A Framework for Understanding LSI Performance

April Kontostathis and William M. Pottenger

Lehigh University
19 Memorial Drive West
Bethlehem, PA 18015
apk5@lehigh.edu
billp@cse.lehigh.edu

Abstract. In this paper we present a theoretical model for understanding the performance of LSI search and retrieval applications. Many models for understanding LSI have been proposed. Ours is the first to study the values produced by LSI in the term dimension vectors. The framework presented here is based on term co-occurrence data. We show a strong correlation between second order term co-occurrence and the values produced by the SVD algorithm that forms the foundation for LSI. We also present a mathematical proof that the SVD algorithm encapsulates term co-occurrence information.

1 Introduction

Latent Semantic Indexing (LSI) (Deerwester, et al., 1990) is a well-known text-mining algorithm. LSI has been applied to a wide variety of learning tasks, such as search and retrieval (Deerwester, et al.), classification (Zelikovitz and Hirsh, 2001) and filtering (Dumais, 1994, 1995). LSI is a vector space approach for modeling documents, and many have claimed that the technique brings out the 'latent' semantics in a collection of documents (Deerwester, et al., 1990; Dumais, 1993).

LSI is based on well known mathematical technique called Singular Value Decomposition (SVD). The algebraic foundation for Latent Semantic Indexing (LSI) was first described in (Deerwester, et al., 1990) and has been further discussed in (Berry, Dumais and O'Brien, 1995; Berry, Drmac, and Jessup, 1999). These papers describe the SVD process and interpret the resulting matrices in a geometric context. The SVD, truncated to k dimensions, gives the best rank-k approximation to the original matrix. In (Wiemer-Hastings, 1999), Wiemer-Hastings shows that the power of LSI comes primarily from the SVD algorithm.

Other researchers have proposed theoretical approaches to understanding LSI. (Zha, 1998) describes LSI in terms of a subspace model and proposes a statistical test for choosing the optimal number of dimensions for a given collection. (Story, 1996) discusses LSI's relationship to statistical regression and Bayesian methods. (Ding, 1999) constructs a statistical model for LSI using the cosine similarity measure.

Although other researchers have explored the SVD algorithm to provide an understanding of SVD-based information retrieval systems, to our knowledge, only Schütze has studied the values produced by SVD (Schütze, 1992). We expand upon this work, showing here that SVD exploits higher order term co-occurrence in a collection. Our work provides insight into the origin of the values in the term-term matrix.

This work provides a model for understanding LSI. Our framework is based on the concept of term co-occurrences. Term co-occurrence data is implicitly or explicitly used for almost every advanced application in textual data mining.

This work is the first to study the values produced in the SVD term by dimension matrix and we have discovered a correlation between the performance of LSI and the values in this matrix. Thus we have discovered the basis for the claim that is frequently made for LSI: LSI emphasizes underlying semantic distinctions (latent semantics) while reducing noise in the data. This is an important component in the theoretical foundation for LSI.

In section 2 we present a simple example of higher order term co-occurrence in SVD. In section 3 we present our analysis of the values produced by SVD. Section 4 presents a mathematical proof of term transitivity within SVD, previously reported in (Kontostathis and Pottenger, 2002b).

2 Co-occurrence in LSI – An Example

The data for the following example is taken from (Deerwester, et al., 1990). In that paper, the authors describe an example with 12 terms and 9 documents. The term-document matrix is shown in table 1 and the corresponding term-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 the rank of the term by document matrix. 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 purpose of dimensionality reduction is to reduce "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 re-computed using the formula $T_kS_k(T_kS_k)^T$. The term-term matrix, after reduction to 2 dimensions, is shown in table 3.

Table 1. Deerwester Term by Document Matrix

	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

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*.

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 is an example of third order co-occurrence. In the next section we present correlation data that confirms the relationship between the term-term matrix values and the performance of LSI.

To completely understand the dynamics of the SVD process, a closer look at table 1 is warranted. We note the nine documents in the collection can be split into two subsets {C1-C5} and {M1-M4}. If the term *survey* did not appear in the {M1-M4} subset, the subsets would be disjoint. The data in table 4 was developed by changing the *survey/m4* entry to 0 in table 1, computing the decomposition of this new matrix, truncating to two dimensions and deriving the associated term-term matrix.

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

Table 4. Modified Deerwester Term by Term Matrix, Truncated to two dimensions

	human	interface	computer	user	system	response	time	EPS	Survey	trees	graph	minors
human	0.56	0.50	0.60	1.01	1.62	0.66	0.66	0.76	0.45	-	-	-
interface	0.50	0.45	0.53	0.90	1.45	0.59	0.59	0.68	0.40	-	-	-
computer	0.60	0.53	0.64	1.08	1.74	0.71	0.71	0.81	0.48	-	-	-
user	1.01	0.90	1.08	1.82	2.92	1.19	1.19	1.37	0.81	-	-	-
system	1.62	1.45	1.74	2.92	4.70	1.91	1.91	2.20	1.30	-	-	-
response	0.66	0.59	0.71	1.19	1.91	0.78	0.78	0.90	0.53	-	-	-
time	0.66	0.59	0.71	1.19	1.91	0.78	0.78	0.90	0.53	-	-	-
EPS	0.76	0.68	0.81	1.37	2.20	0.90	0.90	1.03	0.61	-	-	-
Survey	0.45	0.40	0.48	0.81	1.30	0.53	0.53	0.61	0.36	-	-	-
trees	-	-	-	-	-	-	-	-	-	2.05	2.37	1.65
graph	-	-	-	-	-	-	-	-	-	2.37	2.74	1.91
minors	-	-	-	-	-	-	-	-	-	1.65	1.91	1.33

Notice the segregation between the terms; all values between {*trees*, *graph*, *minors*} subset and the rest of the terms have been reduced to zero. In the section 4 we prove a theorem that explains this phenomena, showing, in all cases, that if there is no connectivity path between two terms, the resultant value in the term-term matrix must be zero.

3 Analysis of the LSI Values

In this section we expand upon the work in (Kontostathis and Pottenger, 2002b). The results of our analysis show a strong correlation between the values produced by the SVD process and higher order term co-occurrences. In the conclusion we discuss the practical applications of this analytical study.

We chose six collections for our study of the values produced by SVD. These collections are described in Table 5. These collections were chosen because they have query and relevance judgment sets that are readily available.

Table 5. Characteristics of collections used in study

Identifier	Description	No. of Docs	No. of Terms	No. Queries	Optimal k	Values of k used in study
MED	Medical Abstracts	1033	5831	30	40	10,25,40,75, 100,125,150,200
CISI	Information Science Abstracts	1460	5143	76	40	10,25,40,75, 100,125,150,200
CACM	Communications of the ACM Abstracts	3204	4863	52	70	10,25,50,70, 100,125,150,200
CRAN	Cranfield Collection	1398	3931	225	50	10,25,50,75, 100,125,150,200
LISA	Library and Information Science Abstracts	6004	18429	35	165	10,50,100,150, 165,200,300,500
NPL	Larger collection of very short documents	11429	6988	93	200	10,50,100,150, 200,300,400,500

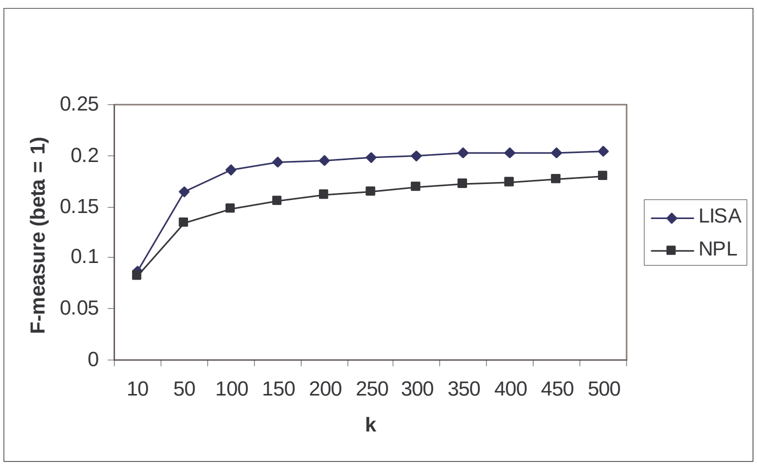

Figure 2. LSI Performance for LISA and NPL

The Parallel General Text Parser (PGTP) (Martin and Berry, 2001) was used to preprocess the text data, including creation and decomposition of the term document matrix. For our experiments, we applied the log entropy weighting option and normalized the document vectors.

We were interested in the distribution of values for both optimal and sub-optimal parameters for each collection. In order to identify the most effective *k* (dimension truncation parameter) for LSI, we used the f-measure, a combination of precision and recall (van Rijsbergen, 1979), as a determining factor. In our experiments we used a beta=1 for the f-measure parameter. We explored possible values from *k*=10, incrementing by 5, up to *k*=200 for the smaller collections, values up to *k*=500 were used for the LISA and NPL collections. For each value of *k*, precision and recall averages were identified for each rank from 10 to 100 (incrementing by 10), and the resulting f-measure was calculated. The results of these runs for selected values of k are summarized in figures 2 and 3. Clearly MED has the best performance overall. To choose the optimal *k*, we selected the smallest value that provided substantially similar performance as larger *k* values. For example, in the LISA collection *k*=165 was chosen as the optimum because the *k* values higher than 165 provide only slighter better performance. A smaller *k* is preferable to reduce the computational overhead of both the decomposition and the search and retrieval processing. The optimal *k* was identified for each collection and is shown in table 5.

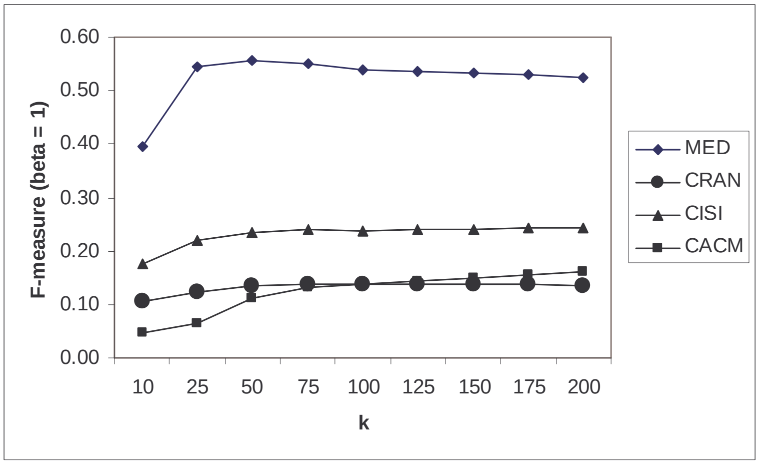

Figure 3. LSI Performance for Smaller Collections

Create the term by document matrix
Compute the SVD for the matrix
For each pair of terms (i,j) in the collection
    Compute the term-term matrix value for the (i,j) element after truncation to *k* dimensions
    Compute the cosine similarly value for the (i,j) element after truncation to *k* dimemsions
    Determine the 'order of co-occurrence'
      If term i and term j appear in same document (co-occur), 'order of co-occurrence' = 1
      If term i and term j do not appear in same document, but i co-occurs with m,
          and m co-occurs with j, then 'order of co-occurrence' = 2
      (Higher orders of co-occurrence are computed in a similar fashion by induction on the number of
          intermediate terms)
Summarize the data by range of values and order of co-occurrence

Figure 4. Algorithm for data collection

3.1 Methodology

The algorithm used to collect the co-occurrence data appears in figure 4. After we compute the SVD using the original term by document matrix, we calculate term-term similarities. LSI provides two natural methods for describing term-term similarity. First, the term-term matrix can be created using $T_kS_k(T_kS_k)^T$. This approach results in values such as those shown in table 3. Second, the term by dimension (T_kS_k) matrix can be used to compare terms using a vector distance measure, such as cosine similarity. In this case, the cosine similarity is computed for each pair of rows in the T_kS_k matrix. The computation results in a value in the range [-1, 1] for each pair of terms (*i*,*j*).

After the term similarities are created, we need to determine the order of co-occurrence for each pair of terms. The order of co-occurrence is computed by tracing the co-occurrence paths. In figure 5 we present an example of this process. In this small collection, terms A and B appear in document D1, terms B and C appear in document D2, and terms C and D occur in document D3. If each term is considered a node in a graph, arcs can be drawn between the terms that appear in the same document. Now we can assign order of co-occurrence as follows: nodes that are connected are considered first order pairs, nodes that can be reached with one intermediate hop are second order co-occurrences, nodes that can be reached with two intermediate hops are third order pairs, etc. In general the order of co-occurrence is n + 1, where n is the number of hops needed to connect the nodes in the graph. Note that some term pairs may not have a connecting path; the LSI term by term matrix for this situation is shown in table 4, the term by term entry will always be zero for terms that do not have a connectivity path.

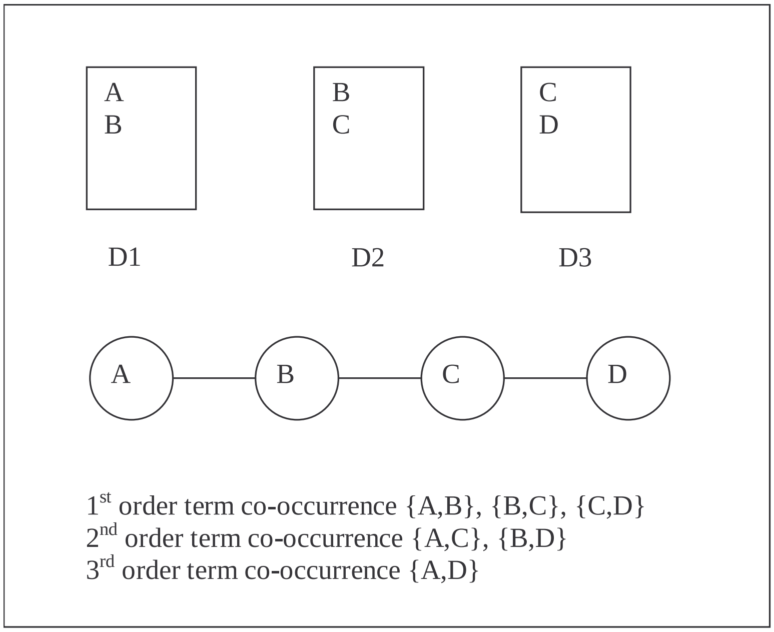

Figure 5. Tracing order of co-occurrence

3.2 Results

The order of co-occurrence summary for the NPL collection is shown in table 6. The values are expressed as a percentage of the total number of pairs of first, second and third order co-occurrences for each collection. The values in table 6 represent the distribution using the cosine similarity. LSI performance is also shown.

Table 7a shows the correlation coefficient for all collections. There is a strong correlation between the percentage of second order negative values and LSI performance for all collections, with the correlations for MED appearing slightly weaker than the other collections. There also appears to be a strong inverse correlation between the positive second and third order values and the performance of LSI. In general the values for each order of co-occurrence/value pair appear to be consistent across all collections, with the exception of the third order negative values for CACM.

Table 6. Distribution summary by sign and order of co-occurrence for NPL

	k=10	k=50	k=100	k=150	k=200	k=300	k=400	k=500
1stOrder								
[-1.0,-.01]	1.4%	2.5%	3.0%	3.1%	3.1%	2.8%	2.2%	1.6%
(-.01,.01)	0.5%	1.7%	2.9%	4.1%	5.0%	6.7%	8.1%	9.4%
[.01, 1.0]	98.1%	95.7%	94.1%	92.8%	91.8%	90.5%	89.7%	89.0%
2nd Order								
	k=10	k=50	k=100	k=150	k=200	k=300	k=400	k=500
[-1.0,-.01]	14.0%	24.6%	28.2%	30.1%	31.2%	32.1%	32.2%	31.8%
(-.01,.01)	2.4%	6.7%	9.9%	12.4%	14.7%	18.9%	22.7%	26.4%
[.01, 1.0]	83.6%	68.6%	61.9%	57.5%	54.2%	49.0%	45.1%	41.7%
3rd Order								
	k=10	k=50	k=100	k=150	k=200	k=300	k=400	k=500
[-1.0,-.01]	44.6%	62.7%	66.9%	69.3%	70.0%	69.7%	68.3%	66.6%
(-.01,.01)	3.9%	8.8%	11.6%	13.5%	15.1%	18.1%	21.0%	23.6%
[.01, 1.0]	51.5%	28.5%	21.6%	17.2%	14.9%	12.2%	10.7%	9.8%
LSI Performance Beta=1								
	k=10	k=50	k=100	k=150	k=200	k=300	k=400	k=500
F-measure	0.08	0.13	0.15	0.15	0.16	0.17	0.17	0.18

Table 7a and 7b. Within Collection Correlation data

Correlation Data for Cosine Similarity
Correlation Coefficients

	1st	2nd	3rd
NPL Cosine			
[-1.0,-.01]	0.38	0.99	0.92
(-.01,.01)	0.88	0.89	0.93
[.01, 1.0]	-0.95	-0.98	-1.00
LISA Cosine			
[-1.0,-.01]	0.55	0.99	0.99
(-.01,.01)	0.79	0.77	0.84
[.01, 1.0]	-0.93	-0.92	-0.97
CRAN Cosine			
[-1.0,-.01]	0.91	0.94	0.97
(-.01,.01)	0.77	0.76	0.78
[.01, 1.0]	-0.88	-0.88	-0.96
CISI Cosine			
[-1.0,-.01]	0.79	0.93	0.99
(-.01,.01)	0.82	0.78	0.77
[.01, 1.0]	-0.92	-0.89	-0.96
MED Cosine			
[-1.0,-.01]	0.71	0.76	0.80
(-.01,.01)	0.52	0.48	0.52
[.01, 1.0]	-0.68	-0.66	-0.74
CACM Cosine			
[-1.0,-.01]	0.99	0.98	-0.21
(-.01,.01)	0.92	0.93	0.93
[.01, 1.0]	-0.96	-0.96	-0.94

Correlation Data for Term-Term Similarity
Correlation Coefficients

	1st	2nd	3rd
NPL TermTerm			
(-9999,-.001]	0.90	0.99	0.99
(-.001,.001)	-0.87	-0.99	-0.99
[.001, 9999)	-0.90	-0.99	-0.99
LISA TermTerm			
(-9999,-.001]	0.90	0.95	0.99
(-.001,.001)	-0.89	-0.95	-1.00
[.001, 9999)	-0.89	-0.95	-0.97
CRAN TermTerm			
(-9999,-.001]	0.90	0.75	0.59
(-.001,.001)	-0.86	-0.73	-0.59
[.001, 9999)	0.81	-0.87	0.46
CISI TermTerm			
(-9999,-.001]	0.89	0.76	0.72
(-.001,.001)	-0.87	-0.77	-0.72
[.001, 9999)	0.84	0.82	0.68
MED TermTerm			
(-9999,-.001]	0.86	0.88	0.82
(-.001,.001)	-0.96	-0.88	-0.82
[.001, 9999)	0.96	0.86	0.87
CACM TermTerm			
(-9999,-.001]	0.98	0.91	0.93
(-.001,.001)	-0.95	-0.88	-0.92
[.001, 9999)	0.94	0.62	0.90

The corresponding data using the term-term similarity as opposed to the cosine similarity is shown in table 7b. In this data we observe consistent correlations for negative and zero values across all collections, but there are major variances in the correlations for the positive values.

Table 8a and 8b. Correlation data by value distribution only

Correlation Coefficients for Cosine Similarity						
Similarity	**NPL**	**LISA**	**CISI**	**CRAN**	**MED**	**CACM**
(-.3,-.2]	-0.74	-0.54	-0.09	-0.28	-0.39	0.76
(-.2,-.1]	0.97	0.96	0.86	0.89	0.92	0.82
(-.1,-.01]	0.78	0.77	0.78	0.76	0.81	0.75
(-.01,.01)	0.98	0.98	0.88	0.90	0.93	0.83
[.01,.1]	-0.36	-0.14	0.25	0.29	-0.21	0.91
(.1,.2]	-0.85	-0.81	-0.59	-0.64	-0.77	0.36
(.2,.3]	-0.98	-0.99	-0.85	-0.90	-0.92	-0.17
(.3,.4]	-0.99	-0.99	-0.97	-0.99	-0.98	-0.48
(.4,.5]	-0.98	-0.97	-1.00	-1.00	-1.00	-0.69
(.5,.6]	-0.96	-0.96	-0.98	-0.97	-0.99	-0.87

Correlation Coefficients for TermTerm Similarity						
Similarity	**NPL**	**LISA**	**CISI**	**CRAN**	**MED**	**CACM**
(-.02,-.01]	0.87	0.71	0.73	0.75	0.47	0.92
(-.01,-.001]	1.00	0.96	0.73	0.76	0.40	0.91
(-.001,.001)	-0.99	-0.95	-0.73	-0.74	-0.41	-0.89
[.001,.01]	-0.99	-0.95	-0.66	-0.93	0.41	0.22
(.01,.02]	0.35	0.93	0.72	0.82	0.50	0.92
(.02,.03]	0.52	0.79	0.69	0.80	0.44	0.93
(.03,.04]	0.95	0.72	0.73	0.78	0.44	0.93
(.04,.05]	0.87	0.69	0.71	0.74	0.45	0.94
(.05,.06]	0.86	0.69	0.72	0.75	0.46	0.94
(.06,.07]	0.84	0.68	0.73	0.78	0.49	0.95
(.07,.08]	0.82	0.68	0.66	0.75	0.51	0.96
(.08,.09]	0.83	0.70	0.69	0.80	0.52	0.95
(.09,.1)	0.84	0.71	0.69	0.77	0.55	0.96
[.1, 9999]	0.87	0.81	0.73	0.84	0.64	0.97

Table 8a shows the values when the correlation coefficient is computed for selected ranges of the cosine similarity, without taking order of co-occurrence into account. Again we note strong correlations for all collections for value ranges (-.2,-1], (-.1,-.01] and (-.01,.01).

Table 8b shows the values when the correlation coefficient is computed for selected ranges of the term-term similarity, without taking order of co-occurrence into account. These results are more difficult to interpret. We see some similarity in the (-.2,-1], (-.1,-.01] and (-.01,.01) ranges for all collections except MED. The positive values do not lend weight to any conclusion. NPL and CACM show strong correlations for some ranges, while the other collections report weaker correlations.

Our next step was to determine if these correlations existed when the distributions and LSI performance were compared across collections. Two studies were done, one holding k constant at k=100 and the second using the optimal k (identified in table 5) for each collection. Once again we looked at both the cosine and the term term similarities. Table 9 shows the value distribution for the cosine similarity for k=100. The correlation coefficients for the cross collection studies are shown in table 10. Note the correlation between the second order negative and zero values and the LSI performance, when k=100 is used. These correlations are not as strong as the correlations obtained when comparing different values of k within a single collection, but finding any similarity across these widely disparate collections is noteworthy. The cross collection correlation coefficients for optimal k values (as defined in table 5) are also shown in table 10. There is little evidence that the distribution of values has an impact on determining the optimal value of k, but there is a correlation between the distribution of cosine similarity values and the retrieval performance at *k*=100.

Table 9. Cross collection distribution by sign and order of co-occurrence, cosine similarity, *k*=100

			1st Order			
	CACM	**MED**	**CISI**	**CRAN**	**LISA**	**NPL**
[-1.0,-.01]	1.8%	1.9%	2.6%	2.5%	2.3%	3.0%
(-.01,.01)	1.9%	1.9%	2.3%	2.6%	2.1%	2.9%
[.01, 1.0]	96.3%	96.2%	95.0%	95.0%	95.6%	94.1%
			2nd Order			
	CACM	**MED**	**CISI**	**CRAN**	**LISA**	**NPL**
[-1.0,-.01]	21.3%	35.0%	31.7%	31.2%	28.7%	28.2%
(-.01,.01)	7.8%	11.4%	9.2%	10.6%	8.5%	9.9%
[.01, 1.0]	71.0%	53.6%	59.1%	58.2%	62.8%	61.9%
			3rd Order			
	CACM	**MED**	**CISI**	**CRAN**	**LISA**	**NPL**
[-1.0,-.01]	55.6%	75.0%	77.3%	72.8%	69.9%	66.9%
(-.01,.01)	17.3%	9.9%	8.7%	12.1%	10.3%	11.6%
[.01, 1.0]	27.1%	15.1%	14.0%	15.2%	19.9%	21.6%
			LSI Performance Beta=1			
	CACM	**MED**	**CISI**	**CRAN**	**LISA**	**NPL**
F-measure	0.13	0.56	0.23	0.14	0.20	0.16

Table 10a and 10b. Cross collection correlation coefficients

Cosine Similarity			
	1st	**2nd**	**3rd**
		Cosine, K=100	
[-1.0,-.01]	-0.40	0.68	0.49
(-.01,.01)	-0.47	0.63	-0.45
[.01, 1.0]	0.44	-0.69	-0.48
		Cosine, K=Optimal	
[-1.0,-.01]	-0.36	0.32	0.23
(-.01,.01)	-0.35	-0.17	-0.34
[.01, 1.0]	0.36	-0.12	0.02

Term-Term Similarity			
	1st	**2nd**	**3rd**
		TermTerm, K=100	
(-9999,-.001]	-0.53	-0.24	-0.29
(-.001,.001)	0.21	0.36	0.32
[.001, 9999)	-0.04	-0.43	-0.38
		TermTerm, K=Optimal	
(-9999,-.001]	-0.43	-0.29	-0.31
(-.001,.001)	0.48	0.36	0.31
[.001, 9999)	-0.49	-0.44	-0.32

3.3 Discussion

Our results show strong correlations between higher orders of co-occurrence in the SVD algorithm and the performance of LSI, a search and retrieval algorithm, particularly when the cosine similarity metric is used. Higher order co-occurrences play a key role in the effectiveness of many systems used for text mining. We detour briefly to describe recent applications that are implicitly or explicitly using higher orders of co-occurrence to improve performance in applications such as Search and Retrieval, Word Sense Disambiguation, Stemming, Keyword Classification and Word Selection.

Philip Edmonds shows the benefits of using second and third order co-occurrence in (Edmonds, 1997). The application described selects the most appropriate term when a context (such as a sentence) is provided. Experimental results show that the use of second order co-occurrence significantly improved the precision of the system. Use of third order co-occurrence resulted in incremental improvements beyond second order co-occurrence.

Zhang, et. al. explicitly used second order term co-occurrence to improve an LSI based search and retrieval application (Zhang, Berry and Raghavan, 2000). Their approach narrows the term and document space, reducing the size of the matrix that is input into the LSI system. The system selects terms and documents for the reduced space by first selecting all the documents that contain the terms in the query, then selecting all terms in those documents, and finally selecting all documents that contain the expanded list of terms. This approach reduces the nonzero entries in the term document matrix by an average of 27%. Unfortunately average precision also was degraded. However, when terms associated with only one document were removed from the reduced space, the number of non-zero entries was reduced by 65%, when compared to the baseline, and precision degradation was only 5%.

Hinrich Schütze explicitly uses second order co-occurrence in his paper on Automatic Word Sense Disambiguation (Schütze, 1998). In this paper, Schütze presents an algorithm for discriminating the senses of a given term. For example, the word *senses* in the previous sentence can mean the physical senses (sight, hearing, etc.) or it can mean 'a meaning conveyed by speech or writing.' Clearly the latter is a better definition of this use of *senses*, but automated systems based solely on keyword analysis would return this sentence to a query that asked about the sense of smell. The paper presents an algorithm based on use of second-order co-occurrence of the terms in the training set to create context vectors that represent a specific sense of a word to be discriminated.

Xu and Croft introduce the use of co-occurrence data to improve stemming algorithms in (Xu and Croft, 1998). The premise of the system described in this paper is to use contextual (e.g., co-occurrence) information to improve the equivalence classes produced by an aggressive stemmer, such as the Porter stemmer. The algorithm breaks down one large class for a family of terms into small contextually based equivalence classes. Smaller, more tightly connected equivalence classes result in more effective retrieval (in terms of precision and recall), as well an improved run-time performance (since fewer terms are added to the query). Xu and Croft's algorithm implicitly uses higher orders of co-occurrence. A strong correlation between terms A and B, and also between terms B and C will result in the placement of terms A, B, and C into the same equivalence class. The result will be a transitive semantic relationship between A and C. Orders of co-occurrence higher than two are also possible in this application.

In this section we have empirically demonstrated the relationship between higher orders of co-occurrence in the SVD algorithm and the performance of LSI. Thus we have provided a model for understanding the performance of LSI by showing that second-order co-occurrence plays a critical role. In the conclusion we describe the applicability of this result to applications in information retrieval.

4 Transitivity and the SVD

In this section we present mathematical proof that the SVD algorithm encapsulates term co-occurrence information. Specifically we show that a connectivity path exists for every nonzero element in the truncated matrix. This proof was first presented in (Kontostathis and Pottenger, 2002b) and is repeated here.

We begin by setting up some notation. Let A be a term by document matrix. The SVD process decomposes A into three matrices: a term by dimension matrix, T, a diagonal matrix of singular values, S, and a document by dimension matrix D. The original matrix is re-formed by multiplying the components, $A = TSD^T$. When the components are truncated to k dimensions, a reduced representation matrix, A_k is formed as $A_k = T_kS_kD_k^T$ (Deerwester et al., 1990).

The term-term co-occurrence matrices for the full matrix and the truncated matrix are (Deerwester et al., 1990):

$$B = TSST^T \quad (1)$$

$$Y = T_kS_kS_kT_k^T \quad (2)$$

We note that elements of B represent term co-occurrences in the collection, and $b_{ij} >= 0$ for all i and j. If term i and term j co-occur in any document in the collection, $b_{ij} > 0$. Matrix multiplication results in equations 3 and 4 for the ij^{th} element of the co-occurrence matrix and the truncated matrix, respectively. Here u_{ip} is the element in row i and column p of the matrix T, and s_p is the p^{th} largest singular value.

$$b_{ij} = \sum_{p=1}^{m} s_p^2 \, u_{ip} u_{jp} \quad (3)$$

$$y_{ij} = \sum_{p=1}^{k} s_p^2 \; u_{ip} u_{jp} \tag{4}$$

B^2 can be represented in terms of the T and S:

$$B^2 = (TSST^T)(\, TSST^T) \; = TSS(T^TT)SST^T = TSSSST^T \; = TS^4T^T \tag{5}$$

An inductive proof can be used to show:

$$B^n = \; TS^{2n}T^T \tag{6}$$

And the element $b_{ij}^{\,n}$ can be written:

$$b_{ij}^{\;n} = \sum_{p=1}^{m} s_p^{2n} \; u_{ip} u_{jp} \tag{7}$$

To complete our argument, we need two lemmas related to the powers of the matrix B.

LEMMA 1: Let i and j be terms in a collection, there is a transitivity path of order <= n between the terms, iff the ij^{th} element of B^n is nonzero.

LEMMA 2: If there is no transitivity path between terms i and j, then the ij^{th} element of B^n ($b_{ij}^{\,n}$) is zero for all n.

The proof of these lemmas can be found in (Kontostathis and Pottenger, 2002a). We are now ready to present our theorem.

THEOREM 1: If the ij^{th} element of the truncated term by term matrix, Y, is nonzero, then there is a transitivity path between term i and term j.

We need to show that if $y_{ij} \neq 0$, then there exists terms $q1, \ldots, qn$, $n >= 0$ such that $b_{i\,q1} \neq 0$, $b_{q1\,q2} \neq 0$, …. $b_{qn\,j} \neq 0$. Alternately, we can show that if there is no path between terms i and j, then $y_{ij} = 0$ for all k.

Assume the T and S matrices have been truncated to k dimensions and the resulting Y matrix has been formed. Furthermore, assume there is no path between term i and term j. Equation (4) represents the y_{ij} element. Assume that $s_1 > s_2 > s_3 > \ldots > s_k > 0$. By lemma 2, $b_{ij}^{\,n} = 0$ for all n. Dividing (7) by s_1^{2n}, we conclude that

$$b_{ij}^{\,n} = u_{i1}u_{j1} + \sum_{p=2}^{m} \frac{s_p^{2n}}{s_1^{2n}} \; u_{ip} u_{jp} \; = 0 \tag{8}$$

We take the limit of this equation as n à ∞, and note that $(s_p/s_1) < 1$ when $2 <= p <= m$. Then as nà ∞, $(s_p/s_1)^{2n}$ à 0 and the summation term reduces to zero. We conclude that $u_{i1}u_{j1} = 0$. Substituting back into (7) we have:

$$\sum_{p=2}^{m} s_p^{2n} \; u_{ip} u_{jp} \; = 0 \tag{9}$$

Dividing by s_2^{2n} yields:

$$u_{i2}u_{j2} + \sum_{p=3}^{m} \frac{s_p^{2n}}{s_2^{2n}} u_{ip}u_{jp} = 0. \tag{10}$$

Taking the limit as nà ∞, we have that $u_{i2}u_{j2} = 0$. If we apply the same argument k times we will obtain $u_{ip}u_{jp} = 0$ for all p such that $1 <= p <= k$. Substituting back into (4) shows that $y_{ij} = 0$ for all k.

The argument thus far depends on our assumption that $s_1 > s_2 > s_3 > \ldots > s_k$. When using SVD it is customary to truncate the matrices by removing all dimensions whose singular value is below a given threshold (Dumais, 1993); however, for our discussion, we will merely assume that, if $s_1 > s_2 > \ldots > s_{z-1} > s_z = s_{z+1} = s_{z+2} = \ldots = s_{z+w} > s_{z+w+1} > \ldots > s_k$ for some z and some $w >= 1$, the truncation will either remove all of the dimensions with the duplicate singular value, or keep all of the dimensions with this value.

We need to examine two cases. In the first case, $z > k$ and the $z \ldots z+w$ dimensions have been truncated. In this case, the above argument shows that either $u_{i\,q} = 0$ or $u_{j\,q} = 0$ for all $q <=k$ and, therefore, $y_{ij} = 0$.

To handle the second case, we assume that $z < k$ and the $z \ldots z+w$ dimensions have not been truncated and rewrite equation (7) as:

$$b_{ij}^{\,n} = \sum_{p=1}^{z-1} s_p^{2n} u_{ip}u_{jp} + \sum_{p=z}^{x+w} s_p^{2n} u_{ip}u_{jp} + \sum_{p=z+w+1}^{m} s_p^{2n} u_{ip}u_{jp} = 0 \tag{11}$$

The argument above can be used to show that $u_{ip}\, u_{jp} = 0$ for $p <= z$-1, and the first summation can be removed. After we divide the remainder of the equation by s_2^{2n}:

$$b_{ij}^{\,n} = \sum_{p=z}^{x+w} u_{ip}u_{jp} + \sum_{p=z+w+1}^{m} \frac{s_p^{2n}}{s_z^{2n}} u_{ip}u_{jp} = 0 \tag{12}$$

Taking the limit as n à ∞, we conclude that $\sum_{p=z}^{x+w} u_{ip}u_{jp} = 0$, and $b_{ij}^{\,n}$ is reduced to:

$$b_{ij}^{\,n} = \sum_{p=z+w+1}^{m} s_p^{2n} u_{ip}u_{jp} = 0 \tag{13}$$

Again using the argument above, we can show that $u_{ip}\, u_{jp} = 0$ for $z+w+1 <= p <= k$. Furthermore,

$$y_{ij} = \sum_{p=1}^{z-1} s_p^{2} u_{ip}u_{jp} + \sum_{p=z}^{x+w} s_p^{2} u_{ip}u_{jp} + \sum_{p=z+w+1}^{k} s_p^{2} u_{ip}u_{jp} = 0. \tag{14}$$

And our proof is complete.■

5 Conclusions and Future Work

Higher order co-occurrences play a key role in the effectiveness of systems used for information retrieval and text mining. We have explicitly shown use of higher orders of co-occurrence in the Singular Value Decomposition (SVD) algorithm and, by inference, on the systems that rely on SVD, such as LSI. Our empirical and mathematical studies prove that term co-occurrence plays a crucial role in LSI. The work shown here will find many practical applications. Below we describe our own research activities that were directly influenced by our discovery of the relationship between SVD and higher order term co-occurrence.

Our first example is a novel approach to term clustering. Our algorithm defines term similarity as the distance between the term vectors in the T_kS_k matrix. We conclude from section 3 that this definition of term similarity is more directly correlated to improved performance than is use of the reduced dimensional term-term matrix values. Preliminary results (preprint available from the authors) show that this metric, when used to identify terms for query expansion, matches or exceeds the retrieval performance of traditional vector space retrieval or LSI.

Our second, and more ambitious, application of these results is the development of an algorithm for approximating LSI. LSI runtime performance is significantly slower than vector space performance for two reasons. First, the decomposition must be performed and it is computationally expensive. Second, the matching of queries to documents in LSI is also computationally expensive. The original document vectors are very sparse, but the document by dimension vectors used in LSI retrieval are dense, and the query must be compared to each document vector. Furthermore, the optimal truncation value (k) must be discovered for each collection. We believe that the correlation data presented here can be used to develop an algorithm that approximates the performance of an optimal LSI system while reducing the computational overhead.

6 Acknowledgements

This work was supported in part by National Science Foundation Grant Number EIA-0087977. The authors gratefully acknowledge the assistance of Dr. Kae Kontostathis and Dr. Wei-Min Huang in developing the proof of the transitivity in the SVD as well as in reviewing drafts of this article. The authors also would like to express their gratitude to Dr. Brian Davison for his comments on a draft. 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, M.W., Drmac Z. and Jessup, E.R. (1999). Matrices, Vector Spaces, and Information Retrieval. *SIAM Review* vol. 41, no. 2. pp. 335-362.

Berry, M.W., Dumais, S. T., and O'Brien, G. W. (1995). Using linear algebra for intelligent information retrieval. *SIAM Review*, vol. 37, no. 4, pp. 573-595.

Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K. and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science,* vol. 41, no. 6, pp. 391-407.

Ding, C.H.Q. (1999). A similarity-based Probability Model for Latent Semantic Indexing. *Proceedings of the 22nd ACM SIGIR '99 Conference.* pp. *59-65.*

Dumais, S.T. (1993). LSI meets TREC: A status report. In: D. Harman (Ed.), *The First Text REtrieval Conference (TREC1*), *National Institute of Standards and Technology Special Publication* 500-207, pp. 137-152.

Dumais, S.T. (1994). Latent Semantic Indexing (LSI) and TREC-2. In: D. Harman (Ed.), *The Second Text REtrieval Conference (TREC2), National Institute of Standards and Technology Special Publication 500-215* , pp. 105-116.

Dumais, S.T. (1995). Using LSI for information filtering: TREC-3 experiments. In: D. Harman (Ed.), *The Third Text REtrieval Conference (TREC3) National Institute of Standards and Technology Special Publication* 500-226. pp. 219-230.

Edmonds, P. (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*. pp. 507-509.

Kontostathis, A. and W.M. Pottenger. (2002a). A Mathematical View of Latent Semantic Indexing: Tracing Term Co-occurrences. Lehigh University Technical Report, LU-CSE-02-006.

Kontostathis, April and William M. Pottenger. (2002b). Detecting Patterns in the LSI Term-Term Matrix. *Workshop on the Foundation of Data Mining and Discovery, The 2002 IEEE International Conference on Data Mining.* pp. 243-248.

Martin, D.I. and Berry, M.W. (2001). Parallel General Text Parser. Copyright 2001.

Schütze, H. (1992). Dimensions of Meaning. *Proceedings of Supercomputing '92.*

Schütze, Hinrich. (1998). Automatic Word Sense Disambiguation. *Computational Linguistics*, Vo. 24, no. 1.

Story, R.E. (1996). An explanation of the Effectiveness of Latent Semantic Indexing by means of a Bayesian Regression Model. *Information Processing and Management,* Vol. 32 , No. 3, pp. 329-344.

van Rijsbergen, CJ. (1979). *Information Retrieval*. Butterworths, London.

Wiemer-Hastings, P. (1999). How Latent is Latent Semantic Analysis? *Proceedings of the 16th International Joint Conference on Artificial Intelligence*, pp. 932-937.

Xu, J. and Croft, W.B. (1998). Corpus –Based Stemming Using Co-occurrence of Word Variants. In *ACM Transactions on Information Systems*, Vol. 16, No 1. pp. 61-81.

Zelikovitz, S. and Hirsh, H. (2001). Using LSI for Text Classification in the Presence of Background Text. *Proceedings of CIKM-01, 10th ACM International Conference on Information and Knowledge Management.*

Zha, H. (1998). A Subspace-Based Model for Information Retrieval with Applications in Latent Semantic Indexing, Technical Report No. CSE-98-002, Department of Computer Science and Engineering, Pennsylvania State University.

Zhang, X., Berry M.W. and Raghavan, P. (2000). Level search schemes for information filtering and retrieval. *Information Processing and Management*. Vo. 37, No. 2. pp. 313-334.