

Essential Dimensions of Latent Semantic Indexing (LSI)

April Kontostathis
 Department of Mathematics and Computer Science
 Ursinus College
 Collegeville, PA 19426
 Email: akontostathis@ursinus.edu

Abstract

Latent Semantic Indexing (LSI) is commonly used to match queries to documents in information retrieval applications. LSI has been shown to improve retrieval performance for some, but not all, collections, when compared to traditional vector space retrieval. In this paper, we first develop a model for understanding which values in the reduced dimensional space contain the term relationship (latent semantic) information. We then test this model by developing a modified version of LSI that captures this information, Essential Dimensions of LSI (EDLSI). EDLSI significantly improves retrieval performance on corpora that previously did not benefit from LSI, and offers improved runtime performance when compared with traditional LSI. Traditional LSI requires the use of a dimensionality reduction parameter which must be tuned for each collection. Applying our model, we have also shown that a small, fixed dimensionality reduction parameter ($k=10$) can be used to capture the term relationship information in a corpus.

1 Introduction

Latent Semantic Indexing (LSI) has been applied to a wide variety of learning tasks involving textual data. LSI is often applied for tasks such as search and retrieval [9, 12], classification [30], and filtering [12, 13]. LSI is a dimensionality reduction approach for modeling documents. It was originally thought to bring out the ‘latent semantics’ within a corpus of documents. However, LSI does not derive ‘semantic’ information per se. It does capture higher order term co-occurrence information [19], and we prefer to state that LSI captures ‘term relationship’ information, rather than ‘latent semantic’ information.

The LSI algorithm is easy to understand and implement (see Section 2), and theoretical work on LSI is ongoing (see Section 3). This theoretical work is important, because LSI does not provide improved retrieval performance across all

corpora. A better theoretical understanding will not only provide a firmer foundation for LSI, but will also explain when use of LSI is appropriate. A natural result of this theoretical work is the development of algorithms which use just the term relationship information from LSI to improve retrieval performance.

One approach towards a better theoretical understanding of LSI is a detailed analysis of the values produced by LSI [23, 20, 19]. This paper builds upon this analysis of values and provides a solution to address three outstanding issues that remain with LSI:

1. The number of dimensions needed is typically large (100 - 300 dimensions). This has significant implications for indexing run time, query run time, and the amount of memory required.
2. The number of dimensions needed must be determined for each collection; therefore training data, in the form of standardized queries with known truth sets, is needed for each collection. This training data often does not exist for large collections.
3. LSI performs worse than traditional vector space retrieval for some collections [20, 17].

In Section 4, we develop a simple model for understanding LSI in terms of traditional vector space retrieval. We then propose an extension to LSI that is based on our model. We refer to this extension as Essential Dimensions of LSI (EDLSI). We show in Sections 5 and 6 that EDLSI addresses all three issues mentioned above:

1. Run time and memory requirements are reduced.
2. The number of dimensions needed is small and fixed across a variety of collections.
3. The retrieval performance of EDLSI is uniformly better for the seven collections we tested.

2 Background

Many algorithms for searching textual collections have been developed. This paper focuses on LSI, which is based on traditional vector space retrieval.

2.1 Traditional Vector Space Retrieval

In traditional vector space retrieval, documents and queries are represented as vectors in t -dimensional space, where t is the number of indexed terms in the collection. Generally the document vectors are formed when the index for the collection is generated. These vectors form a matrix that is often referred to as the term-by-document matrix, A . The query vector, q , is formed when a search of the collection is performed.

In order to determine the relevance of a document to the query, the query vector is multiplied by the term-by-document matrix and a results vector, w , is computed as shown in equation 1. The score in position, j of this vector provides a measure of the similarity of document d_j to q . Generally a search and retrieval system will order the scores in descending order, and return the documents to the user in this order. This provides a *ranking* of the documents for each query. The document with the highest score is given $rank = 1$.

$$w = qA \quad (1)$$

A variety of well-known and effective term weighting schemes [26] can be used when forming the document and query vectors. Term weighting can be split into three types: A *local* weight based on the frequency within the document, a *global* weight based on a term's frequency throughout the dataset, and a *normalizing* weight that negates the discrepancies of varying document lengths. The entries in the document vector are computed by multiplying the global weight for each term by the local weight for the document-term pair. Normalization may then be applied to the document and query vectors.

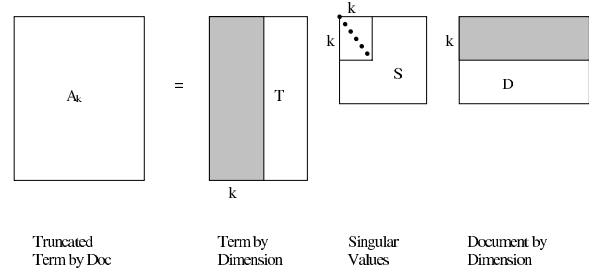
In our studies we employ a log entropy weighting scheme. This approach defines a global term weight for each indexed term, i , using the formula:

$$g_i = 1 + \left(\sum_j p_{ij} \log_2(p_{ij}) \right) / \log_2 n$$

Where n is the number of documents in the collection, the summation traverses all documents in the collection, and p_{ij} is the probability of the term i appearing in the document j and is defined as:

$$p_{ij} = f_{ij} / \left(\sum_j f_{ij} \right)$$

Figure 1. Truncation of SVD for LSI



Where f_{ij} is the number of times term i appears in document j . The local weight of a term in a document or query is then defined as:

$$l_i = \log_2(1 + f_{ij})g_i$$

The purpose of the log entropy weighting scheme is to reduce the relative importance of high frequency terms while giving words that distinguish the documents in a collection a higher weight. In our experiments we normalized each query and document vector for all collections except OHSUMED. A significant portion of the OHSUMED collection contains title data only (as opposed to titles and abstracts), and normalization results in these documents receiving higher weight when they contain a term in common with the query. This problem was previously noted in [16].

2.2 Latent Semantic Indexing

LSI is based on a mathematical technique called Singular Value Decomposition (SVD). The SVD process decomposes a term-by-document matrix, A , into three matrices: a term-by-dimension matrix, T , a singular-value matrix, S , and a document-by-dimension matrix, D . The number of dimensions is r , the rank of A . The original matrix can be obtained, through matrix multiplication of TSD^T . This decomposition is shown in equation 2.

$$A = TSD^T \quad (2)$$

In an LSI system, the T , S and D matrices are truncated to k dimensions. This truncation is accomplished by removing columns $k + 1$ to r of T , columns and rows $k + 1$ to r of S , and $k + 1$ to r of D^T . This process is shown graphically in Figure 1 (taken from [4] – the shaded areas are kept). Dimensionality reduction is thought to reduce ‘noise’ in the term-by-document matrix, resulting in a richer word relationship structure that many researchers claim reveals latent semantics present in the collection [9, 12]. Queries are represented in the reduced space by:

$$qT_k$$

where T_k is the term-by-dimension matrix, after truncation to k dimensions. Queries are compared to the reduced document vectors, scaled by the singular-values ($S_k D_k^T$). This process provides a similarity score for each document for a given query. Thus, the truncated term-by-document matrix is shown in equation 3, and the result vector, w , is produced using equation 4. As with vector space retrieval, the scores will be sorted in descending order and the system will return documents to the user in rank order.

$$A_k = T_k S_k D_k^T \quad (3)$$

$$w = qA_k \quad (4)$$

Choosing an optimal dimensionality reduction parameter (k) for each collection remains elusive. Traditionally, the optimal k has been chosen by running a set of queries with known relevant document sets for multiple values of k . The k that results in the best retrieval performance is chosen as the optimal k for each collection. Optimal k values are typically in the range of 100-300 dimensions [12, 21].

2.3 Datasets and Evaluation.

Retrieval quality for an information retrieval system can be expressed in a variety of ways. In the current work, we primarily use precision and recall to express retrieval performance. Precision is defined as the number of relevant documents returned divided by the total number of documents returned. Recall is the number of relevant documents returned divided by the total number of relevant documents.

In Section 6, we apply these metrics by computing precision at a given recall level. To calculate precision we continue to retrieve documents for a query until a given percentage of correct documents has been retrieved (for example, 30%). The average across all queries is computed to determine the average precision for a collection.

These metrics require the existence of collections that contain a group of documents, a set of standard queries and a truth set for each query (a list of truly relevant documents). We used seven such collections during the course of our study. The collections we used are summarized in Table 1. These collections were downloaded from a variety of sources. MED [7], CISI [7], CRAN [7, 8], NPL [24], and CACM [7, 14] were downloaded from the SMART web site at Cornell University. LISA [28] was obtained from the Information Retrieval Group web site at the University of Glasgow. The OHSUMED [16] collection was downloaded from the Text Retrieval Conference (TREC) web site at the National Institute of Standards and Technology. Not all

of the documents in the OHSUMED collection have been judged for relevance for each query. In our experiments, we calculate precision and recall by assuming that all unjudged documents are not relevant (which is the also the methodology used in the TREC experiments).

Table 1 also shows the optimal k for LSI processing for each collection. The optimal k was identified by measuring the average precision for each collection as we varied k from 5 to 200 (in increments of 5) for the smaller collections (MED, CISI, CRAN, CACM), and from 25 to 500 (in increments of 25) for the larger collections (NPL, LISA, OHSUMED).

3 Related Work

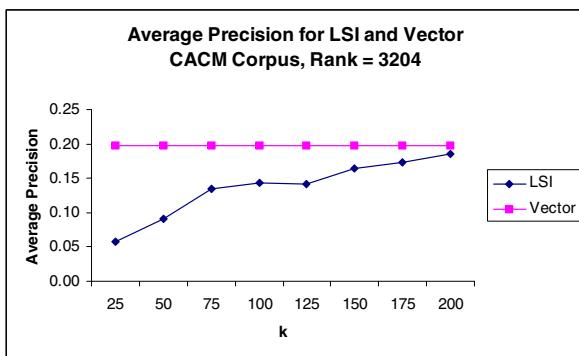
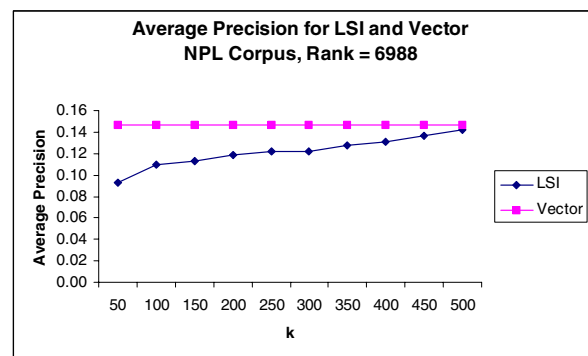
The algebraic foundation for LSI was first described in [9] and has been further discussed by Berry, et al. in [3, 4]. These papers describe the SVD process and interpret the resulting matrices in a geometric context. They show that the SVD, truncated to k dimensions, gives the optimal rank- k approximation to the original matrix. Wiemer-Hastings shows that the power of LSI comes primarily from the SVD algorithm [27].

Other researchers have proposed theoretical approaches to understanding LSI. Zha and Simon describe LSI in terms of a subspace model and propose a statistical test for choosing the optimal number of dimensions for a given collection [31]. Story discusses LSI's relationship to statistical regression and Bayesian methods [25]. Ding constructs a statistical model for LSI using the cosine similarity measure, showing that the term similarity and document similarity matrices are formed during the maximum likelihood estimation, and LSI is the optimal solution to this model [11]. Ding and He show the unsupervised dimension reduction is closely related to unsupervised learning, and use the top SVD dimensions to identify good centroid starting points for a K -means clustering algorithm [10].

Although other researchers have explored the SVD algorithm to provide an understanding of SVD-based information retrieval systems, to our knowledge Schütze was the first to study the values produced by LSI [23]. We later expanded upon this work, showing that the SVD exploits higher order term co-occurrence in a collection, and showing the correlation between the values produced in the term-term matrix and the performance of LSI [19]. We further extended these results to determine the most critical values in an LSI system [20]. In the following sections, we show that the term relationship information can be found within the first few dimensions of the SVD.

Table 1. Summary of Collections Tested

Identifier	Description	Docs	Terms	Queries	Optimal LSI k value
MED	Medical abstracts	1033	5831	30	75
CISI	Information science abstracts	1450	5143	76	145
CACM	Communications of the ACM abstracts	3204	4863	52	200
CRAN	Cranfield collection	1400	3932	225	185
LISA	Library and Information Science Abstracts	6004	18429	35	500
NPL	Larger collection of very short documents	11429	6988	93	500
OHSUMED	Clinically-oriented MEDLINE subset	348566	170347	106	500

Figure 2. LSI vs. Vector Space Retrieval for the CACM collection ($r=3204$)

Figure 3. LSI vs. Vector Space Retrieval for the NPL collection ($r=6988$)


4 Theoretical Model

Mathematically, as the truncation parameter, k , approaches the rank of the term-by-document matrix, r , LSI approaches traditional vector space retrieval. In other words, traditional vector space retrieval is equivalent to LSI when $k = r$. This follows directly from equations 1 through 4.

Figures 2-4 show graphically that the performance of LSI begins to approach the performance of traditional vector space retrieval when k is fairly small with respect to r . For CACM and NPL we see that LSI retrieval performance continues to increase as additional dimensions are added, while retrieval performance of LSI for the MED collection peaks when k is 75 and then retrieval performance begins to degrade back to the level of traditional vector space retrieval. We hypothesize that optimized traditional LSI systems capture the term relationship information within the first few dimensions and then continues to capture the traditional vector space retrieval data until k gets 'close enough' to r .

Figures 2 and 3 show that traditional vector retrieval outperforms LSI on some collections, even for relatively large values of k . Other examples of collections that do not ben-

efit from LSI can be found in [20] and [17].

Our hypothesis suggests that we can use the term relationship information, captured in the first few SVD vectors, in combination with traditional vector retrieval. In Section 6 we get good performance on a variety of collections by using only the first 10 dimensions of the SVD. In our model, we obtain document scores by computing a weighted average of the traditional LSI score computed using only the **essential dimensions**, and the traditional vector space retrieval score. The result vector computation is shown in equation 5, where x is a weighting factor ($0 \leq x \leq 1$) and k is small.

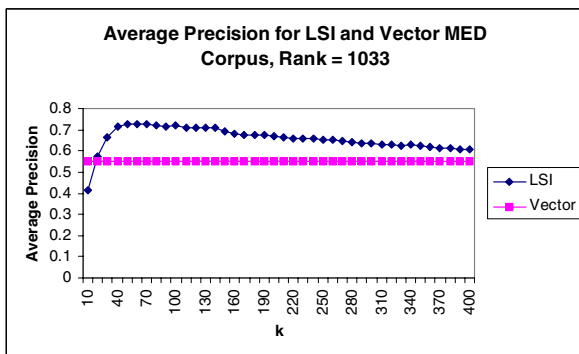
$$w = (x)(qA_k) + (1 - x)(qA) \quad (5)$$

5 Experimental Approach

In this section we describe our approach to testing this model. The results of these tests appear in Section 6.

The term-by-document matrix was created for each collection in Table 1. The terms were words, special characters were removed, a standard english stop list was applied, and stemming was not used. The log entropy weighting scheme

Figure 4. LSI vs. Vector Space Retrieval for the MED collection ($r=1033$)



was applied to the entries in both the query and the document vectors. Query and document vectors were normalized for all collections except OHSUMED.

Next the SVD was performed on the term-by-document matrix. We used the Parallel General Text Processor (PGTP) to perform the SVD [5].

Finally, each query in the set of standard queries was run using traditional vector space retrieval, standard LSI at the optimal k shown in Table 1, and EDLSI. The average precision of each system was calculated as described in the previous section.

This process was repeated using a variety of values for the two parameters: the EDLSI dimensionality reduction parameter, k_{EDLSI} , and the weighting parameter, x .

6 Experimental Results

We present our experimental results in this section.

6.1 Results

The results of our experiments when $k_{EDLSI} = 10$ and the $x = .2$ appear in Table 2. These parameter settings provide consistently good results across all collections. On average, EDLSI improved retrieval performance an average of 12% over traditional vector space retrieval. All collections showed significant improvements, ranging from 8% to 19%. The LSI term-by-document matrix contributes only 20% of the weight to the final similarity score for the results shown in Table 2.

Four of the collections were included in our studies because it has been noted previously that traditional vector space retrieval is preferred over LSI for these collections (CACM, LISA, NPL, OHSUMED) [20]. The proposed method provides an average improvement over traditional vector space retrieval of 14% for these four collections,

when $k_{EDLSI}=10$ and $x = .2$. The improvement over traditional LSI is much higher (see Table 2).

Although $k_{EDLSI} = 10$ and $x = .2$ provide good performance across all the collections, it is possible to tune the system for each collection. For example, $k_{EDLSI} = 35$ and $x = .5$ provides an average precision of .72 for MED, making this technique perform as well as LSI at the optimal setting ($k_{LSI} = 75$), with fewer dimensions. The optimal settings for each collection are shown in Table 3. To obtain this data, we varied the x from .1 to .5 (in increments of .1), and continued to increase the value of k_{EDLSI} in increments of 5, up to $k_{EDLSI} = 100$ for the larger collections and $k_{EDLSI} = 200$ for the smaller collections. The optimal k_{LSI} was also tested (i.e. $k_{EDLSI} = 500$ was tested for LISA, NPL and OHSUMED). In all cases the average precision of traditional LSI at the optimal k_{LSI} was worse than the average precision of EDLSI at the optimal k_{EDLSI} (shown in Table 3) for all weighting factors .1 thru .5. Furthermore, the optimal k_{EDLSI} was always smaller than the optimal k_{LSI} .

6.2 Discussion

The primary advantage of EDLSI is that there is little variability as the parameters change. Figures 5 and 6 give examples of how the average precision changes when dimensionality reduction parameter and the weighting factor are varied independently. In Figure 5, where x has the constant value .2, the average precision varies by only 0.0032 as k_{EDLSI} varies from 5 to 300. We see a bit more fluctuation in Figure 6 where we have set k_{EDLSI} to the constant value 10 and vary x between .1 and .5. The average precision varies by .0195 over the range of values; however, varying from .1 to .3 results in only a .0046 variation in average precision (from max to min).

A comparison of average precision in Tables 2 and 3 show that use of the optimal settings does not greatly affect the average precision on six of the seven collections. CRAN, CISI, LISA, NPL, and OHSUMED derive a small benefit from tuning the parameters, ranging from .003 to .017. CACM obtains no improvement from tuning (when precision is reported to 3 significant digits). The MED collection is the exception, it greatly benefits from tuning as the average precision moves from .583 to .772, but MED is an unusual collection that was derived by combining the results from a series of keyword searches, and is known to greatly benefit from the use of dimensionality reduction because the truncated SVD can easily re-segment the document space [9].

One common problem with information retrieval systems in general, and traditional LSI in particular, is that optimal parameter settings vary widely when different corpora are tested. Figures 2-4 show the great degree of fluctuation

Table 2. Experimental Results at $k_{EDLSI} = 10$, with the LSI matrix representing 20% of the final score

Collection	LSI Avg. Prec. (at Optimal k)	Vector Avg. Prec.	Experimental Avg. Prec.	% Impr. over LSI	% Impr. over Vector
MED	.722	.540	.583	-19%	8%
CISI	.222	.208	.230	4%	11%
CACM	.182	.206	.229	26%	11%
CRAN	.450	.398	.436	-3%	10%
LISA	.277	.302	.344	24%	14%
NPL	.142	.158	.188	32%	19%
OSHUMED	.022	.150	.170	673%	13%

Table 3. Optimal parameter settings and resultant average precision

Collection	k_{EDLSI}	Weighting Factor (x)	Average Precision	% Impr. over LSI	% Impr. over Vector
MED	35	.5	.722	0%	34%
CISI	75	.2	.243	9%	17%
CACM	10	.2	.229	26%	11%
CRAN	125	.5	.453	1%	14%
LISA	80	.3	.360	30%	19%
NPL	15	.2	.191	35%	21%
OSHUMED	95	.3	.174	691%	16%

Figure 5. The affect of varying k_{EDLSI} for OHSUMED, when weighting factor remains stable

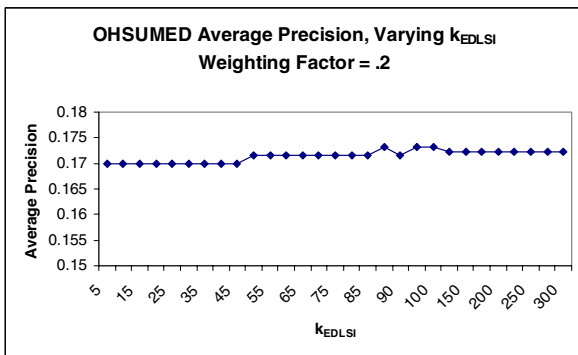
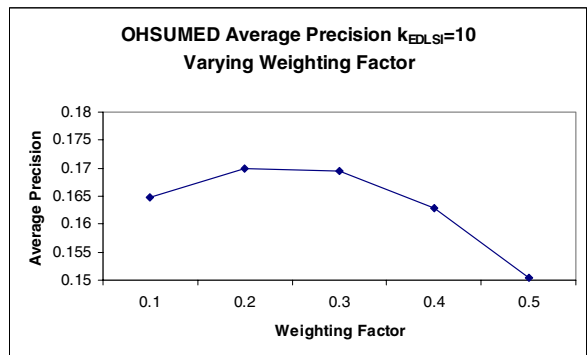


Figure 6. The affect of varying weighting factor for OHSUMED, when k_{EDLSI} remains stable



in the performance of the LSI system that results when the parameter k_{LSI} is changed. Another problem is that the use of gold standard queries and truth sets inevitably results in systems which are tuned to a particular set of queries. We contend that, for general search and retrieval applications, a hope of capturing a small improvement in average precision is not worth the disadvantage of tuning the system to a particular set of queries. Therefore, we are comfortable choosing parameters that work well for all systems. The choice of $k_{EDLSI} = 10$ and $x = .2$ will have a significant positive impact on the average precision for systems using traditional vector space retrieval.

6.3 Run Time Performance and Memory Usage

The computational complexity of SVD approximation algorithms is dependent on the number of factors required (k), the sparseness of the original term-document matrix, and the number of documents in the corpus [6]. Many approximation algorithms for computing only the k largest singular values and their associated vectors exist [1, 2]. Furthermore, the data in Figure 5 of [29] indicates that the complexity of the Lanczos algorithms is close to linear when $k < r/2$ (for the three collections tested). Thus computation of only 10 singular values and their association vectors will have significantly reduced cost when compared to the usual 100-300 dimensions required for traditional LSI.

Furthermore, the combined technique will require minimal extra memory during query run time, when compared to traditional vector space retrieval. The term and document vectors must be stored in memory, but this requires only an additional 79MB of RAM for OHSUMED when $k_{EDLSI} = 10$ ($350,000 \times 10 \times 16 = 53\text{MB}$ for the document vectors and $170,000 \times 10 \times 16 = 26\text{ MB}$ for the term vectors, assuming double precision values are stored). By contrast, a traditional LSI system with 500 dimensions would need 50 times this much, nearly 4GB of RAM. Even at the optimal k_{EDLSI} of 95, OHSUMED would need only an additional 754 MB of RAM.

We chose $k_{EDLSI} = 10$ because it was the smallest k_{EDLSI} which was optimal for any collection. A close look at the data reveals that setting $k_{EDLSI} = 5$ results in very little degradation in precision. Thus, if computational resources are severely limited, it is possible to set k_{EDLSI} to a value smaller than 10 and still obtain results superior to traditional vector space retrieval.

7 Conclusions and Future Work

This paper has contributed both practical and theoretical results to the field of information retrieval.

From a theoretical perspective, we have confirmed that LSI captures term relationship information with only a few dimensions. Future work that involves an analysis of the values produced by LSI, like the studies described in [19] and [18], can focus on fewer singular values and their associated vectors.

From a practical perspective, our Essential Dimensions of LSI technique provides substantial benefit for all collections. The retrieval performance for collections that previous did not benefit from LSI is significantly improved. If a collection is known to have improved retrieval performance using traditional LSI, we can meet or exceed LSI performance with many fewer dimensions. This will reduce indexing and query runtime costs (both CPU time and memory) compared to an optimized traditional LSI system. In the future we would like to apply this technique to other applications that have successfully used LSI or SVD, such as collaborative filtering [15] or text annotation [22].

Acknowledgements

We would like to thank Professor Brian D. Davison at Lehigh University for his invaluable comments on an early draft. This work was partially supported by the National Center for Supercomputing Applications under IRI050005 and utilized the copper system.

References

- [1] M. W. Berry. Large-scale sparse singular value computations. *The International Journal of Supercomputer Applications*, 6(1):13–49, Spring 1992.
- [2] M. W. Berry, T. Do, G. O'Brien, V. Krishna, and S. Varadhan. SVDPACKC (version 1.0) User's Guide. Technical report, 1993.
- [3] M. W. Berry, Z. Drmac, and E. R. Jessup. Matrices, vector spaces, and information retrieval. *SIAM Review*, 41(2):335–362, 1999.
- [4] M. W. Berry, S. T. Dumais, and G. W. O'Brien. Using linear algebra for intelligent information retrieval. *SIAM Review*, 37(4):575–595, 1995.
- [5] M. W. Berry and D. I. Martin. Principal component analysis for information retrieval. *Handbook of Parallel Computing and Statistics*, 2005.
- [6] E. Bingham and H. Mannila. Random projection in dimensionality reduction: applications to image and text data. In *Knowledge Discovery and Data Mining*, pages 245–250, 2001.
- [7] C. Buckley. Implementation of the SMART information retrieval system. Technical Report TR85-686, Ithaca, NY, USA, 1985.
- [8] C. Cleverdon, J. Mills, and E. Keen. Factors determining the performance of indexing systems, vol. 2: Test results. Technical report, Aslib Cranfield Research Project, Cranfield, England, 1966.

- [9] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the American Society of Information Science*, 41(6):391–407, 1990.
- [10] C. Ding and X. He. K-means clustering via principal component analysis. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, page 29, New York, NY, USA, 2004. ACM Press.
- [11] C. H. Q. Ding. A similarity-based probability model for latent semantic indexing. In *Proceedings of the Twenty-second Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*, pages 59–65, 1999.
- [12] S. T. Dumais. LSI meets TREC: A status report. In D. Harman, editor, *The First Text REtrieval Conference (TREC-1)*, National Institute of Standards and Technology Special Publication 500-207, pages 137–152, 1992.
- [13] S. T. Dumais. Latent semantic indexing (LSI) and TREC-2. In D. Harman, editor, *The Second Text REtrieval Conference (TREC-2)*, National Institute of Standards and Technology Special Publication 500-215, pages 105–116, 1994.
- [14] E. A. Fox. Characterization of two new experimental collections in computer and information science containing textual and bibliographic concepts. Technical Report 83-561, Cornell University, 1983.
- [15] D. Hersh and L. Zhukov. SVD based term suggestion and ranking system. In *Proceedings of Fourth IEEE International Conference on Data Mining (ICDM'04)*, pages 391–394, 2004.
- [16] W. Hersh, C. Buckley, T. J. Leone, and D. H. Hickam. OHSUMED: an interactive retrieval evaluation and new large test collection for research. In *Proceedings of the Seventeenth Annual ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 192–201, 1994.
- [17] E. R. Jessup and J. H. Martin. Taking a new look at the latent semantic analysis approach to information retrieval. In *Computational Information Retrieval*, pages 121–144, Philadelphia, PA, USA, 2001. Society for Industrial and Applied Mathematics.
- [18] R. N. Kenmogne. Diploma thesis, Max Planck Institut Informatik, Saarbrücken, Germany. *Understanding LSI via the Truncated Term-Term Matrix*, May 2005.
- [19] A. Kontostathis and W. M. Pottenger. A framework for understanding Latent Semantic Indexing (LSI) performance. *Information Processing and Management*, 42(1):56–73, 2006.
- [20] A. Kontostathis, W. M. Pottenger, and B. D. Davison. Identification of critical values in latent semantic indexing. In T. Lin, S. Ohsuga, C. Liau, X. Hu, and S. Tsumoto, editors, *Foundations of Data Mining and Knowledge Discovery*, pages 333–346. Springer-Verlag, 2005.
- [21] T. A. Letsche and M. W. Berry. Large-scale information retrieval with latent semantic indexing. *Information Sciences*, 100(1-4):105–137, 1997.
- [22] M. J. Martin and P. W. Foltz. Automated team discourse annotation and performance prediction using lsa. In D. M. Susan Dumais and S. Roukos, editors, *HLT-NAACL 2004: Short Papers*, pages 97–100, Boston, Massachusetts, USA, May 2 - May 7 2004. Association for Computational Linguistics.
- [23] H. Schütze. Dimensions of meaning. In *Proceedings of Supercomputing*, pages 787–796, 1992.
- [24] K. Sparck-Jones and C. A. Webster. Research in relevance weighting. British Library Research and Development Report 5553, Computer Laboratory, University of Cambridge, 1979.
- [25] R. E. Story. An explanation of the effectiveness of Latent Semantic Indexing by means of a Bayesian regression model. *Information Processing and Management*, 32(3):329–344, 1996.
- [26] C. van Rijsbergen. *Information Retrieval*. Department of Computer Science, University of Glasgow, 1979.
- [27] P. Wiemer-Hastings. How latent is latent semantic analysis? In *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 932–937, 1999.
- [28] P. Willett. http://ir.dcs.gla.ac.uk/resources/test_collections/lisa/readme. Technical report.
- [29] Y. Yang. Noise reduction in a statistical approach to text categorization. In E. A. Fox, P. Ingwersen, and R. Fidel, editors, *Proceedings of SIGIR-95, 18th ACM International Conference on Research and Development in Information Retrieval*, pages 256–263, Seattle, US, 1995. ACM Press, New York, US.
- [30] S. Zelikovitz and H. Hirsh. Using LSI for text classification in the presence of background text. In H. Paques, L. Liu, and D. Grossman, editors, *Proceedings of CIKM-01, tenth ACM International Conference on Information and Knowledge Management*, pages 113–118, Atlanta, GA, 2001. ACM Press, New York.
- [31] H. Zha and H. Simon. A subspace-based model for Latent Semantic Indexing in information retrieval. In *Proceedings of the Thirteenth Symposium on the Interface*, pages 315–320, 1998.