Combining Content and Collaboration in Text Filtering

Ian M. Soboroff
Department of Computer Science and Electrical Engineering
University of Maryland, Baltimore County
ian@csee.umbc.edu

Abstract

We describe a technique for combining collaborative input and document content for text filtering. This technique uses latent semantic indexing to create a collaborative view of a collection of user profiles. The profiles themselves are term vectors constructed from documents deemed relevant to the user's information need. In initial experiments with a standard text collection, this approach performs quite favorably compared to other content-based approaches. In a larger collection with less possibility for collaboration, the technique does not perform as well.

1 Introduction

Filtering is a process of comparing an incoming document stream to a profile of a user's interests and recommending the documents according to that profile [Belkin and Croft, 1992]. A simple approach for filtering textual content might be to look at each document's similarity to an average of known relevant documents.

Collaborative filtering takes into account the similarities and differences among the profiles of several users in determining how to recommend a document. Typically, collaborative filtering is done by correlating users' ratings of documents. In this approach, a document is recommended to a user because it is highly rated by some other user with whom he or she tends to agree. This can also work for negative ratings; an article may not be recommended because some other "colleague" didn't like it. Examples of such collaborative systems are GroupLens [Konstan *et al.*, 1997] and Ringo [Shardanand and Maes, 1995]. In these "pure" collaborative environments, a document's content is never examined, only its ratings. Such systems have the advantage that they can recommend any kind of content for which one can obtain ratings; however, if a document is unrated, it is difficult to recommend it.

Recently, some exploration has been made into content-based collaborative filtering. In such a system, document content is used in making collaborative decisions. An example of a content-based collaborative filtering environment is Fab [Balabanović and Shoham, 1997]. In Fab, relevance feedback is used to simultaneously mold a personal filter as well as a communal "topic" filter. Documents are discovered and initially ranked by the topic filter according to conformance to their topic, and then sent to users' personal filters. A user then provides relevance feedback for that document, which is used to modify both the personal filter (what the user wants), and the originating topic filter (what matches the topic).

In this paper, we describe a new technique combining content-based and collaborative filtering. This approach compares user profiles and documents in a unified model, which derives relationships between users' interests. The technique uses latent semantic indexing to rearrange a collection of user profiles, so that their commonalities are exploited in the filtering task itself. Candidate documents are routed based on their similarities to the profiles in the LSI space, rather than against the original profiles. We explore the effectiveness of the technique in a batch filtering scenario using a small, standard information retrieval test collection. Finally, we present some results with the TREC collection, which do not illustrate any advantage for this technique, chiefly due to a lack of collaborative potential in the collection.

1.1 Latent Semantic Indexing

Latent semantic indexing, or LSI, is an enhancement to the familiar vector-space model of information retrieval [Deerwester *et al.*, 1990]. Typically, authors will use many words to describe the same idea, and those words will appear in only a few contexts. LSI attempts to highlight these patterns of how words are used within a document collection. By grouping together the word co-occurrence patterns that characterize groups of documents, the "latent semantics" of the collection terms is described. Themes in the document collection arise from subsets of documents with similar word co-occurrences.

Specifically, each document is represented by a vector of terms, whose values are weights related to their importance or frequency of occurrence. The collection of documents, called the *term-document matrix*, is decomposed using the singular value decomposition

$$M = T\Sigma D'$$

The columns of T and D are orthonormal, and are called the *left* and *right singular vectors*. Σ is a diagonal matrix containing the *singular values* σ, ordered by size. If M is $t \times d$ and of rank r, T is a $t \times r$ matrix, D is $d \times r$, and Σ is $r \times r$.

The SVD projects the documents in the collection into an r-dimensional space, in contrast to their t-dimensional representation in the term-document matrix. This LSI space is described by the columns of T, and it is useful to think of $T\Sigma^{-1}$ as a projection for document vectors into the LSI space. In particular, multiplying document vector i from the original term-document matrix by $T\Sigma^{-1}$ yields the ith column of D, the document's representation in the LSI space. We can project any document vector into the LSI space in this way, and compare documents by taking the dot product of their LSI representations.

An important feature of the SVD is that the singular values give an indication of the relative importance of the dimensions, and one can choose how many dimensions to retain by eliminating low-valued dimensions. If all dimensions are kept, then document similarities are the same as they were using the original term-document matrix. If one keeps k dimensions, then the matrix product

$$M_k = T_k \Sigma_k D'_k$$

is the closest rank-k approximation to the original term-document matrix M [Berry *et al.*, 1995]. Document comparisons in this truncated LSI space may be more effective because low-impact term relationships are ignored. Choosing the best value of k is an open problem, and in this work we tried several values to find the best performance.

1.2 LSI for Content Filtering

Latent semantic indexing was applied to filtering as well as ad-hoc retrieval in TREC-3 using a technique similar to ours. The LSI space was first computed from a collection of documents, and then profiles were constructed as centroids of document representations from the LSI space [Dumais, 1995].

Hull, in looking at LSI for use in filtering and routing applications, computed the LSI from a set of documents known to be relevant to the filters [Hull, 1994]. In later experiments [Schütze *et al.*, 1995], a "local LSI" was built from documents similar to a given query. The key insight here is that the LSI projection can be trained from any arbitrary set of document vectors, as long as the vectors we use adequately cover the set of terms we expect to occur in future documents. As Hull observed, the LSI can describe the occurrences of terms across only the documents which are used in the SVD computation. Thus, one should apply the SVD to a document collection that represents the kind of term distributions relevant to the task.

1.3 Combining Content and Collaborative Information

As we mentioned above, collaborative filtering recommends new documents based on correlations of users' ratings of past documents. In general, we can envision a matrix of documents by users, where each user (column) has a set of ratings for some of the documents (rows); this comprises her profile. The goal of a collaborative filtering engine is to fill in the blanks in this matrix with predicted ratings. This is done by computing a correlation coefficient between each user, and predicting a rating of a document to be the weighted sum of other users' ratings of that document by their coefficients [Shardanand and Maes, 1995]. There are other methods for performing this prediction, but this is by far the most common.

In content filtering, the user profile is often constructed from the content of past relevant documents using Rocchio expansion. This may simply yield a centroid of the relevant documents, but more sophisticated techniques are often applied to further refine the profile [Schapire *et al.*, 1998]. These content profiles are closely related to the ratings matrix described above. We can construct a matrix from the ratings matrix such that an entry contains a 1 if the user's rating for that document exceeds some minimum threshold. Each column is then divided by the number of documents exceeding the threshold for that user. Multiplying this profile-construction matrix by the term-document matrix for the document collection produces a matrix of content profiles.

Both of these matrices, the ratings matrix and the content profile matrix, model the universe of user interests. One considers document objects explicitly and separately, while the other pools the document contents to give "ratings" of actual content terms. Neither representation is necessarily collaborative in any way. With the ratings matrix, collaboration occurs when the correlations are used to predict new ratings. For collaboration within the content profile matrix, we use a latent semantic index of the profiles.

Our approach differs from the LSI content filtering approaches in that the LSI space is computed from the collection of profiles, rather than a collection of documents. This means that commonalities between profiles guide the construction of the LSI space. Otherwise, only overlap among documents is able to affect the SVD. Projecting new documents into this space allows us to better view the documents' relation to the group of profiles. Furthermore, collaboration occurs as the LSI describes the relationships of important concepts across profiles.

2 Initial Experiment: Cranfield

We examined our collaborative LSI filtering technique in comparison to two content-based approaches. The first compared incoming documents to the profile centroids using the ordinary term-document matrix. The second was similar to Dumais' approach as described above: an LSI was initially computed from the full document set, and profiles were centroids of some relevant documents within the LSI space. Incoming documents were cast into the LSI space and compared to the profiles.

2.1 Test Collection

Our experiments used the Cranfield corpus, a small test collection of 1400 documents and 225 scored queries. These queries have on average eight relevant documents, and each document is relevant to one or two queries. In order to have a reasonably-sized set of documents for each query, we used a subset of 26 queries which each have 15 or more relevant documents. In this subset, queries had 19 relevant documents on average, with a maximum of 40. The documents in the Cranfield collection are technical scientific abstracts, and are all quite short, between 100 and 4200 bytes in length.

2.2 Document Representation

The documents were indexed using the SMART system [1], which performs stemming of terms and elimination of stop words. The terms were weighted using log-tfidf weighting. All documents were length-normalized for the experiments. Our version of SMART has been modified by the addition of procedures for gathering documents into profiles and computing SVDs of document collections.

2.3 Filter and Test Set Construction

For each query, a training subset and a test subset were derived from the set of relevant documents for that query. This was done by randomly selecting 70% of the relevant documents for use in training, and saving the remaining 30% for testing.

The vectors of the documents in each query's training set were averaged to produce a centroid, which represented the user's profile or filter. The test documents for all queries were pooled to produce a single set of documents against which to test filtering performance. In other words, the test documents for the experiment are each relevant to at least one query. No effort was made to control or induce overlap between queries in the training sets.

[1] `ftp://ftp.cs.cornell.edu/pub/smart`

Two groups of random test and training data were constructed in this way. Each group was then used generate a training matrix (terms by filters) and a test matrix (terms by documents).

2.4 Experimental Tasks

Three different tasks were performed with each test and training set. In the first task, each document in the test collection was ranked against each filter centroid, by taking the dot product between the document and filter vectors. This we call the "content" result, wherein no processing is made on the filters as a group; this is similar to the SIFT system [Yan and Garcia-Molina, 1995].

In the second task, an SVD was applied to the Cranfield term-document matrix. Both the filter centroids and the test documents were cast into this LSI space as described above. The documents were compared to the filters in the LSI space by taking the dot product between them. This we call the "content LSI" result, and is similar to the technique used in [Dumais, 1995].

In the third task, an SVD was applied to the training matrix, and then each document in the test set was cast into the LSI space and ranked against each filter. We call this the "collaborative LSI" result, as here we expected to see the effects of collaboration, as the SVD re-orients the document space along dimensions that highlight common features among the user filters.

As explained above, increasing performance with the SVD requires choosing a dimensionality k smaller than the full rank of the matrix, which will be less than or equal to the smaller of the number of terms and the number of documents. For our experiments, we ran task two using 25, 50, 100, 200, and 500 dimensions out of a possible 1398; and task three with 8, 15, and 18 dimensions out of a possible 26.

3 Results

Using our modified SMART system, the experimental runs yielded a set of ranked scores for each document against each filter. These rankings were used to calculate average precision over fixed levels of recall.

In summary, the collaborative LSI technique performed better than either the content LSI or baseline content approaches. The choice of k was crucial; the best k-value for collaborative LSI allowed it to achieve higher precision than any attempted k-value for content LSI. For most values of k, content LSI did not perform appreciably better than using no LSI at all. Moreover, the best value of k for collaborative LSI was much lower than that needed by content LSI to get good results, meaning that similarities are faster to compute for collaborative LSI.

Figure 1 shows the results of the best runs in each task, chosen by highest overall average prevision. The

Task	k-value	Average Precision	
		Set 1	Set 2
Content (log-tfidf)	–	0.2894	0.2705
Content LSI	25	0.2656	0.1980
	50	0.3136	0.2686
	100	0.3251	0.3053
	200	0.3314	0.3144
	500	0.3302	0.3149
Collaborative LSI	8	0.3136	0.2583
	15	0.4151	0.3745
	18	0.3600	0.3615

Table 1: Overall average precision for each task. This average is over three recall points (0.20, 0.50, 0.80).

overall average precision for all runs in both data sets is shown in Table 1. The figures show precision-recall graphs for the first data set.

Figure 2 shows precision-recall graphs for the content LSI task, compared to the baseline content performance. There isn't a large gain to be made from using LSI here, and in some cases the content LSI approach even performs worse than no LSI at all. In contrast, figure 3 shows precision and recall for the collaborative LSI task, which show a marked gain in performance.

4 Experiments with TREC

The next step was to apply the technique in a larger collection. The TREC-8 routing task was an excellent opportunity for this. Routing at TREC is part of the Filtering track, which for the past two years has used a three-year span of documents from the Financial Times for evaluating filtering techniques.

In the routing task at TREC-8, participants were required to route documents from the 1993-4 Financial Times collection among fifty past TREC topics. These topics have some relevance judgments available in other parts of the TREC collection, including in the Financial Times from 1991-2, and systems were allowed to train their profiles using this information (but none from the test collection).

It's not clear that any collaboration exists a among the topics in TREC, since the topics are not necessarily designed to overlap, either in information interest or in actual relevant document sets. However, several topics this year were 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). There are also groups of topics dealing with medical disorders and thei treatment, pharmaceuticals, and special education.

4.1 Profile Construction

This data set is much larger than the Cranfield collection; the test collection comprises some 140,000 documents. This also meant that the amount of available training data was much less compared to Cranfield. To attempt to overcome this, we employed a more sophisticated profile construction technique which included better term weighting and profile normalization, and unsupervised learning for some training examples.

To build our profiles, we used a technique similar to that used by the AT&T group in TREC-6 [Singhal, 1997] and TREC-7 [Singhal *et al.*, 1998]. First, a training collection was constructed from the FBIS, Los Angeles Times, and Financial Times documents from 1992. We gathered collection statistics here for all future IDF weights. The training documents were weighted with log-tfidf, and normalized using the pivoted unique-term document normalization [Singhal *et al.*, 1996].

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

- 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 Results

We submitted two runs, lists of the 1000 most similar documents for each topic, to be judged; one, *umrqz* routed the test documents using the exact profiles described above. The second, *umrlsi*, first computed the SVD of the collection of profiles, as we did for Cranfield, and routed the test documents in the resulting LSI space.

To choose a dimensionality for the LSI run, we evaluated the routing performance on the training collection. Disappointingly, we did not find that any choice of k gave any benefit to LSI. Arbitrarily, we chose to use 45 out of the possible 50 dimensions.

Overall, both runs performed quite well, with umrqz above the median of all submitted runs 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.

For the majority of queries, however, there was only a very small difference in performance if any between the two runs. We take this to indicate that good overall performance is mostly due to the routing

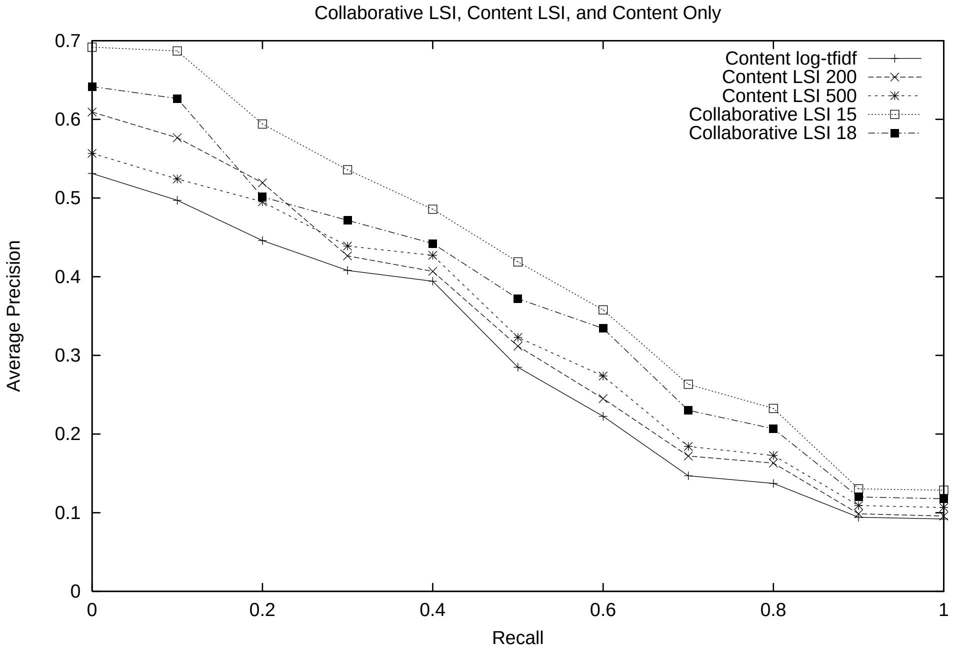

Figure 1: Precision and recall for the top two values of k for content LSI and collaborative LSI, and the content-only task.

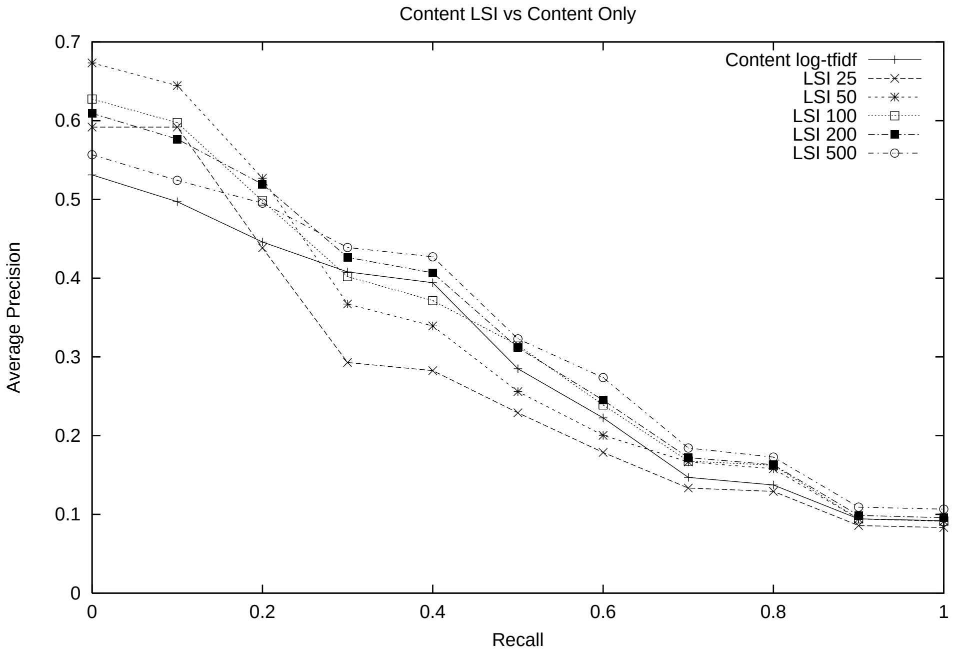

Figure 2: Precision and recall among content LSI runs (k = 25, 50, 100, 200, and 500) in the first data group.

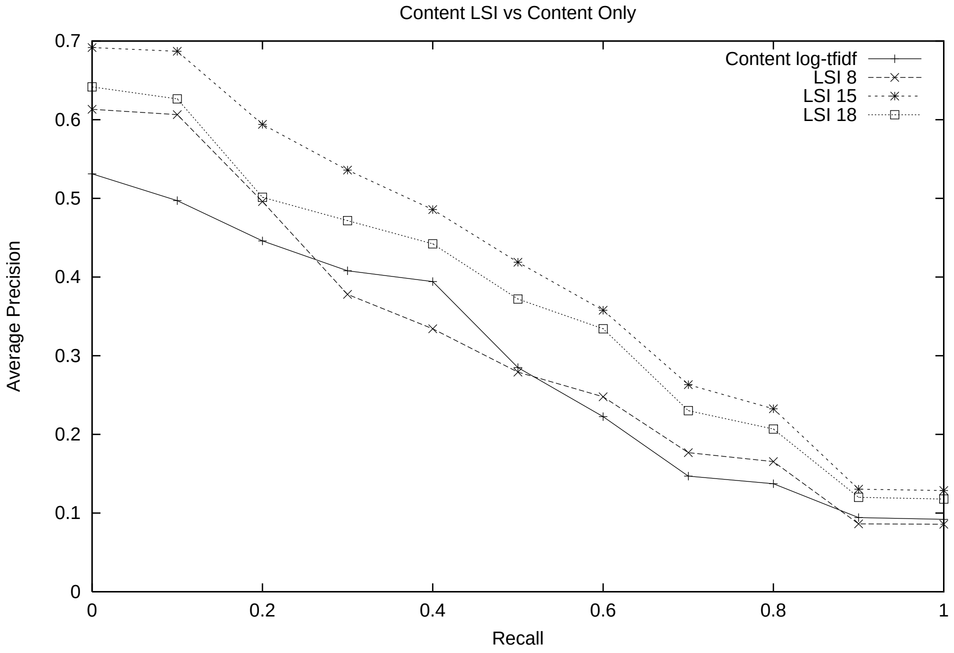

Figure 3: Precision and recall among collaborative LSI runs (k = 5, 15, and 18) in the first data group.

query construction, which uses a combination of approaches shown to work well in previous TRECs. Figure 4 shows the difference in average precision from the mean score for each topic, illustrating the similarity of the results.

This was somewhat 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.

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 non-rotated 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.

To illustrate this, we looked at the distribution of relevant documents among topics, to see if topics indeed share relevant documents. Figure 5 shows, for each run and for the relevance judgments, how many (predicted) relevant topics were given for a document. The "qrels" bars show the actual relevance judgments; one can see that the lions share of documents are relevant to only one topic; less than sixty documents are known to be relevant to more than one topic. If a pure collaborative algorithm were used to predict relevance for these topics, and these relevance judgments were sampled for training data, it would fail miserably because the matrix would be too sparse. The probability of any useful quantity of overlap occurring is very small.

The two charts differ in the method for predicting which documents in the umrqz and umrlsi runs are actually relevant. A routing run contains the highest-scored 1000 documents for each topic, but clearly the system does not expect that all 1000 documents are relevant. Thus, we use only predict as relevant some of the documents in each run. The first picks the top 15 ranked documents; 15 is the median number of relevant documents per topic in the actual relevance judgments. The second picks the top 50.

We can see that our runs tend to spread documents across more topics than are actually relevant. Within the top 15, the qz run distribution is similar to the qrels, and the lsi run gives slightly more overlap. At 50 documents per topic the difference is much greater;

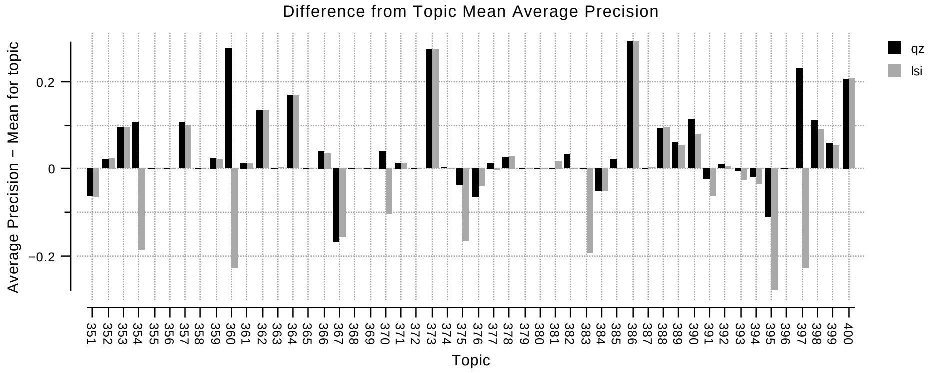

Figure 4: Difference in average precision from mean average precision for each topic. Note that there is very little difference in performance from using LSI.

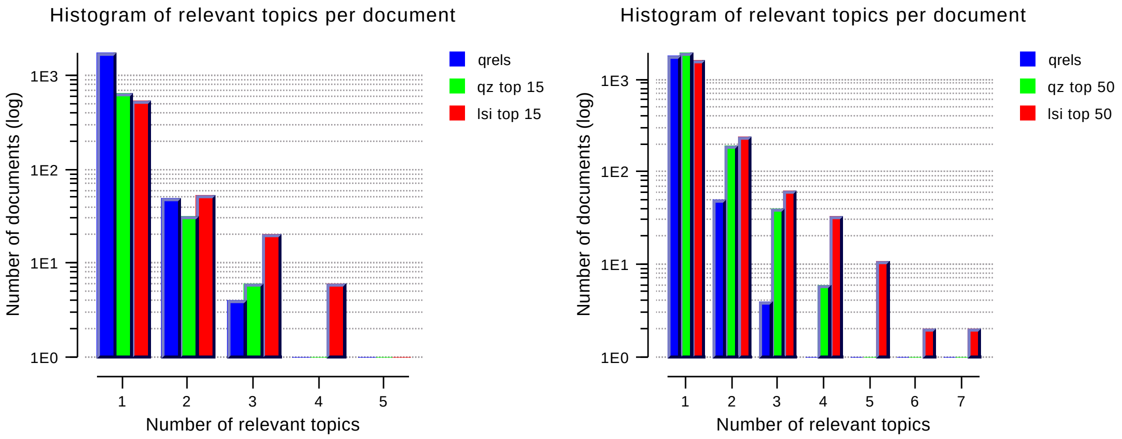

Figure 5: Histograms showing how many topics are relevant to each document, as dictated by the TREC-8 Filtering relevance judgments, and as predicted by the submitted runs. The horizontal axis is the number of relevant topics; the vertical axis is a log scale of the number of documents which are relevant to only that many topics. The chart on the left uses the top 15 submitted documents in each run; the right uses the top 50.

however, for documents that are shared among only two or three topics, the runs are close to each other in overlap.

5 Discussion

The results presented here show that LSI can be a useful tool for text filtering. It is essential that the set of filtering profiles have room for collaboration. The best performance is gained when the right documents are used to generate the best "rose-colored SVD" through which to compare documents to filters. Additionally, some work is needed to discover a good place to truncate the SVD. We currently do this by inspecting some measure of performance at several different k values, but this is obviously rather messy. Other schemes typically involve looking at the decay of the singular values σ_i.

Given that Cranfield is a small and highly specialized collection, we are careful not to generalize too strongly on the results in that collection. The TREC experiment also shows that there are characteristics of the collection which are needed to make collaborative content filtering work. Other, intermediately-sized collections we are examining are the Topic Detection and Tracking corpus[2], and the Reuters-21578 text categorization collection [3].

As far as we know, there are not yet any standard test collections for collaborative filtering whose objects contain a significant amount of text. The most widely used collaborative filtering collection is Digital's EachMovie database [4], but its textual content is limited to names of actors, directors, and the like. This makes it difficult to compare the collaborative aspects of our approach with a more straightforward collaborative filtering algorithm. Test collections will be very important as collaborative filtering research continues to move forward.

Finally, an implicit assumption is often made that interests, user ratings, document similarity, and topical relevance are somehow related. This is evident not only through the use of "did you like this page" ratings data but also in content-oriented information retrieval approaches to filtering. With the use of standard IR collections, the issue is even more pronounced; topical relevance is a much more differently defined and strictly determined concept than that of user preference or even the fulfillment of an information need. These simplifying assumptions are foundational in retrieval and filtering experimentation, but there will certainly be problems when they are applied in real-world filtering systems. It certainly seems, from the early analysis here, that in TREC strict topical relevance may hide the advantages of collaborative filtering.

[2] http://www.ldc.upenn.edu/TDT/

[3] http://www.research.att.com/~lewis/reuters21578.html

[4] http://www.research.digital.com/SRC/eachmovie/

References

[Balabanović and Shoham, 1997] Marko Balabanović and Yoav Shoham. Fab: Content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72, March 1997.

[Belkin and Croft, 1992] Nicholas J. Belkin and W. Bruce Croft. Information filtering and information retrieval: Two sides of the same coin? *Communications of the ACM*, 35(12):29–38, December 1992.

[Berry *et al.*, 1995] 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.

[Deerwester *et al.*, 1990] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

[Dumais, 1995] Susan T. Dumais. Using LSI for information filtering: TREC-3 experiments. In Donna K. Harman, editor, *Proceedings of the Third Text REtrieval Conference (TREC-3)*, pages 219–230, Gaithersburg, MD, November 1995. Also titled "Latent Semantic Indexing (LSI): TREC-3 Report".

[Hull, 1994] 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.

[Hull, 1998] David A. Hull. The TREC-6 filtering track: Description and analysis. In *Proceedings of the Sixth Text REtrieval Conference (TREC-6)*, 1998.

[Konstan *et al.*, 1997] Joseph A. Konstan, Bradley N. Miller, David Maltz, Jonathan L. Herlocker, Lee R. Gordon, and John Riedl. GroupLens: Applying collaborative filtering to Usenet news. *Communications of the ACM*, 40(3):77–87, March 1997.

[Schapire *et al.*, 1998] Robert E. Schapire, Yoram Singer, and Amit Singhal. Boosting and rocchio applied to text filtering. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '98)*, pages 215–223, Melbourne, Australia, August 1998. ACM Press.

[Schütze *et al.*, 1995] Hinrich Schütze, David A. Hull, and Jan O. Pedersen. A comparison of classifiers and document representations for the routing problem. In *Proceedings of the Eighteenth Annual International ACM SIGIR Conference (SIGIR '95)*, pages 229–237, Seattle, WA, USA, July 1995.

[Shardanand and Maes, 1995] Upendra Shardanand and Pattie Maes. Social information filtering: Algorithms for automating "word of mouth". In *Proceedings of CHI'95 – Human Factors in Computing Systems*, pages 210–217, Denver, CO, USA, May 1995.

[Singhal *et al.*, 1996] 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.

[Singhal *et al.*, 1998] 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.

[Singhal, 1997] 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.

[Yan and Garcia-Molina, 1995] Tak W. Yan and Hector Garcia-Molina. SIFT – a tool for wide-area information dissemination. In *Proceedings of the 1995 USENIX Technical Conference*, pages 177–186, 1995.