

Improving Retrieval Performance with Positive and Negative Equivalence Classes of Terms

April Kontostathis
Lehigh University
CSE Department
19 Memorial Drive West
Bethlehem, PA 18015
apk5@lehigh.edu

William M. Pottenger, Ph.D.
Lehigh University
CSE Department
19 Memorial Drive West
Bethlehem, PA 18015
billp@cse.lehigh.edu

ABSTRACT

One of the most pressing problems facing application developers in the area of information retrieval (IR) is the lack of sound mathematical, theoretical frameworks for understanding IR [SIGIR2000]. Although many such frameworks have been proposed, in the final analysis none has been sufficiently well-grounded to attain widespread acceptance in the field. In addition, there is all too often a lack of empirically sound evaluation of such frameworks in an actual application. For this reason we have forayed into the theoretical domain of IR, while at the same time grounded our work in an application of widespread importance, search and retrieval. One need only glance at the statistics of the hit counts of the latest search engines to realize just how important search and retrieval has become. In this paper we present a novel approach to term clustering and its application in improving the performance of search and retrieval. Our approach is firmly grounded in a theoretical framework that we have developed.

Term clustering is an approach that researchers have used to convert the original words of a document into more effective content identifiers. Term clustering algorithms generally consist of two phases. In the first phase term-term similarity is determined. The second phase uses the term-term similarities to develop clusters of terms. Latent Semantic Indexing (LSI) [Deerwester, et al – 1990] is a well-known information retrieval algorithm that is based on Singular Value Decomposition (SVD). The values in the truncated term-term matrix produced by SVD can be treated as similarity measures for input to a clustering algorithm. In this work we present an algorithm that produces clusters of terms that improve retrieval performance (as measured by precision and recall).

We assume that the value in position (i,j) of the term-term matrix represents the *similarity* between term i and term j in the collection. By extension, a negative value represents an *anti-similarity* between term i and term j . Our approach searches for both positive and negative clusters of terms. We show that the positive clusters, when used to expand an initial query, result in significant improvements in recall for a given collection. Furthermore, the negative clusters, when used to prune the result set, result in significant improvements in precision. To our knowledge, these are the first significant results that show that anti-similarity clusters exist and can be used to improve performance of search and retrieval in IR.

1. INTRODUCTION

One of the most pressing problems facing application developers in the area of information retrieval (IR) is the lack of sound mathematical, theoretical frameworks for understanding IR. Although many such frameworks have been proposed, in the final analysis none has been sufficiently well-grounded to attain widespread acceptance in the field. In addition, there is all too often a lack of empirically sound evaluation of such frameworks in an actual application. For this reason we have forayed into the theoretical domain of IR, while at the same time grounded our work in an application of widespread importance, search and retrieval. One need only glance at the statistics of the hit counts of the latest search engines to realize just how important search and retrieval has become. In this paper we present a novel approach to term clustering and its application in improving the performance of search and retrieval. Our approach is firmly grounded in a theoretical framework that we have developed.

Term clustering is an approach that researchers have used to convert the original words of a document into more effective content identifiers [Lewis and Croft –1990]. Term clustering algorithms generally consist of two phases. In the first phase term-term similarity is determined. The second phase uses the term-term similarities to develop clusters of terms. [Lewis and Croft – 1990] presents a historical overview of term clustering approaches. Early work by Sparck-Jones [Sparck-Jones – 1971,1973], and others showed that small clusters of terms were most effective at improving performance, but no significant

advantages were noted, particularly when results were compared to those obtained by the use of manual thesauri. Automatic thesaurus generation approaches are similar to term clustering and [Sanderson and Croft – 1999] presents an overview of the previous research done in this area. The work cited concentrated on the detection of synonyms, hyponyms, hypernyms, and type-of relationships.

Our research program is focused in two areas: the first component is the study of the *anti-similarity* between terms, which can be defined in various ways. The current work defines similarity and anti-similarity based on the Latent Semantic Indexing (LSI) [Deerwester, et al. – 1990, Dumais – 1993] reduced dimensionality term-term matrix. LSI is a well-known information retrieval algorithm that is based on Singular Value Decomposition (SVD). The values in the truncated term-term matrix produced by SVD can be treated as similarity measures for input to a clustering algorithm. This paper describes an algorithm that produces both *positive* and *negative* clusters based on the values in this matrix. This is the first work that uses the LSI term-term values in a term clustering algorithm. An overview of LSI is presented in section 2, along with an explanation of how we use the term-term matrix as similarity and anti-similarity measures between terms. In Section 3, the similarity is used to produce *positive* term clusters and the anti-similarity is used to produce *negative* term clusters. We show in section 4 that these negative term clusters, when used to restrict the number of documents returned by a query, can significantly improve precision. Furthermore, the positive term clusters, when used to expand a query, significantly improve recall.

Secondly, we agree with recent assertions an underlying mathematical, theoretical framework should be provided for research in IR [SIGIR2000], and we want to ensure that any approach we use has a sound theoretical foundation. Toward this end, we consistently provide an underlying mathematical theory that underlies our experimental results. In this paper, we show in section 3 that the clusters of terms that are created by our algorithm can be defined by equivalence relation, and we state this relation. From mathematics, we know that an equivalence relation must enforce the reflexive, symmetric, and transitive properties for the classes it creates. Our results provide support for our claim that clusters that can be associated with an equivalence relation are semantically richer.

An extensive review of the literature uncovered very little research in the area of anti-similarity or negative clusters. [Yu, et al. – 1978] proves mathematically that negative weightings can be used to improve query performance and [Raghavan and Yu – 1979] presents one approach that uses anti-similarity to produce query clusters rather than term clusters. Their paper describes a supervised machine-learning algorithm that uses known query/result pairs to train an algorithm that develops positive and negative term relationships that are then used to cluster queries. The experimental results show an improvement in performance for both the positive and negative metrics, when applied either separately or together.

More recently, [Hoashi, et al. – 2000] have incorporated negative similarities into an algorithm for document filtering. Their approach includes the development of a non-relevant profile, as well as a relevant profile, that increases the precision of the filtering task. The non-relevant profile is used to identify the features of mistakenly retrieved non-relevant documents.

Our literature search did not uncover any unsupervised learning algorithms that developed negative or anti-similarity clusters of terms. To the best of our knowledge, this paper represents the first research to date that proves that these negative clusters exist and can be used to improve performance in a search and retrieval application.

2. Defining term-term similarity and anti-similarity using the LSI term-term matrix

The data for the following example is taken from [Deerwester, et al. – 1990]. In this paper, the authors describe an example with 12 terms and 9 documents. The term by document matrix is shown as table 1, and the corresponding term-by-term matrix is shown in table 2.

The SVD process used by LSI decomposes the matrix into three matrices: T, a term by dimension matrix, S a singular value matrix, and D, a document by dimension matrix. The number of dimensions is $\min(t, d)$ where t = number of terms and d = number of documents. The original matrix can be obtained, through matrix multiplication of TSD^T . The reader is referred to Deerwester, et al – 1990 for the T, S, and D matrices. In the LSI system, the T, S and D matrices are truncated to k dimensions. The dimensionality reduction reduces “noise” in the term-term matrix resulting in a richer word relationship structure that reveals latent semantics present in the collection. After dimensionality reduction the term-term matrix can be approximated

using the formula $T_k S_k (T_k S_k)^T$. A representation of the term-term matrix, after reduction to 2 dimensions ($k=2$) is shown in table 3.

For our purposes, we will assume that the value in position (i,j) of the matrix represents the *similarity* between term i and term j in the collection. As can be seen in table 3, *user* and *human* now have a value of .94, representing a strong similarity, where before the value was zero. In fact, *user* and *human* is an example of second order co-occurrence. The relationship between *user* and *human* comes from the transitive relation: *user* co-occurs with *interface* and *interface* co-occurs with *human* (see table 2); therefore, *user* and *human* co-occur. This demonstrates the value of the LSI system, since queries on the term *user* would correctly result in documents containing the term *human* in the context of this collection.

A closer look reveals a value of 0.15 in the relationship between *trees* and *computer*. Looking at the co-occurrence path gives us an explanation as to why these terms received a positive (although weak) similarity value. From table 2, we see that *trees* co-occurs with *graph*, *graph* co-occurs with *survey*, and *survey* co-occurs with *computer*. Hence the *trees/computer* relationship in the reduced term-term matrix is an example of third order co-occurrence. A complete discussion of the use of higher orders of co-occurrence, and a discussion of how higher-order co-occurrence correlates with the term-term similarity value can be found in [Kontostathis and Pottenger, 2002].

Table 1: Deerwester Term by Document Matrix

Terms	Documents									
	c1	c2	c3	c4	c5	m1	m2	m3	m4	
human	1	0	0	1	0	0	0	0	0	
interface	1	0	1	0	0	0	0	0	0	
computer	1	1	0	0	0	0	0	0	0	
user	0	1	1	0	1	0	0	0	0	
system	0	1	1	2	0	0	0	0	0	
response	0	1	0	0	1	0	0	0	0	
time	0	1	0	0	1	0	0	0	0	
EPS	0	0	1	1	0	0	0	0	0	
Survey	0	1	0	0	0	0	0	0	1	
trees	0	0	0	0	0	1	1	1	0	
graph	0	0	0	0	0	0	1	1	1	
minors	0	0	0	0	0	0	0	1	1	

Table 2: Deerwester Term by Term Matrix

	human	interface	computer	user	system	response	time	EPS	Survey	trees	graph	minors
human	x	1	1	0	2	0	0	1	0	0	0	0
interface	1	x	1	1	1	0	0	1	0	0	0	0
computer	1	1	x	1	1	1	1	0	1	0	0	0
user	0	1	1	x	2	2	2	1	1	0	0	0
system	2	1	1	2	x	1	1	3	1	0	0	0
response	0	0	1	2	1	x	2	0	1	0	0	0
time	0	0	1	2	1	2	x	0	1	0	0	0
EPS	1	1	0	1	3	0	0	x	0	0	0	0
Survey	0	0	1	1	1	1	1	0	x	0	1	1
trees	0	0	0	0	0	0	0	0	0	x	2	1
graph	0	0	0	0	0	0	0	0	1	2	x	2
minors	0	0	0	0	0	0	0	0	1	1	2	x

Note also the negative values produced in the term-term matrix. We will assume a negative value in position (i,j) represents an *anti-similarity*. Note that the anti-similarity values also have a connectivity path in the matrix. For example, the value -.25 for *minors* and *human* has the path *minors* to *survey* to *interface* to *human*, an example of third order co-occurrence.

In the following section we use the similarity and anti-similarity values to produce clusters of terms.

Table 3: Deerwester Term by Term Matrix, truncated to two dimensions

	human	interface	computer	user	system	response	time	EPS	Survey	trees	graph	minors
human	0.62	0.54	0.56	0.94	1.69	0.58	0.58	0.84	0.32	-0.32	-0.34	-0.25
interface	0.54	0.48	0.52	0.87	1.50	0.55	0.55	0.73	0.35	-0.20	-0.19	-0.14
computer	0.56	0.52	0.65	1.09	1.67	0.75	0.75	0.77	0.63	0.15	0.27	0.20
user	0.94	0.87	1.09	1.81	2.79	1.25	1.25	1.28	1.04	0.23	0.42	0.31
system	1.69	1.50	1.67	2.79	4.76	1.81	1.81	2.30	1.20	-0.47	-0.39	-0.28
response	0.58	0.55	0.75	1.25	1.81	0.89	0.89	0.80	0.82	0.38	0.56	0.41
time	0.58	0.55	0.75	1.25	1.81	0.89	0.89	0.80	0.82	0.38	0.56	0.41
EPS	0.84	0.73	0.77	1.28	2.30	0.80	0.80	1.13	0.46	-0.41	-0.43	-0.31
Survey	0.32	0.35	0.63	1.04	1.20	0.82	0.82	0.46	0.96	0.88	1.17	0.85
trees	-0.32	-0.20	0.15	0.23	-0.47	0.38	0.38	-0.41	0.88	1.55	1.96	1.43
graph	-0.34	-0.19	0.27	0.42	-0.39	0.56	0.56	-0.43	1.17	1.96	2.50	1.81
minors	-0.25	-0.14	0.20	0.31	-0.28	0.41	0.41	-0.31	0.85	1.43	1.81	1.32

3. A TERM CLUSTERING ALGORITHM BASED ON AN EQUIVALENCE RELATION

The use of co-occurrence information in textual data has led to improvements in performance when applied to a variety of applications in Information Retrieval, Computational Linguistics and Textual Data Mining. Furthermore, many researchers in these fields have developed techniques that explicitly employ a limited degree of transitivity in the co-occurrence relation. Examples include applications such as Word Sense Disambiguation [Schütze - 1998], Ranking of Relevant Documents [Veling and Van Der Weerd - 1999], and Word Selection [Edmonds - 1997]. Other authors have developed algorithms that implicitly rely on the use of a limited degree of transitivity in the co-occurrence relation for applications such as Search and Retrieval [Deerwester, et al - 1990], and Stemming [Xu and Croft - 1998]. In what follows we refer to various degrees of transitivity in the co-occurrence relationship as orders of co-occurrence – first order if two terms co-occur, second order if two terms are linked only by a third, etc. An example of second order co-occurrence follows.

Assume that a collection has one document that contains the terms *Java* and *applet*. Also, assume that a different document in the collection contains the terms *Java* and *e-commerce*. Furthermore, assume that *applet* and *e-commerce* do not co-occur elsewhere in the collection. It may be useful to gather these three terms in one unique class and infer that *e-commerce* and *applet* are actually related despite the fact that they do not co-occur in any item in the given collection. In other words, due to the strength of the statistical association of these terms, a limited degree of transitivity is imputed to the co-occurrence relation. This example of second-order co-occurrence can be extended to third, fourth, or nth order co-occurrence.

The work described in [Kontostathis and Pottenger – 2002a, 2002b] provides a theoretical foundation for understanding the values produced in the reduced form of the term-term matrix. These results show that LSI's use of higher orders of co-occurrence is a critical component of this foundation. This work demonstrates that LSI uses up to fifth order term co-occurrence. A mathematical proof that a connectivity path exists for every nonzero element in the truncated term-term matrix computed by LSI is also provided. A complete understanding of this term transitivity is key to understanding LSI. These papers extend the work of Schütze, who has studied the values produced by LSI [Schütze - 1992].

```

Input SVD vectors
Create Term-by-Term matrix (TS) (TS)T, discard values between -0.1 and 0.1
For (distance_threshold = max_similarity, distance_threshold > 0, distance_threshold -= 0.001)
  Initialize class list to all zeros
  For (term_iter = 0; term_iter < num_terms; term_iter++)
    If (class_list[term_iter] > 0) continue; // already assigned to a class skip this iteration
    Create list of all terms connected to term_iter with similarity > distance_threshold
    Ensure that terms in list form a completely connected graph with arc weight > distance_threshold
    If terms in list form completely connected graph
      store cluster
      mark terms in class_list
    else
      continue with next term
    end if
  end for
  Evaluate set of clusters formed for this distance threshold, output results
end for
For (distance threshold = min_similarity, distance_threshold < 0; distance_threshold += 0.001)
  Form clusters based on anti-similarity measure using the same algorithm as above

```

Figure 1: Clustering Algorithm

Figure 1 describes an algorithm that can be used to create clusters of term using the truncated term-term matrix. Our clustering approach uses a threshold for term-term similarity (as defined by the term-term matrix, after truncation to k dimensions). All values below the threshold were disregarded and the program searches for clusters of terms that remain completely connected. Clusters based on the negative similarity measures are also created. This work has resulted in the formation of small clusters (few terms in each cluster). Evaluation of these clusters in a search and retrieval application is discussed in section 4.

The clustering algorithm defined in the preceding paragraph can be represented by an equivalence relations that defines the classes of terms. In mathematical terms, let us define $S(A,B)$ as the similarity between term A and term B , and let the threshold be defined as α . Then terms A and B will be in the same cluster if and only if, the $S(A,B) \geq \alpha$ and, for all C , $S(A,C) \geq \alpha \rightarrow S(B,C) \geq \alpha$. The negative similarity clusters are created from the relation: terms A and B will be in the same anti-similarity cluster if and only if, $\alpha < 0$, $S(A,B) \leq \alpha$ and, for all C , $S(A,C) \leq \alpha \rightarrow S(B,C) \leq \alpha$. These definitions are clearly reflexive, symmetric and transitive. It is our intention to update these definitions, as necessary, as we continue to refine the clustering algorithm.

4. EXPERIMENTAL RESULTS

We have run our evaluation algorithm on three small gold standard collections: MED, CRAN and CISI. MED is a commonly studied collection of medical abstracts. It consists of 1033 documents and 5823 terms. CISI is a set of 1460 information science abstracts containing 5609 terms. CRAN is the acronym for the Cranfield collection, one of the earliest collections available to information retrieval researchers. CRAN consists of 1398 documents and 4612 terms.

Each gold standard collection contains a group of documents, a set of standard queries and a set of relevance judgments (list of which documents are relevant to which query). We used standard search and retrieval algorithms (described in section 4), enhancing the queries using the clusters we developed, and reporting the results. Initially we expanded the query using all terms that appear in a cluster with one of the query terms. As expected, recall was improved. The anti-similarity clusters were incorporated as a NOT operator, with the goal of improving precision. Recall is defined as the number of relevant documents retrieved, divided by the total number of relevant documents in a collection. Precision is the number of relevant documents retrieved divided by the total number of documents retrieved.

In order to complete these experiments, we also needed a program to perform the SVD decomposition. The SVDPACK suite [Berry, et al - 1993] that provides eight algorithms for decomposition was selected because it was readily available, as well as thoroughly tested. The sis1 algorithm of the SVDPACK suite was used in our experiments. The SVDPACK algorithms take input in Harwell-Boeing sparse matrix format. The MED, CRAN and CISI data was already in this format and was used without further preprocessing. The sis1 program was then executed for each collection to find the singular values and vectors for k = 100. Current research in Latent Semantic Indexing (LSI) uses experimental methods to determine the appropriate number of dimensions for a given application. Various studies have used a k-value (k = number of dimensions) of 100 to 300 [Dumais - 1993]. The singular values and vectors were then input into the clustering algorithm, which identifies a set of clusters for each distance threshold.

Class:	information	systems		
Class:	journal	journals		
Class:	libraries	library		
Class:	retrieval	system		
Class:	bases	data		
Class:	index	terms		
Class:	science	scientific		
Class:	information	retrieval	system	systems
Class:	document	documents		
Class:	chemical	computer		
Class:	citation	citations		
Class:	precision	recall		
Class:	serial	serials		
Class:	marc	records		
Class:	cable	telephone	television	
Class:	game	games		
Class:	adjective	scales		

Figure 2: Positive Clusters Found for CISI (at any threshold level)

Most of the clusters found using this algorithm contain only one term and are therefore, not very helpful. However, when larger numbers of terms are clustered together, the algorithm finds meaningful clusters. Figure 2 shows the positive clusters found for the CISI collection. This list includes clusters found at any distance threshold, so you will notice that some of the terms are shown twice (for example, Information System and Information Retrieval System Systems). You will also notice that the algorithm finds multiple instances of singular/plural pairs (library, libraries; document, documents), but not as many as a stemming algorithm would identify. The algorithm also identifies pairs that are clearly related, but would not be found by stemming algorithms (information retrieval; bases data; precision recall). We show in the next section that using these clusters to expand a query can result in a significant improvement in recall.

Figure 3 shows the negative clusters for CISI. There does not appear to be an intuitive relationship between the pairs of terms. Clearly, they are not antonyms. More study is required to understand why LSI (or SVD) is assigning a negative similarity weight to these pairs of terms. For this work, we will be content to show that using these terms to limit the number of documents returned by a query will significantly improve the precision of the query.

Class: automatic	serial	Class: coding	references
Class: books	language	Class: dewey	retrieval
Class: fuzzy	social	Class: graph	project
Class: answer	survey	Class: papers	regional
Class: abstracts	libraries	Class: faculty	knowledge
Class: loan	text	Class: medline	methods
Class: cable	titles	Class: microfiche	science
Class: decision	journals	Class: percent	quality
Class: cited	programs	Class: assistance	author
Class: academic	words	Class: center	objects
Class: catalog	psychology	Class: computerized	probability
Class: classification	periodical	Class: periodicals	structure
Class: education	thesaurus	Class: epidemic	service
Class: organizations	selection	Class: medium	studies
Class: code	medical	Class: 1962	system
Class: committee	journal	Class: attitudes	form
Class: publication	relevance	Class: concept	respondents
Class: disciplines	precision	Class: count	systems
Class: network	technique	Class: abstracted	interactive
Class: classified	scientists	Class: acquisitions	citations
Class: medlars	recon	Class: manual	million
Class: annual	indexing	Class: substances	view
Class: compounds	law	Class: entries	librarians
Class: group	memory	Class: manuscripts	terms
Class: international	registry	Class: membership	publications
Class: rules	search	Class: activities	wants
Class: cas	mathematical	Class: empirical	include
Class: cataloguing	cost	Class: full	observation
Class: class	lending	Class: detailed	obsolescence

Figure 3: Negative Clusters for the CISI collection

The evaluation component of the clustering algorithm was specifically designed to evaluate the clusters identified in a given collection in the context of a search and retrieval application. Evaluation thus requires several input files as described in the table 4. These components are readily available for the MED, CRAN and CISI collections. For each corpus, the term-document matrix was loaded, as was the term list. For each query, a list of terms appearing in the query was produced using a simple term extraction algorithm (e.g., no stemming, etc.). The list of related documents for the query was formed by identifying the documents containing the query terms (using the term-document matrix). This list was then compared to the truth set from the gold standard and a baseline precision and recall was calculated using the formulas described above. For some of the queries in these collections, no documents are considered relevant. These queries were removed from our study. After removing these queries, 76 queries remained for the CISI collection, 30 for MED, and 152 for CRAN.

During the evaluation phase, the positive clusters (e.g., see Figure 2) were used to expand the queries. For example, if a query contained term 5, and a positive cluster containing terms 5, 8, and 10 was identified, the query would be expanded to include terms 5 and 10. It was our expectation that this query expansion would result in an improvement in recall, and the results are consistent with this hypothesis (see section 4.2). The negative clusters were used to limit the result set. If terms 5 and 18 formed a negative cluster, the query would be run using term 5 and any documents containing term 18 would be removed from the result set. Use of the negative clusters results in an improvement in precision for the queries (section 4.1).

The positive and negative clusters were evaluated separately, and evaluation of the clusters was performed using two different approaches. First, the set of clusters produced by each distance threshold was evaluated. Second, the union of the

cluster sets for all thresholds was performed and evaluated. The performance results were compared against the baseline performance for each query. As will be seen in this section, improvements in performance (as measured by precision and recall) were identified with each approach.

Table 4: Components needed to evaluate the positive and negative clusters

Input File	Purpose
Term list for collection	List of words that appear in the collection
Doc term list	Matrix that indicates which terms appear in which documents
Queries	Numbered queries
Truth Set	List of queries and the documents that are relevant to each query
Clusters	List of terms that appear in each cluster. If a term does not appear in this file, then its cluster consists of only one term.

4.1 Impact of Negative Clusters on Precision

As mentioned above, one approach to cluster evaluation was to develop the union of all clusters that were created for a collection, regardless of threshold used. In this case, we occasionally had terms appearing in more than one cluster (for example, *information* and *systems* formed a cluster for one threshold, and *information retrieval*, *system* and *systems* formed a cluster for a different threshold). In this case, we used the larger cluster for our evaluation.

The results of our studies were analyzed using a paired two-sample student's t-test designed to determine whether the baseline and experimental means were distinct. This form of the t-test does not assume that the variances of both populations are equal. A paired test is used when there is a natural pairing of observations in the samples, such as when a sample group is tested twice – such as modifying and rerunning a query and comparing the results.

The results of this evaluation for the negative clusters are shown in Table 5. The approach improved the precision of a large number of queries in all three collections. The precision of the CISI collection was improved by an average of 4.19%. The student's t-test determined that this difference was significant at the 99% level. The average precision improvement for the MED collection was determined to 6.07%, and the average precision improvement for CRAN decreased by 1.26%. The MED and CRAN means were not significantly different from the baseline.

Table 5: Impact of Negative Clusters on Precision, when union of all clusters was used

Collection	Number Queries Impacted	Number Queries Improved	Number Queries Worsened	% Impr (average)	Max % Impr	Min % Impr	Sig at 95%	Sig at 99%
CISI	73	60	13	4.19	18.95	-62.31	Yes	Yes
MED	19	13	6	6.07	116.67	-100	No	No
CRAN	94	69	25	-1.26	155.38	-100	No	No

Next we evaluated each cluster for a threshold separately. The results for the CRAN collection are shown in Table 6. Similar results were obtained for MED and CISI. Notice that, although the average precision for the CRAN collection actually decreased, there are several sets of clusters that significantly improved the performance for the collection (.14, .16, .23, .24, .25). The data in tables 5 and 6 shows that the negative clusters can be used to improve precision in a search and retrieval

application. More analysis and more testing with larger collections will allow us to optimize the parameters needed to create effective negative clusters of terms.

Table 6: Impact of Negative Clusters on Precision when single clustering was used, CRAN Results (selected records)

Threshold Cutoff	Number Queries Impacted	Number Queries Improved	Number Queries Worsened	% Impr (average)	Max % Impr	Min % Impr	Sig at 95%?	Sig at 99%
-.14	8	8	0	5.08	20.97	1.64	Yes	Yes
-.15	9	7	2	26.40	96.58	-19.50	No	No
-.16	12	10	2	22.44	96.58	-23.71	Yes	No
-.17	12	9	3	22.78	96.58	-23.71	No	No
-.18	12	9	3	22.78	96.58	-23.71	No	No
-.19	15	9	6	-5.38	24.21	-61.32	No	No
-.20	22	15	7	.21	25.59	-61.32	No	No
-.21	13	8	5	-6.48	17.18	-61.32	No	No
-.22	28	18	10	13.16	198.44	-46.95	No	No
-.23	32	23	9	12.65	198.44	-100.00	Yes	No
-.24	39	32	7	17.00	198.44	-44.26	Yes	Yes
-.25	33	28	5	19.48	198.44	-40.74	Yes	Yes
-.26	33	26	7	.01	41.01	-100.00	No	No
-.27	24	18	6	-4.50	41.01	-100.00	No	No
-.28	24	17	7	1.21	41.01	-100.00	No	No
-.29	21	12	9	-2.86	41.01	-100.00	No	No
-.30	17	11	6	-4.65	31.38	-100.00	No	No
-.31	16	9	7	-7.79	31.38	-100.00	No	No
-.34	23	13	10	-4.06	31.38	-100.00	No	No
-.35	20	10	10	-5.91	31.38	-100.00	No	No
-.36	6	4	2	-6.69	31.38	-100.00	No	No
-.37	5	3	2	-8.73	31.38	-100.00	No	No
-.38	4	2	2	-12.28	31.38	-100.00	No	No
-.39	3	1	2	-24.29	31.38	-100.00	No	No
-.48	0	0	0	n/a	n/a	n/a	No	No

4.2 Impact of Positive Clusters on Recall/Precision

Table 7 reports the impact of the positive clusters on retrieval performance. As expected, the positive clusters significantly improve recall for all three collections, and, in the case of MED and CRAN, improve the recall without severely degrading the precision. Again more study on larger collections is warranted. Similar results were obtained when the positive clusters for each threshold were evaluated.

Table 7: Impact of Positive Clusters on Recall/Precision, when union of all clusters was used

Collection	Number Queries Impacted	Number Queries Recall Improved	Number Queries Precision Worsened	Number Queries Precision Improved	% Impr Recall (average)	Sig at 95%?	Sig at 99%?	% Impr Precision (average)	Sig at 95%	Sig at 99%
CISI	71	37	62	9	17.84	Yes	Yes	-7.72	Yes	Yes
MED	17	7	14	3	2.87	Yes	Yes	-5.00	No	No
CRAN	88	10	78	10	3.42	Yes	Yes	-1.43	No	No

5. CONCLUSIONS

In this paper we have presented an algorithm for developing clusters of terms based on the LSI term-term matrix and the concept of an equivalence relation. This is the first work to interpret the LSI term-term matrix as a similarity measure and to use it in a term clustering algorithm.

Furthermore, we have developed clusters based similarity relationships as well as on anti-similarity relationships. Both types of clusters can be used to improve retrieval performance in a search and retrieval application. We have not found any other approaches that use an unsupervised term clustering algorithm to produce anti-similarity clusters. We have shown that these clusters exist, can be used to improve precision, and can be formed by applying an equivalence relation to the term-term similarity weight. Thus, we have provided a sound mathematical, theoretical foundation for these results.

Further study of the clusters produced by our algorithm is underway. In time we will identify the parameters that produce the best clusters. We also plan to use these clusters to improve performance in other applications, such as emerging trend detection [Kontostathis, et al - 2002]. These preliminary results, though promising, are in the process of being validated with a much larger collection: OHSUMED [Hersh, et al – 1994].

6. ACKNOWLEDGMENTS

The authors gratefully acknowledge the assistance of their colleagues in the Computer Science and Engineering Department at Lehigh University in completing this work. Co-author William M. Pottenger also gratefully acknowledges his Lord and Savior, Yeshua (Jesus) the Messiah, for His continuing guidance in and salvation of his life.

7. REFERENCES

- Berry, Michael, Theresa Do, Gavin O'Brien, Vijay Krishna, and Sowmini Varadhan. 1993. SVDPACKC (Version 1.0) User's Guide. April 1993.
- Deerwester, Scott, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6): 391-407.
- Dumais, Susan T. 1993. LSI meets TREC: A status report. *The First Text REtrieval Conference (TREC1)*, D. Harman (Ed.), National Institute of Standards and Technology Special Publication 500-207, pp. 137-152.
- Edmonds, Philip. 1997. Choosing the Word Most Typical in Context Using a Lexical Co-occurrence Network. *Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics*. Pages 507-509.
- Hersh W., C. Buckley, T. J. Leone, and D. Hickam. *OHSUMED: An Interactive Retrieval Evaluation and New Large Test Collection for Research*. In Proc. ACM SIGIR '9, pages 192-201, 1994.
- Lewis, David D and W. Bruce Croft. 1990. Term Clustering of Syntactic Phrases. *Proc. of the Thirteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*
- Kontostathis, Galitsky, Roy, Pottenger and Phelps. "An overview of Emerging Trends Detection (ETD)", Lecture Notes in Computer Science, Springer-Verlag, 2002.
- Kontostathis, April and William M. Pottenger. 2002a. Transitivity and the Co-occurrence Relation in LSI. *Lehigh University Technical Report. LU-CSE-02-005*. <http://www.cse.lehigh.edu/techreports/>
- Kontostathis, April and William M. Pottenger. 2002b. A Mathematical view of Latent Semantic Indexing: Tracing Term Co-occurrences. *Lehigh University Technical Report. LU-CSE-02-006*. <http://www.cse.lehigh.edu/techreports/>
- Mark Sanderson, Bruce Croft, Deriving concept hierarchies from text, *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, p.206-213, August 15-19, 1999, Berkeley, California, United States
- Schütze, Hinrich. 1992. Dimensions of Meaning. *In Proceedings of Supercomputing '92*.
- Schütze, Hinrich. 1998. Automatic Word Sense Disambiguation. *Computational Linguistics*, Volume 24, number 1.
- SIGIR2000. *Workshop on Mathematical/Formal Methods in Information Retrieval*. Chairs: Sandor Dominich, Mounia Lalmas, Keith van Rijsbergen. July 28, 2000. http://www.acm.org/sigir/forum/S2000/FM_report.pdf
- Karen Sparck Jones. Automatic Keyword Classification for Information Retrieval. Archon Books, 1971.

- Karen Sparck Jones. Collection properties influencing automatic term classification performance. *Information Storage and Retrieval*, 9:499-513, 1973.
- Veling, Anne and Peter van der Weerd. 1999. Conceptual grouping in word co-occurrence networks. *Proceedings of the IJCAI '99*. Volume 2. Pages 694-699.
- Xu, Jinxi and W. Bruce Croft. 1998. Corpus-Based Stemming Using Co-occurrence of Word Variants. In *ACM Transactions on Information Systems*, Volume 16, No 1. January 1998. Pages 61-81.
- C. T. Yu , G. Salton , M. K. Siu, Effective Automatic Indexing Using Term Addition and Deletion, *Journal of the ACM (JACM)*, v.25 n.2, p.210-225, April 1978