TREC-8 Experiments at Maryland: CLIR, QA and Routing

Douglas W. Oard and Jianqiang Wang
College of Library and Information Services
University of Maryland, College Park, MD 20742 USA
{oard,wangjq}@glue.umd.edu

Dekang Lin
Department of Computer Science
University of Manitoba, Winnipeg, Manitoba, R3T2N2CANADA
lindek@cs.umanitoba.ca

Ian Soboroff
Department of Computer Science and Electrical Engineering
University of Maryland, Baltimore County, Baltimore, MD 21250 USA
ian@cs.umbc.edu

Abstract

The University of Maryland team participated in four aspects of TREC-8: the ad hoc retrieval task, the main task in the cross-language retrieval (CLIR) track, the question answering track, and the routing task in the filtering track. The CLIR method was based on Pirkola's method for Dictionary-based Query Translation, using freely available dictionaries. Broad-coverage parsing and rule-based matching was used for question answering. Routing was performed using Latent Semantic Indexing in profile space. This working notes paper presents preliminary results.

1 Introduction

The Eighth Text REtrieval Conference (TREC-8) offered many more attractive evaluation opportunities that our team could have pursued, so we chose to participate in four aspects of the work that are aligned particularly closely with our ongoing work. In Cross-Language Information Retrieval track (CLIR), we focused on rapid retargetability, seeking to learn how well we could do with freely available resources that have more limited vocabulary coverage than those we have used in the past. We also tried out the Inquery synonym operator as a device for selecting the correct translation, an approach introduced by Pirkola [7] but not previously tested at TREC. In the new Question Answering track, we explored the potential for combining broad-coverage parsing with rule-based matching. Our effort for the Routing task in the Filtering track explored the use of Latent Semantic Indexing on a space formed from profiles that aggregate several documents in an effort to understand whether common aspects of the topic space could be automatically identified and exploited. Our participation in the Ad Hoc task was limited to a single run with an off-the-shelf retrieval system—as in past years, we used the Ad Hoc task as a learning opportunity for some of the new members of our team while producing results that might be of some value in enriching the assessment pool.

Our team for the first time included significant participation by visitors from other institutions. Dekang Lin from the University of Manitoba worked on Question Answering while on sabbatical at Maryland. Ian Soboroff from the University of Maryland, Baltimore County worked on the Routing task. Our experience suggests that collaborations of this sort can serve the community well, combining fresh ideas with experience that gives a leg up on climbing the learning curve.

Pair	Source	English Terms	Foreign Terms	Avg Translations
E-G	http://www.quickdic.de	99,357	131,273	1.7
E-F	http://www.freedict.com	20,100	35,008	1.3
E-I	http://www.freedict.com	13,400	17,313	1.3

Table 1: Sources and summary statistics for bilingual dictionaries.

2 Cross-Language Information Retrieval

We participated in the main task of the CLIR track, using an English query to create one single merged ranked list of English, French, German and Italian news stories for each of the 28 topics. We sought to answer three questions:

- What is the best that can be done using freely available resources?
- How well does Pirkola's method for selecting among candidate translations work on the TREC CLIR collection?
- Would building a single index be more effective than building separate indices for each language?

2.1 Freely Available Resources

A purist approach to the first question would have required that we use a freely available retrieval system such as PRISE, SMART or MG. The second question led us to instead choose Inquery, which is inexpensively (but not quite freely) available for research use. We downloaded three bilingual "dictionaries," all of which were actually simply lists of English terms that were paired with some equivalent terms in another language. Here we take "terms" to include both single words and multiword expressions—multiword expressions were common in some of the dictionaries. Table 1 shows the source and summary statistics for each dictionary.

Each of the dictionaries was downloaded in a native machine-readable format that was designed for the originally intended use (typically, interactive access using an associated program). No documentation regarding storage formats was provided with any of the dictionaries, but conversion to a common format (we use one term pair per line, separating the two terms using a tab character) turned out to be quite straightforward in every case. We preserved the order of the original dictionary where possible, and an examination of the results indicates that the known translations for each term are stored in lexicographic order. We typically reorder the translations by their (unconditioned) frequency in the Brown Corpus (for terms that are present in that corpus), but that was not done in this case.

2.2 Pirkola's Technique

Once we had a dictionary in a suitable format, we used it with Dictionary-based Query Translation (DQT) routines that we have previously developed to translate the query from English into the language of one of the four language-specific CLIR subcollections (no translation was needed for the English subcollection). In DQT, each query term for which at least one translation is known is replaced with one or more of the known translations. Since query terms may have more than one translation, some selection heuristic is needed. In the past we have tried retaining Every Translation (DQT-ET) or just the First Translation (DQT-FT), finding that sometimes one approach yields better average precision and sometimes the other does. We thus elected to try both and to select the best of the two as our baseline for evaluating Pirkola's technique.

Pirkola used structured queries to attack the problem of translation ambiguity. With Inquery's synonym operator, he automatically grouped all Finnish translations for an English query term into the same "facet." This approach yielded substantial improvements in average precision when compared with an approach similar to our DQT-ET technique [7]. Specific terms, which are quite useful for searching, typically have

Run ID	Official	Queries	Translation	Index	Avg Prec
umd99b1	Yes	Long	Pirkola	Monolingual	0.1616
umd99b2	Yes	Long	DQT-FT	Monolingual	0.1555
umd99b3	Yes	Long	DQT-ET	Monolingual	0.1344
umd99c1	Yes	Title	Pirkola	Monolingual	0.1003
umd99c2	Yes	Title	Pirkola	Multilingual	0.1030
umd99c3	No	Title	DQT-ET	Monolingual	0.1142
umd99c4	No	Title	DQT-ET	Multilingual	0.0938
umd99c5	No	Title	DQT-FT	Monolingual	0.0966
umd99c6	No	Title	DQT-FT	Multilingual	0.0982

Table 2: Official and unofficial CLIR runs.

relatively few translations. But with DQT-ET, the more translations a query term has, the more weight it will get because every possible translation will appear in the query. With Pirkola's structured queries, translations of the same term are treated as instances of the same term. In this way, important query terms get relatively more weights. In our experiment, we implemented Pirkola's technique by grouping all translations for each query terms using the Inquery synonym operator #syn(). No #syn() operator was used for terms with a single translation. All of the groups and single terms were then combined using Inquery's sum operator #sum().

2.3 Multilingual Indexing

As in TREC-7, we usually built a separate index for the documents in each language (English, French, German, and Italian), produced separate ranked lists for each language for each topic using queries translated into only that language, and then applied a uniform merging strategy strategy in which we took n documents from the top of the English list for every 1 document that we took from each other list [6]. In preliminary experiments with TREC-7 data, we found $n = 2$ to be optimal for DQT with these dictionaries. That contrasts markedly with our conclusion at TREC-7 that $n = 10$ was best when queries were translated using a commercial machine translation system. We have not yet investigated this effect in detail, but in the results reported below we use a uniform 2:1:1:1 merge in which each block of 5 documents in the merged list contains 2 English documents, 1 French document, 1 German document, and 1 Italian document.

Good results have also been reported with a unified multilingual index [3], so we also tried that approach. In that case, all documents were indexed together regardless of language, and the translated queries in each language (including the untranslated English queries) were combined on a topic-by-topic basis. The approach results in a single ranked list, so no merging strategy is required. In order to maximize the comparability of our results across the two indexing techniques, we did not use stopword list in either case.

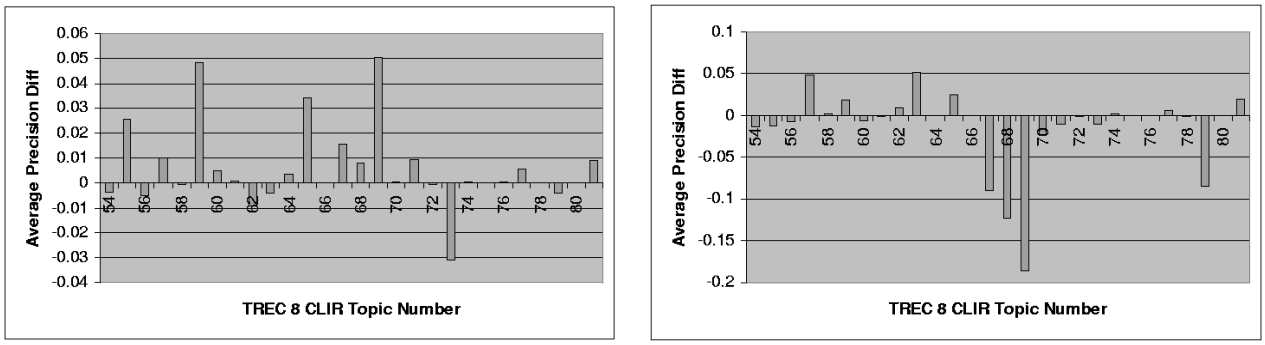
2.4 Results

We submitted five official CLIR runs and scored an additional four unofficial runs locally, as shown in Table 2. Only the "umd99b1" and "umd99c1" runs contributed to the relevance assessment pools for each topic. All runs were in the automatic category. Title queries were formed automatically using the words in the title field from each topic. Long queries were formed using all words in the topic except SGML markup and field titles.

Our results regarding the utility of freely available bilingual dictionaries are necessarily tentative because the only evaluation results that we have examined to date are for the multilingual case in which the effects of poor performance in one or more language pairs could easily be masked by monolingual English results and by the results in language pairs for which richer resources were used. Nonetheless, we did observe large

variations in average precision achieved on individual queries when we changed translation techniques. Some cross-language effects are thus clearly evident, so at least some of our resources must have been useful.

We obtained inconclusive results for the effectiveness of Pirkola's method. It produced a 4% relative improvement in average precision over DQT-FT, the best of the two DQT techniques that we tried, when long queries were used. That difference barely missed statistical significance at the 0.05 level with a two-tailed paired t-test ($t = 1.92, p = 0.065$). Figure 1(a) compares the two techniques on a per-query basis, showing that topics for which Pirkola's technique is better are considerably more common. Pirkola's technique is quite slow, however, requiring about 8 minutes per long French query on a SPARC 20 (compared with about 1 minute per long French query for either DQT-FT or DQT-ET). We note with some concern that this slowdown occurred with a dictionary in which multiple translations were relatively rare (averaging only 1.3 translations per term). As Figure 1(b) illustrates, with short queries, the best DQT technique (in this case, DQT-ET) often outperforms Pirkola's technique under the same conditions (monolingual indices, uniform 2:1:1:1 merge). The 13% relative loss in average precision incurred by Pirkola's technique in this case is not statistically significant, however ($t = -1.47, p = 0.15$).



(a) (b)

Figure 1: Comparative results by query, obtained by merging ranked lists produced using monolingual indices. (a) Long queries: Pirkola's method better above zero, DQT-FT better below. (b) Title queries: Pirkola's method better above zero, DQT-ET better below.

We also were not able to find a statistically significant difference between the use of a single multilingual index and our uniform 2:1:1:1 merging strategy for results obtained using separately constructed monolingual indices. We use the multilingual index only with title queries in our experiments. Neither the 3% relative improvement that resulted from multilingual indexing with Pirkola's technique nor the 2% relative improvement that resulted from monolingual indexing with DQT-FT showed any sign of significance ($t = -0.24, p = 0.81$ and $t = -0.10, p = 0.92$ respectively). The situation was reversed when DQT-ET was used, with a 22% relative advantage for merged results obtained using multilingual indices. That difference is not statistically significant, however ($t = 1.09, p = 0.28$). Figure 2 compares the two indexing strategies on a per-query basis for DQT-ET.

The common factor in the two relatively large differences observed above is the surprisingly good performance of run "umd99c3." Examination of Figures 1(b) and 2 reveals that this results from good performance of Pirkola's technique on title queries with any indexing strategy for topic 68, but only with merged monolingual indices for topics 67, 69 and 79. We will need to look more closely at the results to see what it is about the title queries for these four topics that produces the observed effects.

3 Question Answering

Many natural language systems are organized as a stream of processing modules. A parser is usually one of the upstream modules. The resulting parse trees are typically used to guide the processing in downstream modules. For example, a semantic interpreter may rely on the parse trees to identify the atomic components

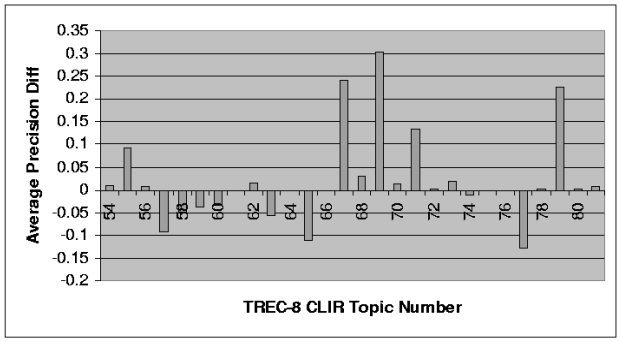


Figure 2: Comparative results by query for DQT-ET on title queries, merging monolingual ranked lists better above zero, single multilingual index better below.

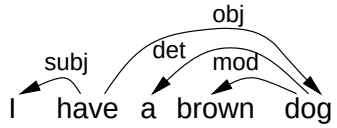
that are semantically interpretable and then combine them according to the parse tree structure to obtain the interpretation for larger chunk of text. We call such processing syntax-guided. A problem with syntax-guided processing is the heavy reliance of the downstream modules on the parse trees. Without the parse trees, a syntax-guided module is usually unable to produce any output.

SyncMatcher adopts a syntax-constrained approach where parse trees are used as a source of constraints for downstream modules. Without the constraints, the downstream modules are still functional. The difference is that they will be faced with more ambiguous inputs, which increases the likelihood of error in the output.

The parser used in SyncMatcher is MINIPAR, a principle-based broad-coverage parser [5]. Although MINIPAR uses a constituency grammar internally, its outputs are dependency structures. For each word in the sentence, a dependency structure specifies the governor of the word. For example, (1a) is a dependency structure of a sentence. The root of the dependency tree is "have" and there are 4 dependency relationships in the tree as shown in (1b).

(1)

a.



b.
```
(have subj I)
(have obj dog)
(dog mod brown)
(dog det a)
```

Given a query and a stream of documents, SyncMatcher matches sentences in the documents against the query using the dependency trees as constraints. Each match is assigned a score, which is used to rank the answers extracted from the documents. The outputs for each query are top-5 distinct answers.

To find the best match between a query and a sentence in the documents, SyncMatcher first establish the set of potential correspondence between the words in the query and the words in the documents according to the following rules:

- a word may match another word with identical root form.
- two words match if the result of stemming them with the Porter stemmer is the same.
- A wh-word matches proper nouns that have the same semantic tag as the wh-word. For example, "who" matches named entity that is classified as PERSON.

After collecting the set of potential matching pairs of words, SyncMatcher tries to find a subtrees of the dependency trees of the query and an input sentence that satisfies the following constraints:

(2)

a. If a node B is on the (undirected) path between two nodes A and C in the dependency tree of the query and A', B' and C' are nodes in the dependency trees of an input sentence that corresponds to A, B and C respectively, then B' must be on the (undirected) path between A' and C' in the dependency tree.

b. If A' and C' are nodes in the dependency tree of an input sentence and A' and C' corresponds to A and C in the query respectively, there must not exist another node on the path between A' and C' that may also correspond to A or C.

3.1 Semantic Tagging of Wh-words

SyncMatcher answers queries by extracting named entities from the documents. Therefore, we must first determine the type of named entity that the answer belongs to. If the wh-word in the query is "who", "when", "where", "how many" or "how much", the answer is usually a PERSON, a TIME/DATE, a LOCATION, a NUMBER or an AMOUNT, respectively. When the wh-word in the query is "which", "what" or "how", the semantic category of the wh-word is determined by their governor in the dependency tree. For each type of named entity, we constructed a list of common nouns that typically refer to them. For example, the list of common nouns for LOCATION include

> country, nation, city, region, republic, island, province, state, town, area, community, territory, capital, world, South, neighborhood, village, land, colony, camp, ...

A wh-word in a query is tagged as type X if its governor belongs to the list of common nouns of type X. For example, in the query "Which country is Australia's largest export market?", the governor of "which" is "country". So, "which" is tagged as LOCATION. In another query "Which former Ku Klux Klan member won an elected office in the U.S.?", the governor of "which" is "member". Since "member" belongs to a list of words that are very similar to "person", "man", etc., the word "which" is tagged as PERSON.

The dependency trees generated by MINIPAR also encodes the following types of coreference relationships.

- traces and zero pronouns and their antecedents
- personal pronouns and their antecedents
- proper names and their antecedents

The first type coreference relationships are identified during parsing. The other two types are identified by the coreference recognizer borrowed from a University of Manitoba's MUC system.

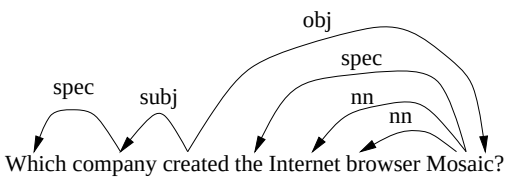
3.2 A Walkthrough Example

Consider the following query:

> Q.108 Which company created the Internet browser Mosaic?

The dependency tree for the query is as follows:

(3)



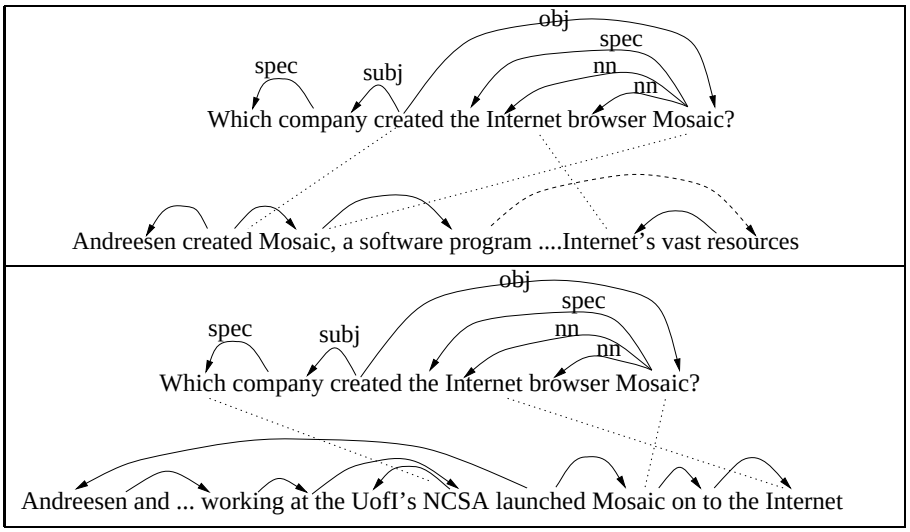
Consider the following fragments from one of the documents:

> .... Then he met Marc Andreesen. A 23-year-old cyber-star computer science graduate, Andreesen created Mosaic, a software program that enables even computer novices to explore the Internet's vast resources. Since Andreesen and a group of fellow students working at the University of Illinois' National Center for Supercomputing Applications launched Mosaic on to the Internet last year, it has been used by an estimated 2m people.

The word "Andreesen" is not found in the lexicon in SyncMatcher. However, there is the coreference relationship between "Andreesen" and "Marc Andreesen" earlier in the document. Since "Marc" is a known first name in the lexicon, "Marc Andreesen" is recognized as a person. Therefore, "Andreesen" is tagged as a PERSON.

Since the governor of "which" in the query is "company", "which" is tagged as an ORGANIZATION. It can only correspond to words in documents that are also tagged as organizations, such as "University of Illinois" and "National Center for Supercomputing Applications".

SyncMatcher identified the following two matches from the above paragraph.



Both matches involve three words in the query. The second match involves the wh-word in the query and is consequently scored higher. SyncMatcher then returns the matching element for the wh-word, "National Center for Supercomputing Applications" as the answer.

The phrase "University of Illinois" also matches "which" in Q.108. However, because "NCSA" is on the path between it and other matching words, such as "Mosaic", the constraint (2b) rules it out.

3.3 Results

We used the documents collected by AT&T Labs using a search engine, which contains 200 documents per query. The total size of the document collection is about 200MB (32M words). The document are ordered according to the relevance score obtained from the search engine. However, this information is currently ignored. SyncMatcher parsed all the sentences in the documents except those in the headers or footers. The total processing time is about 40 hours on a 233MHz Pentium II with 160MB memory and 6GB disk, running Linux. This is roughly equivalent to 222 words per second.

Out of the 198 questions in the Q&A Track, SyncMatcher returned the correct answer as one of its top 5 answers in 80 cases. The distribution of the answers is shown in the following table.

1st	2nd	3rd	4th	5th	not found
47	14	7	7	5	118

4 Routing

We have been exploring a filtering technique which combines content and collaborative aspects [11], and TREC-8 is its first exposure with a large collection. We expected this technique to give some advantage to related families of topics, while not harming performance on other topics.

Since our work has focused on the basic technique and not on adaptation, we only submitted results for routing. While adaptation and profile construction are probably not orthogonal, we hope that this can help show if our technique works aside from any benefits gained from adaptive filtering.

4.1 Collaborative LSI

We first construct our routing queries using a sophisticated relevance feedback approach. All queries are then collected together, and a latent semantic index of the query collection is computed. Test documents are routed in the reduced-dimension LSI space, with hopes that this space highlights common interests among the queries, and diminishes noise.

Latent semantic indexing [1] has been used before in the TREC Routing task [2]. The key difference in our approach is that we compute the latent semantic index from a collection of queries, rather than a collection of simple documents. Specifically, we collect our routing queries for topics 351-400 into a single term-query matrix, and compute an SVD of this matrix. This should give two advantages over a straightforward application of LSI. First, the LSI space is oriented towards features of the queries, rather than the documents, making it better suited to a routing environment with few saved documents and persistent queries. Second, the LSI space highlights commonalities among queries, so that if queries are similar they can benefit from each other.

In Dumais' approach, the LSI transformation highlights common features among documents, giving dimensions where groups of documents share co-occurrence patterns of certain weighted terms. This is simply too general, and not related to our problem, which is not to choose among documents but to choose among queries.

Hull [4] described a "local LSI" technique, which rather than computing the LSI from the entire collection, computed it from the top n documents in an initial retrieval on the query. This is closer to a query-centric LSI than Dumais, but does not allow for collaboration among queries.

It's not clear that any collaboration takes place in TREC filtering, since the topics are not necessarily designed to overlap, either in information interest or in actual relevant document sets. However, several topics this year are closely related, from a reading of the topic descriptions. One group might be "clothing sweatshops" (361) and "human smuggling" (362). Another is "hydrogen energy" (375), "hydrogen fuel automobiles" (382), and "hybrid fuel cars" (385).

To build our profiles, we use a technique similar to that used by the AT&T group in TREC-6 [8] and TREC-7 [10]. First, a training collection is constructed from the FBIS, Los Angeles Times, and Financial Times documents from 1992. We gather collection statistics here for all future IDF weights. The training documents are weighted with log-tfidf, and normalized using the pivoted unique-term document normalization [9].

We then build a routing query using query zoning. An initial query is made from the short topic description, and the top 1000 documents are retrieved from the training collection. The results from this retrieval are used to build a Rocchio feedback query, using:

- The initial short-description query (weighted $\alpha = 3$)
- All documents known to be relevant to the query in the training collection (weighted $\beta = 2$)
- Retrieved documents 501-1000, assumed to be nonrelevant (weighted $\gamma = -2$)

4.2 Implementation Details

Our system for routing is based on SMART, with routines added by us for pivoted document length normalization weights, construction of the LSI vector space, and the similarity computations needed to build a

ranked list. The LSI code is based on software written at the University of Maryland,[1] and on SVDPACKC from the NETLIB archive.[2] Our experiments were run on a Intel Pentium II-based system running Linux 2.2 with 512MB of RAM and 36 GB of local SCSI-II disk storage.

Two runs were submitted. The first, "umrqz," used only the routing queries as described above. The second, "umrlsi," computed an LSI from the collection of these routing queries, and routed the test documents in the resulting LSI space. For LSI to give any benefit, the dimensionality must be reduced below the maximum (in this case, 50 dimensions). We are not aware of any proven principled method for choosing this dimensionality besides trying several levels and seeing what gives the best performance. We thus ran our LSI queries against the training collection at several dimensions, and found that no dimensionality choice seemed to show any benefit for LSI. For the official submission, we arbitrarily chose a 45 dimensions.

4.3 Results

Overall, both runs performed quite well, with umrqz above the median for 27 queries, and umrlsi for 23. For five queries, we produced the best performance, and for four of those, the LSI gave the maximum score. Most of this is due to the routing query construction, which uses a combination of approaches shown to work well in previous TRECs. We also looked at using the top ten documents from the query zone as unsupervised positive examples, and at using the known negative judgments as supervised negatives, but these did not give as good performance on retrieving the training set.

For the majority of queries, there was only a very small difference in performance if any between the two runs. This was expected, because since the topics are mostly different, with little opportunity for overlap, the LSI should have been unable to help most queries. However, for the example candidate topic "clusters" described above, the difference in average precision from using LSI was negligible. There were also several topics related to drugs and pharmaceutics, although beyond this their foci were rather different; none of these topics improved with LSI.

For 18 queries where the difference in average precision between the non-LSI and LSI routing was more than 0.009, in 11 cases the difference was quite small relative to the whole span of scores. In the other seven, the difference was more marked, and in all but one (381) against LSI. For one query (360), LSI gave the minimum performance and the nonrotated query gave the maximum.

Furthermore, in the twenty topics where average precision in the umrlsi run was high (> 0.5), precision without LSI was either the same or slightly higher.

In eight topics, the LSI average precision was less than 60% of that achieved without LSI. These topics have a fair range of relevant document set sizes and in only one of these topics was performance across all systems poor. One topic in this group was 375, "hydrogen energy", and three were drug-related (drug legalization, food/drug laws, mental illness drugs). It may be that the drug-related topics contained a lot of shared terms, but this caused LSI to bring out a lot of false friends.

4.4 Discussion

A more in-depth analysis is needed to see where LSI is failing. There are at least three possible points of failure in this approach:

- The topics have no content overlap that would indicate collaborative potential. (bad topics)
- The topics may collaborate, but the queries don't weight the right terms correctly. (bad queries)
- Collaborating queries share highly-weighted content terms, but these aren't prominent enough for the LSI to highlight them. (bad collaboration)

As to the first possibility, we could look at overlap in terms, training documents, and test documents among the topics. This should give us a better view of where to expect LSI to make gains, but on the other hand this is what the LSI is supposed to do for us. It might be instructive to look at the LSI dimensions and the terms which characterize them, to see what exactly what patterns the LSI is finding.

[1] The LSI code is available at http://www.glue.umd.edu/~oard

[2] SVDPACKC is available from http://www.netlib.org

In the second case, the queries may be ill-suited to finding collaboration with the LSI. It might be that the quantity or variety of negative examples throws it off, in which case tuning of the query-zoning approach or the Rocchio weights might be in order. The choice of parameters was tuned using the non-LSI queries against the training set, so performance was not necessarily optimized for LSI.

Lastly, it could be that there are topics which could collaborate, and in fact there is term co-occurrence across their queries which we'd expect the LSI to find, but these patterns aren't prominent relative to the rest of the collection. This might happen because there aren't enough terms co-occurring, or the pattern doesn't span enough queries. In our three example groups, only drug-related topics represent a large segment of the topic collection, and this grouping is vague.

5 Ad Hoc Task

For TREC-7 and TDT-2 we had been using PRISE, but our interest in trying out Pirkola's technique for CLIR led to our choice of Inquery for CLIR TREC-8. The Ad Hoc task provides a useful opportunity for us to get new people familiar with the tools that we will be using in the CLIR track—this year we submitted a single official Ad Hoc run using Inquery 3.1p1 with the default settings. Queries were automatically formed from the title and description fields, and we automatically performed limited stop structure removal based on a list of typical stop structure observed in earlier TREC queries (e.g., "A relevant document will contain").

6 Conclusion

Our investment in TREC this year was rewarded with a rich set of insights. In Cross-Language Information Retrieval, we learned that we can construct passable systems using freely available resources but that a more efficient implementation of Pirkola's method may be needed before interactive applications will be practical. In this first year of the Question Answering track, we learned that the techniques we have been working with have good potential and that the evaluation methodology is quite tractable. In the Routing task, we achieved competitive results using a new approach, and recognized some promising directions for future work. The great frustration of TREC is that there are so many important and well framed questions being explored, but that practical considerations make it necessary for each team to focus on only a few. We have explored the potential for building a larger team through both on-campus and off-campus collaborations this year, and have been quite pleased with the result. Perhaps the strongest legacy of this effort, then, will be the closer personal and professional ties that we have forged.

Acknowledgments

The CLIR work has been supported in part by DARPA contract N6600197C8540 and a Shared University Research equipment grant from IBM. The QA work has been supported in part by Communication Security Establishment under contract W2213-9-0001/001/SV awarded to Nalante, Inc. and by Natural Sciences and Engineering Research Council of Canada grant OGP121338.

References

[1] Michael W. Berry, Susan T. Dumais, and Gavin W. O'Brien. Using linear algebra for intelligent information retrieval. *SIAM Review*, 37(4):573–595, December 1995.

[2] Susan T. Dumais. Using LSI for information filtering: TREC-3 experiments. NIST Special Publication 500-225, pages 219–230, Gaithersburg, MD, November 1995. National Institute of Standards and Technology. Also titled "Latent Semantic Indexing (LSI): TREC-3 Report".

[3] Fredric C. Gey, Hailing Jiang, Aitao Chen, and Ray R. Larson. Manual queries and machine translation in cross-language retrieval and interactive retrieval with Cheshire II at TREC-7. In *The Seventh Text REtrieval Conference*, pages 527–540, November 1998. http://trec.nist.gov.

[4] David Hull. Improving text retrieval for the routing problem using latent semantic indexing. In *Proceedings of the Seventeenth Annual International ACM SIGIR Conference (SIGIR '94)*, pages 282–291, Dublin, Ireland, July 1994.

[5] Dekang Lin and Randy Goebel. Context-free grammar parsing by message passing. In *Proceedings of PACLING 93*, 1993.

[6] Douglas W. Oard. TREC-7 experiments at the University of Maryland. In *The Seventh Text REtrieval Conference (TREC-7)*, pages 541–545, November 1998. http://trec.nist.gov.

[7] Ari Pirkola. The effects of query structure and dictionary setups in dictionary-based cross-language information retrieval. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 55–63, August 1998.

[8] Amit Singhal. AT&T at TREC-6. In E. M. Voorhees and D. K. Harman, editors, *The Sixth Text REtrieval Conference*, NIST Special Publication 500-240, pages 215–226, Gaithersburg, MD, November 1997. National Institute of Standards and Technology.

[9] Amit Singhal, Chris Buckley, and Mandar Mitra. Pivoted document length normalization. In W. Bruce Croft and C. J. van Rijsbergen, editors, *Proceedings of the Nineteenth Annual Internation ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 21–29. Association for Computing Machinery, August 1996.

[10] Amit Singhal, John Choi, Donald Hindle, David D. Lewis, and Fernando Pereira. AT&T at TREC-7. In E. M. Voorhees and D. K. Harman, editors, *The Seventh Text REtrieval Conference*, NIST Special Publication 500-242, pages 239–252, Gaithersburg, MD, November 1998. National Institute of Standards and Technology.

[11] Ian M. Soboroff and Charles K. Nicholas. Combining content and collaboration in text filtering. In Thorsten Joachims, editor, *Proceedings of the IJCAI'99 Workshop on Machine Learning in Information Filtering*, pages 86–91, Stockholm, Sweden, August 1999.