Technical Report TR-CS-98-01

Collaborative Filtering with LSI: Experiments with Cranfield

Ian M. Soboroff

Department of Computer Science and Electrical Engineering
University of Maryland, Baltimore County
1000 Hilltop Circle
Baltimore, MD 21250

10 November 1998

Collaborative Filtering with LSI: Experiments with Cranfield

Ian M. Soboroff

10 November 1998

Abstract

This document describes an initial experiment in using latent semantic indexing to perform collaborative filtering among a group of user profiles. I applied the technique to a set of 26 queries from the Cranfield corpus, and compared results from different n-gram and term-weighting strategies. For nearly all queries, the LSI approach outperforms a baseline routing technique using centroids.

1 Introduction

Filtering, or routing, is a process of comparing an incoming document stream to a profile of a user's interests and ranking the documents according to that profile [2]. Collaborative filtering utilizes the filtering activity of a community of users to enhance filtering performance.

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 [5][8] and Ringo [9]. In these "pure" collaborative environments, a document's content is never examined, only its ratings. This has the advantage that such a system can recommend any kind of content for which one can obtain ratings; however, if a document is unrated, it typically cannot be recommended.

Recently, some exploration has been made into content-based collaborative filtering. In such a system, document content is used in making collaborative decisions. A premier example of a content-based collaborative filtering environment is Fab [1]. 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 report, I explore a new content-based collaborative filtering technique. This approach compares user profiles and documents in a unified model, which derives relationships between users' interests. The technique uses latent semantic indexing [4] to rearrange a collection of user profiles, so that their commonalities are exploited in the filtering task itself. This involves first constructing user profiles from centroids of documents relevant to the profile, and gathering the profiles into a single term-by-profile matrix. I then compute a reduced-rank SVD of this matrix, yielding the LSI space for the profiles. Candidate documents are routed based on their similarities to the profiles in the LSI space, rather than against the original profile centroids.

Dumais [4] has done work closely related to that described here. In her TREC-3 experiment, she first computed the LSI space from a collection of documents, and then constructed profiles as centroids of document representations from the LSI space. My approach differs in that the LSI space is computed from the collection of profiles, rather than the collection of documents. This means that overlap between profiles is a factor the construction of the LSI space, and commonalities between profiles can be highlighted in the LSI's reduced-dimension representation. In Dumais' work, only overlap among documents was able to affect the LSI space.

2 Experimental Design

I evaluated LSI as a collaborative filtering technique by testing it against a baseline approach which routes documents against the initial profile centroids. My experiments used the Cranfield corpus, a small (1400 document) collection with 225 judged 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, I used a subset of 26 queries for this experiment: queries 1, 2, 23, 46, 47, 57, 65, 67, 72, 73, 90, 125, 132, 156, 157, 186, 201, 202, 203, 204, 212, 217, 218, 219, 220, and 221. These test queries have between 15 and 40 relevant documents each, with an average of 19. The documents in the Cranfield collection are technical scientific abstracts, and are all quite short, between 100 and 4200 bytes in length.

2.1 Document Representation

The documents were represented with a vector-space model, using n-grams as terms. N-grams are overlapping character sequences of length n. N-grams preserve inter-word spaces similarly to stem pairs; however, they are not only easier to collect, but are more resilient to noise and language issues. 3-grams and 5-grams were used as terms in separate tests.

I examined two different term-weighting strategies. The first used only the term frequency, or "tf", which is simply the count of occurrences of the n-gram in each document. The second applied the inverse document frequency to the tf weight ("tf.idf"); this takes into account how often a term is used across the collection. Most experiments with IR and filtering in general, and also studies of LSI, show improvement with tf.idf and other term-weighting approaches over tf or no term weighting. All documents and profiles were length-normalized for the experiments.

2.2 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 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.

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), using either 3- or 5-grams with tf or tf.idf weighting.

2.3 Experimental Tasks

Two tasks were performed with each set of test and training matrices. 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 I call the "baseline" result, wherein no processing is made on the filters as a group; this is similar to SIFT [12].

In the second task, an SVD was first applied to the training matrix, and then each document in the test set was ranked against each filter in the LSI space. This was done by first "casting" the test document into the LSI space by taking the weighted sum of its LSI term vectors, and then comparing the new document representation to each filter using their dot product. I call this the "LSI" result, and here I expected to see the effects of collaboration, as the SVD re-orients the document space along dimensions that highlight common features of the documents, which in this case are user filters. I hypothesized that filters with high degrees of overlap would achieve higher precision, since they should form high-profile communities in the LSI space. A base assumption is that collaborative filtering should give better performance than filtering in isolation, especially for filters which have interests in common.

An important property of LSI is that, if all dimensions are preserved, document similarities in LSI space are equal to document similarities in the original vector space [7]. In other words, if any improvement in performance is to be gained from LSI, one must remove some "low-impact" dimensions of the LSI, those associated with the smallest singular values. The difficulty is to decide

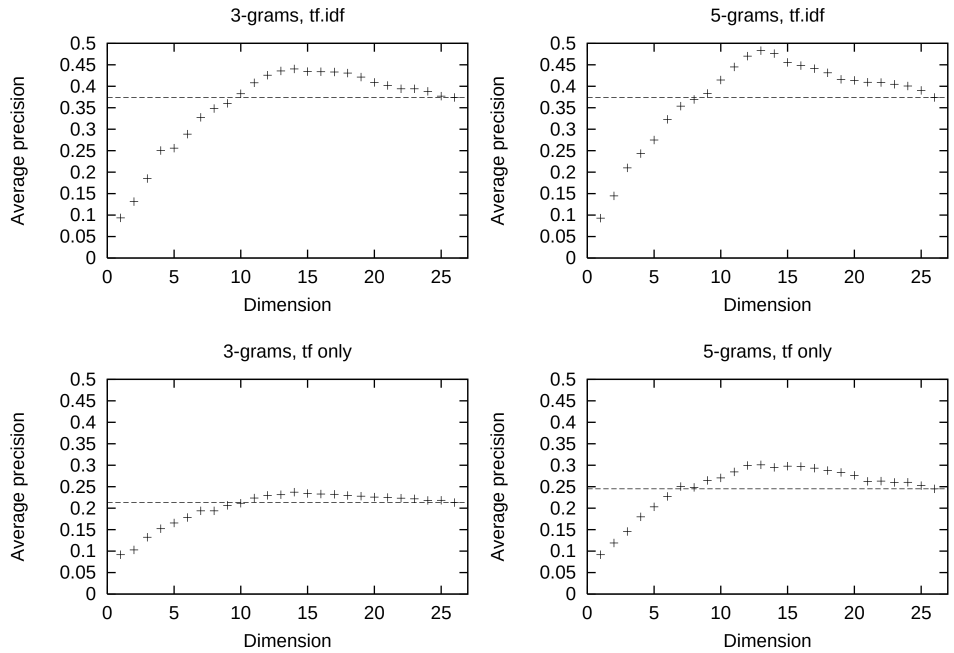

Figure 1: Performance of LSI collaborative filtering at each level of dimensionality. The metric is average uninterpolated precision over all filters. The horizontal line is the baseline performance, which coincides with LSI's performance at full rank.

how many dimensions to retain, since performance falls quickly below that of the original vector space if too few dimensions are retained. To find the optimal number of dimensions, for each combination of n-gram length and term weighting, I evaluated LSI filtering at each level of dimensionality. To evaluate the performance in this scenario, I compared the average precision over all filters. Plots of these values for the first test group, versus the baseline average precision is shown in Figure 1.

Table 1 shows the dimensionality used in each test combination. For 5-grams, 13 dimensions was always optimal, while for 3-grams, 14 dimensions usually gave the best precision. It should be noted that choosing an optimal dimensionality in a "real-world" filtering environment is impossible, since an administrator can not know beforehand how the users will rate documents. Since there is no convenient set of complete relevance judgments available for tuning the system, one would need to apply heuristic approaches. One straightforward approach would be to use all the ratings on past documents, but this assumes that new documents will be similar to ones already seen.

Data Group	Input	Best Rank
Group 1	3-grams, tf	14
	3-grams, tf.idf	14
	5-grams, tf	13
	5-grams, tf.idf	13
Group 2	3-grams, tf	17
	3-grams, tf.idf	14
	5-grams, tf	13
	5-grams, tf.idf	13

Table 1: The number of dimensions used in the LSI filtering task in each test set. These are the dimensions with the highest overall uninterpolated precision.

3 Results

The data from the experimental runs, which consisted of a set of scores for each document by each filter, were analyzed to calculate precision at fixed levels of recall (10%, 20%, ... 100%).

In summary, filtering using LSI profiles performed better than centroids for nearly all filters. The first test group yielded better performance in both LSI and centroid tasks than the second group. However, LSI still surpassed the baseline for most filters in both groups. The results were most significant when using tf.idf weighting and 5-grams. Longer terms and term weighting improved performance, although when using tf.idf weights, use of 5-grams did not contribute as much as might be expected. Finally, examining individual filter performance shows that a few filters do not do well in the LSI task, but usually in these cases the baseline precision is also quite low.

3.1 Average Precision Among all Filters

As an initial overview, precision at fixed recall levels was averaged for all filters. Figure 2 shows graphs of the average precision between the four variables in the first data group. Using this measure, LSI consistently performs above the baseline in each combination. The best performance is seen when tf.idf weights are used; interestingly enough, the use of longer terms did not seem to give a large performance boost over and above tf.idf weighting in the LSI task. This seems to hint that when using LSI for collaborative filtering, it may be more worthwhile to spend time on better weighting, than space for more unique terms.

Table 2 compares performance among all variables in both data groups. The measure used is precision averaged over all queries. The precision at each query is taken as the average precision interpolated over fixed levels of recall. As we have seen above, longer terms and global weights improve performance in both tasks, but LSI consistently performs well above the baseline.

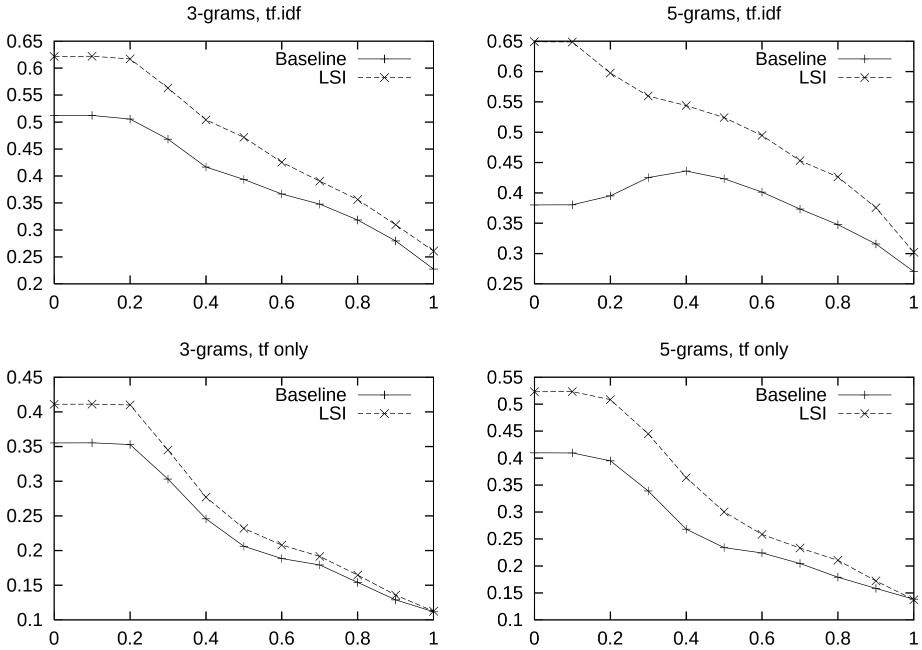

Figure 2: Precision averaged over all filters, for each combination of parameters of term length and weighting. These graphs are for the first data group.

Data Group	Input	Average Precision	
		Baseline	LSI
Group 1	3-grams, tf	0.2346	0.2635
	3-grams, tf.idf	0.3953	0.4675
	5-grams, tf	0.2691	0.3341
	5-grams, tf.idf	0.3771	0.5068
Group 2	3-grams, tf	0.2637	0.2755
	3-grams, tf.idf	0.3467	0.4004
	5-grams, tf	0.2728	0.3163
	5-grams, tf.idf	0.3494	0.4279

Table 2: Average precision (interpolated over fixed levels of recall) over all filters, in each group and choice of input.

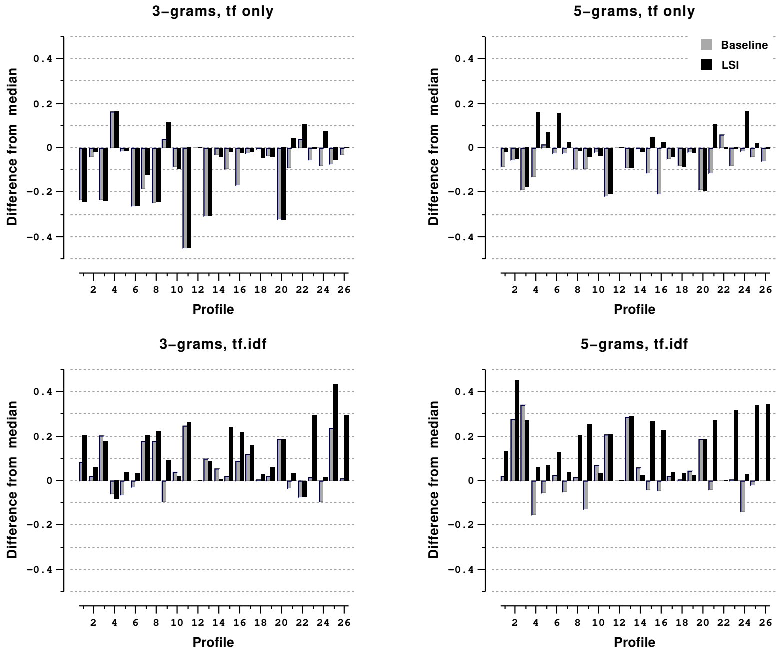

Figure 3: Difference from the median average precision for each filter, in the first data group.

3.2 Performance of Individual Filters

Figures 3 and 4 show each individual filter's average precision, and how that measure changes across the various term lengths and weights. The bars indicate the difference from the filter's median average precision over all runs in a data group.

For most filters, LSI improves filtering performance. These figures also support the impression from figure 2 that the most dramatic improvements come with tf.idf weighting. The best performance is usually seen with 5-grams and tf.idf weights, although better weighting seems more helpful than longer terms. In some cases, 5-grams perform worse than 3-grams when using the same weighting strategy.

It is important to assess whether LSI gives a statistically significant advantage as measured. To gauge the significance, the data would typically be tested using a pooled t-test to see if the mean average precision of LSI filtering is greater than the baseline. The resulting number is the probability that the difference in precision could occur randomly.

A basic assumption of the t-test is that the observations come from a normal distribution, and for such a small sample size, testing this normality assumption is crucial. This can be done by visual inspection of a quantile plot, which shows the cumulative distribution of the sample. If the curve is close to a straight line,

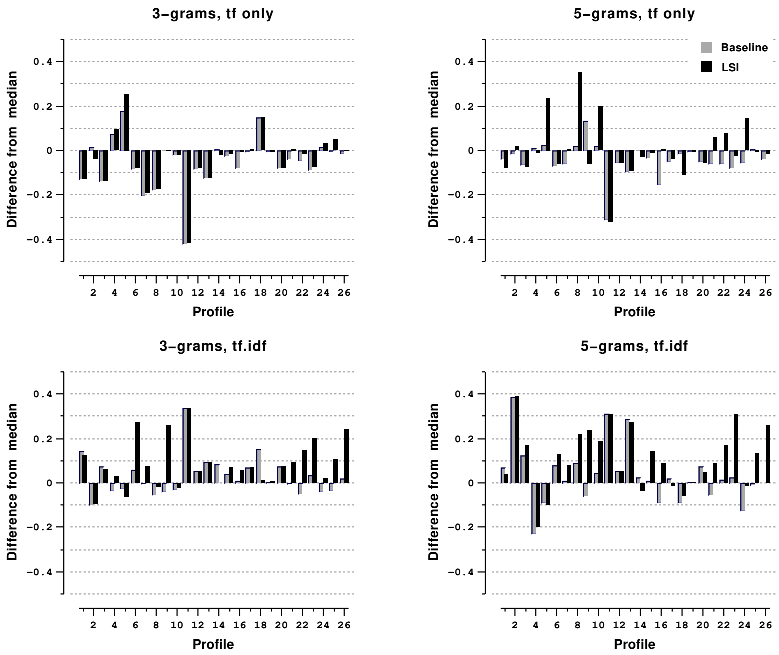

Figure 4: Difference from the median average precision for each filter, in the second data group.

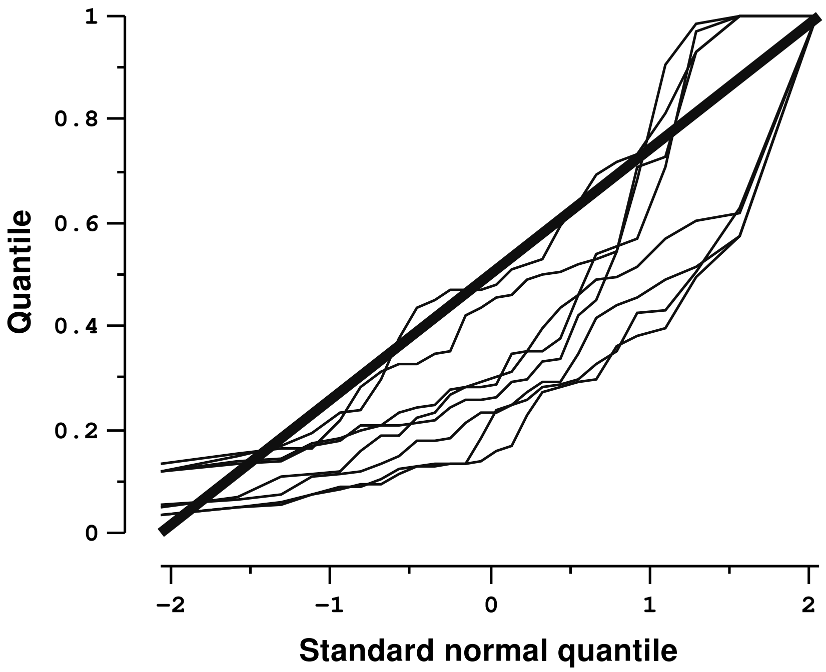

Figure 5: Quantile plots of the average precision data illustrated in figure 3, and linear plot for comparison. Only two lines, which are from 3-gram and 5-gram tf.idf LSI runs, are close to linear.

the plot suggests that the data come from a normal distribution. Figure 5 shows the quantile plots for all results in the first data group, along with a straight line for comparison. One can see that the quantile data from most of the runs are not linear; aside from the tf.idf LSI data, which are the two curves closest to the straight line, the normality assumption may not hold.

In such a case, a non-parametric test is better suited to the data. A non-parametric test does not assume any distribution, but instead examines the relative rankings of the samples themselves. Such tests lack some of the power of a parametric test such as the t-test, but are nearly as efficient in cases where data is non-normal. I applied two non-parametric tests, the sign test and the Wilcoxon signed-ranks test. For both cases, the test statistic is the difference from the baseline average precision. The sign test looks at the quantity of positive and negative differences. The Wilcoxon test also takes into account the magnitude of the differences, and is thus more statistically efficient. Table 3 shows the results of both tests.

The test results show the probability that LSI performs at the baseline level,

Data set	Sign test		Wilcoxon Test	
	Group 1	Group 2	Group 1	Group 2
3gram-tf	0.1147	0.0073	0.0173	0.0029
5gram-tf	0.0216	0.2121	0.0009	0.0463
3gram-tf.idf	0.0113	0.0173	0.0004	0.0075
5gram-tf.idf	0.0053	0.0113	0.0003	0.0008

Table 3: Results from sign tests and Wilcoxon tests on average precision. The null hypothesis is that the mean average precision of LSI filtering is equal to the baseline. The numbers shown are the probability that H_0 is true, i.e., that LSI affords no significant advantage.

as opposed to performing above it. This probability is barely significant in all but two cases. We can be most certain that LSI will perform above the level of the centroid approach when tf.idf weights and 5-grams are used. One should keep in mind that as with Table 2 and Figure 2, this analysis looks at all the filters as a whole, and does not highlight individual profile performance. Additionally, these values do not show how large the difference is, merely how confident we are that the difference *is* significant at all.

When tf.idf weights are used, it is not clear that employing longer terms provides a significant gain. In testing for a significant advantage for tf.idf-weighted 5-grams over 3-grams, the probability values from the Wilcoxon test are .0614 and .3991 for the first and second test groups respectively. This indicates a reasonably high chance that there is little advantage to using 5-grams when tf.idf weights are used.

3.3 Assessing Opportunity for Collaboration

While it has been shown that most filters perform better using LSI, it is not clear that these results can be attributed to obvious collaborative potential. Table 4 shows the number of documents in each query's training set which are shared with some other query's training set. From this cursory examination, there does not seem to be any correlation between filter performance in the LSI task and quantity of training overlap. To properly test whether overlap and task performance are independent, a larger sample is needed.

Table 4 also shows the number of training documents for each query. The number of documents used to construct the filters do not seem to correlate with LSI's performance in this experiment.

4 Discussion and Analysis

The results of the experiment do seem to indicate that using LSI as an engine for collaborative filtering is a promising approach. However, "low-level" questions

Query	Pct	Train	Task	Query	Pct	Train	Task	Query	Pct	Train	Task
1	30	20	LSI	10	57.1	14	Base	19	80	10	Base
2	29.4	17	LSI	11	0	11	LSI	20	0	10	Base
3	8.7	23	Base	12	0	11	—	21	54.5	11	LSI
4	54.5	11	LSI	13	10	13	Base	22	81.8	11	Base
5	70	10	LSI	14	46.4	28	Base	23	61.5	13	LSI
6	40	10	LSI	15	72.7	11	LSI	24	64.3	14	LSI
7	63.6	11	LSI	16	54.5	11	LSI	25	69.2	13	LSI
8	80	10	LSI	17	70	10	LSI	26	35.3	17	LSI
9	75	12	LSI	18	70	10	—				

Table 4: Percent of documents in each query's training set which are shared with other queries' training sets. "Train" is the actual size of the training set. "Task" names the task which performed best in the majority of input variations for that query, from the query's average precision in the first data group.

of term selection and weighting, as well as questions about filter construction and collaborative potential have yet to be answered.

The length of n-grams was not as large an influence on task performance as term weighting, although both helped. In particular, when tf.idf weighting is used, 5-grams don't provide a large additional benefit. In some cases, in fact, using 5-grams resulted in a lower average precision than when using 3-grams! In prior IR experiments with n-grams from more prosaic text, greater performance variation is observed when varying the size of n, although the relative virtues of large or small n are application-dependent [10].

It may be that this affect is due to the Cranfield corpus, because documents are short and filled with very specialized terms. Further experiments with different classes of documents will explore this hypothesis; since shorter n-grams result in fewer unique n-grams in a collection, good performance with simpler terms could be quite advantageous.

The greater issue is that the experiment described here does not show that LSI is explicitly fostering collaboration, at least by the primitive measure used. While it might be assumed that any process which takes a community of user models into account is collaborative, it follows that a collaborative process should show the greatest improvement for those users who are able to collaborate the most! I again attribute this lack of apparent collaborative impact to the uniformity of the Cranfield collection.

This "uniformity" can be illustrated by a scree plot, a graph of the sorted singular values of a matrix. Figure 6 shows these plots for the SVD from the first data group's filters. One can see that the singular values drop off very sharply at the second value, showing very little variation in the "collection's" term usage. Note also that the graphs for the tf data sets do not seem to correspond to the dimensionalities chosen for LSI filtering from Figure 1. I would expect that there would be reasonably high values corresponding roughly

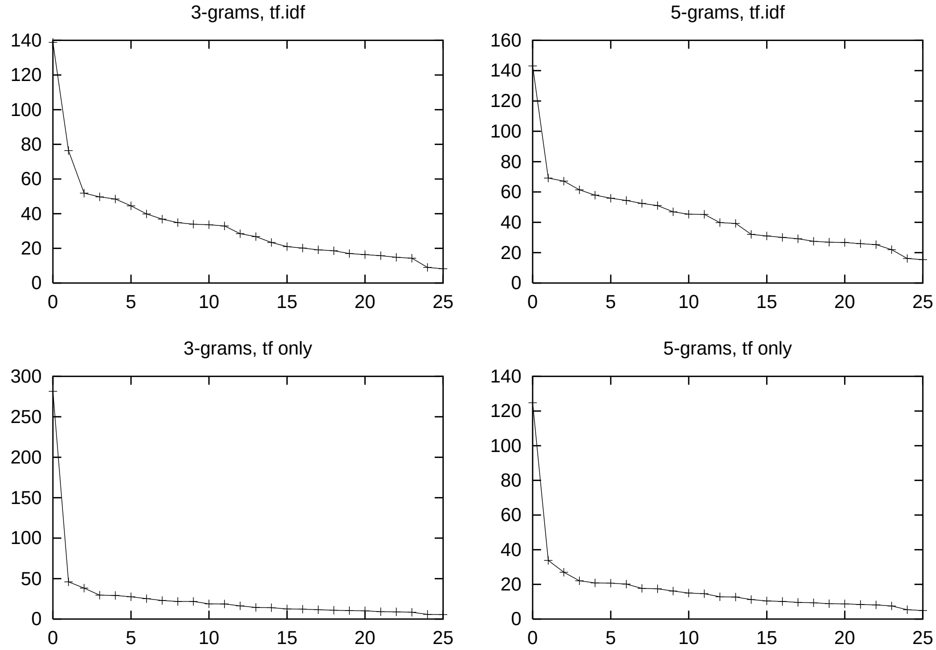

Figure 6: Plots of the singular values of the SVD's in the first data group.

to the number of distinct filtering interests in the community, and thus interpret these plots as indicating that the filters are rather homogeneous as far as the LSI is concerned. A further experiment should look directly at collaborative potential between filters.

5 Conclusion and Future Work

The use of latent semantic indexing as a method of collaborative filtering is very promising. These experiments have shown that computing the SVD of a matrix of filters can significantly enhance performance relative to the use of the filters alone. However, I have not shown the specific aspects of a filter that may cause it to work well with LSI, and issues of term length and weighting strategies need to be further explored.

Given that Cranfield is a small and highly specialized collection, I suspect that the results might be different on a larger and more general corpus. This might also better reflect the filtering environment of actual users in a hypothetical system using the LSI collaboration approach. I would like to apply similar experiments to Lewis' Reuters collection [6], and/or the TDT Corpus [3] from the Linguistic Data Consortium.

Since the articles in these collections are more prose-like, containing more stop words and less jargon as a proportion of the total terms, I expect that LSI will work even better than it has in this experiment. Moreover, Reuters and TDT have an order of magnitude more documents than Cranfield.

The TDT collection provides an opportunity for a different experimental approach. The documents in TDT are ordered chronologically, so I could define the filtering interests, and consider the test documents as a stream over time, applying relevance feedback as an actual user might. In this case, a more sophisticated performance measure than average precision over the entire task is required. Additionally, by examining precision and recall over time, the time needed to train the filters can be determined, as well as their susceptibility to overfitting.

References

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

[2] 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.

[3] Linguistic Data Consortium. The TDT pilot study corpus. `http://www.ldc.upenn.edu/ldc/catalog/html/text_html/tdt.html`, October 1997.

[4] 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".

[5] 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.

[6] David Lewis. The Reuters-21578 text categorization test collection. `http://www.research.att.com/~lewis/reuters21578.html`. Checked on 11 May 1998; Timestamp Tue Jan 20 21:07:21 EST 1998.

[7] Douglas W. Oard. *Adaptive Vector Space Text Filtering for Monolingual and Cross-Language Applications*. PhD thesis, University of Maryland College Park, 1996.

[8] Paul Resnick, Neophytos Iacovou, Mitesh Suchak, Peter Bergstrom, and John Riedl. GroupLens: An open architecture for collaborative filtering of netnews. In *Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work*, pages 175–186, Chapel Hill, NC, 1994. ACM.

[9] 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.

[10] Ian M. Soboroff, Charles K. Nicholas, James M. Kukla, and David S. Ebert. Visualizing document authorship using n-grams and latent semantic indexing. In *Proceedings of the Workshop on New Paradigms in Information Visualization and Manipulation (NPIV '97)*, Las Vegas, NV, USA, November 1997. ACM Press.

[11] Ronald E. Walpole and Raymond H. Myers. *Probability and Statistics for Engineers and Scientists.* Macmillan Publishing Company, fifth edition, 1993.

[12] 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.