Information Retrieval, 3, 313–338, 2000

Improving English and Chinese Ad-Hoc Retrieval: A Tipster Text Phase 3 Project Report

K.L. KWOK kwok@ir.cs.qc.edu
Computer Science Department, Queens College, CUNY, 65-30 Kissena Boulevard, Flushing, NY 11367

Received March 22, 1999; Revised December 20, 1999; Accepted February 1, 2000

Abstract. Both English and Chinese ad-hoc information retrieval were investigated in this Tipster 3 project. Part of our objectives is to study the use of various term level and phrasal level evidence to improve retrieval accuracy. For short queries, we studied five term level techniques that together can lead to good improvements over standard ad-hoc 2-stage retrieval for TREC5-8 experiments. For long queries, we studied the use of linguistic phrases to re-rank retrieval lists. Its effect is small but consistently positive.

For Chinese IR, we investigated three simple representations for documents and queries: short-words, bigrams and characters. Both approximate short-word segmentation or bigrams, augmented with characters, give highly effective results. Accurate word segmentation appears not crucial for overall result of a query set. Character indexing by itself is not competitive. Additional improvements may be obtained using collection enrichment and combination of retrieval lists.

Our PIRCS document-focused retrieval is also shown to have similarity with a simple language model approach to IR.

Keywords: language model and PIRCS retrieval model, ad-hoc two-stage retrieval, pseudo-relevance feedback, collection enrichment, segmentation and Chinese IR

1. Introduction

As increasing amounts of computer-readable texts are becoming available on the web or on optical disks, document searching and retrieval has become an indispensable tool for information users and analysts of all walks of life. Up till the late 1980's, research in text retrieval has been using mainly collections of the order of thousands of items. Since 1990, with the foresight of the TIPSTER (e.g., http://www.itl.nist.gov/iaui/894.02/related_projects/tipster/) and TREC (e.g. Voorhees and Harman 1998) programs, substantial progress has been made to advance the state-of-the-art in ad-hoc information retrieval (IR) and text detection in general. Examples include: availability, experimentation and standard evaluation of large gigabyte-size collections, term weighting improvements, 2-stage 'pseudo-feedback' retrieval strategy, recognition of difficulties of short queries versus long, use of phrases, studies of multi- and cross-language retrieval, among others. This project builds upon previous findings to bring further advances in this field using our PIRCS retrieval system as the platform.

We have participated in all past TREC experiments, and since 1996, also participated in the TIPSTER Text Phase 3 program. This report serves to summarize work that has been done during the program, and some of the important findings for both English and Chinese IR. Section 2 and 3 gives an overview of our PIRCS system and the 2-stage retrieval strategy.

Section 4 presents our work for English ad-hoc retrieval employing term and phrasal level evidence. Section 5 describes various Chinese retrieval methodologies and their evaluation. Section 6 has the conclusions.

2. Pircs retrieval system

2.1. Basic PIRCS' retrieval model

The software used for our investigations is PIRCS (acronym for Probabilistic Indexing and Retrieval-Components-System), a document retrieval system that has been developed in-house since 1990. It is based on the probabilistic retrieval approach (Robertson and Sparck Jones 1976), which ranks document d with respect to a query q via a retrieval status value (RSV) which is the log odds that given document d, it will be found relevant to q, viz.: log [Pr(relevant to $q \mid d$)/Pr(not relevant to $q \mid d$)]. In probabilistic retrieval, a document is represented as a pattern of binary term usage, i.e. $d = (t_1, t_2, \ldots, t_m)$ with $t_k = 1$ or 0 denoting presence or absence of term k. After applying Bayes' Theorem and ignoring unknown prior odds of relevance, this RSV is usually cast as proportional to the log likelihood ratio that given samples of d relevant to q one finds pattern of term usage like those in d, viz.: log [$Pr(d \mid$ relevant to $q)/Pr(d \mid$ not relevant to $q)$]. After making the term independence assumption and some manipulation, this can be written as the familiar formula:

$$\text{RSV-}Q(q,\ d) = \sum_k x_k w_k, \quad \text{with}$$
$$w_k = \log\,[r_k/(1 - r_k)(1 - s_k)/s_k] \tag{1a}$$

Here, RSV-Q $(\cdot)$ means a query-focused RSV, x_k is the binary weight of term k in d, and w_k is the query term weight based on knowing $r_k = Pr(t_k = 1 \mid$ relevant to $q)$, and $s_k = Pr(t_k = 1 \mid$ not relevant to $q)$. The probabilities r_k are not calculable unless some sample documents relevant to q are given, and usually this is not available at the initial stage of an ad-hoc retrieval. Various heuristics like (Croft and Harper 1979, Robertson and Walker 1997) are used to estimate them instead. On the other hand, if some n documents in a collection of size Nd are known relevant to q (as in relevance feedback), and term k appears in $n_k <= n$ of them and has document frequency D_k in the collection, one can estimate the probabilities as: $r_k = n_k/n$, and $s_k = (D_k - n_k)/(\text{Nd} - n)$. The factor $\log(1 - s_k)/s_k$ is approximately the inverse document frequency (IDF) weight $\log \text{Nd}/D_k$, since usually $n_k \ll D_k$, and $n \ll \text{Nd}$. With these values of r_k and s_k, the weight then takes the form:

$$w_k = \log\,[n_k/(n - n_k) \times (\text{Nd} - n - D_k + n_k)/(D_k - n_k)] \tag{1b}$$

PIRCS approaches this initial unknown parameter problem by treating each item (document or query) as non-monolithic but constituted of conceptual components, and lets each item learn its own term weights. As approximation each content indexing term, including repeats, is considered an independent conceptual component, and represented as a term pattern of all 0s except for a single 1 for the term itself. When all documents are broken

up into components, we end up with a component universe that is a multiset of size Nw, the number of tokens in a collection. To weight the terms of query q without relevance information, we argue that q should be relevant to its own constituting components, i.e. item self-relevance. We augment the component universe with those of the query, and imagine q retrieving against the augmented universe. Self-relevance means each query has a naturally relevant set which is its own set of conceptual components, and a non-relevant set which is the component universe (augmented universe minus the query components). This way we can bootstrap the probabilities without any external relevance information by estimating $r_k = q\text{tf}_k/L_q$, and $s_k = F_k/\text{Nw}$. L_q is the number of conceptual components in query q (what we call the length of q), $q\text{tf}_k$ is the query term frequency of k, and $F_k = \sum_{\text{all docs}} \text{tf}_k$ is the collection frequency of term k in the whole collection. The factor $\log(1 - s_k)/s_k$ in w_k is approximately $\log(\text{Nw}/F_k)$ which has been called ICTF (inverse collection term frequency) weight (Kwok 1990). It differs from IDF in having all term frequencies counted, not just binary. After the query has been weighted, a document d that is to be evaluated with respect to q is also considered as constituted of conceptual components. When one of its component k matches a query term k, it receives a weight w_k. The total weight received by d including components occurring tf_k times (i.e. term frequency in d) is averaged by the document length L_d, and the RSV corresponding to (1a) becomes:

$$\text{RSV-}Q(q, d) = \sum_k (\text{tf}_k/L_d) w_k, \quad \text{with} \tag{2a}$$

$$w_k = \log\left[q\text{tf}_k/(L_q - q\text{tf}_k) \times (\text{Nw} - F_k)/F_k\right] \tag{2b}$$

PIRCS' approach differs from probabilistic retrieval in accounting for document length and term frequency information in (2a) vs (1a). It also provides query term weights (2b) vs (1b) in the absence of relevance information, and counts using components rather than documents.

Additionally, in PIRCS we regard queries as (short) documents or vice versa. They can switch roles in the above development and we would be ranking queries with respect to a given document. This scenario occurs in a routing environment where a set of user profiles is employed to screen documents one by one. Analogously, we let each document d learn its own term weights by assuming it is a query and retrieving from the component universe. Each document also has a natural self-relevant set (which consists of its own components), and an irrelevant component set which is the universe minus the components of d. After going through similar development, we obtain an RSV-D corresponding to (2a) that is document-focused as follows:

$$\text{RSV-}D(d, q) = \sum_k (q\text{tf}_k/L_q) w_k, \quad \text{with} \tag{3a}$$

$$w_k = \log\left[\text{tf}_k/(L_d - \text{tf}_k) \times (\text{Nw} - L_d - F_k + \text{tf}_k)/(F_k - \text{tf}_k)\right] \tag{3b}$$

Since Nw (size of the component universe) $\gg$ any other variables, we can approximate (3b) by:

$$\begin{aligned} w_k &\sim \log[(\text{tf}_k/L_d) \times (\text{Nw}/F_k) \times (1 - \text{tf}_k/F_k)^{-1} \times (1 - \text{tf}_k/L_d)^{-1}] \\ &\sim \log[1 + \text{tf}_k \times \text{Nw}/(L_d \times F_k) \times ((1 - \text{tf}_k/F_k)^{-1} \times (1 - \text{tf}_k/L_d)^{-1} \\ &\quad - L_d \times F_k/(\text{tf}_k \times \text{Nw}))] \end{aligned} \tag{3c}$$

Equation (3c) will be discussed in the next section. In general, (2a) and (3a) provide different retrieval lists, and usually it is beneficial to combine the two with mixing parameter α as is done in PIRCS to give a combined RSV for ranked retrieval as follows:

$$\text{RSV}\ (q, d) = \alpha \times \text{RSV-}D(q, d) + (1 - \alpha) \times \text{RSV-}Q(q, d) \tag{4}$$

2.2. Similarity to a simple language model approach

Recently a number of groups have proposed linguistically-motivated approaches to IR including (Ponte and Croft 1998, Miller et al. 1999, Hiemstra and Kraaij 1999). Their basis for ranking documents is to assume a language model for each document from which one attempts to generate the query statement under attention. The RSV is taken as the log of the probability of realizing the query given the document and its language model. As a first approximation, the independent unigram model is employed. A particularly simple approach is that of (Hiemstra and Kraaij 1999) who evaluates this probability for a query q with terms $t_1, t_2, \ldots, t_\text{m}$ as:

$$\Pr(t_1, t_2, \ldots, t_\text{m} \mid d) = p(t_1 \mid d) \times p(t_2 \mid d) .. \times p(t_\text{m} \mid d)$$

with each $p(t_k \mid d)$ approximated as:

$$p(t_k \mid d) = \alpha_2 \times \text{tf}_k / L_d + \alpha_1 \times D_k / \text{Nd}.$$

The first factor is the unigram generation model given d, and the second factor smoothes the first probability using the global collection statistics because documents are usually too short to provide a reliable language model for estimation, and also some query terms may not exist in d. By taking the logarithm and factoring, the RSV for ranking becomes:

$$\text{RSV}(q,\ d) = \sum_k q\text{tf}_k \times w_k, \quad \text{with} \tag{5a}$$

$$w_k = \log[1 + (\alpha_2/\alpha_1) \times (\text{tf}_k / L_d) \times (\text{Nd}/D_k)]. \tag{5b}$$

Here α_1, α_2 distributes importance to the two factors, and a constant sum, $\sum_k \log(\alpha_1 \times D_k/\text{Nd})$, dependent only on the query has been ignored. Because this approach arises from the observed linguistic behavior of a document and the collection, it does not require any relevance information to estimate the weights just like PIRCS' approach.

This RSV bears close similarity to that of PIRCS' Eq. (3). Both involve the query term frequency $q\text{tf}k$; the query length factor L_q in (3a) is a constant for the query. In w_k of (5b), the inverse document frequency factor (Nd/D_k) was used. In fact, from the point of view of a unigram model for the collection, the formula (Nw/F_k) seems more appropriate and has been employed in (Ponte and Croft 1998). This is because tf_k/L_d is used with documents for generation and a document is a sample from the collection. If (Nw/F_k) were used instead, then (5b) is similar in form to (3c), with

$$\alpha_2/\alpha_1 \quad \text{replaced by the factor}$$
$$(1 - \text{tf}_k/F_k)^{-1} \times (1 - \text{tf}_k/L_d)^{-1} - L_d \times F_k/(\text{tf}_k \times \text{Nw})$$

which varies with the variables. Typical values of the variables in a 2GB collection of TREC7 data are: Nw $\sim$ 150M, $L_d \sim$ 50 to 500 (sub-documents), $F_k \sim$ 100 to 100K, $\text{tf}_k \sim$ 1 to 6. If both F_k and L_d are $\gg \text{tf}_k$, which is often the case for most terms, the factor may be approximated as $1 + \text{tf}_k/F_k + \text{tf}_k/L_d - F_k \times L_d/(\text{tf}_k \times \text{Nw})$ in place of α_2/α_1 after ignoring second order terms. Thus, the PIRCS combination RSV (4) may be seen as combining both probabilistic retrieval and a type of simple language model approach.

2.3. Network realization of PIRCS approach

Shown in figure 1(a) is a network view of PIRCS approach and has three layers of query Q, term T and document D nodes connected with bi-directional weighted edges (Kwok 1995). Unit activation initiated from a document spreads towards a query via common terms and multiplied by intervening edge weights such as tf_k/L_d and w_k respectively (Eq. (2a)). Activation summed from the common terms and deposited at a query realizes RSV-Q. The opposite Q-T-D activation flow and summed at a document realizes RSV-D. The network also supports query adaptation based on knowing some n sample relevant documents as shown in figure 1(b). Adaptation can lead to edge weight changes that would rank documents similar to the given sample more effectively, and can also lead to expanding the query with additional edges linked to terms based on selecting terms from the sample relevant documents.

The number of terms in the expanded query is a user-supplied parameter. The terms are selected first based on frequency of occurrence within the training set, and ties are broken using the average probability of a term in the sample set, i.e. $p_k = (\Sigma\ \text{tf}/L)/n$. Some of these terms k may already exist in the query; we train the existing T-Q edges via $r_k = \eta \times p_k$ and the Q-T edges via $\kappa \times p_k + (1 - \kappa) \times q\text{tf}_k/L_q$. When term k and hence the associated edges are new, the Q-T edges are set to p_k and T-Q edges have $r_k = \eta' \times p_k$. η, η', κ are learning constants.

Our system also uses two-word adjacency phrases as terms to improve on the basic single-stem representation. Documents of many thousands or more words long can have adverse effect on retrieval. PIRCS deals with the problem by simply segmenting long documents into approximately equal sub-documents of 550-word size and ending on a paragraph boundary. For the final retrieval list, retrieval status values (RSV) of the top three sub-documents of the same document are combined with decreasing weights to return a final RSV. This in effect favors retrieval of longer documents that contain positive evidence in different sub-parts of it. PIRCS has participated in all previous TREC 1-8 blind retrieval experiments and consistently returned highly competitive results (see e.g. Voorhees and Harman 1998).

3. Two-stage ad-hoc strategy

Automatic ad-hoc retrieval with natural language queries is convenient for the user. It is difficult for a retrieval system because the query wordings are unknown beforehand, and its topical content is unpredictable. Moreover, there will not be any example relevant documents that a system can rely on for training purposes as in a routing situation.

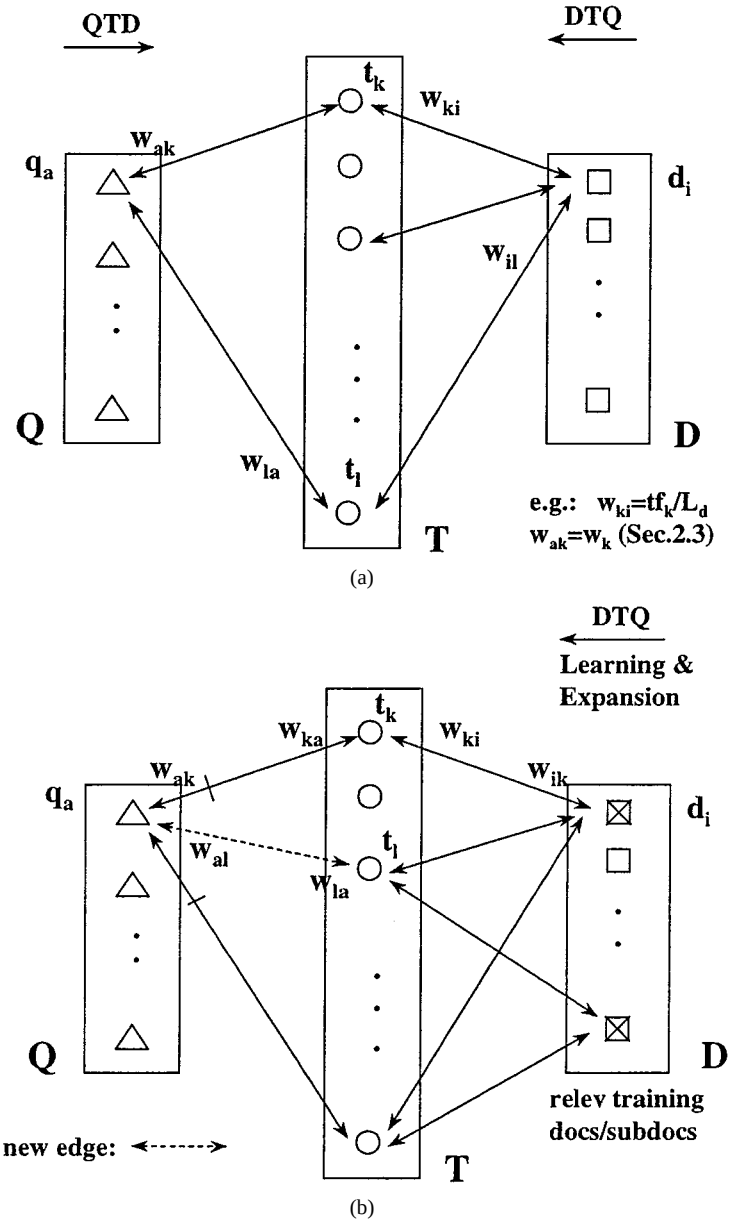

Figure 1. a) 3-layer PIR network; b) DTQ learning with term expansion.

To improve the accuracy of ad-hoc retrieval, it is now a common practice to adopt a 2-stage retrieval strategy. Under the right circumstances substantial improvements over single stage are obtained. In a 1-stage retrieval, the raw query which is a user-provided description of information needs is directly employed by the retrieval algorithm to assign a retrieval status value (RSV) to each document in a collection, and the ranked list of documents is interpreted as the final retrieval result. In a 2-stage strategy, this initial ranked list is interpreted as but an intermediate step. The set of n top-ranked documents of the initial retrieval is assumed relevant, even though the user has not made any judgment. These 'pseudo-relevant' documents are used to modify the weight of the initial query according to some learning procedure, as well as to expand the query with terms from these documents based on some selection criteria like frequency of occurrence. The modified query is then used to do a second retrieval, and the resultant ranked list becomes the final result. This helps because if the raw query is reasonable and the retrieval engine is any good, the initial top n documents can be considered as defining the topical domain of the user need and should have a reasonable density of relevant or highly related documents, and the procedure simulates real relevance feedback. The history of 2-stage retrieval dates back to (Attar and Frankel 1977, Croft and Harper 1979). It was limited to small collections and not too successful. Since TREC2 & 3, the technique has been applied by various groups effectively with large collections.

The process of a 2-stage retrieval is depicted in figure 2. The shaded boxes represent ranked document output from the 1st and 2nd retrieval. Selected top-ranked documents from 1st stage are used as feedback (thick arrow) to augment the raw query to form expanded query, which produces the final retrieval list. The numbers indicate the point where methods discussed in Sections 4.1 and 4.2 are applied to improve this 2-stage retrieval scheme.

Traditionally, real relevance feedback can give very large improvements in average precision, like 50 to over 100% (Salton and Buckley 1990). Experiments with our PIRCS system

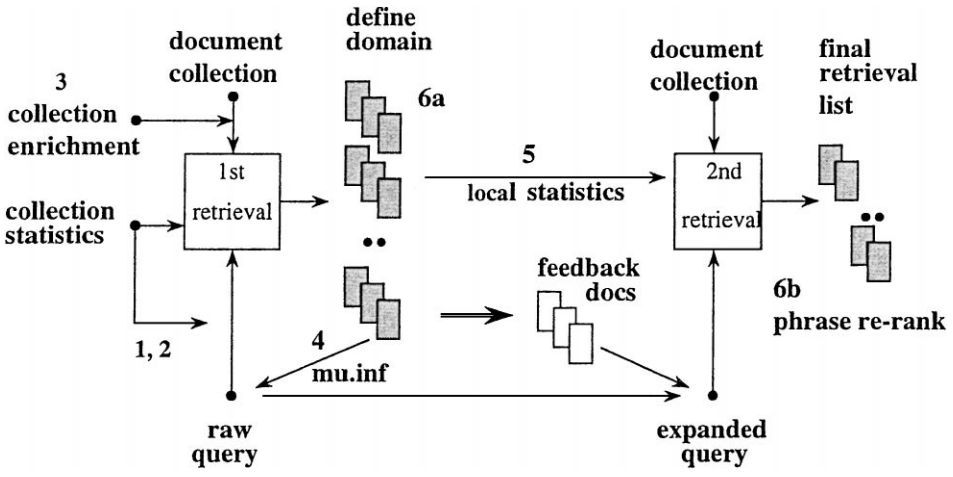

Figure 2. Tow-stage retrieval and methods of improvements.

have shown that this 2-stage of ad-hoc method works more often than not, and the average precision for a set of queries can improve a few to over 20%. From a network point of view, 2-stage retrieval may be seen as activation spreading in the fashion Q-T-D—T-D. After the normal Q-T-D processing, the n top documents are identified. These documents then activate terms that are most common within n, and together with activation values from the query, achieve both query re-weighting and query expansion as discussed in Section 2.3. The activated terms then spread activation back to the documents (i.e. $-T$-D) to achieve the 2nd stage RSV-D. Additionally, activation can spread—T-Q through the newly activated terms to achieve 2nd stage RSV-Q.

In all of our work, this 2-stage approach is used in our retrieval experiments. Some tables below also show initial 1st stage results for comparison.

4. English ad-hoc retrieval

An important finding in the TREC experiments is that short queries have substantially different retrieval properties from long ones. We consider short queries as those with a few content terms and are popular in casual environments such as web searching. For our purpose, short queries are obtained from the title section of a TREC topic. Serious users wanting more exhaustive and accurate searching should issue longer paragraph-size queries with some related conceptual terms. They usually return better results (except for very specific queries like 'hydroponics', 'osteoporosis', etc.) because longer exposition can reduce ambiguity due to homographs and descriptive deficiency due to synonyms. The 2-stage retrieval approach has been shown in several years of TREC experiments to improve over 1-stage for both query types. Our work has investigated additional methods to enhance retrieval accuracy for this strategy.

4.1. Term level evidence

We studied several methods for improving our approach of 2-stage pseudo-relevance feedback retrieval for short queries (Kwok and Chan 1998). These are related to using single term statistics and evidence, and include (see figure 2 to see when they are applied): 1) avtf query term weighting, 2) variable high frequency Zipfian threshold, 3) collection enrichment, 4) enhancing term variety in raw queries, and 5) using retrieved document local term statistics. Short queries normally use terms once and there is no frequency information to distinguish which ones are more important. Avtf employs the average occurrence statistics of terms in the collection to weight them for queries. Thus, the $q\text{tf}_k/L_q$ value in Eq. (3a) is replaced by the following: $(F_k/D_k)/\log[\max(\text{cutoff}, D_k)]$ and normalized over the query to one as discussed in Kwok (1996). Variable high frequency threshold defines statistical stopwords based on query length. When queries are short, even higher frequency terms (which normally would be excluded based on a Zipf threshold) may be useful. Both of these methods involve minimal costs as the term frequency statistics already exist and easily obtained for calculation.

Collection enrichment adds external related collections to the target collection under investigation so as to improve the chance of ranking a larger number of related or relevant

documents in the top n for the 2nd stage pseudo-feedback process. Related documents can help suggest useful terms for query expansion. This process may involve processing several times the volume of the target collection, but this is done only once. Increased collection size will also cost more time during retrieval. Typical external collections we have used are the TREC-provided newspaper collections such as Associated Press, Wall Street Journal, and foreign broadcasts such as FBIS articles, etc.

While 2nd stage query expansion in our system is a large scale (40–80) term addition process, we also experiment with pin-pointing a small number (1–10 depending on query length) of more specific terms to add to raw queries. This we call adding term variety. These specific terms are highly associated ones from the domain-related top n documents based on their having large mutual information values to the whole query, quite similar to (Xu and Croft 1996). This ends up with a multi(3)-stage retrieval, doing two steps for improving 1st stage retrieval before the 2nd, and is expensive. Retrieved document local statistics is a method to re-weigh terms in the 2nd stage using a set of retrieved but assumed irrelevant documents (such as those ranked from 500 to 1000) of the 1st stage as the irrelevant set rather than the whole collection as used at the beginning. (Singhal et al. 1997) has shown that more widespread terms within the domain-related documents may be weighted more appropriately this way than using the whole collection as irrelevant. This method involves moderate costs of re-calculating weights.

Results using these methods are tabulated in Table 1. This table is abstracted from (Kwok and Chan 1998) except that significant differences are added and TREC-8 data is new. The measure RR (or Rel-ret) is the number of relevant documents returned after retrieving 1000 documents (and totaled over all topics). This should be compared to the Total Relevant available in the heading. AvPre is the mean over all topics of the average non-interpolated precision; and R.Pre is the mean over all topics of the recall precision at the point where the number retrieved is exactly equal to the number of relevant documents. We highlight these three as the main measures for comparison. In addition, P@n documents—the precision at $n = 10, 20, 30$ documents retrieved are also shown. As in most IR experiments, differences in evaluation measures between two methods that are statistically significant are not easy to achieve, see e.g. (Tague-Sutcliffe and Blustein 1995). We mostly show percentage improvements only. In cases where the differences achieve a 5% or less level of significance based on the 1-tail rank test, the 'number of cases greater/number of cases different' ratio is also displayed. The '#' symbol is used to denote the basis from which improvements for the following right hand table column are measured.

It can be seen that standard 2-stage strategy performs about 9% to 26% better than initial retrieval using the AvPre measure as reference (TREC-6 .240 vs. .220, TREC-7 .226 vs. .180). The other techniques successively bring further improvements, accumulating to about 8 to near 50% over the standard 2nd stage retrieval results (TREC-7 .243 vs. .226, TREC-5 .241 vs. .161). Recall-oriented RR measure also shows 0 (TREC-7) to over 20% (TREC-5) increases. Collection enrichment brings in most of the gains and its AvPre improvements over normal 2nd stage retrieval are statistically significant. It is an attractive technique since searchable texts are increasingly available nowadays. We envisage that so long as the external text falls within similar topical domain of the query, it could be helpful as an enrichment tool. It is found that collection enrichment also works somewhat for long

Table 1. Term level retrieval enhancement for TREC 5–8 experiments.

	1st Retr	2nd Retr		Avtf	Var.Th	Collection Enrichment		Mut.Inf	Local Stat
	←TREC-5 50 Medium Queries; Average Unique Terms = 6.3; Total Relevant = 5524→								
		% improv.				% improv.			% improv.
RR	**1763 #**	**2279 +29**	**31/39; #**	**2335**	**2635**	**2732 +20**	**24/34; #**	**2787**	**2792 +2**
AvPre	**.140 #**	**.161 +15**	**#**	**.181**	**.214**	**.234 +45**	**34/48; #**	**.239**	**.241 +3**
P@10	.290 #	.284 —2	#	.326	.372	.382 +35	#	.404	.406 +6
P@20	.230 #	.255 +11	#	.276	.310	.329 +29	#	.339	.341 +4
P@30	.205 #	.225 +10	#	.245	.285	.301 +34	#	.307	.310 +3
R.Pre	**.179 #**	**.191 +7**	**#**	**.210**	**.249**	**.270 +41**	**28/39; #**	**.271**	**.271 +0**
	← TREC-6 50 Short Queries; Average Unique Terms = 2.6; Total Relevant = 4611 →								
RR	**2188 #**	**2272 +4**	**26/37; #**	**2384**	**2517**	**2656 +17**	**27/37; #**	**2738**	**2739 +3**
AvPre	**.220 #**	**.240 +9**	**#**	**.258**	**.258**	**.284 +18**	**37/49; #**	**.289**	**.291 +2**
P@10	.334 #	.372 +11	#	.402	.388	.444 +19	#	.442	.450 +1
P@20	.292 #	.306 +5	#	.334	.332	.377 +23	#	.386	.387 +3
P@30	.255 #	.275 +8	#	.309	.303	.337 +23	#	.350	.350 +4
R.Pre	**.262 #**	**.264 +1**	**#**	**.291**	**.287**	**.312 +18**	**28/37; #**	**.311**	**.315 +1**
	← TREC-7 50 Short Queries; Average Unique Terms = 2.6; Total Relevant = 4674 →								
RR	**2299 #**	**2964 +29**	**33/47; #**	**2960**	**2989**	**2971 +0**	**#**	**2984**	**2983 +0**
AvPre	**.180 #**	**.226 +26**	**#**	**.228**	**.227**	**.236 +4**	**36/50; #**	**.242**	**.243 +3**
P@10	.386 #	.408 +6	#	.406	.398	.434 +6	#	.446	.448 +2
P@20	.342 #	.370 +8	#	.365	.357	.368 -0	#	.372	.377 +3
P@30	.302 #	.335 +11	#	.333	.327	.325 -3	#	.329	.335 +3
R.Pre	**.231 #**	**.252 +9**	**#**	**.257**	**.255**	**.262 +4**	**#**	**.268**	**.271 +3**
	← TREC-8 50 Short Queries; Average Unique Terms = 2.5; Total Relevant = 4728 →								
RR	**2713 #**	**3133 +15**	**29/39 ; #**	**3133**	**3211**	**3279 +5**	**#**	**3299**	**3307 +0**
AvPre	**.239 #**	**.273 +14**	**#**	**.273**	**.279**	**.302 +10**	**33/50; #**	**.310**	**.310 +3**
P@10	.420 #	.448 +7	#	.448	.464	.478 +7	#	.484	.490 +3
P@20	.365 #	.404 +11	#	.408	.425	.443 +7	#	.445	.444 +0
P@30	.324 #	.365 +13	#	.365	.376	.397 +9	#	.403	.407 +3
R.Pre	**.291 #**	**.296 +2**	**#**	**.301**	**.309**	**.330+13**	**29/37; #**	**.333**	**.332 +1**

queries (see next Section). Mutual information and local statistics can bring about a further 0–3% improvement only. It is probably not worth doing because of the associated costs.

4.1.1. Example of 2-stage retrieval with varying top documents and query expansion. With large collections, 2-stage blind feedback can improve substantially over 1-stage initial retrieval as shown in Table 1. It depends on two parameters to be set: the number of top-ranked documents to be selected for feedback and the number of terms from these documents to be used for query expansion. It is difficult to obtain optimal choices under all circumstances without learning data. We show in Table 2 below an example of varying these two parameters for TREC-8 short-query retrieval. To emphasize the effects due to this operation, we show results without collection enrichment and subsequent enhancements:

Table 2. Influence of number of top-ranked documents and terms on pseudo-feedback for TREC-8 short query retrieval.

#doc\ #term	6	10	20	40	60	80
6	3012/.271/.298	3056/.272/.304	3132/.281/.313	3122/.281/.311	3133/.281/.308	3142/.279/.311
12	3020/.269/.301	3109/.278/.304	3115/.281/.305	3160/.283/.305	3185/.281/.306	3198/.282/.309
24	3047/.268/.292	3099/.275/.303	3194/.284/.313	**3184/.286/.308**	3193/.282/.313	**3211/.279/.309**
36	3047/.268/.292	3122/.277/.306	3155/.283/.309	**3205/.286/.306**	3202/.283/.303	3220/.284/.302

Values in cells are RR/AvPre/R.Pre. Initial retrieval has 2731/.244/.297 for comparison.

that is experiments at the Var.Th column of Table 1. The initial retrieval at that point has values for RR/AvPre/R.Prec equal to 2731/.244/.297 (not shown in Table 1), and should be used as the basis for comparison. Number of top-ranked documents chosen are 6, 12, 24 and 36, and for each document set, the number of terms selected vary from 6 to 80.

It is observed that for at least a ten-term expansion, effectiveness measures varies about 5% (mostly less) depending on the parameter choices. The cell in bold at (24 document, 80 term) is our choice for the TREC-8 blind experiments. Even though it is not optimal, it still improves over initial retrieval by more than 14% using the AvPre measure (.279 vs .244). Recall-related RR measure improves even more by over 17%. Even using six terms and six top-ranked documents is sufficient for $>10\%$ improvement for both AvPre and RR over 1-stage. Expanding with 80 (or more) terms can bring higher RR recall values, but AvPre begins to decline. For this experiment, a choice of around 40 terms appears best for the AvPre measure and is rather insensitive to the number of top-ranked documents used. This illustrates that 2-stage blind feedback seems to be a robust strategy for large collections and is well worth doing.

4.1.2. Example of collection enrichment with varying external documents. Collection enrichment depends on many factors including the agreement between the target and the external collections in topical domain and coverage, the quality of the pseudo-feedback documents and the number of such documents used for feedback. For example, given many candidate external collections one would need an effective strategy for choosing a suitable subset for enrichment use. An in-depth study of these factors is beyond the scope of this paper, but an illustration of how the number of pseudo-feedback documents might affect enrichment results is given in Table 3 below for TREC-8 short queries.

The baseline retrieval result for comparison is the bottom row with no enrichment. The bolded row is our blind experimental submission to TREC-8. There, we tried to limit the number of external documents for feedback to a maximum of 10 per query, hoping to avoid them overwhelming the target collection feedback, and perhaps diminish topic drifting. It can be seen for this TREC-8 collection and queries that best result is to use all top-ranked documents without restriction (top row), leading to improvements of about 8% in AvPre and 2% in RR over the bottom row. This uses about 10 external feedback documents per query. Even a small amount of external feedback documents (average 4.6 per query) can help a

Table 3. Influence of external feedback documents on TREC-8 short query retrieval.

#of external feedback docs per affected query	RR	AvPre	P@10	P@20	P@30	R.Pre
506d/50q = 10.1	3279	.302	.478	.443	.397	.330
443d/50q = 8.9	3266	.296	.476	.434	.396	.328
403d/50q = 8.1	**3273**	**.299**	**.474**	**.437**	**.393**	**.329**
229d/50q = 4.6	3241	.290	.460	.429	.384	.318
no enrichment	3211	.279	.464	.425	.376	.309

Total #of feedback documents used = 24.

little, leading to about 4% in AvPre and 1% in RR increases respectively. The improvement is not monotonic with respect to the number of external feedback items used. Naturally, the quality of the documents being brought in for feedback is important and not just the number.

4.2. Phrase level evidence

Investigators in IR are aware of the simplistic and inadequate representation of document content based on a bag of single word stems or some 2-word adjacency phrases. To a certain extent this is dictated by the requirements that text retrieval systems have to support large scale environments as well as unpredictable, diverse needs. Many previous attempts such as (Fagan 1987, Smeaton et al. 1995, Strzalkowski and Carballo 1996), have been made to include more sophisticated phrasal representation in order to improve retrieval results. They have not worked as well as content terms or results have generally been inconclusive.

We also investigated phrasal evidence for retrieval, but only to the extent that it is used to refine result that has been obtained via term level retrieval. Only long queries with three or more phrases are considered since short ones may not provide sufficient evidence to work with. Documents remain indexed using single stems but with sentence and position information captured for defining windows to match with query phrases. Specifically, we use phrasal evidence to re-rank a retrieved list with the aim to promote more relevant documents earlier in the list (Ruge 1992). This could lead to higher density of true relevant documents in the 1st stage retrieval, thereby improving 'pseudo-feedback' for the 2nd stage downstream. The 2nd stage retrieval list could similarly be re-ranked to return better final precision effectiveness.

Trying to weight the matching of phrases between a query and a document involves many considerations and setting many parameters. First, one needs to define what constitute a phrase in the query. Then one has to decide whether to account for phrase importance. Because queries are usually a few sentences only, we simply assume that every phrase is equally important. Thus, duplicate phrases are removed. Moreover, when a phrase embeds in another, we eliminate the shorter one because the longer one presumably is more specific, and also we will consider partial matching. Also, one needs to decide what constitutes a matching of a query phrase in a document. We assume that whenever two or more stems

in a query phrase occur within a window of one to three sentences, we call it a match with varying weights. Weights depend on the window size as well as the proportion of the query phrase matched. The order of the stems does not matter but multiple matching in a document are counted. In addition, we need to consider how the phrases of a query is covered by a document. If only few of the query phrases are weakly matched, we threshold it and assume no match. Finally, one also needs to consider how much weight to add to the original term level RSV. We assume that some good phrase matching is evidence that the whole overall matching is good, and simply add a weight proportional to the RSV rather than figuring out the weight from the individual stems. Thus, no particular single stems nor their weights need enter into the phrase weight calculation.

A query is processed into variable length noun phrases using a POS-tagger from Mitre and simple bracketing of adjacent tag patterns of adjectives and nouns. (We have also experimented with the BBN tagger before). Given a retrieved document, each noun phrase concept of the query is matched within up to a three sentence context anywhere in the document and re-weighing as follows:

$$\text{RSV} = \text{RSV} \times (1 + \text{phrase-factor})$$
$$\text{phrase-factor} = \text{constant} \times g(\text{phrase\_matching}) \times h(\text{query\_phrase\_coverage})$$

The phrase factor is ineffective unless it passes a query coverage test: $h(\cdot) = (\text{count2} + \text{count3})/(\text{const} + q\_\text{num\_phr}) > \text{threshold}$. q_num_phr is the number of unique and non-embedding noun phrases in a query having two or more single stems. count2 and count3 count the number of distinct query phrases that have been matched in document windows containing two or more than two stems of a query phrase. $h(\cdot)$ is arranged so that the threshold is passed when $\text{count3} > 1$, or $\text{count3} = 1$ and $\text{count2} > 1$, or $\text{count3} = 0$ and $\text{count2} > 2$. Matching three terms in a phrase is rare and we consider it more important; two term matchings are less useful and more of them is needed to be considered indicator of conceptual coverage.

$g(\cdot)$ attempts to assign importance based on how the query phrases occur in documents. Given a query phrase of q_len terms, some x of its terms may occur within one sentence in a document $n1$ times, or in two to three sentences $n2$ times. If a document window matches three or more query terms, we assign an importance of $1 + \log 2(n1)$ or $1 + 0.5 \log 2(n2)$ depending on the window size; if it matches two query terms within one sentence, the value is 1/2. Otherwise it is set to 0. This is then multiplied by the fraction of the query being matched x/q_len. All query phrase and window contributions are summed.

Empirical studies for the TREC5-8 long query environment were performed and results are shown in Table 4 (TREC-5 to TREC-7 results have appeared in our TREC-7 report; significant differences and TREC-8 data are new). Columns 1 to 3 also include results of standard 2-stage retrieval with and without collection enrichment for long queries. It is seen that enrichment also works for long queries but seems less effective than for short, improving from 3% to 15%. RR measure practically remains unchanged. Phrase re-ranking processing is done both for 1st stage in order to improve quality in the top n retrieved documents, and also for the final retrieval list of the 2nd stage (see figure 2 steps 6a, b). The attempt brings small but consistent improvements in AvPre varying from 0% for TREC8 to 4% for TREC5.

Table 4. Collection enrichment and phrase re-ranking: long queries.

	1st Stage		2nd Stage		Coll.Enrch		Phrase Re-rank	
<-- TREC-5 50 Long Queries; Average Unique Terms = 22.0 Total Relevant = 5524 -->								
		% improv		% improv		% improv		% improv
RR	**2463**	#	**3077 25,30/38**	#	**3034 -1**	#	**3072 +1**	
AvPre	**0.2197**	#	**0.2529 +15**	#	**0.2617 +3**	#	**0.2729 +4**	
P@10	0.404	#	0.414 +2	#	0.438 +6	#	0.444 +1	
P@20	0.326	#	0.367 +13	#	0.384 +5	#	0.399 +4	
P@30	0.2893	#	0.332 +15	#	0.3407 +3	#	0.356 +4	
R.Pre	**0.2579**	#	**0.2771 +7**	#	**0.2917 5,20/27**	#	**0.2959 +1**	
<-- TREC-6 50 Long Queries; Average Unique Terms = 21.3; Total Relevant = 4611 -->								
RR	**2537**	#	**2947 16,31/37**	#	**3043 3,21/29**	#	**3072 +1**	
AvPre	**0.2371**	#	**0.2643 11,34/50**	#	**0.3052 +15**	#	**0.308 +1**	
P@10	0.402	#	0.452 +12	#	0.492 +6	#	0.49 -0	
P@20	0.337	#	0.381 +13	#	0.399 +5	#	0.406 +2	
P@30	0.3067	#	0.338 +10	#	0.3607 +3	#	0.366 +1	
R.Pre	**0.278**	#	**0.2958 +6,28/41**	#	**0.3258 +5**	#	**0.3311 +2**	
<-- TREC-7 50 Long Queries; Average Unique Terms = 17.2; Total Relevant = 4674 -->								
RR	**2717**	#	**3142 16,37/44**	#	**3165 +1**	#	**3162 -0**	
AvPre	**0.2137**	#	**0.2569 +20**	#	**0.2658 +3**	#	**0.2723 +2**	
P@10	0.466	#	0.486 +4	#	0.502 +3	#	0.496 -1	
P@20	0.396	#	0.403 +2	#	0.434 +8	#	0.434 0	
P@30	0.3433	#	0.368 +17	#	0.3873 +5	#	0.3947 +2	
R.Pre	**0.262**	#	**0.2834 +8**	#	**0.2928 3,24/36**	#	**0.296 +1**	
<-- TREC-8 50 Long Queries; Average Unique Terms = 12.8; Total Relevant = 4728 -->								
RR	**2821**	#	**3278 16,40/50**	#	**3344 +2**	#	**3350 +0**	
AvPre	**0.2652**	#	**0.3064 16,33/40**	#	**0.3241 +6**	#	**0.3249 +0**	
P@10	0.458	#	0.474 +3	#	0.494 +4	#	0.504 +2	
P@20	0.395	#	0.425 +8	#	0.453 +7	#	0.447 -1	
P@30	0.338	#	0.3867 +14	#	0.408 +6	#	0.4033 -1	
R.Pre	**0.3119**	#	**0.3324 +7.30/45**	#	**0.3441** +4	#	**0.3421 -1**	

5. Chinese ad-hoc retrieval

Our investigation continues the work of other investigators on Chinese IR during Tipster 1&2 (e.g. Boisen et al. 1996). PIRCS can handle the 2-byte encoding of Chinese characters according to the GB2312 convention. During processing, our system can handle both English and Chinese present simultaneously in documents and queries.

5.1. Differences between Chinese and English for IR

The Chinese language is substantially different from alphabet-based languages like English. The basic unit of Chinese is a character, also called an ideograph. Some of these can historically be traced to a pictorial interpretation. The number of unique characters is well over ten thousand, but a smaller set consisting of about 6573 of the most commonly used is encoded as the GB2312 set. (Although a character can further be considered as constituted

of components called radicals, current popular encoding methods do not distinguish these finer details and therefore they are not useable for IR.) A character by itself can be a word with meaning, but is usually ambiguous. Most common words are composed of a string of two to four characters, with 2-character words constituting a majority of ∼70%. Longer ones are usually idioms, proper names or phrases. The boundaries between words and phrases are often unclear.

Chinese texts consist of strings of characters tightly juxtaposed with each other with no white space and ends on a punctuation mark or space. It is an agglutinative language with no delimiters between words. If English had this property, this sentence would look like this:

IfEnglishhadthisproperty, thissentencewouldlooklikethis:

The human brain can separate the words with little difficulty, but a computer program would have great problem. For example, a bad program might segment it with the following output:

If English hadt his proper ty, this sent encewo uldloo klik et his:

and it is not clear how effective IR can result from indexing these 'words'. However, if segmentation is done systematically (even though English is not agglutinative) such as the consecutive overlapping n-gram experiments in Damashek (1995), Cavnar (1995), it has been shown that $n = 4$ to 6 can give reasonable results. For Chinese, many methods have been proposed for word segmentation (e.g. Sproat et al. 1996, Wu and Tseng 1995), but the goal of perfect segmentation is elusive partly because even native speakers quite often disagree on what is the correct answer. Consequently, many Chinese IR approaches are based on n-grams with $n = 1$ or 2. They work to a certain extent because, as alluded to earlier, single characters do carry meaning and 2-grams can exhaustively cover some 70% of the correct words. Still, it is useful to explore Chinese IR with word segmentation because further linguistic investigations such as parsing for deeper language understanding, translation, summarization, etc. all require accurate word boundary detection. As an example, we show below a Chinese title sentence from one of the TREC queries being segmented based on: a) manual word segmentation together with its English translation; b) character; c) bigram and d) an example of incorrect segmentation resulting randomly in some meaningful words that are wrong in the context:

Query #CH11:

联合国驻波斯尼亚维和部队

a) Correct manual segmentation and an English translation:

联合国	驻	波斯尼亚	维和	部队
UN	station in	Bosnia	peace-keeping	force

b) Character segmentation:

联 合 国 驻 波 斯 尼 亚 维 和 部 队

c) Bigram segmentation:

联合 合国 国驻 驻波 波斯 斯尼 尼亚 亚维 维和 和部 部队

d) Example of a wrong segmentation:

联合	国	驻	波斯	尼亚维	和	部队
unite	country	station in	Persia	?	and	force

5.2. *Chinese IR with segmented words*

At the time when this project started, we believed that word segmentation is important for effective Chinese IR. Since efficient word segmentation software for large collections were not available, we developed an approximate short-word segmenter based on rules and statistical processing (Kwok 1997). However, for all segmented words two or more characters long we also add their single character components for representation. This is done for two reasons. First, because the segmenter may not be sufficiently accurate, use of characters helps in guarding against null matching between query and document. Secondly, the Chinese language is rich in abbreviations such as: 核电站 for (核子发电站 'nuclear generate electricity plant'), 伤亡 for (受伤死亡 'injuries and deaths'), 贩毒 for (贩卖毒品 'sell drug'), 美军 for (美国军队 'american soldiers'). These abbreviations are formed from selected characters from word components and are so ubiquitous that they become common words themselves.

Using single characters can also guard against null matching if the query employs one form while a document uses another. Earlier work has used word segmentation on queries only and relied on character representation for documents with operators to combine characters for matching query words (Boisen et al. 1996). Once the text has been processed into terms they are indexed, with stopword removal based on high frequency threshold only, and stored for retrieval as for English.

Table 5 (abstracted from Table 2 in Kwok (1999) with significant differences added) for both TREC-5 and 6 show that our short-word plus character indexing method (sw.c) works

Table 5. 1st, 2nd stages and combination Chinese retrieval results.

	TREC5 (Total Relevant = 2182)									
	← 28 Long Queries #Uniq Terms 35.9					→← 28 Short Queries #Uniq Terms 7.0				→
	sw.c 1st Stage	sw.c 2nd Stage	% improv	sw.c+bi	% improv	sw.c 1st Stage	sw.c 2nd Stage	% improv	sw.c+bi	% improv
RR	**1950 #**	**2061 +6**	**7/10; #**	**2111 +2**		**1818 #**	**1972 +8**	**13/13; #**	**1985 +1**	
AvPre	**.390 #**	**.463 +19**	**26/28; #**	**.471 +2**		**.345 #**	**.417 +21**	**23/28; #**	**.425 +2**	
P@10	.539 #	.604 +12	#	.621 +3		.429 #	.554 +29	#	.539 -3	
P@20	.491 #	.563 +15	#	.582 +3		.418 #	.480 +15	#	.516 +8	
P@30	.464 #	.525 +13	#	.539 +3		.391 #	.456 +17	#	.492 +8	
R.Pre	**.403 #**	**.461 +14**	**19/21; #**	**.468 +2**		**.357 #**	**.413 +16**	**21/24; #**	**.423 +2**	

	TREC6 (Total Relevant = 2958)									
	← 28 Long Queries #Uniq Terms 37.7					→← 28 Short Queries #Uniq Terms 6.0				→
	sw.c 1stStage	sw.c 2nd Stage	% improv	sw.c+bi	% improv	sw.c 1st Stage	sw.c 2nd Stage	% improv	sw.c+bi	% improv
RR	**2730 #**	**2786 +2**	**13/18; #**	**2784 -0**		**2243 #**	**2578 +15**	**20/21; #**	**2611 +1**	
AvPre	**.537 #**	**.607 +6**	**22/26; #**	**.633 +4**		**.375 #**	**.498 +33**	**23/26; #**	**.514 +3**	
P@10	.804 #	.877 +9	#	.869 -1		.615 #	.719 +17	#	.750 +4	
P@20	.735 #	.812 +10	#	.829 +2		.565 #	.700 +24	#	.710 +1	
P@30	.685 #	.771 +13	#	.782 +1		.517 #	.646 +25	#	.662 +2	
R.Pre	**.528 #**	**.567 +7**	**17/22; #**	**.582 +3**		**.404 #**	**.462 +14**	**20/23; #**	**.496 +7**	

very well, and have enabled us to return the best *automatic* retrieval results for both TREC-5 & 6 (Kwok and Grunfeld 1997, Kwok et al. 1998). It also demonstrates that the PIRCS retrieval model can handle both English and Chinese languages equally good.

It can be seen that two-stage retrieval is good for Chinese (as well as for English), leading to improvements in AvPre of some 6% (TREC-6 long) to 33% (TREC-6 short) over initial 1st stage retrieval and the differences are statistically significant. RR also improves from 2% (TREC-6 long) to 15% (TREC-6 short). Long queries perform better than short ones as in English, between 11% and 22% (TREC-5 .463 vs. .417 and .607 vs. .498). These Chinese queries return surprisingly good results even though the segmentation is approximate. Column sw.c+bi results will be discussed later in Section 5.5.

5.3. Comparing segmenters

Word segmentation is important for the Chinese language since linguistics-strong applications such as POS tagging, sentence parsing, machine translation, text to voice, etc. are all dependent on words being accurately identified to do well. It would therefore be interesting to see how better word segmentation could lead to more accurate retrieval. We have done manual analysis of our Queens approximate segmentation algorithm for correctness using the 54 TREC-5 & 6 topics and concluded that its recall and precision measures for segmenting TREC topics into short-words are about mid to high 80%. These figures are approximate because even native speakers sometimes disagree on the correct segmentation. We have also analyzed a segmenter from UMASS (Ponte and Croft 1996) that is based on a unigram model. It is trained from a collection that has been segmented based on a lexicon list. It segments a test sentence by evaluating possible choices and selecting the one with the highest probability of the trained model. Our opinion is that its recall and precision values vary between about 90% to low-90%, approximately 5% better than ours. We used both segmenters on the Chinese collection and did retrieval using PIRCS under the same parameter settings. The result is presented in Table 6 below.

The UMASS segmenter seems to show better AvPre values for short queries, while our's works better for long. This makes sense since short queries have less redundancies and

Table 6. Comparing Queens & Umass segementers.

	TREC-5 (total relevant 2182)						TREC-6 (total relevant 2958)					
	28 Long Queries			28 Short Queries			26 Long Queries			26 Short Queries		
	UMASS	Queens	% improv.	UMASS	Queens	% improv.	UMASS	Queens	% improv.	UMASS	Queens	% improv.
Rel_ret:	**2070**	**2059**	**-0**	**1991**	**1972**	**-1**	**2761**	**2791**	**-1**	**2488**	**2547**	**+2**
Interpolated-prec Avg:												
0.10	0.6734	0.7009	+4	0.6113	0.5892	-4	0.8782	0.8821	+0	0.7843	0.7355	-6
0.30	0.5402	0.5619	+4	0.4923	0.499	+1	0.7444	0.7555	+1	0.6308	0.6234	-1
0.50	0.4770	0.4837	+1	0.4301	0.4404	+2	0.6189	0.6494	+5	0.498	0.4942	-1
0.70	0.3931	0.3816	-3	0.3538	0.3705	+5	0.4633	0.4906	+6	0.3676	0.3617	-2
0.90	0.2540	0.2582	+2	0.2332	0.2461	+6	0.2654	0.2812	+6	0.2296	0.2268	-1
Av.prec (non-interpol):												
	0.4598	**0.4677**	**+2**	**0.4144**	**0.4166**	**+1**	**0.5871**	**0.6034**	**+3**	**0.4908**	**0.4755**	**-3**
Precision at x docs:												
10:	0.5893	0.6143	+4	0.5607	0.5536	-1	0.85	0.8692	+2	0.75	0.7115	-5
20:	0.5446	0.575	+6	0.5107	0.4804	-6	0.8077	0.8135	+1	0.6942	0.6692	-4
30:	0.5238	0.5262	+0	0.4881	0.456	-7	0.7551	0.7564	+0	0.6462	0.6192	-4
R-Precision:												
Exact:	**0.4529**	**0.4646**	**+3**	**0.4121**	**0.413**	**+0**	**0.5569**	**0.5666**	**+2**	**0.4757**	**0.463**	**-3**

are therefore more sensitive to accuracy in index term determination. However, it is a bit surprising to see that average results of the two segmenters are very similar. The interaction between word segmentation and IR is complex. In what follows we attempt to offer some explanation by comparing the index terms of individual short queries and their achieved effectiveness for the two segmenters, bearing in mind that our system removes stopwords via high frequency and capture single characters as index terms. First, 26 of the 54 queries have identical segmentation for the two. Two more differ only in having numbers captured by the UMASS system. Other queries may roughly be classified into three types as follow:

1) Queries insensitive to segmentation errors: these are cases where wrong segmentation occurs only in regions with high frequency terms (bigrams generally) that are non-content bearing. Whether those characters are treated as singles or segmented correctly into larger units does not matter because they are eventually removed as stopwords. Examples are words like 有关 'related' 情况 'condition' (query #5), 问题 'issues' (#9), 方面 'aspect' (#27), 造成 'leading to' (#34), etc. They are counted as errors if wrongly segmented. Thus, not all segmentation errors would adversely affect retrieval.
2) Queries where good segmentation does lead to better retrieval: these are cases where the incorrect segmentation involves content-bearing terms. Examples include: query #4 (.. 新 发现 的 油田 'newly discovered oil fields' captured by UMASS, Queens output as 新发 现 的 油田 mixing up the words 'newly' and 'discovered'); query #20 (越战 失踪 美军 'Vietnam war missing American soldiers' captured by Queens, UMASS output as 越 战 失踪 美 军 missing the commonly adopted bigram expressions and eventually loosing all single characters because of their high frequency usage except 越); query #32 (.. 贩毒 集团 'drug selling cartel' captured by Queens, UMASS output as 贩 毒 集团 which misses the common bigram abbreviation for drug selling); query #33 (两岸 劫机 'two shore hijack airplane' captured by UMASS, while Queens output 两 岸劫 机, with the single characters eventually removed leaving a meaningless bigram); query #35 (..... 二 十七日 南非 .. '27 day South Africa' captured by UMASS, Queens output as .二 十 七 日南 非 ... where the numerals are to be thrown away, but wrongly aggregating 'day with south', thereby missing South Africa). In these situations, because wrong segmentation leads to important concepts to be absent, it does cause substantial loss in effectiveness for these short queries.
3) Queries where good segmentation does not necessarily lead to good retrieval: there are at least three sub-classes that we observed. One sub-class involves low frequency non-content bearing words that got segmented correctly as bigrams and were kept because their bigram frequencies are below the stopword threshold. They actually introduce noise and depress retrieval results. A less accurate segmenter may separate these into single characters which may become stopwords and got removed, giving less noisy retrievals. Examples include: query #40 (某些 meaning 'some', captured by UMASS), query #48 (之后 meaning 'afterwards', captured by Queens). There are similar examples of English terms like 'whereabouts', etc. that are low frequency but not meaningful. One may avoid this by using a stopword list; but the list needs to be exhaustive. For our Chinese experiments we rely only on a high frequency threshold for stopword removal.

Another sub-class commonly involves foreign names. Query #39 (阿尔及利亚 'Algeria') was captured by UMASS, but our segmenter has it wrongly split into two fragments: 阿尔及 ('Alge') 利亚 ('ria'). But because the first three-character fragment is quite unique, it is sufficient to represent the 5-character string 'Algeria' well. The second fragment of two characters could act randomly as noise or signal depending on the characteristics of the collection and document. It can affect final RSV weighting because the length of an item is changed. Many foreign names may be transliterated into fairly long strings of 4 or more characters. But so long as the segmentation gives fragments that are two or more characters long and are reasonably uncommon, and that this is consistently done in both the query and the document collection, accurate segmentation may not be crucial for this situation.

A third sub-class involves segmented terms that are semantically correct but do not help retrieval results. One also encounters such cases in English IR where a semantically good word may not function as well as a less content-bearing term or its absence (see query track report: (Kwok et al. 20xx)). This may involve the peculiarities of collection statistics and the weighting used. Examples include: query #3 (营运 'operation'), captured by Queens; query #6 (支持 'support') captured by UMASS. Their presence actually leads to lesser performance.

More accurate segmentation does lead to better retrieval when they involve content-bearing terms. However, there are situations such as those above that act otherwise and may average out the results, especially for long queries where there are sufficient redundancies to remedy the situation. This might be a reason why average results are similar for the two segmenters. It is also possible that the two segmenters are not sufficiently different in accuracy to result in more noticeable differences of effectiveness. A very high quality segmenter (perhaps 95% or better) together with specially selected queries would tell a different story.

5.4. Bigram representation

We have further experimented with using simpler representation methods such as single characters and bigrams (consecutive overlapping two character) for retrieval. Bigram representation does not need any lexicon nor segmentation rules (e.g. Chien 1995, Chen et.al. 1997), but often over-generates a large number of indexing terms that are noise terms and not meaningful to humans. Character indexing is even simpler, but they are highly ambiguous since there are only 6763 distinct characters in the GB2312 scheme. Table 7 (abstracted from Table 2 in Kwok (1999)) shows examples of retrieval results using character and bigram representation. Surprisingly results with single characters are reasonable, though not competitive; and bigram results can rival those of short-words (Table 5) when the queries are long. This has important ramifications since it may mean that for effective Chinese IR, one need not deliberate too much about which segmentation method to use. Bigrams may be employed for initial coarse screening. Accurate segmentation can then be used on the much smaller number of retrieved documents for applications such as extraction, translation, etc. For large-scale collections, bigram segmentation is also more efficient time-wise since no lexicon checking is required, although it is more expensive space-wise. The 'bi.c' column in Table 7 will be explained in the next section.

Table 7. Characters, bigram and combination retrieval results.

TREC5 (Total Relevant = 2182)								
←	28 Long Queries				→←	28 Short Queries			→
	Char c	Bigram	bi.c	%		Char c	Bigram	bi.c	%
#Uniq Terms	19.3	bi 74.2	78.3	improv		7.0	bi 11.1	12.9	improv
RR	2007	2126 #	2126	0		1757	1971 #	1981	+0
AvPre	.381	.457 #	.454	-1		.318	.372 #	.387	+4
P@10	.539	.618 #	.600	-3		.421	.479 #	.511	+7
P@20	.491	.570 #	.566	-1		.407	.448 #	.477	+6
P@30	.455	.531 #	.523	-2		.388	.446 #	.451	+2
R.Pre	.403	.459 #	.456	-1		.351	.382 #	.405	+6

TREC6 (Total Relevant = 2958)								
←	26 Long Queries				→←	26 Short Queries			→
	Char c	Bigram	bi.c	%		Char c	Bigram	bi.c	%
#Uniq Terms	20.2	bi 65.2	80.9	improv		6.0	bi 10.3	12.2	improv
RR	2612	2735 #	2806	+3		2304	2342 #	2521	+8
AvPre	.512	.574 #	.627	+9		.432	.459 #	.489	+7
P@10	.785	.827 #	.858	+4		.723	.658 #	.739	+12
P@20	.744	.775 #	.810	+5		.639	.617 #	.664	+8
P@30	.683	.713 #	.759	+6		.599	.589 #	.618	+5
R.Pre	.507	.547 #	.575	+5		.433	.460 #	.482	+5

5.5. Combining representations

Since short-word with character and bigram representations separately returns comparable good results, this leads us to investigate whether they can perhaps reinforce each other (Kwok 1999). Short-words provide effective term matching between a query and a document, but one might have wrong segmentation. Bigrams however are exhaustive and can remedy the situation. Given a collection, we index it both ways. For each query we also index it both ways and perform separate retrievals. Their retrieval lists are then combined based on the RSV of each document as follows (with $\beta = 1/2$):

$$\text{RSV}_i = \beta \times \text{RSV}_{i1} + (1 - \beta) \times \text{RSV}_{i2}$$

The result, included in Table 5 as 'sw.c+bi' column, was a further improvement of about 2 to 4% compared with the 2nd stage average precision without combination, which are already at quite high values. RR however remains practically the same. If for some applications the last bit of effectiveness is important, this is a viable approach. Moreover, this strategy could be realized by having both retrievals performed in parallel on separate hardware, thus without affecting retrieval time too much.

Also included in Table 7 as the 'bi.c' column is the result of adding characters to bigram indexing, just like adding characters to short-words. Compared to bigram by itself, it is seen that this is also useful in 3 out of 4 cases. It appears that when it does not work it leads to small degradation in precision (TREC-5 long: −1%; .454 vs 0.457). When it works it leads to larger improvements of 4 to 9% (TREC-6 long: +9%; 0.489 vs. .459) over bigram results. Characters are highly ambiguous as indexing terms but there are actual Chinese words that are single character, and using bigrams only would not lead to correct term matching. Adding characters to bigrams makes query and document representations longer and therefore lengthen retrieval time proportionately, but it does not involve two separate retrievals.

5.6. Collection enrichment for Chinese IR

Section 4.1 shows that collection enrichment is an effective strategy to improve English ad-hoc retrieval, especially for short queries. Here, we see if this is also true for Chinese. The TREC Chinese collection came from two sources: 24,988 documents from XinHua News Agency (xh) and 139,801 from Peoples' Daily newspaper (pd). In PIRCS, they were segmented into sub-documents of 38,287 and 193,240 items respectively. We use the combined TREC5 and 6 queries numbering 54, and do retrieval with the xh collection as the target but enriched with pd, and vice versa. Some queries do not have any relevants in one of the sub-collections and the actual number of queries for evaluation is less. This is done for both long and short (title only) versions of the queries. This is a limited experiment compared to English as we do not have access to additional Chinese collections. Results are tabulated in Table 8.

It is seen that collection enrichment has practically no effect on RR values. For AvPre, in three out of four cases we notice small improvements of between 3 to 4% over the standard 2nd retrieval without enrichment. The exception is for the long query case retrieving on pd and enriched with xh where the measure practically remains unchanged (.499 vs. .500). It should be noticed that the xh collection is only 1/5 the size of the pd, and the value of about 50% AvPre is quite high already. More experimentation with bigger collections need to be

Table 8. Chinese IR with collection enrichment.

	Target: xh (enriched by pd, total relevant 1739)						Target: pd (enriched by xh, total relevant 3401)					
	52 Long Queries			52 Short Queries			53 Long Queries			53 Short Queries		
	Base	Coll. Enrich.	% improv.	Base	Coll. Enrich.	% improv.	Base	Coll. Enrich.	% improv.	Base	Coll. Enrich.	% improv.
Rel_ret %	**1709**	**1715**	**+0**	**1586**	**1592**	**+0**	**3264**	**3269**	**+0**	**3066**	**3052**	**-0**
Interpolated-prec avg												
0.10	0.7618	0.7839	+3	0.6745	0.6851	+2	0.7535	0.7419	+2	0.64	0.6631	+4
0.30	0.6538	0.6581	+1	0.5497	0.572	+4	0.6145	0.6156	+0	0.5012	0.5172	+3
0.50	0.5629	0.5833	+4	0.4866	0.5122	+5	0.5217	0.5189	-1	0.407	0.4166	+2
0.70	0.4455	0.4652	+4	0.358	0.3827	+7	0.4001	0.3993	-0	0.3206	0.3244	+1
0.90	0.3140	0.3259	+4	0.2323	0.2487	+7	0.2688	0.2649	-1	0.215	0.2131	-1
Av.prec (non-interpol)												
	0.5325	**0.5457**	**+2**	**0.445**	**0.4622**	**+4**	**0.4997**	**0.499**	**-0**	**0.4043**	**0.4156**	**+3**
Prec at x doc												
10:	0.5712	0.5885	+3	0.4942	0.5038	+2	0.6642	0.6849	+3	0.5415	0.5811	+7
20:	0.4856	0.5	+3	0.4173	0.425	+2	0.5906	0.5887	-0	0.4991	0.516	+3
30:	0.4321	0.4353	+1	0.3635	0.3705	+2	0.5484	0.5465	-0	0.4616	0.473	+2
R-Precision:												
Exact:	**0.5056**	**0.5131**	**+1**	**0.4303**	**0.4433**	**+3**	**0.4849**	**0.4839**	**-0**	**0.4133**	**0.4198**	**+2**

Table 9. Influence of external collection (pd) size on xh retrieval.

External coll. size	#of feedback doc per affected query	RR	AvPre	P@10	P@20	P@30	R.Pre
whole pd	1007d/52q = 19.4	1592	.462	.504	.425	.371	.443
∼65MB	840d/52q = 16.2	1594	.465	.508	.430	.370	.442
∼33MB	646d/52q = 12.4	1601	.461	.508	.424	.369	.444
∼16MB	419d/52q = 8.1	1584	.455	.492	.420	.369	.440
∼8MB	248d/48q = 5.2	1591	.449	.479	.410	.367	.432
∼4MB	153d/42q = 2.9	1580	.443	.489	.413	.363	.418
∼2MB	86d/39q = 1.7	1578	.445	.489	.415	.367	.426
no enrich	0.0	1586	.445	.494	.417	.364	.430

Total #of feedback documents used = 30.

performed, but the indication is that collection enrichment appears to work also for Chinese collections.

To illustrate how external collection property might influence collection enrichment results, we have performed further experiments on short queries retrieving the XinHua (xh) collection enriched by different amounts of the Peoples' Daily (pd) documents. Short query retrieval has shown larger influence by collection enrichment. The external collection is chosen as every second, fourth, eighth, etc. document from pd resulting in external collections of sizes approximately 1/2, 1/4, 1/8, 1/16, 1/32, and 1/64 of 130 MB. However, it is the quality and number of pd documents fetched for feedback that is meaningful; the external collection size has only indirect influence. It is difficult to determine document quality, but tabulated in Table 9 is the variation with size. The bottom row shows results without collection enrichment and can be used as a basis. For example, using 1/2 of pd as external collection (size ∼65 MB) leads to 840 feedback documents affecting 52 queries for an average of ∼16.2 external feedback documents per query. It has effectiveness similar to using the whole pd for collection enrichment (AvPre .465 vs .462). Improvements are less than those in Table 3 probably because the basis is already high. Enrichment effect begins to be noticeable (AvPre .455 vs basis AvPre .445) when the average number of feedback documents reach about 419/52 = 8, approximately 1/4 of the total of 30 feedback documents used. More experimentation needs to be done with other varied collections and queries to draw firmer conclusions.

6. General discussion of efficiency issues

This paper has concentrated on looking at ways to improve the effectiveness of ad-hoc retrieval. In real life, efficiency is also an important issue even though costs of processor speed, disk, and memory space have dropped dramatically. Search systems for local optical disks or text databases of an organization may have to support retrieval for half to tens of gigabyte size data efficiently. Wide-use retrieval systems such as those for web searching

have to be designed to face extremely stringent requirements in order to support may be millions of inquiries a day on hundreds of gigabytes of text and data. In this section we lightly discuss whether the methods proposed before can be efficiently implemented.

One may imagine a large sparse N (documents) $\times M$ (terms) matrix with cells containing term weights to represent a processed collection. We assume that term positions in documents are not captured since phrase operation is expensive and not too effective. If the matrix is stored column-wise, document identities containing a term are gathered into a posting list under that term. The totality of all term lists constitutes an inverted file organization (Frakes and Baeza-Yates 1992) that still seems to be the most popular design for IR. This way, given a query with t distinct terms, t posting lists need to be extracted from the inverted file. These lists are then merged and each document on the lists can be given an RSV based on the query terms present and the retrieval model used. Assuming that the merging operation can be done in memory, then other things being equal, the time of retrieval (disk accesses) would be proportional to the number of distinct terms in a query, each term having an average size posting list. Thus, $\mathrm{T1} = \mathrm{a} + \mathrm{b} \times t$ where T1 is the time cost for an initial retrieval, a is a constant pertaining to the fixed costs of executing a retrieval, and b is another constant related to accessing an average size posting list.

In this scenario, collection enrichment for initial retrieval means increasing N by the size of the external collection (and adding storage cost proportionately), with the result of lengthening the average size of posting lists. If the lists are well-positioned on a disk, retrieving it may not need additional access time, only transfer time. The time cost for initial retrieval with collection enrichment would then be $\mathrm{T1}' = \mathrm{a} + \mathrm{b}' \times t$, where $2 \times \mathrm{b} > \mathrm{b}' > \mathrm{b}$ can be assumed. 2nd stage retrieval complicates things much further. After n top-ranked documents are known from the first stage, one needs to obtain the various terms contained in such documents. The data can be efficiently obtained only if the collection matrix has been stored row-wise as well, doubling the storage cost. From these documents one can obtain the m query expansion terms via list merging in memory. Thus, 2nd stage retrieval would entail time cost proportional to fetching the n document lists plus another retrieval with m that is larger than the original t terms. This time cost would be $\mathrm{T2} = \mathrm{a} + \mathrm{b} \times (m + n)$ if we consider document term lists have the same average size as posting lists. As an illustration, assume T1 for a 2-term initial query is $\mathrm{a} + 1$ seconds, then using collection enrichment may bring this operation to something like $\mathrm{a} + 1.5$ seconds. Employing 6 feedback documents with a 8-term 2nd stage query would likely result in a retrieval taking $\mathrm{a} + 7$ seconds. This leads to a total of $2 \times \mathrm{a} + 8.5$ seconds compared to $\mathrm{a} + 1$. If the fixed cost a is a few seconds, this should be quite acceptable for a local retrieval system, and can buy an effectiveness improvement of $>10\%$ using TREC-8 as an example (Table 2). For web search engines where retrieval time for one query may be optimized to tenths of a second or less, 2-stage strategy may be considered too expensive with current disk access speeds.

For Chinese retrieval, the storage cost of the posting lists is vastly different for characters, bigrams, and short-word with characters. In GB coding, since there are only 6573 different characters, one would expect much longer 1-gram posting lists. Bigrams on the other hand have many more unique patterns and one would expect more lists with much shorter average length. Segmented short words would lie in between. After removing high frequency terms as stopwords it turns out that for the collection matrix, bigram representation takes about

80% more space than short-word with characters. There are however more than 5 times as many posting lists for bigrams, but each list is on average 1/3 that of the short-word with character representation (see Table 1 in Kwok (1999)). When a raw query of t characters is given (such as #CH11 of Section 5.1), one would form $t-1$ bigrams and many of these will be kept as query terms (non-stopwords). When the same query is processed as short-words with characters, it will produce t characters plus at most $t/2$ segmented words. However, most of the single characters as well as some segmented short-words are high frequency and will be removed. The resultant query size is actually shorter than for bigrams on average: about 6.5 to 11 ratio for short queries and 36 to 72 for long (Kwok 1999). The effect on retrieval is that short-word with characters may need about 0.5 to 1.5 times the number of disk accesses for bigrams depending on how longer posting lists affect disk accesses.

7. Conclusion

A 2-stage retrieval strategy with pseudo-feedback often returns better ad-hoc results than 1-stage alone. We have investigated further term-and phrase-level evidence methods for improving retrieval accuracy in this situation. We showed that five term-level methods, operating at different points on the pseudo-feedback loop, together are effective for enhancing ad-hoc short query 2^{nd} stage results some 8 to over 40% for TREC5-8 experiments. A particularly useful technique is collection enrichment, which simply adds domain-related external collections to a target collection to help improve 2^{nd} stage retrieval downstream. It brings substantial improvements in many cases and does not hurt much in others. It works more often than not for short as well as long queries in both English and Chinese IR. For long queries, we studied an approach of employing linguistic phrases from queries to match within document windows as further evidence to re-rank retrieval output. It can lead to fairly small, consistent improvements.

Before the Tipster and TREC programs, Chinese IR was considered a difficult problem because of the additional word segmentation problem. These experiments showed that, contrary to expectation, simple representation methods such as bigrams or short-words (via approximate segmentation) with characters can lead to highly effective results. Our short-word segmentation algorithm is approximate; but when it was compared to one that is about 5% more accurate, better average result was not observed. We offered some explanation. Further small gains are possible using collection enrichment, or combination of retrieval lists from bigrams and short-word with character retrievals. These observations however must be regarded as tentative until they are confirmed with much bigger scale experimentation. So far, Chinese data consists only of 54 queries judged against 170 MB of text, compared to English data that is an order of magnitude larger. Additional evaluated data is essential to maintain progress in Chinese IR.

For the future, one needs to have more accurate segmentation software and experiment with carefully selected queries to show the effects of segmentation on retrieval results. Collection enrichment seems to be a simple and effective tool to improve ad-hoc retrieval. Variations of this technique such as discovering methods to select the most useful external collections could enhance the technique further. Both collection enrichment and query expansion need to be studied on a query by query basis. Phrase-level evidence conceptually

should work for IR, but has not been demonstrated to be useful so far. It could be fruitful to continue studying how to weight phrases in harmony with the term-level weights for both English and Chinese IR. Perhaps the latest linguistics-oriented approach to IR may throw new light on this old problem.

Acknowledgments

This work was partially supported by a contract from the U.S. Department of Defense MDA904-96-C-1481 and a PSC-CUNY grant. I like to express my appreciation to R. Weischedel for use of the BBN POS tagger; L. Hirschman for the Mitre POS tagger and W.B. Croft for the UMASS Chinese segmenter. Also to my students J.H. Xu for help in Chinese processing, to M. Chan for experiments in Section 4.1. I am also indebted to the anonymous reviewers for many valuable suggestions.

References

Attar R and Frankel AS (1977) Local feedback in full-text retrieval systems. J. of the ACM, 24(3):397–417.

Boisen S, Crystal M, Petersen E, Weischedel R, Broglio J, Callan J, Croft B, Hand T, Keenan T, and Okurowski M (1996) Chinese information extraction & retrieval. In: Proceedings of Tipster Text Program (Phase 2), Sept. 1996, pp. 109–119.

Cavnar WB (1995) Using an *n*-gram-based document representation with a vector processing retrieval model. In: Harman DK, Ed., Overview of the Third Text REtrieval Conference (TREC-3). NIST Special Publication 500-234, GPO, Washington, DC, pp. 269–277.

Chien LF (1995) Fast and quasi-natural language search for gigabytes of Chinese texts. In: Proc. of 18th Ann. Intl. ACM SIGIR Conf. on R&D in IR. pp. 21–28.

Chen A, He J, Xu L, Gey F and Meggs J (1997) Chinese text retrieval without using a dictionary. In: Proc. of 20st Ann. Intl. ACM SIGIR Conf. on R&D in IR. pp. 42–49.

Croft WB and Harper D (1979) Using probabilistic models of information retrieval without relevance information. J. of Documentation, 35:285–295.

Damashek M (1995) Gauging similarity via *n*-grams: Language independent categorization of text. Science, 246:843–848.

Fagan JL (1987) Experiments in automatic phrase indexing for document retrieval: A comparison of syntactic and non-syntactic methods. PhD Thesis, Department of Computer Science, Cornell Univeristy, TR 87-868.

Frakes WB and Baeza-Yates R (Eds.) (1992) Information Retrieval—Data Structures and Algorithms. Prentice-Hall, Englewood Cliffs, NJ.

Hiemstra D and Kraaij W (1999) TREC-7 working notes: Twenty-one in ad-hoc and CLIR. In: Voorhees E and Harman DK, Eds., The Seventh Text REtrieval Conference (TREC-7). NIST Special Publication 500-242, GPO, Washington, DC, pp. 133–142.

Kwok KL (1990) Experiments with a component theory of probabilistic information retrieval based on single terms as document components. ACM Transactions on Office Information System, 8:363–386.

Kwok KL (1995) A network approach to probabilistic information retrieval. ACM Transactions on Office Information System, 13:324–353.

Kwok KL (1996) A new method of weighting query terms for ad-hoc retrieval. In: Proc. 19th Annual Intl. ACM SIGIR Conf. on R&D in IR. pp. 187–195.

Kwok, K.L. (1997) Lexicon effects on Chinese information retrieval. Proc. of 2nd Conf. on Empirical Methods in NLP, pp. 141–148.

Kwok KL (1999) Employing multiple repesentations for Chinese information retrieval. J. of the American Society for Information Science, 50(8):709–723.

Kwok KL and Chan M (1998) Improving two-stage ad-hoc retrieval for short queries. In: Proc. 21st Ann. Intl. ACM SIGIR Conf. on R&D in IR. pp. 250–256.

Kwok KL and Grunfeld L (1997) TREC-5 English and Chinese retrieval experiments using PIRCS. In: Voorhees EM and Harman DK, Eds., Information Technology: The Fifth Text REtrieval Conference (TREC-5). NIST Special Publication 500-238. GPO, Washington, DC, pp. 133–142.

Kwok KL, Grunfeld L and Chan M (20xx) TREC-8 ad-hoc, query and filtering track experiments using PIRCS. To be published by NIST.

Kwok KL, Grunfeld L and Xu JH (1998) TREC-6 English and Chinese retrieval experiments using PIRCS. In: Voorhees E and Harman DK, Eds., The Sixth Text REtrieval Conference (TREC-6). NIST Special Publication 500-240, GPO, Washington, DC, pp. 207–214.

Miller DRH, Leek T and Schwartz RM (1999) A hidden Markov model information retrieval system. In Proc. 22nd Ann. Intl. ACM SIGIR Conf. on R&D in IR. pp. 214–221.

Ponte J and Croft WB (1996) Useg: a retargetable word segmentation procedure for information retrieval. In: Symposium on Document Analysis & Information Retrieval (SDAIR 1996).

Ponte J and Croft WB (1998) A language modeling approach to information retrieval. In: Proc. 21st Ann. Intl. ACM SIGIR Conf. on R&D in IR. pp. 275–281.

Robertson SE and Sparck Jones K (1976) Relevance weighting of search terms. J. of American Society for Information Science, 27:129–146.

Robertson SE and Walker S (1997) On relevance weights with little relevance information. In: Proc. 20st Ann. Intl. ACM SIGIR Conf. on R&D in IR. pp. 16–24.

Ruge G (1992) Experiments on linguistically-based term association. Info. Proc. and Mngmt, 28:317–332.

Salton G and Buckley C (1990) Improving retrieval performance by relevance feedback. J. of American Society for Information Science, 41(4), 288–97.

Singhal A, Mitra M and Buckley C (1997) Learning routing queries in a query zone. In: Proc. 20st Ann. Intl. ACM SIGIR Conf. on R&D in IR. pp. 25–33.

Smeaton AF, O'Donnell R and Kelledy F (1995) Indexing structures derived from syntax—TREC3 system description. In: Harman DK, Ed., Overview of The Third Text REtrieval Conference (TREC-3). NIST Special Publication 500-225, GPO, Washington, DC, pp. 55–68.

Sproat R, Shih C, Gale W and Chang N (1996) A stochastic finite-state word-segmentation algorithm for Chinese. Computational Linguistics, 22:377–404.

Strzalkowski T and Carballo JP (1996) Natural language information retrieval: TREC-4 report. In: Harman DK, Ed., The Fourth Text REtrieval Conference (TREC-4), NIST Special Publication 500-236, GPO, Washington, DC, pp. 245–258.

Tague-Sutcliffe J and Blustein J (1995) A statistical analysis of the TREC-3 data. In: Harman DK, Ed., Overview of The Third Text REtrieval Conference (TREC-3), NIST Special Publication 500-225, GPO, Washington, DC, pp. 385–398.

Voorhees E and Harman D (1997) Overview of the Fifth Text REtrieval Conference (TREC-5). In: Voorhees E and Harman DK, Eds., Information Technology: The Sixth Text REtrieval Conference (TREC-5), NIST Special Publication 500-238, GPO, Washington, DC, pp. 1–28.

Voorhees E and Harman D (1998) Overview of the Sixth Text REtrieval Conference (TREC-6). In: Voorhees E and Harman DK, Eds., Information Technology: The Sixth Text REtrieval Conference (TREC-6), NIST Special Publication 500-240, GPO, Washington, DC, pp. 1–24.

Wu Z and Tseng G (1995) ACTS: An automatic Chinese text segmentation system for full text retrieval. Journal of the American Society for Information Science, 46:83–96.

Xu J and Croft WB (1996) Query expansion using lcoal and global document analysis. In: Proc. 19st Ann. Intl. ACM SIGIR Conf. on R&D in IR. pp. 4–11.