j} w_{t_i} + j^k * w_{t_1t_2...t_j}$	$\sum_{i=1}^{j} w_{t_i}$
$Weight_1$	$w_{t_1t_2...t_j}$	$\sum_{i=1}^{j} w_{t_i} - j^k$
$Weight_2$	$w_{t_1t_2...t_j}$	$\sum_{i=1}^{j} w_{t_i}$
$Weight_3$	$\sum_{i=1}^{j} w_{t_i} + log\frac{\#(t_1 \wedge t_2 \wedge \cdots \wedge t_j)}{\#(t_1\ adj\ t_2\ adj \cdots adj\ t_j)}$	$\sum_{i=1}^{j} w_{t_i}$
where $\#(t)$ indicates the number of documents containing the term t and k is another tuning constant		

None of these functions has a very strong probabilistic basis, beyond the attempt to satisfy the condition. Each has two versions, one applied to words and the other to characters, so there are eight functions in all. The "natural" weights come from equation 5: w_{t_i} is obtained by applying the equation to term t_i, and $w_{t_1t_2...t_j}$ comes from applying the same equation to the combined term $t_1t_2...t_j$.

In the Chinese TREC experiments, we chose two retrieval results which use $Weight_0$ as the weighting method for evaluation. Further experiments suggested that $Weight_1$ might be preferable, and the BM25 results reported below also use $Weight_1$.

Query processing

There are two kinds of automatic methods for query processing in our Chinese text retrieval system. One is character-based method, which uses character, character pairs and multi-character adjacencies as retrieval keywords. The other is word-based method. Given a word segmentation system, similar methods for characters can be applied to words as a way of allowing phrases to contribute to the matching. We refer to this method as the word-based query processing. Table 8 shows nine methods which have been implemented in our systems.

We use M_5 and M_6 which are word-based in our Chinese TREC experiments. The text in all parts of the Chinese topics were treated in the same way. Text segmentation is applied to segment the topics into the words and phrases. Only pairs of the adjacent segmented terms which occur in the same subfield of the topic are regarded as new potential phrases. All these segmented terms and new potential phrases are ranked by the values of their weights multiplied by the within-query frequencies. The top 36 terms are used as keywords for searching the word index and for searching the character index using the adjacency operator.

Table 8: Illustration of retrieval algorithms

Algorithms	Query processing	Database	Number of weighting methods
M_0	characters	character	1
M_1	characters+character-pairs	character	8
M_2	characters+multi-character adjacencies	character	8
M_3	words	character	8
M_4	words	word	1
M_5	words+word-pairs	character	8
M_6	words+word-pairs	word	4
M_7	words+multi-word adjacencies	character	8
M_8	words+multi-word adjacencies	word	4

6.3 Experimental results

The two submitted Chinese results are shown in Table 9 and Table 10. Both runs are around the median score of the groups participating in the Chinese task. The results for the two runs based on the BM25 weighting function are shown in Table 10. By using a combination of the BM25 function and $Weight_1$ for phrases, the word and character approach perform 18% and 21% better than the word approach city96c1 respectively. Taking the word approach city96c1 as baseline, the character method city96c2 shows a slight improvement. However, the change to BM25 (essentially the equation given as 1) produces a much greater improvement.

Table 9: Comparative Chinese results

Run	Best	> median	= median	< median
$city96c1$	4	9	3	12
$city96c2$	0	11	7	10

Table 10: Chinese ad hoc results

Run	Average Precision	Total Rel Retrieved	R Precision	Precision 100 docs
$city96c1$	0.3256 (+0.0%)	1891	0.3418	0.2900
$city96c2$	0.3336 (+2.5%)	1950	0.3493	0.2939
$city96c12.BM25$	0.3828 (+17.6%)	1997	0.3893	0.3175
$city96c24.BM25$	0.3950 (+21.3%)	2015	0.4053	0.3250

6.4 Conclusions

The application of probabilistic retrieval methods in free-text Chinese language material is very successful. In terms of average precision and total number of relevant retrieved documents, the character approach does better than the word approach. As the results indicate in Table 10, the combination of BM25 and $Weight_1$ for phrases makes a significant positive contribution to the quality of retrieval.

7 Interactive track experiment

The task definition and the type of users conducting the searches were the main differences for this round of Okapi interactive TREC. The set task for the TREC–5 interactive track was defined as finding relevant documents which covered as many different aspects of a topic as possible within a twenty minute search session. The eight searchers were postgraduate students in the Information Science Department. They had some experience of information retrieval systems, but acted as end-users. Each of the test topics was carried out by two searchers. Past performance in interactive searching had shown that the query expansion facility in the Okapi system was effective in retrieving more similar items and it was envisaged that the task of finding different aspects would be more taxing. The same GUI interface was used as in TREC–4 with some modifications to allow for greater user control in the search interaction as indicated below:

Incremental query expansion In previous versions of the interactive interface query expansion had to be explicitly requested. Clicking on an EXPAND button caused the current query to be merged with all terms extracted from new relevant documents. In incremental query expansion, introduced in the current interface for TREC–5, once the searcher makes a positive relevance judgment, extracted terms are automatically added to the working query and current query terms are dynamically re-weighted and re-ordered if appropriate (See Appendix A.2.6). The intention was to make the underlying relevance feedback process more visible by enabling the searcher to relate relevance judgments more closely to the working query.

Changing relevance judgments and restoring query terms In manipulating the working query, searchers were given greater flexibility to 'undo' previous decisions. Firstly, they could change previous relevance judgments and switch between "Full Document" or "Passage Only" for term extraction. Secondly, terms removed by the searcher from the current query were displayed in a separate window so that they could easily be reinstated in the query if required at a later stage.

As in the previous interactive round, searchers filled in a post-search questionnaire indicating their evaluation of each of the query tasks. The results of the questionnaire are found in Appendix D. The analysis in the following section is an attempt to classify the level of difficulty for each of the twelve queries according to the searchers' perceptions and to compare and correlate queries deemed difficult or easy with searching behaviour characteristics.

7.1 Topic classification

In five of the post-search questions searchers were asked to rate different aspects of the search task as 'easy', 'moderately easy', 'difficult' or 'very difficult'. These ratings related to the following questions (See Appendix D):

- interpreting the topic initially (Q1),
- generating initial query terms (Q2),
- finding relevant items from the initial hitlist (Q3),
- finding documents on different aspects of the topic (Q5), and
- rating the overall search task (Q9).

Points were allocated for each of the responses ranging from 0 for the 'easy' category to 3 for the 'very difficult' category. Table 11 shows the resulting score and ranking for each of the twelve test topics with the most difficult scoring 21 and the least difficult scoring 4. For the three most difficult and the three easiest topics there was a high degree of agreement between the two searchers who undertook the individual topics. However for the three most difficult topics, searchers were much less consistent in indicating how difficult it was to interpret in the first instance. Difficult and easy topics were evenly distributed amongst the different blocks allocated to the searchers.

Table 11: Difficulty rating and ranking of topics

topic	256	292	260	255	254	284	286	299	264	293	294	258
score	21	21	17	16	14	13	11	11	10	9	7	4

7.2 Searching behaviour for 'difficult' and 'easy' topics

An analysis of the three highest and three lowest ranking topics was undertaken to compare any associated differences in searching behaviour. The characteristics considered included: the generation of initial query terms, the use of query expansion, the manipulation of the query term sets, and the examination of retrieved documents

7.2.1 Generation of initial query terms

Table 12 compares the source of initial query terms for the 'difficult' and 'easy' topics undertaken by the pairs of searchers, as well as the overall figures for all twelve topics. It appears that for 'difficult' topics more query terms were generated initially than for the 'easy' topics but that a smaller proportion of query terms were derived from the topic specification as given. Moreover searchers used a greater proportion of phrasal terms to generate 'easy' queries than 'difficult' ones. As in TREC–4, problems arise in expressing abstract ideas and the length of the topic specification itself seems to have a direct bearing on the number of terms used for the initial query formulation. The mean for TREC–5 was 7.06 compared to 4.72 for the shorter TREC–4 topics.

Table 12: Source of terms for the initial query

Topic category		No. of terms	Topic spec.	User generated	No. of phrases
Difficult	Mean	9.5	6	3.5	2.3
	Median	10	5.5	3.5	2
	Range	4—13	3—9	0—8	0—5
	Total	57	36 (63%)	21 (58%)	14 (25%)
Easy	Mean	5.8	4.5	1.3	2.3
	Median	6	5	1	1.5
	Range	1—10	1—7	0—3	0—5
	Total	35	27 (77%)	8 (23%)	14 (40%)
All topics	Mean	6.9	5.2	1.8	2.6
	Median	6	5	1.5	2
	Range	1—19	1—15	0—8	0—13
	Total	166	124 (75%)	42 (25%)	63 (38%)

7.2.2 Query expansion and manipulation of query term sets

The query expansion facility was used by all the searchers for all the topics except one. The number of iterations (i.e. new search based on an expanded query with system extracted terms) for each search session averaged 3.63. This is comparable to the interactive searches for TREC–4 (3.56) and shows that there is little difference between expert and end-users in their dependency on the system to support query reformulation. There was also little difference between the mean number of iterations undertaken for 'difficult' (3.8) and 'easy' (4.2) topics.

Overall query expansion was deemed by searchers to be helpful in finding relevant documents in two-thirds of the topics (Table 13). A breakdown of the two categories of topics shows little agreement between searchers, although query expansion may appear to be more helpful in the case of 'easy' topics.

Table 13: Perceived effectiveness of query expansion

Topic category	Helpful	Not helpful	No expansion
Top 3 difficult	3	3	0
Bottom 3 easy	4	2	0
All topics	15	8	1

Searchers could manipulate query term sets in three ways: by removing system generated candidate terms, by adding their own new terms or restoring previously deleted terms. In this current round of TREC users removed a substantially lower number of terms from the working query, an average of 25.22 compared to 39.32 in TREC–4. Although the top 20 terms of an expanded query were displayed in both cases, the incremental query expansion may have produced less noise. However Table 14 shows that five time as many terms were removed from term sets for the 'easy' topics as opposed to the 'difficult' ones. This could perhaps indicate that for 'easy' topics searchers had a much better idea of what terms were appropriate for the task. Indeed the inappropriateness of terms is given as the most prominent reason for deleting query terms overall (Table 15). The least prevalent reason was because aspects of the topic had already been covered. Searchers of 'easy' topics also agreed that the automatic display of new query terms helped them identify new aspects of the topic whereas those undertaking 'difficult' topics were more divided.

Table 14: Removal of terms from working query

Topic category	Total	Mean	Median	Range
Difficult	55	9.2	3	1—41
Easy	275	46.0	29.5	7—109
All topics	166	6.9	6	1—109

Table 15: Reasons for term removal

Topic category	Proper names	Numbers	Inappro-priate	Already covered	None removed
Difficult	1	2	5	1	0
Easy	4	3	5	2	0
All topics	11	11	20	3	1

By contrast user generated query terms were rarely deleted and the number of removed terms restored was also negligible. Searchers also concentrated on the working query as presented by the system and introduced few of their own query terms in the course of a search (See Appendix C.2).

7.2.3 Examination of retrieved documents

A hitlist of fifty documents was displayed at each iteration. On average searchers scrolled through a total of 41.2 titles in a search session of approximately twenty minutes. This is proportionately comparable to the average of 54.2 items scrolled in the thirty minute sessions in TREC–4. However the very different nature of the interactive retrieval task set in the two rounds is manifested by the number of full documents seen. In TREC–5 the mean was almost half of that in TREC–4, 14.13 documents as opposed to 31.12. Evidence that searchers were very selective in examining full documents for the TREC–5 interactive task is also provided by the fact that a number (19.5%) of search iterations led to hitlists from which no full documents were examined. With regard to 'difficult' and 'easy' topics there was little difference between the number of documents 'viewed' and 'seen', or how far searchers scrolled.

Searchers rejected 51% of 'seen' documents. Although for feedback purposes searchers were given the choice of assigning relevance to full documents or best passages, in 87% of instances the full document was chosen.

7.3 Results of interactive searches

The precision and aspectual recall for the interactive searches (excluding Topic 284) were 0.487 and 0.330 respectively. The average precision and aspectual recall for 'difficult' and 'easy' topics indicate a lower performance for 'difficult' topics, 31.3% compared to 43.1% for precision and 22.3% compared to 38.1% for recall.

Because of the different nature of the interactive task these results are not comparable with other TREC results. However one observation has emerged which requires further consideration. Disagreement between assessors' aspectual judgments and those of the searchers was found for 136 out of a total of 583 documents. There was also disagreement between the initial binary relevance assessments by the assessors and their subsequent assessment of aspects. Ten of the documents were judged to be relevant but to contain no aspects, whilst 126 documents were judged as not relevant but to contain aspects.

A comparison between the 'difficult' and 'easy' topics taking account of this inconsistency, reveals that there was a total of 47 disagreeing judgments for the former and 11 for the latter. It would seem that an inherent feature of a difficult topic or search is one where there are discrepancies on how the search task is perceived by the searchers as well as disagreements on relevance judgments. There was, however, no apparent association between the number of aspects identified and the difficulty of the topics.

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: [13] (pp.21–30).

[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: [14]. p21–34.

[7] Robertson S E *et al.* Okapi at TREC–3. In: [15]. p109–126.

[8] Robertson S E *et al.* Okapi at TREC–4. In: [16]. p73–96.

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

[10] Wu Z and Tseng G. Chinese text segmentation for text retrieval: Achievement and Problems. *Journal of the American Society For Information Science 44* (9) p532–541 1993

[11] Chen G. On single Chinese character retrieval system. *Journal of Information 11* (1) p11–18 1992 (in Chinese).

[12] Robertson S E, Walker S and Hancock-Beaulieu M. Large test collection experiments on an operational, interactive system: OKAPI at TREC. *Information Processing & Management 31* (3) p345–360 1995.

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

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

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

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

A Interactive System Description

A.1 GUI interface

The interface is an adaptation of the Trec 4 GUI to the BSS written in C, C++ and TCL/TK. Figures 2 and 3 show screen dumps of the running system. It is composed of six main windows some of which contain context-enabled 'function' buttons.

A.1.1 Query Entry

A.1.2 Working Query

A scrollable, ranked list of the terms that constitute the current working query.

A.1.3 Removed Terms

A scrollable list of any terms removed by the user from the working query, displayed in the order of removal.

A.1.4 Hitlist

A scrollable, ranked hitlist for the current iteration. Once documents have been seen from this list, the entry at the top of the window is the next ranked document after the last one seen.

A.1.5 Pool of Positive Relevance Judgments

A scrollable, ranked list of positive user relevance judgments.

A.1.6 Full Document

A scrollable window in which full documents selected from the hitlist are displayed. Items A.1.1 to A.1.5 are all displayed in one window together with two context-sensitive buttons to:

- **SEARCH**: Search on the current working query to form the hitlist for the next iteration. This button is enabled every time the working query is modified, either by the entry of new user terms, or the (automatic) addition/removal of system extracted terms.
- **EXIT**: Always enabled.

Window A.1.6 is a pop-up text window for the display of full records. At the bottom of the window are three buttons for making relevance judgments. These are described in more detail in Section A.2.4.

A.2 User interaction

A.2.1 User input of query terms

Users enter one or more terms, optionally followed by an adjacency operator (a '+' sign) as the last non-space character in the line, into the query entry box.

- No adjacency operator. Each term is treated as a single member of the query.
- An adjacency operator. The set of terms are treated as:

  ADJ : a phrase, possibly including intervening stopwords. The terms are taken in input order.

  SAMES : within the same sentence. The terms may occur in any order within the sentence.

  For example, the input line "laws of the sea" would find documents that included, amongst others, both "laws of the sea" and "The right of innocent passage in Black Sea waters is not provided for by Soviet law". Any set of terms input with an adjacency operator is referred to as a 'PHRASE'. Internally two sets are generated, S(A) and S(S), with number of postings and weights N(A), W(A) and N(S), W(S) respectively. These are then combined into one set according to the rules:

Sets Generated	Use
1. N(A) = N(S) = 0	both discarded
2. N(A) = 0, N(S) > 0	S(S), N(S), W(S)
3. N(A) = N(S), N(A) > 0	S(A), N(A), W(A)
4. N(A) < N(S), N(A) > 0	S(A), N(A), W(A), S(S)-S(A), N(S)-N(A), W(S) (Count as one term only.)

  The weight calculated for each term is a Robertson/Sparck Jones F4 predictive weight, with halves (equation 2). In addition, user entered terms — both single terms and phrases — are given a loading; r and R increased by 4 and 5 respectively. Terms are displayed in the working query window in descending order of Robertson Selection Value [9]. Information displayed with each term is (a) the number of postings, and (b) the number of documents judged as relevant that contain the term. Phrases are followed by a letter indicating:

  A Adjacency only

  S Same sentence occurrence only

  B Both adjacency and same sentence

  G An indexed phrase.

A.2.2 Searching

The working query is composed of N terms ($N \leq 20$) from the entire termset made up of user-entered and extracted terms. Before each reformulation of the working query the termset is first ordered by two sort keys:

- USER_TYPE (**U**ser or **E**xtracted) descending
- RSV descending

The rules for the inclusion of terms from the termset in the working query are as follows. Suppose the entire termset is composed of T terms made up of U user-entered terms and E extracted terms. i.e. $T = U + E$. There are two cases to consider:

U < 20: All of the user-entered terms are allowed into the working query. For extracted terms (if any) to take up the remaining $20 - U$ places they must satisfy the conditions that:

1. They occur in at least two relevant documents.
2. The RSV of the term is at least 60 percent of the average RSV of the current working query.

U ≥ 20: The working query will be composed of the top 20 user-entered terms.

A search on the working query entails combining the terms in a best match operation (bm25). Passage retrieval (Section 3.1) is applied to the document set generated with parameters $p\_unit = 4$, $p\_step = 2$, $k_1 = 1.6$ and $b = 0.7$. The weight of the document is taken to be the higher of the weights of the full record or the best passage.

A.2.3 Hitlist generation

For a search performed on any iteration, documents are included in the hitlist only if they satisfy the two conditions:

1. they do not exist in a set generated by a previous iteration.
2. they satisfy a length threshold condition, i.e. the full record or, for longer documents, the best passage, must be less than 10K characters in length (in contrast to Trec 4 where all documents retrieved were included).

The hitlist is made up of the top H ranked documents ($H \leq 50$) that satisfies both of these two conditions. An entry for each document consists of:

A header line. ⟨record_no⟩ ⟨docid⟩ ⟨normalised_weight⟩ ⟨document_length⟩ [⟨passage_length⟩]
The ⟨normalised_weight⟩ is the system weight mapped onto the range 1..1000. The ⟨document_length⟩ and ⟨passage_length⟩ are given in pages, where one page is approximately 2000 characters.

A system generated title. Since generally records have no distinct title field, a "title" for display in the hitlist window is made up from approximately the first 150 characters from the headline field (if any) and the text field.

Query term occurrence information. This may (apparently) contain some duplicate information since the information is obtained by counting the highlighted terms in the document, which may contain different sources for the same stemmed terms.

A.2.4 "Seeing" documents

A document, selected for "seeing" by double-clicking anywhere in its hitlist entry, is displayed in a pop-up, scrollable text window. The best passage (or the whole document if the same) is highlighted in light-cyan; query terms are highlighted in green. The document is displayed either at the start line of the best passage, or, if (i) the best passage is the whole document or (ii) there is no best passage, the line containing the first query term. At the bottom of the text window are three buttons to allow users to make a relevance judgment.

1. **Full Document**: Relevant, terms are extracted from the text field of the entire document.
2. **Passage Only**: Relevant, terms are extracted only from the best passage.
3. **Not Relevant**: Not relevant.

Searchers must make a relevance judgment before they may go onto to any other part of the search process. A relevance judgment can be altered (e.g. F -> P, P -> F, [F | P] -> N, or N -> [F | P]) at any time until a new search is made, in which case the set of extracted terms is altered accordingly.

A.2.5 Relevance judgments pool

The normalised ranked hitlist information for all documents currently judged as relevant. Any member of the current relevance judgments pool that exists in a document set generated by a new search, has its weight set to that which it has in the latest document set; the display order is adjusted accordingly.

A.2.6 Incremental Query Expansion

Once the searcher has made two or more positive relevance judgments — "Full Document" or "Passage Only" — extracted terms are automatically added to the working query if they satisfy the conditions specified in A.2.2. After each relevance judgment all terms extracted from the appropriate section of the document are merged with the existing termset and the relevance information, including weights and RSVs, is adjusted accordingly.

A.2.7 Removing terms

Terms may be removed from the working query by double clicking on its entry in the query window. Removed terms are displayed in the removed terms window (A.1.3). It is possible that in the entire set of user-defined and extracted terms there are more than 20 candidate terms for the working query. Thus, as terms are removed, other terms may be promoted to take their place. A removed term may be reinstated in the query by double-clicking on its entry in this window.

A.2.8 Quitting

Quitting the search is achieved by clicking once on the "Exit" button.

B Experimental Conditions

B.1 Searcher characteristics

Eight searchers, all post-graduate students in the Department of Information Science, were divided into two groups — A and B. Group A consisted of one male and three females; Group B consisted of two males and two females. Their ages ranged from middle 20s to late 30s. The searchers were end-users who had (a) no specialist knowledge of any of the subject domains covered by the topics, and (b) were not familiar with the Okapi system of searching.

B.2 Task description/training

Within each group each searcher was randomly assigned one of the four blocks of three topics: i.e. there were two searchers allocated to each block. Group A carried out their training and searches during the week beginning 29th August, Group B during the week beginning 9th September. Each searcher was given training which consisted of:

1. a set of guidelines which included information such as:
   (a) the task definition as defined for the interactive track,
   (b) procedures to follow in the event of a system crash,
   (c) suggestions on strategies which could be adopted for carrying out searches, e.g. to remove terms deemed as irrelevant from extracted lists of terms before carrying out an expanded search.
2. the opportunity to familiarize themselves with the interface and the task by carrying out three trial runs on topics taken from blocks that they would not be searching themselves.

Three searchers carried out their training and performed their runs on the same day. The other five searches performed their runs on the day after their training. After each "real" search, each searcher was asked to fill in an evaluation questionnaire for the search before proceeding on to the next one.

C Search Process

Unless otherwise stated the column 'Type' entries **N**, **+** and **A** refer to:

N: terms defined with no adjacency operator

+: terms defined with an adjacency operator

A: all terms defined.

C.1 Clock Time

Mean	Median	Variance	Range
20.6	20.8	3.1	13.9–22.9

Times are given to the nearest tenth of a minute.

C.2 Number of User Defined Terms

	At all iterations				After the first iteration			
Type	**Mean**	**Median**	**Variance**	**Range**	**Mean**	**Median**	**Variance**	**Range**
N	4.96	3.5	21.78	0–16	2.38	1	10.77	0–14
+	3.83	3	9.80	0–13	1.21	1	3.74	0–9
A	8.79	8	24.26	2–18	3.59	2.5	15.04	0–15

"Phrases" Defined By Searchers

Phrases generated: 94 **Phrases used:** 74

C.3 Number of Terms Used In Queries

	Initial query				Final query			
Type	**Mean**	**Median**	**Variance**	**Range**	**Mean**	**Median**	**Variance**	**Range**
N	4.56	2.5	6.96	0–13	15.13	17	21.33	4–20
+	2.50	1.5	5.50	0–13	3.08	2	7.56	0–13
A	7.06	5.5	20.30	1–19	18.21	20	11.48	8–20

C.4 Number of Iterations

An iteration, i.e. a new query formulation, was taken to be marked by each 'search' command. Expansion was performed incrementally. No data was kept to indicate how many times and when each working query was altered by the inclusion/exclusion of extracted terms.

Mean	Median	Variance	Range
3.63	3	1.98	1–7

C.5 Documents "Viewed" (hitlist) and "Seen" (full text)

"Viewing" a document consisted of seeing a header line, a system generated title, and query term occurrence information as described in Appendix A.2.3. The figures represent the percentage distance scrolled through the hitlist by the searcher. "Seeing" a document consisted of showing the full record in a scrollable window.

Viewed				Seen			
Mean	**Median**	**Variance**	**Range**	**Mean**	**Median**	**Variance**	**Range**
60.58	67	835.99	8–98	14.13	13.5	15.42	9–21

C.6 Relevance Judgments

In the following table the relevance judgments column ('Rel') are given as: **F** (Full document), **P** (Passage) and **N** (Not relevant).

Rel	Mean	Median	Variance	Range
F	5.96	6	9.69	0–14
P	0.92	0.5	2.17	0– 6
N	7.25	7	10.46	2–16
All	14.13	13.5	15.42	9–21

C.7 Use of System Features

Command	Mean	Median	Variance	Range
define - N	4.96	3.5	21.58	0–16
+	3.83	3	9.80	0–13
A	8.79	8	24.96	2–18
search	3.63	3	1.98	1–7
show	14.13	13.5	15.42	9–21
remove	25.22	13.5	1119.18	0–116
restore	0.85	0	3.40	0–8

C.8 Number of User Errors

Data on user errors were not collected. However, there was one search during which the system "crashed". This occurred almost at the end of the 20 minute period and so this was taken as the end of the search.

C.9 Topic 274

C.9.1 Search Narrative

The initial query terms entered by the searcher for this topic, concerned with latest developments in the production of electric automobiles, were 'electric', 'automobiles', 'batteries', 'production', 'feasibility' and 'recharging', all of which were chosen from the topic specification itself. The searcher also included a term and phrase of her own, 'cars' and 'alternative power'. Only one document was viewed and judged relevant from the first hitlist. The term 'problems' was added for the second iteration, which produced a fruitful hitlist. Eight out of the top ten ranking documents were examined and seven were deemed relevant. The rejected document on battery technology had been judged as relevant initially and was reconsidered in the light of other documents concerned with batteries.

As each positive relevance judgment could affect the weighted list of extracted candidate query terms, in the second iteration the searcher deleted a total of 107 terms from the term set on seven occasions in the course of finding 10 relevant documents. These included proper nouns, dates as well as other terms which were deemed not be appropriate for the topic. The rationale given by the searcher was "to avoid narrowing down the query or taking it into a different direction". For example, 'prototype', 'Los Angeles' and 'Chrysler' were considered to be too narrow whereas 'driving', 'characteristics' and 'highway' were judged as not directly relevant.

Following the retrieval of seven relevant items from the top ranks, it became more difficult to find others which dealt with different aspects of the topic and the searcher had to scroll as far as the 38th item in the hitlist to find three more relevant documents. More terms were also removed from the term set at this later stage. The searcher indicated that the replacement of removed terms by others lower down the ranking was annoying and she was prompted to delete them as a matter of course. However when the term 'environmental' came up the searcher commented, "I was surprised that such a relevant term didn't come up earlier". Apparently she had not considered introducing it herself. The ranking of system generated terms above user generated ones caused some disquiet as well as the fact that some relevant terms were removed by the system following a subsequent positive relevance judgment.

In the third iteration seven more full documents were seen and three more were judged relevant. Only two more terms were removed from the term set. Some of the documents rejected were relevant but did not add new information for the topic. In all cases in making positive relevance judgments, the searcher chose to mark the full document as relevant for relevance feedback purposes as opposed to the highlighted best passage only. In viewing the printed output of the search the searcher admitted that due to time constraints, she had judged relevance in the latter stages of the test on the strength of terms present and their distribution throughout the document.

The search started off very successfully and was generally considered to be easy. The display of extracted query terms helped to identify new aspects of the topic and query expansion led to finding more documents and improve the coverage of the topic.

C.9.2 Breakdown of command usage

Define (N)	2	Define (+)	1	Search	3	Show	17	Remove	109	Restore	0

D Search Evaluation: questionnaire results

1. How difficult was it to interpret initially as given?

Easy	Moderately easy	Difficult	Very difficult	TOTAL
9 (37%)	10 (42%)	2 (8%)	3 (13%)	24 (100%)

2. How difficult was it to generate initial search terms?

Easy	Moderately easy	Difficult	Very difficult	TOTAL
9 (37%)	9 (37%)	5 (21%)	1 (4%)	24 (100%)

3. How difficult was it to find relevant items from the initial hitlist?

Easy	Moderately easy	Difficult	Very difficult	TOTAL
3 (12.5%)	12 (50%)	6 (25%)	3 (12.5%)	24 (100%)

4. How did you identify the different aspects of the topic?

From the initial specification	21	88%
From the initial hitlist	2	8%
By examining documents	10	42%

5. How difficult was it to find documents on the different aspects of the topic?

Easy	Moderately easy	Difficult	Very difficult	TOTAL
3 (12.5%)	9 (37.5%)	7 (29%)	5 (21%)	24 (100%)

6. If you removed terms from the extracted term lists on what basis did you do so?

Proper names	Numbers	Difficult Inappropriate	Aspects already covered
11 (46%)	11 (46%)	20 (83%)	3 (12.5%)

7. Did the automatic display of new query terms help identify new aspects of the topic?

Yes	No	TOTAL
13 (54%)	11 (46%)	24 (100%)

8. Did the use of expanded searching lead to finding documents covering new aspects of the topic?

No expansion	Expansion helpful	Expansion unhelpful	TOTAL
1 (4%)	15 (63%)	8 (33%)	24 (100%)

9. How would you rate the overall difficulty of the topic question?

Easy	Moderately easy	Difficult	Very difficult	TOTAL
1 (4%)	12 (50%)	7 (29%)	4 (17%)	24 (100%)

10. How would you rate the success of your search?

Successful	Moderately successful	Not successful	TOTAL
5 (21%)	13 (54%)	6 (25%)	24 (100%)

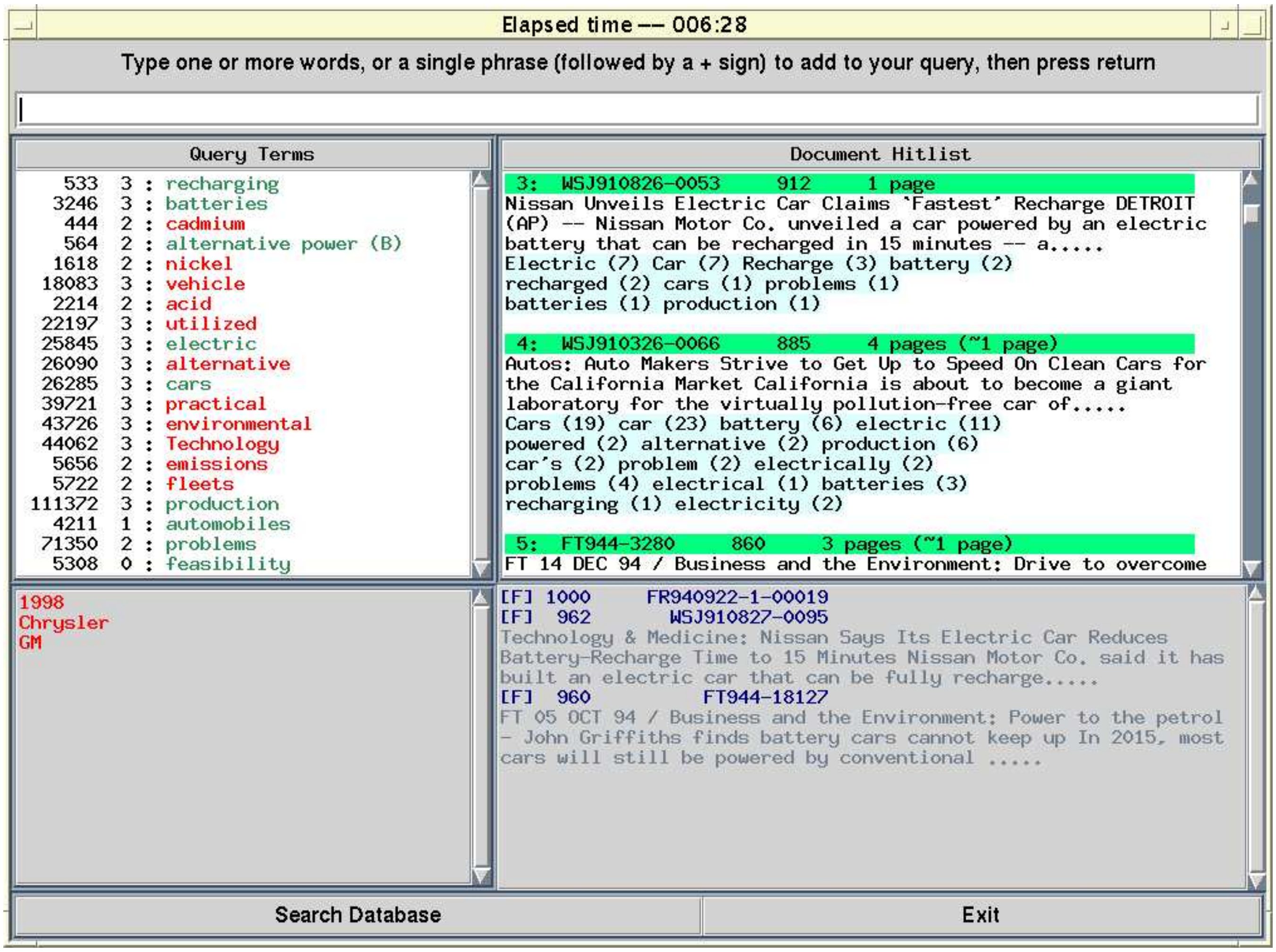
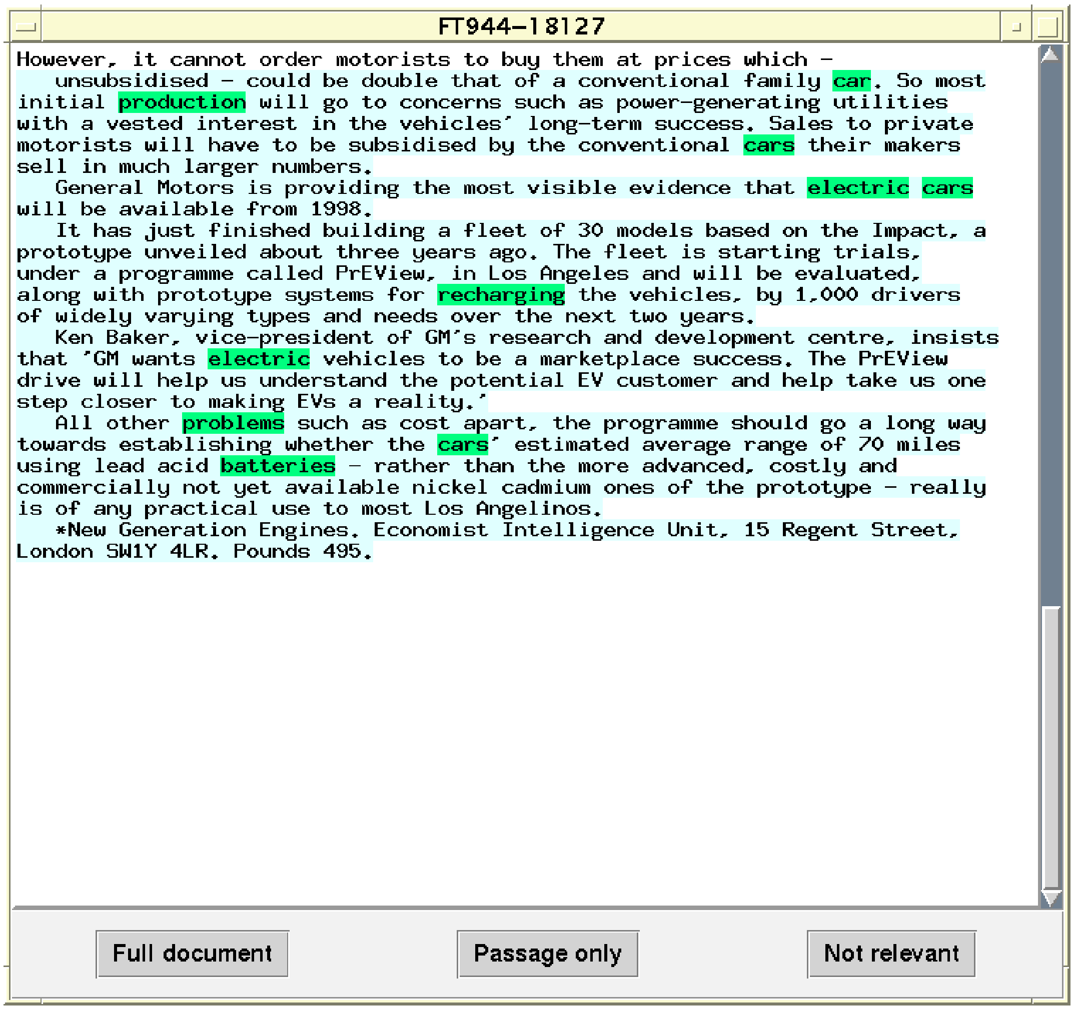


Figure 2: Interactive interface: main search screen



FT944-18127

However, it cannot order motorists to buy them at prices which – unsubsidised – could be double that of a conventional family car. So most initial production will go to concerns such as power-generating utilities with a vested interest in the vehicles' long-term success. Sales to private motorists will have to be subsidised by the conventional cars their makers sell in much larger numbers.

General Motors is providing the most visible evidence that electric cars will be available from 1998.

It has just finished building a fleet of 30 models based on the Impact, a prototype unveiled about three years ago. The fleet is starting trials, under a programme called PrEView, in Los Angeles and will be evaluated, along with prototype systems for recharging the vehicles, by 1,000 drivers of widely varying types and needs over the next two years.

Ken Baker, vice-president of GM's research and development centre, insists that 'GM wants electric vehicles to be a marketplace success. The PrEView drive will help us understand the potential EV customer and help take us one step closer to making EVs a reality.'

All other problems such as cost apart, the programme should go a long way towards establishing whether the cars' estimated average range of 70 miles using lead acid batteries – rather than the more advanced, costly and commercially not yet available nickel cadmium ones of the prototype – really is of any practical use to most Los Angelinos.

*New Generation Engines. Economist Intelligence Unit, 15 Regent Street, London SW1Y 4LR. Pounds 495.

Figure 3: Interactive interface: full record display