INQUERY and TREC-8

James Allan[†], Jamie Callan[*], Fang-Fang Feng[†], and Daniella Malin[†]

[†]Center for Intelligent Information Retrieval
Department of Computer Science
University of Massachusetts
Amherst, Massachusetts USA

[*]Language Technologies Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, Pennsylvania USA

This year the Center for Intelligent Information Retrieval (CIIR) at the University of Massachusetts participated in seven of the tracks: ad-hoc, filtering, spoken document retrieval, small web, large web, question and answer, and the query tracks. We spent significant time working on the filtering track, resulting in substantial performance improvement over TREC-7. For all of the other tracks, we used essentially the same system as used in previous years.

In the next section, we describe some of the basic processing that was applied across most of the tracks. We then describe the details for each of the tracks and in some cases present some modest analysis of the effectiveness of our results.

1 Tools and Techniques

Although UMass used a wide range of tools, from Unix shell scripts, to PC spreadsheets, four major tools and techniques were applied across almost all tracks: the Inquery search engine, the InRoute filtering engine, query processing, and a a query expansion technique known as LCA. This section provides a brief overview of each of those so that the discussion does not have to repeated for each track.

1.1 Inquery

All tracks other than the filtering track used Inquery[Callan et al., 1992] as the search engine, sometimes for training, and always for generating the final ranked lists for the test. We used Inquery V3.2, an in-house development version of the Inquery system made available by the CIIR (V3.1). The differences between the two are not consequential for this study.

The current belief function used by Inquery to calculate the belief in term t within document d is:

$$w_{t,d} = 0.4 + 0.6 \times \frac{\mathrm{tf}_{t,d}}{\mathrm{tf}_{t,d} + 0.5 + 1.5\frac{\mathrm{length}(d)}{\mathrm{avg\ len}}} \times \frac{\log \frac{N+0.5}{n_t}}{\log N + 1}$$

where n_t is the number of documents containing term t, N is the number of documents in the collection, "avg len" is the average length (in words) of documents in the collection, length(d) is the length (in words) of document d, and $\mathrm{tf}_{t,d}$ is the number of times term t occurs in document d.

[*]The work reported was carried out while Jamie Callan was at the University of Massachusetts.

1.2 InRoute

The InRoute filtering system is based on the same Bayesian inference network model as InQuery. It is designed to operate efficiently in high-volume filtering environments, where incoming documents must be processed rapidly, one at a time. It uses the same document indexing techniques, query language, and scoring algorithms as InQuery. Corpus statistics for the document stream are estimated using an archival corpus or are learned as documents stream by. InRoute also incrementally learns improved profiles [Allan, 1996] and improved thresholds [Callan, 1998], as relevance judgments become available for documents that have been disseminated.

InRoute was used only in the filtering track.

1.3 Query Processing

The processing of queries—i.e., the transformation from TREC topic to InQuery query—was handled very similarly to the way it was managed for TREC-7, though there were some small changes. It includes three steps:

1. Basic Query Processing removes stop words and phrases (e.g., "relevant documents will include"), and stop structures—i.e., that are sentences discussing criteria of non-relevance in the narratives. For removing the stop structures the processor simply segments each sentence first then removes the sentences that contain the stop structure (e.g., "Documents discussing ... are not relevant"), but keep those clause which stands on the negative part of the removed sentence (such as in "... not relevant, unless ...", where the "unless" clause is not be removed).

2. Query Formalizing identifies noun phrases (as in TREC-6 and 7), and proper names. The proper names were transformed to use the ordered proximity-one operator (e.g., `#passage25(#1(Golden Triangle)`, which requires that the two words occur immediately adjacent in the text in that order; the passage operator affects the weighting of the feature. For noun phrases we not only used the phrase operator but also duplicated the single terms used by the phrase (e.g., `#passage25(#phrase(tropical storms)), tropical, storms`). Note that for proper names, the single term duplication was *not* done.

   The query formalizing also identifies compound words, like *wildlife* and *airport*, that are formalized with the synonym operator (e.g., `#syn(#1(air port) airport)`). While the query is fomalized, if there is any word concerned with foreign countries, like *international*, *world*, or *Europe*, a token `#foreigncountry` will be added to the query. If there is a term concern with the United States, then a token `#usa` will be added. If both `#foreigncountry` and `#usa` are found in a query, all such tokens are removed.

   Finally a query is formed with the weighted sum operator (`#wsum`) with a weight for each term. For those terms occurring in the title and description fields the weight 1.0 is used, while 0.3 is used for those terms occurring in the narrative field. That is, we trust the title and the description more than the narrative.

3. Query Expansion adds 50 LCA concepts to each query. These 50 concepts are collected from top 30 passages (the passage database are built with TREC volumns 1 through 5). This is the same process that we used in TREC-7, but we did not use "filter-required" on the title words this year. LCA is described next.

1.4 Local Context Analysis (LCA)

In SIGIR '96, the CIIR presented a query expansion technique that worked more reliably than previous "pseudo relevance feedback" methods.[Xu and Croft, 1996] That technique, Local Context Analysis (LCA), locates expansion terms in top-ranked passages, uses phrases as well as terms for expansion features, and weights the features in a way intended to boost the expected value of features that regularly occur near the query terms.

LCA has several parameters that affect its results. The first is the choice of LCA database: the collection from which the top ranked passages are extracted. This database could be the test collection itself, but is often another (perhaps larger) collection that it is hoped will broaden the set of likely expansion terms. In the discussion below, if the LCA database is not the test collection itself, we identify what collection was used.

LCA's other two parameters are the number of top passages used for expansion, and the number of expansion features added to the query. The LCA features were put into a query construct that allows a weighted average of the features. Assuming n features, f_1 through f_n, they are combined as:

$$\begin{array}{lll} \texttt{\#wsum(1.0} & 1.0 & f_1 \\ & \vdots & \vdots \\ & 1-(i-1)*0.9/s & f_i \\ & \vdots & \vdots \\ & 1-(n-1)0.9/s & f_n \text{)} \end{array}$$

Here, s is scaling factor that is usually equal to n. The weighted average of expansion features is combined with the original query as follows:

$$\texttt{\#wsum(1.0} \text{ 1.0 original-query} \quad w_{\text{lca}} \text{ lca-wsum)}$$

where w_{lca} is the weight that the LCA features are given compared to the original query. Note that the final query is a weighted combination of the original query and the expansion features. As will be discussed below, in the SDR track the combination was unintentionally done differently, slightly shifting the balance between the original query and the expansion concepts.

2 Ad-hoc Track

Other than minor changes to the query processing, we did not investigate any ideas in the ad-hoc track this year. We submitted four runs, each of which used the complete set of query processing techniques outlined in Section 1.3. The runs and their average precision were:

INQ601	title only	0.2325
INQ602	description only	0.2492
INQ603	title plus description	0.2659
INQ604	title, description, and narrative	0.2809

INQ603 (comparatively, our best run) is below the average for 13 topics of the 50 topics. We have not analyzed those runs in any additional detail.

3 Filtering Track

Our goals for the Filtering track were to test the InRoute filtering system and a new threshold learning algorithm. We felt that performance in TREC-7 was quite poor for three reasons:

1. disseminating too many non-relevant documents early on;
2. not monitoring query performance during the run; and,
3. using a threshold learning method that could only raise thresholds.

We account for each of these in TREC-8 resulting in substantial performance improvement. We dealt with them as follows:

- The first problem was that too many documents were disseminated at an early stage. Wishing to mimic closely a real world situation where we cannot know an appropriate threshold initially, we tried to learn thresholds quickly from the first retrieved documents. This proved too difficult. We were not able to find a reasonably effective threshold without retrieving (and taking the hit for) a sizeable number of documents, most of which would turn out to be non-relevant. As a solution, we ran each query on a retrospective database and chose an initial threshold on the basis of these alternative document scores. This turned out to be highly effective. This technique was adopted from CLARIT's TREC-7 submission.[Zhai et al., 1998]
- Our second problem was that we had not implemented any sort of performance monitoring mechanism to watch queries during the course of the run and "shut off" queries for which performance was far below the target precision. This year, we simply stopped retrieving documents for queries retrieving more than N non-relevant documents and *MIN* or fewer relevant documents. *MIN* was somewhat arbitrarily set to 2. Any less than that would not have been reasonable given the threshold learning method we chose (described below). Because of the effectiveness of our initial thresholds, we were able to set N to a fairly small number; N was also the parameter we varied between submissions. We set $N = 9$ and $N = 15$ for our official submission for LF1 and LF2 respectively. These values were chosen on the basis of test runs using AP data and TREC-7 queries. A more elegant solution would have been to chose N and *MIN* based on some function of the target precision as embodied in utility metric being used for evaluation. Unfortunately, we did not have time before the submission deadline to discover what this function should be.
- In our previous threshold learning method, thresholds could only go up and never down during the course of a run. This meant that the threshold learning was insensitive to new trends in scores. It also increased the importance of the initial phases of learning. We needed to retrieve many documents at the beginning of the run in order to get a good start. This year we opted for a threshold learning method that could adjust thresholds both up and down. In doing so, we gave up both modifying the queries on the basis of retrieved documents and dynamically adjusting term idfs.

Our method this year was to set thresholds to the lowest document score, in a list of scores sorted highest to lowest, at which precision was equal to or greater than the target precision. We collected document scores and after each new arrival, sorted the scores and computed precision at each point. We then chose the first precision point to equal or exceed the target precision. Since this process would be futile if none of the documents retrieved were relevant, we used our initial thresholds until at least *MIN* (two) documents had been retrieved. Document scores of initial documents would have vary little in common with document scores of subsequent documents if the queries and/or idfs had changed during the course of the run. In the interest of threshold sensitivity, we chose not to modify our queries or adjust idfs. Our indexing process did not make use of the controlled-language field in the FT database.

3.1 Queries for filtering

We used the title, description and narrative parts of the topic processed the queries using all three of the query processing steps described for our ad-hoc runs. However, `#usa` and `#foreigncountry` tokens were never used, regardless of whether the query suggested they should be.

	FT92	FT93	FT94	FT92-94
INQ610 LF1	35	39	42	32
INQ611 LF2	36	30	33	25
INQ612 LF2	38	36	38	32
INQ613 LF1	34	31	34	28

Table 1: Number of queries at or above the median for filtering

	FT92	FT93	FT94	FT92-94
INQ610 LF1	258	29	30	350
INQ611 LF2	88	-14	24	74
INQ612 LF2	125	29	9	139
INQ613 LF1	127	-186	-129	-155

Table 2: Sum of our scores minus sum of median scores for filtering

3.2 Filtering Results

The CIIR submitted four runs, two for evaluation using the LF1 utility metric and two for evaluation using the LF2 utility metric. As discussed above, we varied the number N, of non-relevant retrieved documents we allowed the system to retrieve before shutting off the query. We allowed that number to be greater in our primary run for LF2 than LF1 since LF2 is the more lenient of the two metrics. In the end, the smaller value for N outperformed the larger value for both metrics. The runs are as follows:

INQ610	$N = 9$	LF1 primary run
INQ611	$N = 15$	LF2 primary run
INQ612	$N = 9$	LF2 secondary run
INQ613	$N = 15$	LF1 secondary run

Tables 1 and 2 show the performance of this run compared to the median.

4 Spoken Document Retrieval Track

We applied the full query processing technique and ran the resulting queries. Note that unlike in past years, we did the LCA query expansion on an external collection rather than on the SDR collection itself. We participated only in the “quasi-SDR” version of the track.

5 Small Web Track

For the two gigabyte Web track, our results were reasonable. Of the 50 topics, 42 were better than the median and only 7 worse. Three of our scores were the maximum score for that query. The run, INQ620 used the same query set as Ad-hoc INQ603, i.e, the title and description portions of the topic, with the same query processing. We did not take advantage of link structure in any way. The average precision of the run was 0.3327 with precision at 10 documents of 0.5040 and at 20 of 0.4130.

6 Large Web Track

We submitted one run to the Large Web Track. That run, INQ650, achieved a modified average precision of 0.3927. Its precision at 10 documents retrieved was 0.4960 and at twenty documents, it was 0.5000.

7 Question and Answer Track

Our work in the Q&A track is in loose combination with Sheffield University. We used straightforward IR techniques to isolate passages of text that were highly likely to contain the answer to the questions. We submitted those passages as our "answers" and also handed them to Sheffield to do some further processing. At this point, we do not have comparitive information to show the impact of their work.

We used Inquery's best passage operator to try to locate the answer for a particular question in the 5 top retrieved documents. Inquery's best passage operator is flexible with respect to the size of the best passage the user is looking for. This setting is in word count rather than bytes. We set the best passage size to 10 words for the 50-byte runs and 50 words for the 250-byte passage runs. We then post-processed the answer to pull out something of the correct size. If a word was split by the 50- or 250-byte boundary, we did not include the partial word in our "answer."

CIIR submitted four runs, two for the 50-byte limit and two for the 250-byte limit. The difference between these runs was whether or not we used LCA expansion on the questions (i.e., whether step 3 of the query processing in Section 1.3 was applied). We found that LCA expansion improved performance. Overall we did quite well:

Runid	$>$ med	$\geq$ med	$<$ med	Our average	Average of median	% Difference
INQ634 - 50 bytes, with expansion	43	185	20	0.18737	0.11970	+57.1%
INQ635 - 250 bytes, with expansion	58	172	33	0.37828	0.28081	+34.7%
INQ638 - 50 bytes, no expansion	29	176	29	0.12556	0.12485*	+0.5%
INQ639 - 250 bytes, no expansion	45	165	40	0.33283	0.28081	+18.5%

(*)The average of the median is different between these two 50 byte runs apparently because NIST carried the second evaluation out to two rather than just one significant digit.)

8 Query track

The CIIR contributed a large set of queries to the Query track. They were generated as part of a class assignment for a database class in the Fall of 1998.

9 Acknowledgements

This material is based on work supported in part by the National Science Foundation, Library of Congress and Department of Commerce under cooperative agreement number EEC-9209623. It was also supported in part by Defense Advanced Research Projects Agency/ITO under ARPA order number D468, issued by ESC/AXS contract number F19628-95-C-0235. Additional support came in part from Air Force Research

Laboratory Contract number F30602-98-C-0110. Any opinions, findings and conclusions or recommendations expressed in this material are the authors' and do not necessarily reflect those of the sponsors.

References

[Allan, 1996] Allan, J. (1996). Incremental relevance feedback. In *Proceedings of the 19th annual international ACM SIGIR conference on research and development in information retrieval*, pages 270–278, Zurich. Association for Computing Machinery.

[Callan, 1998] Callan, J. (1998). Learning while filtering documents. In *Proceedings of the 21st annual international ACM SIGIR conference on research and development in information retrieval*, Melbourne. Association for Computing Machinery.

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

[Xu and Croft, 1996] Xu, J. and Croft, W. B. (1996). 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. Association for Computing Machinery.

[Zhai et al., 1998] Zhai, C., Jansen, P., Stoica, E., Grot, N., and Evans, D. (1998). Notes on threshold calibration in adaptive filtering: The CLARIT system TREC-7 filtering report. In *The Seventh Text REtrieval Conference TREC-7*.