Improving Two-Stage Ad-Hoc Retrieval for Short Queries

K.L. Kwok and M. Chan

Computer Science Department, Queens College, CUNY
Flushing, NY 11367, USA
http://ir.cs.qc.edu/

Abstract Short queries in an ad-hoc retrieval environment are difficult but unavoidable. We present several methods to try to improve our current strategy of 2-stage pseudo-relevance feedback retrieval in such a situation. They are: 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. Avtf employs collection statistics to weight terms in short queries. Variable high frequency threshold defines and ignores statistical stopwords based on query length. Collection enrichment adds other collections to the one under investigation so as to improve the chance of ranking more relevant documents in the top n for the pseudo-feedback process. Enhancing term variety to raw queries tries to find highly associated terms in a set of documents that is domain-related to the query. Making the query longer may improve 1st stage retrieval. And retrieved document local statistics re-weight terms in the 2nd stage using the set of domain-related documents rather than the whole collection as used during the initial stage. Experiments were performed using the TREC 5 and 6 environment. It is found that together these methods perform well for the difficult TREC-5 topics, and also works for the TREC-6 very short topics.

1 Introduction

Automatic ad-hoc retrieval using short queries is a knotty yet unavoidable problem in IR. It is knotty because in ad-hoc environment, the query is unpredictable and one does not have any history of the information need. This precludes the training advantage that one may have such as, for example, in a routing environment. A short query means a statement of one or a few words, and certainly not a paragraph of description, and therefore a retrieval system has very scarce clue to work with for predicting relevance of a document. Yet, in reality, short queries are very common. It has been reported that users in Internet searches average close to one word per query. This may simply be due to users generally prefer to minimize the chore of typing, or that users have misconceptions of the power of computers, thinking that computers can know what one wants based on just a word or two.

This paper explores a few approaches that may help in such a situation. Some results based on the TREC collections are also presented as evidence of the viability of these methods. In all cases, the PIRCS retrieval system was used as the experimental platform. It has been described in [Kwok95] and the TREC proceedings [e.g. KwGX9x].

2 2-Stage Retrieval

A popular method in TREC experiments to enhance ad-hoc retrieval is to use a 2-stage strategy, and under the right circumstances it can give substantial improvements in effectiveness. In 1-stage retrieval, a raw query that 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 considered relevant, even though no user judgment is done. These pseudo-relevant documents are then used to modify the weight of the initial query according to some probabilistic procedures, 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 become the final retrieval result. The process of the 2-stage retrieval is depicted in Fig.1.

When one deals with short queries of a few words that are also not very specific in nature, the initial retrieval list can be quite poor. Because no user judgment is done on this list, there can be many non-relevant documents mixed in the top n (often chosen to be $6<n<50$) pseudo-relevant set. When n' terms (often chosen to be $10<n'<80$) are then selected to expand the query, many of them will be of dubious usefulness. It is therefore surprising that this process works at all. Yet experiments with our PIRCS system have shown that it works more often than not, about 2 out of 3 times (35 queries in TREC-5 and 32 in TREC-6), and the average precision for a set of queries can improve a few to over 20%. It appears quite often that even though the documents used for pseudo-feedback are not relevant, they fall in the general topical area of the raw query. The retrieval algorithm works to the extent that documents of the same general area are biased to rank high. These documents then provide a source of terms that can augment the initial query in topical description. This expanded query can pull documents of the same topical area and improve the chance of having relevant documents ranked high in the second retrieval.

Nothing beats having genuinely relevant documents in the top n for pseudo-feedback because then it would behave much

like true relevance feedback. True feedback has traditionally return much higher effectiveness than no feedback – like 100% and better [SaBu90]. In this case of pseudo-feedback, one can only hope for a high percentage of relevant documents in the top n. However, there are cases where the number of relevant documents in the collection is actually very small – say 2 or 3. No matter how good the initial retrieval is, one cannot bring more relevants to the top n ranked list. In such cases, because of the low precision of the top n documents, second stage retrieval should be avoided. But one does not know how many relevants there are for each query without a manual judgment of documents.

In any case, improving the precision of the initial top n documents appears to be crucial for the second stage to work well. A raw short query lacks two properties that can give reasonable initial retrieval results, namely, adequate term weighting and sufficient user need description. Terms in short queries are 'flat' in that they seldom repeat, and so their within-query frequencies are not much use for distinguishing term importance. Shortness of queries also means lack of term variety and term redundancy in expressing the information needs. This leads to difficulties in matching with documents because of the synonym and homograph problems.

We focus on some automatic methods that may help in such situations. What follows discuss each of these methods in turn: a) avtf query term weighting - Section 3; b) variable high frequency Zipf threshold - Section 4; c) collection enrichment - Section 5; d) adding term variety to raw query - Section 6; and e) retrieved document local term statistics - Section 7. The conclusion is in Section 8.

3 avtf Query Term Weighting

In short queries, terms are all practically used once and their usage frequency does not provide discriminative term importance. In [Kwok96] we introduced the average term frequency (avtf) weighting for such situation. The idea is that the way these terms are used in the collection may give a clue as to how such terms might appear in a query if the author were asked to write the same query longer. In the collection, the average frequency of term k given that it occurs in a document is: F_k/D_k , where F_k is the collection frequency of term k and D_k is the corresponding document frequency. For use in short-query term weighting, we introduce also consideration of how the term behaves in the collection via the inverse log max factor, and the actual weight used for term k is given by:

$$avtf_k = (F_k/D_k)^{1.5} * 1/\log \max[cutoff, D_k]$$

where cutoff is a constant set to 2000 so that low frequency terms do not get over-weighted. In addition, within each query a factor is used to normalize the sum of all query term weights to one.

This weighting is assigned at the 1st stage retrieval shown as '1' in Fig.1. They get modified during 2nd stage pseudo-feedback. In Figure 2a,b we show how such weighting can improve average precision for both TREC-5 and TREC-6 short queries from Method 0, which is the default retrieval of our system without avtf, to Method 1. The lower curve plots the 1st stage results while the upper curve displays the 2nd stage. Compared to Method 0 (2nd stage), avtf weighting improves over 7% for TREC-6 and over 12% for TREC-5. It can also be seen that 2-stage retrieval improves substantially over 1-stage retrieval in both cases and is the recommended strategy for ad-hoc.

More detailed measures from the TREC evaluation program such as relevants-retrieved, precision at x documents, etc. are also given in Table 1 columns marked Method 0 and 1. Percentage improvements using Method 0 1st stage as the baseline are also given. For example, relevants retrieved improves from 1763 to 2335 for TREC-5 (32%) and from 2188 to 2384 for TREC-6 (9%).

4 Variable High Frequency Threshold

In IR, it is well known that median frequency terms are best for retrieval. It is usual practice to define two frequency thresholds - one low and one high - for screening out index terms that may not be useful based on the Zipfian behavior of word usage in a large collection of text. For example, words occurring once, twice or three times only in a collection of about 1 million usually mean that they are probably typographic errors. Even if they do carry meaning, their rare occurrence in the collection means that they also would be used rarely in queries, and hence in the long run they would not substantially affect retrieval on average. According to the Zipf Law, half of the unique words seen will have frequency 1 only, and one sixth of them will have frequency two. Screening them out will save substantial space for storage as well as processing time. High frequency words on the other hand will be so prevailing that they will not be very useful for discriminating relevant documents from irrelevant ones. They may be regarded as statistical stopwords. Leaving them out also will lead to a more efficient system by eliminating these words with long postings, and may also lead to better effectiveness by suppressing a source of ambiguity. There is fair consensus as to the low threshold value, but the high threshold is quite problematic. For large TREC collections, a value like 60K have served us well. This works out to about 8% of the ¾ million sub-documents generated in a typical collection 2 gigabytes in size.

For short ad-hoc queries of a few words, we have concern that after stopword removal and high frequency filtering, there may not be much left of a query - a phenomenon of 'term loss'. For example, query #286 ('Why does the cost of paper rises?') would be left with just one word: 'paper' if a high threshold of 60K is used. Another example is query #318 ('Best retirement country'), which would also be left with the word: 'retir'.

Since adequate description of an information need is important, it may be a good idea to retain as much of the raw query words as possible when queries are short. High frequency terms still can carry some indication of the topic, when there is not much else to use. After some experimentation, we found that a simple two-grade value can help: a 60K threshold for queries with 12 words or longer, and 100K for short ones. This method is shown as '2' in Fig.1.

Results of Method 2 are shown in both Fig.2 and Table 1. For TREC-5, it leads to substantial improvements in the 2nd stage retrieval to a precision of 0.2139 (about 53% compared to baseline) even though the first stage only performs slightly better than not using it. For TREC-6, since none of the very

short queries are longer than 12 words, the 2-grade value does not apply. Using a higher threshold of 100K versus 65K practically leaves the precision unchanged but improves the number of relevants retrieved substantially, as it should in both cases (Table 1: TREC5: 2335 to 2635, TREC-6: 2384 to 2517).

5 Collection Enrichment

As discussed in Section 2, some queries have few relevant documents in the whole collection under investigation. Other queries behave poorly and few relevant documents got ranked high. In such circumstances, the top n documents of an initial retrieval will have low precision and the second stage retrieval may not work. A technique to remedy this situation is to borrow from other collections with similar documents in order to enrich the current retrieval collection and improve the probability that there are relevant documents for the queries under investigation [WaRH9x]. [ACCB9x] also makes use of external documents to improve their local context analysis for expansion of their queries. In TREC-1, we made the following observation when we compare retrieval results between large (like Wall Street Journal WSJ) and small collections (like CACM):

> "At low recall region, however precision of WSJ is comparable or better than the small collections. Why is that? First, ... Second, when a collection is large, there is a very good chance that a number of relevant documents exist using closely the same terms as the queries describing their content, especially if the queries are well-worded. These documents will rank high, ..."

The borrowed collections should be reasonably similar to the current collection, else we might introduce only noise. It essentially increases the collection size and improves the probability of having more relevant documents ranked high. We assume that all the ad-hoc collections on the TREC disks satisfy this criteria. Thus, for each query of the TREC5 ad-hoc task (which retrieves against disk 2 and 4), we first do a 1-stage retrieval against disk 1, 3 and 5 to pull out the first 200 sub-documents of each sub-collections of the disks. These form a 'miscellaneous sub-collection' and augment the retrieval collections on disk 2 & 4 during the first stage retrieval. Some of these miscellaneous documents may be relevant and using the same wordings as the query, and therefore got ranked within the top n and become 'pseudo-relevant' feedback documents for the second stage retrieval. Some are not relevant but could be on similar topical area of the query and use related wordings, and also got ranked within the top n. This way, the set of feedback documents may become enriched. For the TREC6 task (which retrieves against disk 4 and 5), we enrich it with disks 1, 2 & 3. This method is shown as '3' in Fig.1.

From Figure 2, we observe that this Method 3 is quite successful, leading to average precision of 0.2339 for TREC-5 and 0.2835 for TREC-6 (improvements to 67.4% and 28.7% respectively over baseline). Similar additional improvements are also observed in other measures as tabulated in Table 1.

6 Adding Term Variety to Raw Queries

After an initial retrieval, we trust our retrieval engine to return a set of n top-ranked documents and their term usage to define the topical domain of a query. If we employ the raw query and obtain highly associated terms within this domain, the chance of getting some related terms could be good. The idea is to improve the description and redundancy of the raw query. Other people have tried to add terms by various means with mixed results. For example: [Voor94] uses the synsets of terms in Wordnet; [XuCr96] adds in a large scale phrases from the whole collection that are highly similar to the query phrases; [SmvR83] uses statistically related terms from the whole collection as well as from judged relevant documents. There are many measures of term association such as based on raw co-occurrence frequency or correlation coefficient. We use the term presence factor of the expected mutual information measure as described in [vanR79].

For each raw query term, we identify candidate associated terms in the context of a sub-document according to:

$$MuInf = P(t1, t2) \log [P(t1, t2) / (P(t1)P(t2))]$$

where P(t1, t2) = probability that both a query term and a document term will appear together in a single sub-document, and P(t1), P(t2) = probability that a query term or document term will appear in a sub-document.

The average MuInf value of a document term to the query terms is then used to rank the candidates. Some of these candidates can have high document frequency in the collection and we have previous experience that adding to a query broadly-used terms may actually be counter-productive. On the other hand, not all high frequency terms are useless (see examples in Kwok96]). We have devised a quality test that helps us select these candidates based on the avtf formula discussed in Section 3.

The quality test is implemented as follows: if the document frequency of a candidate is less than a threshold (chosen as 11000), it is selected because we believe it would be sufficiently specific in meaning; else it would be rejected unless it has avtf value >= 0.17. These thresholds are chosen after some experimentation.

In addition, a table is used to decide a variable number of terms to add according to the query length. The rationale is to guard against error of commission since a longer query can absorb a couple of wrong terms without changing its original intent too much while this would not be true for queries of one word or two. With many trials, we decide to use the following table to determine the maximum number of candidate terms to add to a query as listed in the following:

Query Len	Add	Query Len	Add
<=2	0	7 to 8	4
3	1	9 to 10	5
4	2	11 to 12	6
5 to 6	3	>12	7

Some examples of adding term variety to queries are given in the table below. Query 252 concerns '.. steps taken by government or private entities world-wide to stop the smuggling of aliens' has length 9 and we allow a maximum of 5 terms to be added. Query 255 needs 'countries that do not practice or ignore environmental protective measures' considers 4 terms but skips 'requir' because of its low avtf

value. Query 259 concerns 'new theories about the 1960's assassination of president kennedy' brings in 3 terms but also omit 'john'. Query 265 is on 'domestic violence in the U.S.' has length of 3 and we only add 1 term. Query 285 concerns '.. the number of submarines, both nuclear-powered and conventional, in the inventories of all countries in the world' adds 4 terms. Of these queries, 255 & 285 lead to worse result after adding the terms, while the others improve.

q#	len	candidate term	avg. muinf	docfq	avtf	select
252	9	illeg	.470	11911	.184	
		immigr	.261	7058	.374	
		polit	.248	70412	.181	
		in	.223	2604	.341	
		border	.203	16086	.197	
255	7	impact	.325	44491	.195	
		determin	.298	97244	.212	
		requir	.277	70526	.154	no
		epa	.275	15914	1.08	
		health	.248	57014	.372	
259	5	conspiraci	.424	3377	.198	
		john	.379	59281	.128	no
		conspiraci -theori	.285	201	.162	
		kill	.260	28794	.192	
265	3	victim	.360	13239	.190	
		displac	.353	2839	.244	no
285	7	plant	.699	39959	.314	
		reactor	.549	4756	.483	
		license	.435	6184	.613	
		weapon	.326	15248	.265	

This process of adding term variety is different from query expansion for the 2nd stage. Query expansion makes use of a small number of pseudo-feedback documents, and the expansion is large scale: we expand with 30-80 terms. In this situation, we add only a few highly associated terms and the purpose is to improve the quality of the top n documents in the initial retrieval. In effect, the initial retrieval has been broken up into two retrieval steps, resulting in a 3 stage process. This method is shown as '4' in Fig.1.

Results with adding term variety is also shown in Fig.2 and Table 1 under Method 5. Additional effect as measured from the baseline is small but positive, about 4% for TREC-5, and 2% for TREC-6 where queries are shorter. Considering the cost of an extra retrieval process and the small gains in precision, it may not be worth doing.

7 Retrieved Document Local Statistics

Using local statistics for term weighting was introduced in [SiMB97] in the context of routing, and we have implemented the method here to see if it can bring further improvements in short query ad-hoc circumstances. The idea is that the discriminative power of term k in a collection is initially approximated by an Inverse Collection Term Frequency formula in our PIRCS system as follows:

$$w_{k|global} = \log (1-q_k)/q_k$$

where

$$q_k = \Pr(\text{term k occurs} \mid \text{non-Relevant to query}) \cong F_k / N_w$$

F_k is the collection term frequency of term k, and N_w is the total number of terms in the collection. (The complementary factor $\log p_k/(1-p_k)$ that depends on relevant document learning and is the basis of our 2-stage retrieval is kept unchanged).

This term weight $w_{k|global}$ is designated as global because it is obtained using whole collection statistics, and is useful for isolating a domain-related set of documents (as is the case with our initial retrieval set) from the general corpus. However, for further retrieval refinement within this set, the continued use of this weighting may not be appropriate anymore. For example, the word 'computer' may help a user isolate documents relating to computing from general news articles in a heterogeneous corpus, but once within the computing domain where many articles will have the word 'computer', it should have less discriminating use and its weight should be downgraded. This would happen if the above weight is re-calculated within a set of initial retrieved documents that are deemed irrelevant. We chose this set to be the 500 documents ranked from 501 to 1000 in the initially retrieved set. Using this set, the F_k, N_w and $w_{k|local}$ values are defined.

It turns out that a direct replacement of $w_{k|global}$ by $w_{k|local}$ causes too drastic a change. A ratio of the old and new weight is used instead as follows:

$$w_k = (1-x) * w_{k|global} + x * w_{k|local}$$

with $x = 0.06$ to 0.1.

This method is marked '5' in Fig.1, and its effect is shown in Fig.2 and Table 1 as Method 5a,b. 5a shows result using local statistics after applying Method 4 and increases average precision from 0.2392 to 0.2405 in TREC-5, while 5b skips Method 4 and also increases precision from 0.2835 to 0.2902 for TREC-6. The effect is small or about unchanged in both cases. Perhaps in an ad-hoc short query environment without real user judgment, the division of relevant and irrelevant documents is extremely blurred and the method may not be as applicable as in a routing situation.

8 Conclusion

Retrieval via short queries in an ad-hoc environment presents difficulties for an IR algorithm. Yet it cannot be ignored because it is very common. We try out several approaches to improve the retrieval effectiveness in such a situation. We are able to make successful accumulative improvements of 32 to 72% in precision automatically when compared with simple 1-stage retrieval with the raw queries. However, some of these are obtained with complexity and at the expense of retrieval time. For example, Method 4 (adding term variety) requires a 3rd retrieval and the gains of both Method 4 and 5 may be too small to be significant.

Acknowledgment This work is partially supported by a contract from the Department of Defense (MDA904-96-C-1481). We are also grateful to N. Dinstl who contributed in extending the PIRCS system for this project.

References

[ACCB9x] Allan, J, Callan, J, Croft, W.B, Ballesteros, L, Byrd, D, Swan, R & Xu, J. (199x). InQuery does battle with TREC6. (TREC-6 draft paper).

[Kwok96] Kwok, K.L. (1996). A new method of weighting query terms for ad-hoc retrieval. In: Proc. 19th Ann. Intl. ACM SIGIR Conf. On R&D in IR. Aug 18-22, 1996. Frei, H.P., Harman, D. Schauble, P. & Wilkinson, R. pp.187-195.

[Kwok95] Kwok, K.L. (1995). A network approach to probabilistic information retrieval. ACM Transactions on Office Information Systems. 13:325-353.

[KwGX9x] Kwok, K.L., Grunfeld, L. & Xu, J.H. (199x). TREC-6 English and Chinese experiments using PIRCS. In: The Sixth Text REtrieval Conference (TREC-6). To be published.

[SaBu90] Salton, G & Buckley, C (1990). Improving retrieval performance by relevance feedback. Journal of American Society for Information Science. 41(4): 288-297, 1990.

[SiMB97] Singhal, A, Mitra, M & Buckley C (1997). Learning routing queries in a query zone. In: Proc. of 20th Ann. Intl. ACM-SIGIR Conf. On R&D in IR. Belkin, N.J, Narasimhalu, D, & Willett, P (eds). pp.25-32.

[SmvR83] Smeaton, A.F. & van Rijsbergen C.J. (1983). The retrieval effects of query expansion on a feedback document retrieval system. The Computer Journal, 26, pp.239-246.

[vanR79] van Rijsbergen, C.J. (1979). Information Retrieval, 2nd Edition. London: Butterworths.

[Voorh94] Voorhees, E.M. Query expansion using lexical-semantic relations. In: Proc. of 17th Ann. Intl. ACM-SIGIR Conf. On R&D in IR. Jul 3-6, 1994. Croft, W.B. & van Rijsbergen, C.J., (eds). pp.61-69.

[WaRB9x] Walker, S., Robertson, S.E. & Boughanem, M. (199x). Okapi at TREC-6: automatic ad hoc, VLC, routing and filtering. (TREC-6 draft paper).

[XuCr96] Xu, J. & Croft, W.B. (1996). Query expansion using local and global document analysis. In: Proc. 19th Ann. Intl. ACM SIGIR Conf. On R&D in IR. Aug 18-22, 1996. Frei, H.P., Harman, D. Schauble, P. & Wilkinson, R. pp.4-11.

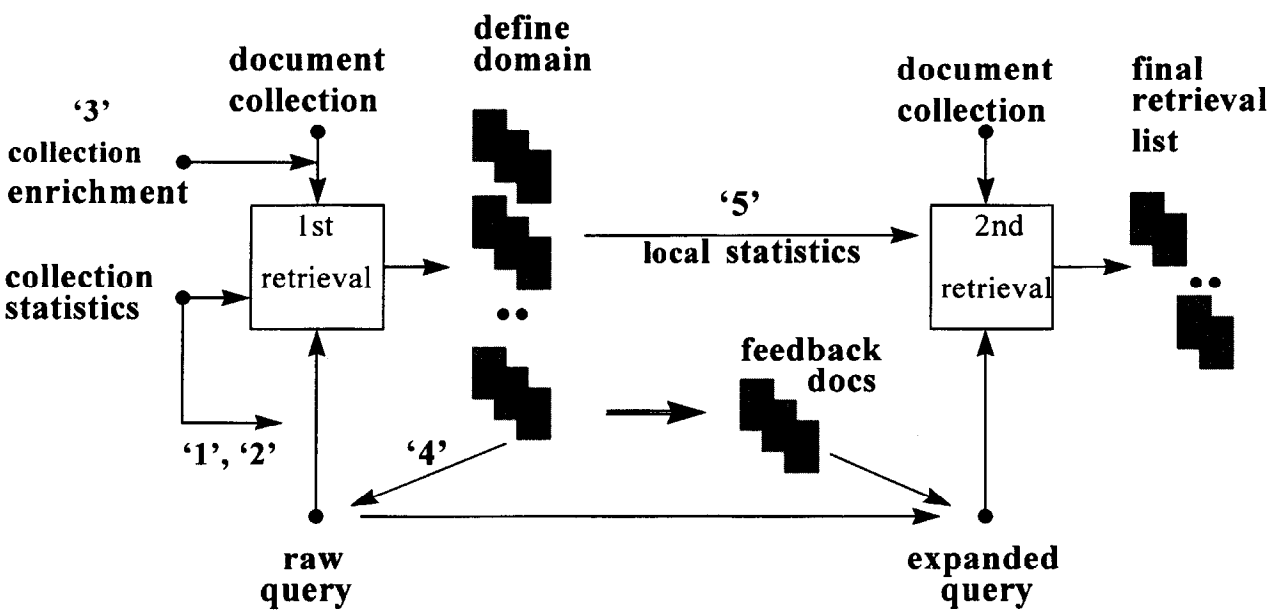

Fig.1 Two-Stage Retrieval and 5 Methods of Improvements

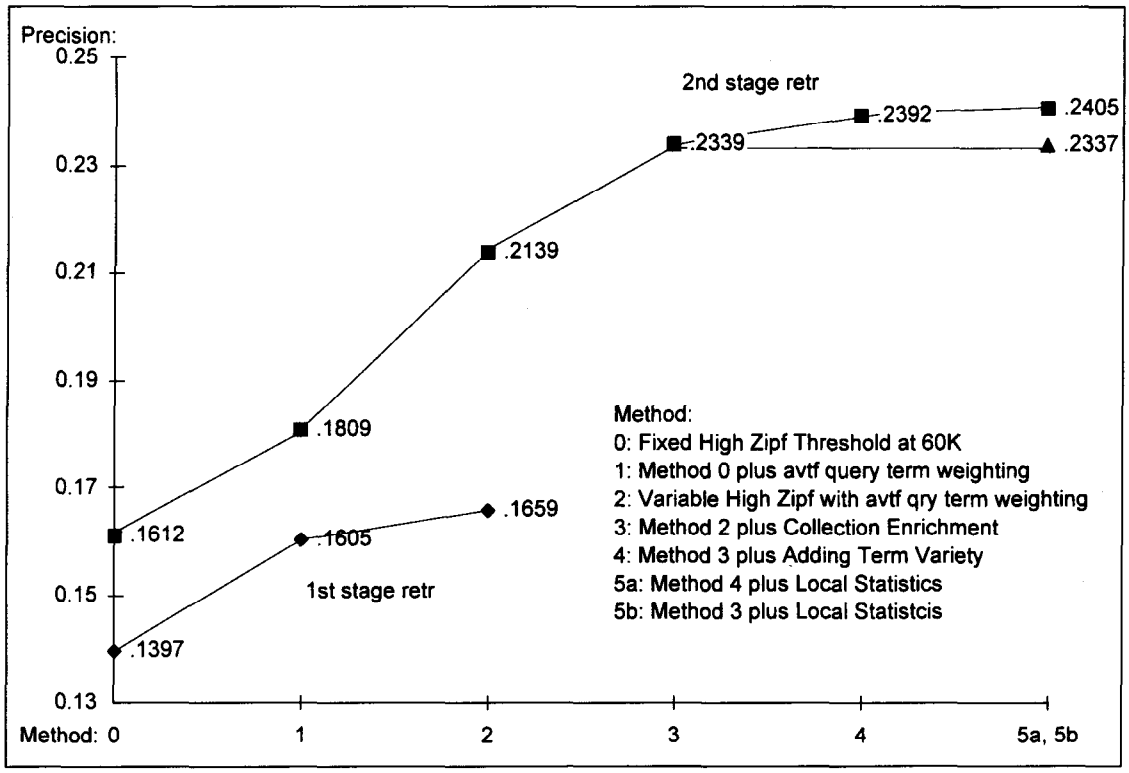

Figure 2a: TREC-5 results for 50 short queries: various methods

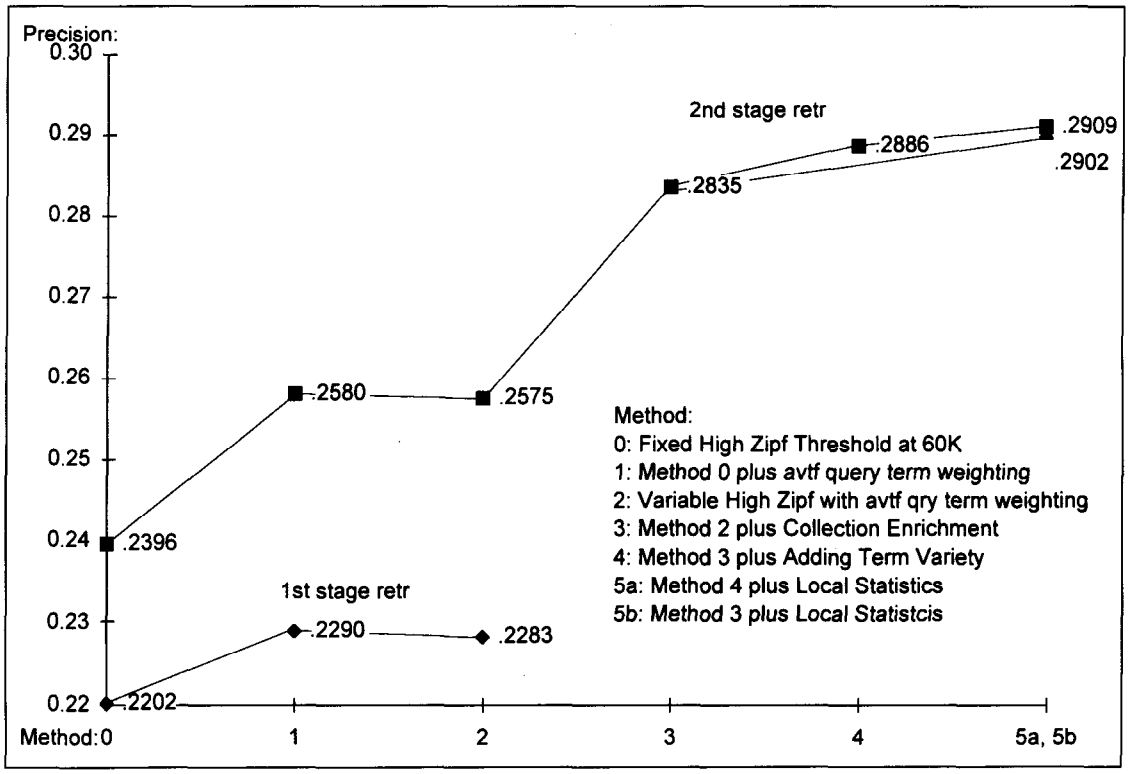

Figure 2b: TREC-6 results for 50 short queries: various methods

	Fixed High Zipf Threshold: 60K				Variable High Zipf Threshold (max) : 100K					
	No avtf query term weighting		With avtf query term weighting		With avtf query term weighting		With Collection Enrichment	Add Term Variety	With Local Statistics Add Term Variety	No Term Variety
Method:	0		1		2		3	4	5a	5b
Retr. stage:	1st	2nd	1st	2nd	1st	2nd	2nd	2nd	2nd	2nd
Relevant:	5524	5524	5524	5524	5524	5524	5524	5524	5524	5524
Rel_ret:	1763	2279	1990	2335	2057	2635	2732	2787	2792	2735
Improvement:	0%	+29%	+13%	+32%	+17%	+49%	+55%	+58%	+58%	+55%
Interpolated Recall - Precision Averages:										
at 0.10:	.3143	.3205	.3221	.3306	.3282	.3855	.4030	.4196	.4181	.3989
at 0.30:	.1796	.2129	.2021	.2372	.2218	.2809	.3037	.3116	.3153	.3051
at 0.50:	.1135	.1552	.1510	.1823	.1506	.2154	.2335	.2336	.2354	.2354
at 0.70:	.0691	.0985	.0948	.1255	.0960	.1515	.1622	.1651	.1651	.1620
at 0.90:	.0267	.0377	.0401	.0443	.0445	.0546	.0635	.0604	.0611	.0640
Average precision (non-interpolated) over all rel docs:										
	.1397	.1612	.1605	.1809	.1659	.2139	.2339	.2392	.2405	.2337
Improvement:	0%	+15%	+15%	+29%	+19%	+53%	+67%	+71%	+72%	+67%
Precision:										
At 10 docs:	.2900	.2840	.3120	.3260	.3260	.3720	.3820	.4040	.4060	.3780
At 20 docs:	.2300	.2550	.2460	.2760	.2570	.3100	.3290	.3390	.3410	.3330
At 30 docs:	.2053	.2247	.2173	.2453	.2360	.2853	.3013	.3067	.3100	.3047
At 100 docs:	.1362	.1554	.1420	.1728	.1560	.1960	.2084	.2124	.2124	.2068
R-Precision (precision after R (= num_rel for a query) docs retrieved):										
Exact:	.1788	.1913	.1951	.2097	.2043	.2494	.2697	.2707	.2705	.2722
Improvement:	0%	+7%	+9%	+17%	+14%	+39%	+51%	+51%	+51%	+52%

Table 1a: TREC-5 (Title Only Query) Ad-Hoc Retrieval Results (Averaged over 50 queries)

	Fixed High Zipf Threshold: 65K				Variable High Zipf Threshold (max) : 100K					
	No avtf query term weighting		With avtf query term weighting		With avtf query term weighting		With Collection Enrichment	Add Term Variety	With Local Statistics Add Term Variety	No Term Variety
Method:	0		1		2		3	4	5a	5b
Retr. stage:	1st	2nd	1st	2nd	1st	2nd	2nd	2nd	2nd	2nd
Relevant:	4611	4611	4611	4611	4611	4611	4611	4611	4611	4611
Rel_ret:	2188	2272	2126	2384	2173	2517	2656	2738	2739	2691
Improvement:	0%	+4%	-3%	+9%	-1%	+15%	+21%	+25%	+25%	+23%
Interpolated Recall - Precision Averages:										
at 0.10:	.4367	.4540	.4463	.5094	.4677	.4813	.5372	.5343	.5417	.5457
at 0.30:	.2753	.3067	.2906	.3297	.2813	.3292	.3791	.3938	.3971	.3901
at 0.50:	.2191	.2346	.2254	.2470	.2139	.2502	.2731	.2861	.2867	.2849
at 0.70:	.1363	.1763	.1414	.1784	.1400	.1852	.1765	.1797	.1792	.1778
at 0.90:	.0473	.0713	.0498	.0771	.0487	.0826	.0812	.0819	.0823	.0849
Average precision (non-interpolated) over all rel docs:										
	.2202	.2396	.2290	.2580	.2283	.2575	.2835	.2886	.2909	.2902
Improvement:	0%	+9%	+4%	+17%	+4%	+17%	+29%	+31%	+32%	+32%
Precision:										
At 10 docs:	.3340	.3720	.3600	.4020	.3660	.3880	.4440	.4420	.4500	.4440
At 20 docs:	.2920	.3060	.3170	.3340	.3150	.3320	.3770	.3860	.3870	.3860
At 30 docs:	.2553	.2747	.2740	.3087	.2773	.3033	.3367	.3500	.3500	.3493
At 100 docs:	.1772	.1858	.1848	.2084	.1832	.2088	.2168	.2244	.2264	.2210
R-Precision (precision after R (= num_rel for a query) docs retrieved):										
Exact:	.2619	.2638	.2731	.2912	.2631	.2866	.3115	.3114	.3150	.3147
Improvement:	0%	+1%	+4%	+11%	+0%	+9%	+19%	+19%	+20%	+20%

Table 1b: TREC-6 (Title Only Query) Ad-Hoc Retrieval Results (Averaged over 50 queries)