INQUERY at TREC-5

James Allan, Jamie Callan, Bruce Croft
Lisa Ballesteros, John Broglio, Jinxi Xu, Hongmin Shu

Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts
Amherst, Massachusetts 01003, USA

The University of Massachusetts participated in five tracks in TREC-5: Ad-hoc, Routing, Filtering, Chinese, and Spanish. Our results are generally positive, continuing to indicate that the techniques we have applied perform well in a variety of settings.

Significant changes in our approaches include emphasis on identifying key concepts/terms in the query topics, expansion of the query using a variant of automatic feedback called "Local Context Analysis", and application of these techniques to a non-European language.

The results show the broad applicability of Local Context Analysis, demonstrate successful identification and use of key concepts, raise interesting questions about how key concepts affect precision, support the belief that many IR techniques can be applied across languages, present an intriguing lack of tradeoff between recall and precision when filtering, and confirm once again several known results about query formulation and combination.

Regrettably, three of our official submissions were marred by errors in the processing (an undetected syntax error in some queries, and an incomplete data set in an another case). The following discussion analyzes corrected runs as well as those (not particularly meaningful) submitted runs.

Our experiments were conducted with version 3.1 of the INQUERY information retrieval system. INQUERY is based on the Bayesian inference network retrieval model. It is described elsewhere [5, 4, 12, 11], so this paper focuses on relevant differences to the previously published algorithms.

1 Ad-Hoc Experiments

The emphasis in this year's Ad-Hoc experiments was on how to create very effective queries from short natural language topics. In particular, how to create *several* rather different queries from the Description field of the topic. The underlying assumption, based on past experience, was that a combination of several "pretty good" queries would be more effective (or, at least easier to create) than one "excellent" query.

Three techniques were explored. The first, which we call *basic query processing* in this paper, is an extension of the query processing techniques we used in previous TREC evaluations. The second, which we call *core query processing*, attempted to identify and emphasize the most critical query terms. The third, *local context analysis*, is a new approach to query expansion [14]. Each technique is discussed in more detail in the subsections that follow.

Two Ad-Hoc runs were submitted. Each query in INQ301 was a combination of three queries created from the Description (<desc>) fields of topics 251-300. Each query in INQ302 was a combination of two queries created from the Description, Narrative (<narr>), and <title> fields of the topics.

Experiments were run on TREC volumes 2 and 4, using the released version of Inquery 3.1. No special changes were made for TREC. The most significant changes in Inquery relative to last year's TREC experiments is the adoption of the Okapi term frequency formula ($\frac{tf}{tf+K}$) [10] and of the Kstem dictionary-based stemming algorithm.[8]

1.1 Basic Query Processing

As in past years, our *basic query processing* is a series of processing steps that produce a "words" part of the query and a "phrase" part of the query.

The "words only" portion of the query is generated by removing negation and "stop" phrases (e.g., "To be relevant, a document should contain"), converting hyphenated terms into an OR of hyphenated and un-hyphenated forms, and recognition and expansion of country names.

The "phrase" portion of the query is initially processed in a similar manner, except that words are not discarded immediately, because doing so would interfere with later syntactic processing. Instead, they are marked as "stopped" for later deletion.

Syntactic processing is based on a few assumptions about full-text queries: when a purpose clause is present, it is the most important part of the query. After that, there are certain prepositions that tend to signal significant concepts, for example, "of" and "for."

In addition, different kinds of questions require slightly different treatment. In queries of the form "What is the X of Y?" and "How X is Y?", Y tends to be the important concept. For the incomplete sentence "X done by Y for Z", Z and Y are likely to be significant. However, a laconic sentence fragment such as "X of Y" suggests equal treatment for both X and Y. Weighting of terms reflected all of this. In addition, phrases were built out of significant noun groups. Time constructions were for the most part deleted or down-weighted. Unless a date is spelled out and found in the focus position (e.g., "What happened on month day, year?") it is unlikely to be reliable.

In order to do this processing, the queries were tagged with part-of-speech information and a loose parse was constructed, based on constituents such as noun phrases and prepositional phrases. Conjunctions were handled as completely as possible so that a preposition with multiple objects would receive credit for each object, as in "the cost of military outfitting and equipping." Some kinds of appositives were recognized, such as acronyms.

Two shortcomings of the current state of development:

1. No advantage is taken of groupings revealed by constituents and conjunctions beyond that of noun group phrasing; and
2. Enumerative appositives are not distinguished and handled.

The latter problem is not difficult to address. The major issue is to distinguish whether appositives are restrictive, adding little new information, or whether they give additional examples of subjects of interest. The surprising thing is that a great deal can be done with syntax alone, time permitting. A lexicon with even simple argument structure, or use of collection statistics to infer the same information, would enhance the syntactic analysis at crucial decision points.

1.2 Core Query Processing

The *core query processing* was an extension of basic query processing intended to improve precision at low recall. It is based on the observation that in many queries, there are one or two terms that *must* be in a document for the document to be relevant. Our goal was to identify such terms automatically, and then to ensure their presence in the top-ranked documents.

The procedure used to select the required, or *core*, terms was elaborate, and is probably more complex than necessary. The goal was to apply several forms of (uncertain) evidence to identify core terms. The procedure consisted of the following steps applied to the Description field of the topic.

1. Discard words on the query stopword list (e.g., "relevant", "document"). Rank the remaining terms by $w \times avg\_tf^{0.7} \times idf$, where w is the basic query processing weight, avg_tf is the term's average frequency in documents where it occurs, and idf is Inquery's inverse document frequency.[1] The top ranked term is considered the *core* term.

2. If the core term is part of a phrase identified by basic query processing, the phrase becomes the core term.

3. Cluster the query terms using EMIM, as in [6]. If the cluster contains multiple terms, replace the core term with a proximity operator (unordered window) containing the clustered terms. The size of the window ranged from 20 to 200, depending upon the number of matches in the corpus (smaller when there are many matches).

4. If the term weighted most highly by basic query processing is not in the core cluster, add it.

5. If the proximity operator matches fewer than 10 documents, it is too strict. Relax it by discarding terms with low weights until at least 10 documents are matched.

Although complex, this procedure consists of essentially three steps. Query terms are ranked by two methods (one statistical, one linguistic), and clustered by one. The results are combined to produce a proximity operator of core terms that retrieves at least 10 documents, but preferably not too many more. If this approach proves useful, it can be made substantially simpler and more efficient.

The set of documents retrieved by the proximity operator is always a subset of the documents retrieved by basic query processing. Documents are ranked within each set by their basic query processing scores. The two sets are then combined such that documents retrieved by the proximity operator (a small set, we hope) are ranked ahead of documents retrieved by basic query processing.

1.3 Local Context Analysis

Local Context Analysis (LCA) is an "automatic feedback" technique that expands the query by including terms which occur in the top-ranked documents, but only if they occur near query terms. It is explained in detail elsewhere.[14]

LCA first runs a query against the database to get the top n documents. The concepts from those documents are ranked based on the their co-occurrence (in the top ranked documents) with the individual query terms. (The hypothesis is that good expansion concepts tend to co-occur with all query words in the top ranked documents.) The top ranked m concepts are used for query expansion.

For the Ad-hoc track, we used the 100 top-ranked 300-word passages rather than the entire document. Concepts that were available for expansion were noun phrases as recognized by Jtag[13]. 70 concepts were added to each query with tapering weights w_i for concept i:

$$w_i = 1.0 - 0.9 * (i - 1)/70$$

[1] The approach was inspired by Kwok[7], but the formula is original.

1.4 Combining queries

The three types of queries—basic, core, and expansion—were combined in two different ways. For INQ301, the automatic run starting from short (description only) topics, the combination was:

Expanded Query = `#wsum`(1.0 1.0 basic-query
2.0 core-query
4.0 expansion-query)

That is, the belief from the expanded portion of the query was four times as strong as that from the basic query.

For INQ302, the basic query processing used the description and narrative sections, and the core concept processing was omitted. Those queries were formed as:

Expanded Query = `#wsum`(1.0 1.0 basic-query
2.0 expansion-query)

2 Routing Experiments

We submitted only one run in the routing track, and made few changes from last year's run; our focus was on work in the other tracks. The differences between this year and last were in original term weighting, application of Dynamic Feedback Optimization, and INQUERY's belief score.

Queries were expanded by adding terms, adjacent word pairs, and nearby word pairs. The selected concepts were chosen from a large candidate set by comparing their occurrences in relevant and non-relevant training documents. Weights were assigned using a Rocchio combination of "term belief". Finally, the weights were adjusted by fitting them more closely to the training data using a technique very similar to one described by Buckley and Salton[2].

For the final run, as required by the routing guidelines, collection wide statistics such as "idf" and "average document length", were taken from the training collections. Collection frequency information in the test database was not used in any way.

2.1 Term selection

Training data for routing came from TREC disks 1–3 as well as the TREC-4 routing disk. For each query, we used all known relevant training documents for that query as well as the same number of top-ranked non-relevant documents identified by running the original query against the training databases which had previously been judged for that query. Because we consider any unjudged document to be non-relevant, this process required building 5 training databases. (The "original query" refers to the result of automatically creating an INQUERY structured query from the original TREC topic, without reference to relevance judgments. A more complex version of that process is discussed in the Ad-hoc description above.)

For each query, all terms occurring in at least 5 relevant documents were identified, and were then ranked by their relative occurrences in the relevant and non-relevant documents. That is, by:

$$\frac{df_{rel}}{n_r} - \frac{df_{nonrel}}{n_{nr}}$$

where df_{rel} is the total number of relevant documents containing the term, df_{nonrel} is that count in non-relevant documents, n_r is the number of relevant documents, and n_{nr} is the number of non-relevant (recall that here $n_r = n_{nr}$). The top 50 terms in that order were chosen and weighted using a Rocchio formula:

$$\beta \cdot \frac{1}{df_{rel}} \sum_{rel} belief - \gamma \cdot \frac{1}{df_{nonrel}} \sum_{nonrel} belief$$

where $\beta = 8$, $\gamma = 2$, and the belief for term t in document d was calculated by the formula:

$$\frac{tf_{t,d}}{tf_{t,d} + 0.5 + 1.5 \frac{doclen}{avgdoclen}}$$

where $tf_{t,d}$ is the number of occurrences of term t in document d, *doclen* is the length of document d in words, and *avgdoclen* is the average length (in words) of the documents in the training collection for this query. This equation is the "tf" portion of the belief function used by INQUERY version 3.1; it was adopted from Okapi[10].

The use of top-ranked non-relevant documents for negative feedback information is the approach that we used last year also,[1] prefering to balance the number of relevant and non-relevant used in training. An alternate approach is to use all of the collection other than the known relevant documents for negative information. Buckley et al[3] have recently adopted an approach similar to ours which they call "query zoning."

2.2 Additional concepts

The same process described above was applied to find concepts based upon pairs of terms also. This is the same technique that we applied last year.[1] In this case, candidate pairs were found considering *only* the 200-word passage of the training document which best matched the original query. From those passages, 50 adjacent term pairs (ordering significant) were chosen. In addition, 50 each of word pairs within 5, 20, or 50 words (order *in*significant) were added. Selection and weighting were done exactly as described above.

In all, each query was augmented with 250 new concepts, though there was some overlap. In query 240, for example, "anti terror" appeared in every category.

To create the expanded query, the original query was first flattened to remove some of its structure. Phrases, pairs of words in proximity, synonym operators, and other "concept creation" operators were left intact, but evidence-combining operators such as `#sum` were removed. The query was then a set of concepts which were each assigned a weight of 10 times the number of occurrences of that concept in the original query.

The 250 new concepts were then added to that list and assigned the weight as described above. Note that this means that original query terms tended to have a weight that was 2-3 times that of the new concepts. "Mistakes" in that weighting were corrected in the next step.

Buckley et al[3] explored term co-occurrences in their recent work, too. They used twice as many non-relevant documents for statistics-gathering, and used different measures for term selections, but the idea is similar.

2.3 Weight adjustments

We again used the Dynamic Feedback Optimization approach of Buckley and Salton[2] to adjust the chosen weights to achieve higher effectiveness in the training data, predicting that this effort will result in better effectiveness in the test documents.

The approach starts by evaluating the query on the training data. Then some concept's weight is adjusted and the slightly different query is evaluated. If the effectiveness has improved, the change is retained; if the new weight hurts effectiveness, the original is restored. In both cases, the next concept weight is tried. This process repeats until all concept weights have been "tweaked" once, completing a pass.

In TREC-4 we repeated each pass until no more improvement was found, but that caused a small problem with overfitting. For TREC-5, we reverted to Buckley and Salton's original approach and just ran a pass of adjustments once for each possibility: increase the weight by 100%, then 50%, 25%, 12.5%, and finally 6.25%. So each concept had the opportunity to be adjusted 5 times.

For efficiency reasons, the evaluations were done using only the top 20,000 documents retrieved from that query's training set in response to the new query (prior to reweighting). Effectiveness of a "tweak" was measured by average precision among those 20,000 documents.

3 Spanish Experiments

The Spanish retrieval experiments built upon the work done in TREC-4 [1]. This year's focus was on combining global analysis and local feedback for query expansion via Local Context Analysis as described in Section 1.3.

3.1 Query Formation

Query processing for the Spanish topics is similar to that of the English topics and was used to generate base queries for retrieval. Base queries were then expanded using Local Context Analysis. When expanding, the top 31 concepts were added with multi-term concepts wrapped in a `#phrase` operator with the restriction that all terms be found within a window of 25 terms. The top concept was given a weight of 1.0 with all subsequent concepts down-weighted by 1/100 for each position further down in the rank (so the second one had a weight of 0.99, then 0.98, and so on).

Concepts (single terms or phrases) containing terms occurring in the original query (called "duplicate phrases") and phrases recognized in the original parsing of the base queries (called "query phrases") are considered to be more important. These concepts were identified and given more weight.

Two sets of queries were generated, one using only topic descriptions (SIN300) as the base for expansion and the other using both descriptions and narratives (SIN301). The original query and additional concepts were combined in two ways with the following basic structure:

```
#wsum(1.0 1.0 original-query
          4.0 LCA-concepts
          4.0 duplicate-phrases and query phrases)
```

For run SIN300 the third group contained both duplicate phrases and query phrases. The SIN301 run was done similarly, but the third group of concepts contained only duplicate phrases.

4 Chinese Experiments

INQUERY required few modifications to support Chinese information retrieval.

1. Each Chinese character is indexed as a term. Exceptions are made for the characters making up numbers and the elements of dates, which are indexed as a group.

2. Queries are segmented to separate them into words. The hidden Markov model segmenter[9] is used for this purpose.

4.1 Segmentation

To allow for flexibility in segmentation at query time, the Chinese documents were indexed by treating each Chinese character as an individual term.

The queries were then segmented automatically with a segementer based upon hidden Markov models[9]. In the resulting query, each segmented Chinese word was represented by a proximity operator which required the glyphs in the word to be located immediately adjacent and in order.

To compensate for possible segmenter errors, sequences of single characters were grouped together in a `#phrase` operator, which is evaluated as a proximity of `#3` if the group of glyphs appears commonly in the textbase, or as a sequence of single glyphs if the sequence is infrequent. Single characters not found in a sequence were downweighted.

4.2 Query Construction

Title and Description fields were used. Each word in the description was weighted as a single term (except for isolated single characters), but the whole title was weighted as one term. The segmenter was augmented with a name recognizer to reduce errors of name segmentation.

The queries were then expanded by a slight modification of the Local Context Analysis method described in the section on Adhoc English query processing[14]. For purposes of expansion, any segmented "word" in the top-ranked documents was a candidate. Query concepts were segmented words found in the title or description text of the topic. Concepts for expansion were taken from the 10 top-ranked documents and only 30 concepts were added. Concept i was assigned weight:

$$w_i = 1.0 - (i - 1)/60$$

Two runs were done to test the effect of the expansion terms. One run (called HIN300) weighted the expansion section at half the weight of the original query. The other run (HIN301) gave the expansion section the same weight as the original query. (In the English Ad-hoc track, the expansion terms were given two or four times the weight of the original query.)

5 Filtering Experiments

Our approach to the filtering track leveraged heavily off of the routing track's results. After the final routing queries were generated (as described above), we ran each query on its *training* database. For a given query, we then found the point in its ranked list which yielded the optimum value for each of the styles of filtering—that is, where including all documents ranked from 1 to that point resulted in a maximum utility score.

We then ran the queries on the *test* database and for each style of filtering, used the calculated threshold to decide which documents were filtered. Because any collection-specific information that was used in ranking (e.g., idf and average document length) was taken from the training database, the scores were directly comparable. Clearly, if the test database's idf value were used, this approach could not possibly work.

Note that the submitted documents were therefore top portions of the same ranked lists submitted for the routing run—with the exception that in some cases more than 1,000 documents were filtered because the threshold was so low.

	nl	B*	C*	B+C	L*	B+L	C+L	B+C+L*
1-1-1	0.1442	0.1796	0.1852	0.1851	0.1604	0.1919	0.1990	0.1964
1-2-4	–	–	–	0.1851	–	0.1801	0.1978	0.2001
1-2-4?	–	–	–	–	–	–	–	0.2046

Table 1: Average precision breakdown for various combinations of runs. nl is the description only, B is basic query processing, C is core concepts, and L is the LCA. Table 2 shows more detail on the starred runs. BCL is the same as INQ301. Rows show different weights assigned to B, L, and C components; "4?" means either 4 or 0, whichever is best.

6 Ad-Hoc Results and Discussion

Two sets of results, INQ301 and INQ302, were evaluated in the ad-hoc document retrieval evaluation. Both sets of results were based on completely automatic processing of the TREC topic statement into two queries, and automatic query expansion. The difference is that INQ301 was formed from the Description (<desc>) field only, while INQ302 was formed from both the Description and Narrative (<narr>) fields.

A query processing error caused every query in INQ302 to reduce to the useless, but valid, query "2.0". Hence the output from that run is also useless: all it demonstrates is that UMass did not examine any documents prior to submitting results to NIST. In the table below, INQ302c is an unofficial run showing what we intended to submit (the "corrected" run).

The INQ301 adhoc run ("Description only") was treated as the baseline run. The official evaluations, and one unofficial run, are summarized below, using the results for all 50 queries.

Query Type	Average Precision			
	5 Docs	30 Docs	100 Docs	11-Pt Avg
INQ301	.38	.28	.17	.20
INQ302	.02 (−95.7%)	.01 (−98.3%)	.00 (−98.5.%)	.00 (−97.9%)
INQ302c	.40 (+7.5%)	.30 (+9.6%)	.21 (+21.0%)	.23 (+13.3%)

As expected, using the Narrative field to create queries is helpful, producing a 13.3% improvement in average precision. We do not know yet whether the improvement is due to creating an improved base query, or whether it is due to improved query expansion.

Tables 1 and 2 show the contributions of the various components in the INQ301 queries, as compared with a baseline using all (and only) the words in the Description field. Both basic and core query processing techniques produced significant improvements in precision at all levels of recall. Query expansion terms/phrases, when used by themselves as a query, are also significantly better than the words in the Description field at all recall levels, although they are significantly worse at ranks 1–20.

It is somewhat surprising that the core query processing technique improved recall instead of precision (as compared with basic query processing). This technique was intended to "promote" a small number documents that contain key query terms. The danger was that it would promote many irrelevant documents. The results suggest that it promoted a few extra irrelevant documents, depressing low-recall precision slightly. It is not immediately clear how it improved precision at high recall.

As expected, a weighted combination of basic query processing, core query processing, and local context analysis terms/phrases was more effective than any of its components at 10% recall and

Recall	Precision (50 queries)								
	Desc Only	Base Query Processing (B)		Core Query Processing (C)		Local Context Analysis (L)		INQ301 (B+C+L	
0	0.5644	0.6482	(+14.8)	0.6449	(+14.3)	0.4779	(−15.3)	0.5919	(+ 4.9)
10	0.3294	0.3733	(+13.3)	0.3808	(+15.6)	0.3097	(−6.0)	0.3811	(+15.7)
20	0.2593	0.2921	(+12.6)	0.2817	(+ 8.6)	0.2670	(+3.0)	0.3071	(+18.4)
30	0.2071	0.2399	(+15.8)	0.2419	(+16.8)	0.2259	(+9.1)	0.2586	(+24.9)
40	0.1650	0.2055	(+24.5)	0.2229	(+35.1)	0.1987	(+20.4)	0.2359	(+43.0)
50	0.1220	0.1739	(+42.5)	0.1746	(+43.1)	0.1727	(+41.6)	0.1955	(+60.2)
60	0.0781	0.1193	(+52.8)	0.1253	(+60.4)	0.1322	(+69.3)	0.1682	(+115.4)
70	0.0523	0.0861	(+64.6)	0.0913	(+74.6)	0.0961	(+83.7)	0.1138	(+117.6)
80	0.0350	0.0618	(+76.6)	0.0720	(+105.7)	0.0484	(+38.3)	0.0856	(+144.6)
90	0.0181	0.0379	(+109.4)	0.0494	(+172.9)	0.0291	(+60.8)	0.0643	(+255.2)
100	0.0161	0.0241	(+49.7)	0.0337	(+109.3)	0.0081	(−49.7)	0.0378	(+134.8)
avg	0.1442	0.1796	(+24.5)	0.1852	(+28.4)	0.1604	(+11.2)	0.2001	(+38.8)

Table 2: Interpolated precision at 11 recall points for the parts of the the query. Table 1 compares other runs.

above. At 0% recall, the poor low-recall performance of local context analysis caused the combined query to be worse than basic query processing. These results suggest that the weighting used (1 × basic query processing, 2 × core query processing, 4 × local context analysis) was the opposite of what was required. Further tests have confirmed that LCA should not have been so heavily weighted, though the impact of "better" weights is not large.

In considering possible weightings of the three components, we also determined the best weighting on a per-query basis and ran that. We found a 27.7% improvement in average precision (from 0.2001 to 0.2555) when doing that. Such an improvement is unlikely to be found outside a retrospective analysis, but it does suggest that better weight balancing would be helpful in general.

7 Spanish Results and Discussion

Two sets of results, SIN300 and SIN301, were evaluated in the Spanish track. Both sets were based on automatic processing of TREC topics SP51-SP75 into queries and automatic query expansion. The official results are summarized below.

Query Type	Average Precision			
	5 docs	30 docs	100 docs	11-Pt Avg
SIN300	0.6720	0.5253	0.4044	0.4551
SIN301	0.6800 (+ 1.2)	0.5893 (+ 12.2)	0.4532 (+ 12.1)	0.5058 (+ 11.1)

SIN301 was significantly better above the 5 document cutoff. Recall that the primary difference between the query sets is that SIN300 was based only on topic descriptions, while SIN301 was based on both descriptions and narratives. The narratives contained good concepts which alone improved results by 14%. The combination of global and local analysis by Local Context Analysis is also effective and yields improvements of an additional 15%. This approach is significantly better than last year's use of InFinder.

8 Chinese Results and Discussion

Precision and recall tables for Chinese experiments follow. The queries were evaluated on the combined Xinhua and Peoples Daily Datasets from TREC-5 (1996). As the results indicate, query expansion makes a significant positive contribution to the quality of retrieval. Stopword removal had little impact.

Regrettably, the original INQUERY submission for Chinese TREC evaluation was incorrect because the query set was run on only the Xinhua text. Missing from the database on the official run was the Peoples Daily text which represents almost 4/5 of the total collection!

Interpolated Recall-Precision Averages				
	Unstopped	Stopped	Expanded 0.5 vs 1 (HIN300)	Expanded 1 vs 1 (HIN301)
at 0.00	0.6905	0.6911	0.7421	0.7490
at 0.10	0.5462	0.5682	0.6125	0.6069
at 0.20	0.4905	0.4924	0.5557	0.5618
at 0.30	0.4304	0.4314	0.4754	0.4849
at 0.40	0.3716	0.3721	0.4272	0.4425
at 0.50	0.3327	0.3308	0.3565	0.3626
at 0.60	0.2607	0.2657	0.2958	0.3109
at 0.70	0.1955	0.1990	0.2387	0.2598
at 0.80	0.1635	0.1660	0.1951	0.2143
at 0.90	0.1069	0.1104	0.1288	0.1422
at 1.00	0.0246	0.0248	0.0301	0.0439
Average	0.3139	0.3170	0.3523	0.3651
Change:		1.0%	12.2%	16.3%
Precision				
At 10 docs:	0.4526	0.4579	0.4737	0.4789
At 100 docs:	0.2558	0.2579	0.2800	0.2905
At 1000 docs:	0.0648	0.0648	0.0679	0.0687
R-Precision	0.3392	0.3380	0.3737	0.3803

9 Routing Results and Discussion

The following table summarizes the results of our routing run:

Set	AvgPrec	R-Prec	Prec@10	Prec@100
45 topics	0.3359	0.3549	0.5156	0.3269
39 topics	0.3633	0.3966	0.5872	0.3756

The "45 topics" are the 45 judged topics which had at least one relevant document. The "39 topics" are the 39 of those which had more than 5 relevant documents.

The following table shows the effect of different parts of the processing. These figures vary from the reported numbers above because they include all 45 topics that had relevant documents, and because the analysis was done with a slightly modified version of Inquery.

	Base	+terms	+#1	+#uw5	+#uw20	+#uw50	+DFO
Total number of documents over all queries							
Rel_ret:	3665	3860	3864	3867	3867	3872	3998
Interpolated Recall - Precision Averages:							
at 0.00	0.6476	0.7412	0.7331	0.7566	0.7577	0.7564	0.8001
at 0.10	0.5116	0.5687	0.5835	0.5901	0.5916	0.5961	0.6540
at 0.20	0.4134	0.4594	0.4650	0.4702	0.4750	0.4777	0.5271
at 0.30	0.3531	0.3937	0.4007	0.4050	0.4088	0.4114	0.4662
at 0.40	0.2996	0.3323	0.3409	0.3445	0.3482	0.3523	0.3974
at 0.50	0.2579	0.2907	0.2986	0.3023	0.3060	0.3081	0.3504
at 0.60	0.1845	0.2158	0.2174	0.2171	0.2184	0.2225	0.2529
at 0.70	0.1123	0.1442	0.1425	0.1400	0.1421	0.1437	0.1805
at 0.80	0.0809	0.0901	0.0908	0.0942	0.0950	0.0966	0.1285
at 0.90	0.0543	0.0592	0.0590	0.0644	0.0643	0.0621	0.0876
at 1.00	0.0380	0.0421	0.0421	0.0418	0.0418	0.0418	0.0494
Average precision (non-interpolated) over all rel docs							
	0.2502	0.2825	0.2861	0.2891	0.2912	0.2932	0.3333
% Change:		12.9	14.4	15.6	16.4	17.2	33.2
Precision:							
At 5 docs:	0.4622	0.4800	0.4800	0.4889	0.4844	0.4800	0.5111
At 10 docs:	0.4178	0.4511	0.4511	0.4467	0.4511	0.4467	0.4822
At 15 docs:	0.3867	0.4104	0.4148	0.4207	0.4193	0.4207	0.4578
At 20 docs:	0.3611	0.3889	0.3878	0.3933	0.3911	0.3944	0.4344
At 30 docs:	0.3244	0.3511	0.3548	0.3533	0.3630	0.3630	0.3993
At 100 docs:	0.2527	0.2751	0.2760	0.2778	0.2798	0.2804	0.2969
At 200 docs:	0.1944	0.2103	0.2113	0.2116	0.2123	0.2144	0.2246
At 500 docs:	0.1232	0.1314	0.1312	0.1315	0.1319	0.1322	0.1370
At 1000 docs:	0.0814	0.0858	0.0859	0.0859	0.0859	0.0860	0.0888
R-Precision (precision after R (= num_rel for a query) docs retrieved):							
Exact:	0.2816	0.3114	0.3143	0.3260	0.3272	0.3300	0.3677

The results show that expanding the query by 50 terms ("+terms) yielded a 13% gain. Expansion by pairs of terms (the next four columns) yielded a modest amount, but nothing of substance. The feature reweighing step (Dynamic Feedback Optimization) yielded another healthy improvement. For TREC-5, term expansion and feature reweighing were the key elements.

In TREC-4, we found roughly a 10% improvement from expanding with pairs of terms, so were surprised they they provided so little benefit this year. We have discovered that our term-pair selection code is errorful and are investigating the problem: the selected pairs not only provide minimal improvement on the test collection; they provided a modest *drop* in effectiveness on the training data!

10 Filtering Results and Discussion

Statistics about our results are summarized in the following table:

	Prec	Bal	Rec
Avg pool util	-12.2	1.9	86.0
Avg sampled util	13.9	33.5	159.9
Total retrieved	1276	2844	4623
Total relevant	5808	5808	5808
Total rel ret	807	1469	2209
Avg prec	0.4417	0.4793	0.4843
Avg recall	0.0667	0.1332	0.2466

These results are not particularly pleasing on the surface, but further analysis indicates that the results were relatively good compared to other systems (matching the best reported performance in 40% of the cases). Given that we applied such a simple technique, we are hopeful that further work will improve the utility numbers.

We are intrigued that our high-recall run achieved a higher average precision than our high-precision run but have not yet done the analysis to understand what happened.

11 Summary and Conclusions

Our results this year are very encouraging, not only because they continue to support our approaches, but because they have presented several puzzles for investigating. The conclusions we draw from this work are:

1. Our Local Context Analysis "automatic feedback" technique is consistently helpful across queries, across databases, and across languages.
2. Our techniques for locating the "core concepts" in a query are helpful, though there are several questions their use raises.
3. Combining multiple forms of queries continues to be a successful approach.
4. Longer queries—for example, those including "narrative" prose—once again show themselves to be more useful for retrieving relevant documents.
5. The query expansion and reweighting techniques applied for routing in the past continue to work with new queries and a new routing collection.
6. All of our approaches are successful in languages other than English, and can be applied with surprisingly little difficulty.

We are particularly intrigued by the odd results we found identifying core concepts in the Ad-hoc track, and by the unusual relationship between precision and recall we encountered in the Filtering track. We are pleased that TREC-5 reaffirmed our beliefs, but much more excited by the new areas of exploration raised.

Acknowledgments

Thank you to Bill Curtis for his help with with some of the test runs and Victor Lavrenko for help with results analysis.

This paper is based on research supported by the NSF Center for Intelligent Information Retrieval at the University of Massachusetts, Amherst. This material is also based on work supported in part by the National Science Foundation, Library of Congress and Department of Commerce under cooperative agreement number EEC-9209623. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsor.

References

[1] James Allan, Lisa Ballesteros, James P. Callan, W. Bruce Croft, and Zhihong Lu. Recent experiments with INQUERY. In *Fourth Text REtrieval Conference (TREC-4)*, 1995. Forthcoming.

[2] Chris Buckley and Gerard Salton. Optimization of relevance feedback weights. In Edward A. Fox, Peter Ingwersen, and Raya Fidel, editors, *Proceedings of the 18th annual international ACM SIGIR conference on research and development in information retrieval*, pages 351–357, Seattle, Washington, July 1995. ACM.

[3] Chris Buckley, Amit Singhal, and Mandar Mitra. Using query zoning and correlation within SMART : Trec 5. In D. Harman, editor, *Proceedings of the Fifth Text REtrieval Conference (TREC-5)*. National Institute of Standards and Technology, 1996.

[4] J. P. Callan, W. B. Croft, and S. M. Harding. The INQUERY retrieval system. In *Proceedings of the Third International Conference on Database and Expert Systems Applications*, pages 78–83, Valencia, Spain, 1992. Springer-Verlag.

[5] W. B. Croft, J. Callan, and J. Broglio. TREC-2 routing and ad-hoc retrieval evaluation using the INQUERY system. In D. Harman, editor, *Proceedings of the Second Text REtrieval Conference (TREC-2)*. National Institute of Standards and Technology Special Publication 500-215, 1994.

[6] W. Bruce Croft and Jinxi Xu. Corpus-specific stemming using word form co-occurence. In *Symposium on document analaysis and information retrieval (SDAIR '95)*, 1995.

[7] K.L.Kwok. A new method of weighting query terms for ad-hoc retrieval. In *Proceedings of the 19th annual international ACM SIGIR conference on research and development in information retrieval*, pages 187–195, Zurich, 1996. Association for Computing Machinery.

[8] Robert Krovetz. *Word Sense Disambiguation for Large Text Databases*. PhD thesis, University of Massachusetts, Amherst, 1995.

[9] J. Ponte and W. B. Croft. USeg: A retrargetable word segmentation procedure for information retrieval. In *Symposium on document analaysis and information retrieval (SDAIR '96)*, 1996.

[10] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford. Okapi at TREC-3. In D. K. Harman, editor, *The Third Text REtrieval Conference (TREC-3)*, Gaithersburg, MD, 1995. National Institute of Standards and Technology, Special Publication 500-225.

[11] Howard Turtle and W. Bruce Croft. Inference networks for document retrieval. In Jean-Luc Vidick, editor, *Proceedings of the 13th International Conference on Research and Development in Information Retrieval*, pages 1–24. ACM, September 1990.

[12] Howard R. Turtle and W. Bruce Croft. Efficient probabilistic inference for text retrieval. In *RIAO 3 Conference Proceedings*, pages 644–661, Barcelona, Spain, April 1991.

[13] J. Xu, J. Broglio, and W. B. Croft. The design and implementation of a part of speech tagger for english. Technical Report IR-52, Center for Intelligent Information Retrieval, Department of Computer Science, University of Massachusetts, 1994.

[14] J. Xu and W. B. Croft. Query expansion using local and global document analysis. In *Proceedings of the 19th annual international ACM SIGIR conference on research and development in information retrieval*, pages 4–11, Zurich, 1996. Association for Computing Machinery.