Incorporating Latent Semantic Indexing into a Neural Network Model for Information Retrieval

Inien Syu
Department of Computer Science
Embry-Riddle University
Daytona Beach, FL 32114-3900
syui@db.erau.edu

S. D. Lang and Narsingh Deo
Department of Computer Science
University of Central Florida
Orlando, FL 32816-0362
{lang, deo}@cs.ucf.edu

Abstract

We incorporate the Latent Semantic Indexing (LSI) technique into a competition-based neural network model for information retrieval. The original neural network model was based on a causal inference network, incorporating Roget's Thesaurus, that connects the index terms and related documents. Since the process of creating or updating a thesaurus is rather expensive, we apply the LSI technique to provide an automated procedure that captures the semantic relationship between the documents and index terms. Our experimental results using four standard text collections show that the LSI-based model generates appreciable improvement in retrieval effectiveness with faster query evaluation over the thesaurus-based model.

1 Introduction

In conventional information retrieval models, such as the Boolean models, vector space models, and probabilistic models, documents and queries are represented by a set of subject terms or keywords, sometimes with associated weights. The Boolean models determine the relationship between a document and a user query by matching the document terms and the exact combination of the search terms specified in the query. The vector space models calculate the similarity measures between documents and the user query based on the term weights which determine the degree of importance of the terms. The probabilistic models rank the documents by the probability that each document would be judged relevant to a given query [TC91]; the probability is estimated by considering the presence or absence of certain terms in the document while comparing to the terms in a user query, together with the information about term distribution in the document collection [Sal89].

Since the individual terms and keywords are not adequate discriminators of the semantic content of the documents and queries [FLGD87], the performance of the conventional retrieval models often suffers from either missing relevant documents which are not indexed by the keywords used in the query, but by synonyms; or retrieving irrelevant documents which are indexed by unintended sense of the keywords in the query [BCB92]. Therefore, there has been great interest in text retrieval research that is based on semantics matching instead of strictly keyword matching.

Latent Semantic Indexing (LSI) using Singular-Value Decomposition (SVD) is a particular approach to overcoming some of the deficiencies of term-matching retrieval techniques. The LSI technique performs truncated SVD to analyze the conceptual structure of the word usage across documents [BD94]. Using the singular values and the associated vectors obtained from the truncated SVD, a high-dimensional vector space representing term-document associations is mapped onto a vector space of a lower dimension which reflects the major associative patterns in the data, while ignoring the less important associations. Thus, terms which occur in similar documents will be near each other in the reduced vector space, and documents may be retrieved to satisfy a user query when they share terms that are close in the reduced space. Since documents are represented in the reduced vector space by the statistically derived conceptual indices, instead of by individual words, the LSI technique overcomes some of the drawbacks of keyword matching techniques. Also, since the statistically derived vectors are more robust indicators of the semantic meaning than individual words, the retrieval performance based on the reduced vector space may be better than that of the original space [BCB92, BD94].

Several recent papers report the use of the LSI method in information retrieval. In [DDFL90], the LSI method equaled or outperformed the standard vector space method. The results reported in [Dum91] show that the LSI performance can be substantially improved using either the differential term weighting scheme or relevance feedback. In [DN92], the LSI method was used to automating the assignment of submitted manuscripts to reviewers of the *Hypertex'91* conference. Based on the interests of each reviewer, a set of relevant manuscripts were retrieved and sent to the reviewer. The results demonstrated that this automated assignment method achieved better matching between the reviewers

CIKM 96, Rockville MD USA
© 1996 ACM 0-89791-873-8/96/11 ..$3.50

and their interests, when compared to the assignments produced by the human experts. In [BCB92], the LSI method was compared to Multidimensional Scaling (MDS) algorithms. MDS is a class of data analysis techniques for representing data points in a multidimensional real-value space. Using MDS, the objects are represented so that the inter-point similarities in the space match inter-object similarity information provided by the researcher. The results demonstrate that the document representations given by LSI are equivalent to the optimal representations found using MDS. The LSI method was compared to Metric Similarity Modeling (MSM) [Bar94]. The results show that the optimal MSM solution is identical to the document indexing solution provided by LSI. Recently, LSI is applied to the **routing** and **adhoc** retrievals using large collections in the TREC conferences [Dum94, Dum92, Dum95]. The results were comparable to the best results submitted to those conferences. Furthmore, these results demonstrated that the large, sparse SVD problems could be solved without concerns for numerical convergence. In [BD94], a survey of the computation requirements for managing LSI-encoded databases for information retrieval was presented. Recently, LSI has also been applied to information filtering and text categorization [Fol90, Hul94, WPW95].

In this paper, we present the technique of incorporating Latent Semantic Indexing into a neural network model for text retrieval. The original neural network model was based on a causal inference network that connects the terms and related documents. The model also used Roget's Thesaurus to relate synonymous index terms [SL94b]. Using four standard document collections, CACM, CISI, ADINUL, and CRANFIELD, we demonstrated that the neural network model's retrieval performance, in terms of precision and recall, was comparable to or better than that of the recent text retrieval models [SL94b]. However, in a thesaurus-based information retrieval model, the semantic information embodied is reflected by the terms in its thesauri and the documents stored in its database [HM86]. When new documents are indexed and stored in the database, the indexing vocabulary needs to be updated to account for the changes in the domain knowledge it covers. Since the 1911 edition of Roget's Thesaurus we used for constructing the original model lacks many crucial index terms [SL94a], and the process of merging or updating thesauri is rather expensive, we incorporated the Latent Semantic Indexing technique into our neural network model, in stead of using a thesaurus, in an attempt to capture the semantic relationship between the documents and the index terms. Our results reported here show that by incorporating the LSI method, the neural network model generates an appreciable improvement over the thesaurus-based model.

The remainder of this paper is as follows. Section 2 provides a overview of the Latent Semantic Indexing method as applied to information retrieval. Section 3 briefly reviews our original neural network model. Section 4 reports the experimental results comparing the LSI-based model and the thesaurus-based model in their retrieval performance. Section 5 is the conclusion.

2 Overview of the LSI Method

In the conventional vector space models, the representations of documents and terms are explicitly taken into account for the result of the retrieval. Using Latent Semantic Indexing (LSI), which is an extension of the vector space retrieval method, it is assumed that there is some underlying or "latent" association in the pattern of terms or keywords used across documents [DN92], and this latent association can be estimated by using statistical techniques. Singular-Value Decomposition (SVD) is a technique closely related to eigenvector decomposition and factor analysis used in statistics [CW85], and Latent Semantic Indexing (LSI) using SVD is a particular approach to modeling the latent semantic relationships between the documents and the index terms. This approach performs singular-value decomposition on a term-by-document matrix, generating a reduced space with lower dimension. In the reduced space, the semantic association between two documents is captured based on how frequently the index terms used in each of the documents co-occur in other documents. Similarly, the semantic association between two index terms can be captured based on how frequently they are used in the similar contexts (documents). Using the LSI representation, documents are retrieved to satisfy a user query when they share terms of similar semantic meaning. As a result, LSI overcomes some of the deficiencies of term-matching retrieval, and provides an automated procedure that relates synonymous index terms without the need for constructing or updating a thesaurus. Since the dimension of the resulting semantic space is typically much smaller than the number of unique index terms used in a document collection (e.g. 100 to 300 vs. several thousands [Dum94]), a retrieval model using LSI can also benefit from requiring less time and memory for query processing.

We now briefly explain the properties of SVD, and describe the conversion of document and query representations from the original vector space to the reduced vector space.

2.1 Singular-Value Decomposition (SVD)

It is known that SVD is the most reliable tool available for matrix factorization [KMS89]. For any matrix A, $A^T A$ has nonnegative eigenvalues. The nonnegative square roots of the eigenvalues of $A^T A$ are called the *singular values* of A, and the number of the non-zero singular values is equal to the rank of A, $rank(A)$ [Ort87]. If A is an $m \times n$ matrix and $rank(A) = r$, the singular-value decomposition of A is defined as

$$A = UWV^T, \tag{1}$$

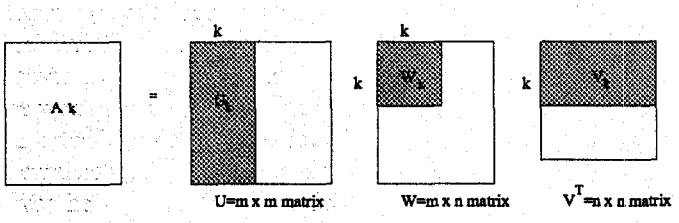


Figure 1: A schema of the truncated SVD of a term-by-document matrix A.

where the sizes of U, V, and W are, respectively, $m \times m$, $n \times n$, and $m \times n$; both U and V^T are orthogonal matrices, i.e., $U^TU = I_m$, and $VV^T = I_n$; W is a diagonal matrix consists of the singular values of A:

$$W = \begin{pmatrix} \sigma_1 & & & & \\ & \sigma_2 & & & \\ & & \cdots & & \\ & & & \sigma_{r-1} & \\ & & & & \sigma_r \\ & & 0 & & \end{pmatrix}.$$

The σ_i's are the singular values of A, $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0$, and $\sigma_i = 0$ for $i \geq r+1$.

In order to perform SVD in the LSI retrieval model, a term-by-document matrix A which represents the documents in a collection must be constructed. Using SVD, there is a simple strategy for generating optimal approximation of the document representation specified by the matrix A. Since the singular values in W are ordered by size, the first k largest may be kept and the remaining smaller ones are set to zero. As a result, the representations of the matrices U, V, and W can be reduced as follows: 1) Obtain a new diagonal matrix W_k by removing column and rows which are zeros from W; 2) Obtain a matrix U_k by removing the $(k+1)$st to the mth columns from U; and 3) Obtain a matrix V_k by removing the $(k+1)$st to the nth rows from V. The product of the resulting matrices is a matrix A_k which is an approximation of the matrix A (see Eq. 2), and $rank(A_k) = k$. Figure 1 presents a schema of the truncated SVD of matrix A.

$$A_k = U_k W_k V_k^T \tag{2}$$

The LSI method using SVD can be viewed as a technique for deriving a set of uncorrelated indexing variables or factors (i.e. the singular values) [DDFL90]. The documents and queries are then represented by vectors of factor values, instead of the individual index terms. The use of the k-largest factors captures most of the important latent associations between documents and index terms, and avoids unintended sense in word usage. A more detailed account of the mathematical properties of SVD can be found in [Bas94, Gol89].

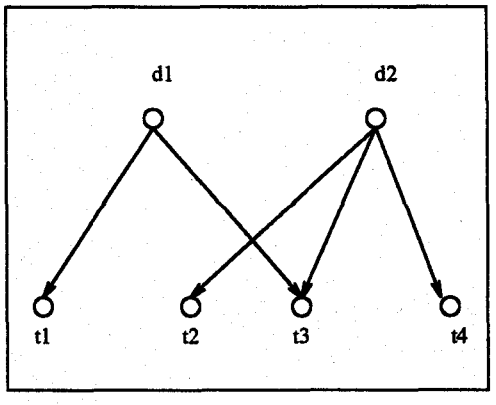


Figure 2: An example of the connectionist model.

2.2 Document and Query Representations

Using the singular-value decomposition (Eq. 2), a term-by-document matrix A is mapped into a reduced $k \times n$ matrix represented by $W_k V_k^T$, which relates k factors to n documents. Similarly, a user query q, treated as a single document m-vector, can be converted into a k-factor vector q' using Eq. 3.

$$q' = (q^T U_k W_k^{-1})^T \tag{3}$$

3 The Neural Network

The original neural network model was developed based on the causal inference network described in [PR89]. The implementation of the original model was explained in detail in [SL94b]. In this section, we briefly describe the network structure and the activation mechanism used in the original model.

The neural network model is a two-layer network (see Figure 2): the document layer D (output), the index term layer $\mathcal{T}$ (input), and a relation R connecting D and $\mathcal{T}$ [PR89]. There are no inhibitory or intra-set links. Based on the relation R, two sets, $effects(d_i)$ and $causes(\tau_j)$, are defined for each $d_i \in D$ and each $\tau_j \in \mathcal{T}$: $effects(d_i) = \{\tau_j \mid < d_i, \tau_j > \in R\}$, and $causes(\tau_j) = \{d_i \mid < \tau_j, d_i > \in R\}$. Intuitively, $effects(d_i)$ contains all the index terms caused by document d_i, and $causes(\tau_j)$ contains all the documents that can cause index term τ_j. Each document node $d_i \in D$ has an activation level at time t during the computation, denoted $d_i(t) \in [0, 1]$; each index term node τ_j has an activation level $\tau_j(t) \in [0, 1]$.

Initially, $d_i(0) = p_i$, where

$$p_i = \frac{1}{total\ number\ of\ documents}, \tag{4}$$

which represents the prior probability that a document d_i appears relevant to a given query. Each index term node $\tau_j \in \mathcal{T}$ is marked to be present ($\tau_j \in \mathcal{T}^+$) if it is present in

the user's query; otherwise, it is marked absent ($\tau_j \in \mathcal{T}^-$). Each link (connection) is associated with a constant weight representing the approximate implication strength r_{ij} of term τ_j on document d_i:

$$r_{ij} = \frac{tf_{ji} \cdot idf_j}{max\_tf_i \cdot \log(total\ number\ of\ documents)}, \quad (5)$$

where tf_{ji} is the frequency that a term τ_j appears in a document d_i, idf_j is the inverse document frequency corresponding to τ_j, and max_tf_i is the maximum tf value of the index terms in the document d_i [TC91].

To process a user query, the index terms that are present in the query, or synonymous to any that is present, are marked in the neural network. (We denote the set of marked index term nodes $\mathcal{T}^+$.) A "winners-take-all" competition algorithm is then used which iteratively updates the activation levels of the nodes in $\mathcal{T}^+$, followed by updating the activation levels of the document nodes in D. This process continues until an equilibrium is reached at time t_e, at which point each $d_i(t_e)$ is approximately equal to 0 or 1. A subset of the entire document collection, $D_s = \{d_i \mid d_i(t_e) \approx 1.0\}$, is taken to be the retrieval for a given user query.

We now briefly describe the equations used in updating the activation levels. Assuming a discrete representation of time, the index term nodes in $\mathcal{T}^+$ are updated using the activation rule

$$\tau_j(t) = 1 - \prod_{d_i \in causes(\tau_j)} (1 - r_{ij} d_i(t)). \quad (6)$$

To update the activation levels for the document nodes, a function $in_i(t)$ indicates the desired direction of change for $d_i(t)$ in order to obtain local optimization. The value $in_i(t)$ is determined by the rule

$$in_i(t) = K_i \cdot \prod_{\tau_j \in \mathcal{T}_i^+} \left(1 + r_{ij} \frac{1 - \tau_j(t)}{\tau_j(t) - r_{ij} d_i(t)}\right), \quad (7)$$

where $\mathcal{T}_i^+ = \mathcal{T}^+ \cap effects(d_i)$; K_i is a constant factor representing the influence of $\mathcal{T}^-$ nodes, and prior probabilities p_i, on document activations. The constant K_i is computed once for each document node d_i at the beginning of the neural network computation, and is defined by

$$K_i = \prod_{\tau_j \in \mathcal{T}_i^-} (1 - r_{ij}) \cdot \left(\frac{p_i}{1 - p_i}\right) \quad (8)$$

where $\mathcal{T}_i^- = \mathcal{T}^- \cap effects(d_i)$. Also, a ramp function is used to bound the change rate of $d_i(t)$ in $[-1, 1]$. The ramp function $f(x) = 1$ if $x > 1$; -1 if $x < -1$; x otherwise. Finally, the activation level $d_i(t)$ is defined by the following equation

$$d_i(t+1) = d_i(t) + f(in_i(t) - 1) \cdot (1 - d_i(t)) \cdot \Delta, \quad (9)$$

where Δ is a constant controlling the rate of change; we set it to 0.1. If $d_i(t+1)$ is less than 0.0 from Eq. 9, then $d_i(t+1)$ is set to 0.0. Thus, a desired $d_i(t)$ is guaranteed to be in [0, 1] at any time t.

When the computation reaches an equilibrium and outputs a retrieval set of documents (i.e. those with $d_i(t) \approx 1.0$), we often need to rank these documents based on their relevance to the given query. We used Eq. 10 to compute the document ranking values. The derivation of Eq 10 is described in detail in [SL96].

$$q_i(d_i(t)) = \prod_{\tau_j \in \mathcal{T}_i^+} [1 - \prod_{d_k \in D_l} (1 - r_{kj})] \cdot \prod_{\tau_h \in \mathcal{T}_i^-} \prod_{d_k \in D_l} (1 - r_{kh}) \quad (10)$$

4 Experiments and Results

In this section, we explain the construction of the neural network which is based on the document representation derived from SVD. We also explain the method to compute the precision and recall values in our model, and show the experimental results and the performance comparisons. We then derive formulas which estimate the actual time required for our experiments.

4.1 Network Construction and the Experiments

Since both the theoretical foundation and empirical studies are important issues in measuring the effectiveness of information retrieval models, we used four standard document collections, CACM, CISI, CRANFIELD, and ADINUL, to test the retrieval performance of our original neural network model [SL96]. In order to compare the performance of the LSI model with that of the original model, the same document collections were used in the experiments reported here. These collections contain information of the authors, titles, abstracts, and citations of the articles published in different research journals. Each collection consists of a set of documents and queries. The document-query relevance judgements are also provided. Table 1 shows the pertinent statistics for these collections. The performance of our LSI model is also compared with that of a vector space model (SVM) which uses the cosine measure to estimate the similarity between the document vectors and the query vectors.

To evaluate the neural network's retrieval performance on these document collections, we need to extract the index terms of each collection and generate a term-by-document matrix $A = [a_{ij}]$ as the initial representation of the documents. Only noun index terms are selected from each collection by using Roget's Thesaurus. The connection strength values a_{ij} in matrix A are computed using Eq. 5.

In a collection of n documents and m unique index terms, the initial representation of the collection is an $m \times n$ matrix. After the initial matrix is generated, we performed singular-value decomposition on the matrix to obtain a reduced matrix of rank k (see Section 2). The choice of the rank value k is critical to the retrieval performance. Idealy, the value of k should be large enough to represent the real

	CACM	CISI	CRAN-FIELD	ADINUL
Number of Documents	3204	1460	1398	82
Number of Queries	50	112	225	35
Number of Unique Index Terms	1649	1725	1276	376
Average Number of Relevant Documents per Query	16	28	8	5
Average Number of Relevant Queries per Document	0.1	0.1	0.2	0.5

Table 1: Collection statistics.

information in a collection, and also small enough so that the unimportant details will not be included [DDFL90]. The choice of a proper k value is an open issue in the literature on Factor Analysis [Gol89, Bas94]. However, the experimental results reported in [DDFL90, Dum94] have shown that the best results are obtained when $100 \leq k \leq 300$ for small collections (e.g. 1000-2000 abstracts) and for large collections (e.g. collections outlined in the TREC conference). In our experiments, we used various rank values to test the performance of our networks for each collection. For the CACM, CISI, and CRANFIELD collections, we used rank values 10, 50, 100, 150, 200, 300, and 400. For the ADINUL collection, since its initial representation is a 376×82 matrix, we used rank values 10, 20, 30, 40, 50, 70, and 82. Since computing the truncated SVD of large term-by-document matrices is very time-consuming, it was executed once for each collection, and the resulting matrices U, W and V (see Eq. 1) were saved in disk files. These files were then used to generate the reduced matrices U_k, W_k and V_k for each tested rank value k. The SVD program was adapted from [PTVF92] and was run on a Sun SparcStation System 600.

For each rank value k, a two-layer neural network is constructed based on a $k \times n$ matrix $W_k V_k^T$: the document layer of n document nodes, and the factor layer of k factors. The value a_{ij} of the matrix $W_k V_k^T$ can be viewed as the connection strength value r_{ij} between a document d_i and a factor f_j. After a neural network is constructed, a given query is processed and is represented as an m-element vector q. The query vector q is mapped into a k-factor vector q' using Eq. 3. The query vector q' is then used to mark the k factor nodes in the neural network. We note that by substituting the index terms with factors, the equations used in our original model for computing the index term and document activation levels (Eqs. 6 and 9) can be adapted to compute the activation levels of the factor and document nodes, respectively.

In each of the experiments we performed, the neural network algorithm retrieved a set of documents for each given query, then these documents were sorted by their ranking values using Eq. 10. A common approach to evaluating retrieval performance is to report the precision percentages at different recall levels. Thus, to compute the retrieval precision at the $x\%$ recall level, $1 \leq x \leq 100$, we scanned the list of the retrieved documents in ranking order, using the correct retrieval set provided by the test collection as basis, until the $x\%$ recall is met. At that point, the precision value is calculated as the percentage of relevant documents within the list of the retrieved documents scanned so far. Since our model returns a retrieval set of documents for a user query, instead of ranking all the documents in the collection, there is one modification required for computing the precisions in our model. It is possible that the $x\%$ recall is not met even after the entire list of the retrieved documents is scanned. In that case, it is reasonable to report that the corresponding precision value is zero.

4.2 Results and Performance Comparison

For each test collection, the overall performance was determined by computing the average precision at 10 recall points of 0.1, 0.2, ..., 1.0. Our experiment results show that the neural networks of rank 100 (consisting of 100 factor nodes) outperform the networks using other rank values for the CISI and CRANFIELD collections; the neural network of rank 150 outperforms the networks of other rank values for the CACM collection; and the neural network of rank 70 outperforms the networks of other rank values for the ADINUL collection. Figures 3 and 4 plot the performance comparisons among the LSI model, the original neural network model with Roget's Thesaurus, and the neural network model without Roget's Thesaurus. For the LSI model, the figures include the results of three rank values: the optimal rank, the smallest, and the largest ranks. Figures 3 and 4 demonstrate that the neural network model using Roget's Thesaurus outperforms the neural network model without using Roget's Thesaurus (as reported in [SL96]), and the LSI model using the optimal rank value is better than the neural network model with Roget's Thesaurus. Therefore, the semantic association between documents and index terms can be better represented using the LSI method. Tables 2 to 5 show the percentage changes in precision at different recall levels when comparing the LSI model (using the optimal rank) with the original neural network model using Roget's Thesaurus.

We also compare the performance of our LSI neural network model with that of a LSI vector space model using the cosine similarity measure The performance of the LSI vector space model was also tested using various rank values for each collection. The experimental results show that the optimal rank values for CACM, CISI, CRANFIELD, and ADINUL are 200, 200, 100, and 70, respectively. The comparisons between the optimal performance of our LSI neural network model and that of the LSI vector space model

	Precision (% change) CACM	
Recall	Original	LSI
10	66.4	71.0 (+6.9)
20	58.7	62.2 (+5.9)
30	53.0	54.8 (+3.4)
40	42.1	47.3 (+12.1)
50	35.2	39.4 (+11.9)
60	32.4	35.2 (+8.6)
70	26.5	30.8 (+16.2)
80	22.7	26.6 (+17.2)
90	12.6	19.1 (+51.5)
100	9.9	12.0 (+21.2)
average	36.0	39.8 (+10.6)

Table 2: Comparison of LSI model (rank 150) and the thesaurus-based model for CACM.

	Precision (% change) CISI	
Recall	Original	LSI
10	52.1	62.4 (+19.8)
20	47.7	57.7 (+21.0)
30	35.4	41.1 (+16.1)
40	25.7	31.7 (+23.3)
50	20.2	28.1 (+39.3)
60	15.4	25.2 (+63.6)
70	16.7	20.8 (+24.6)
80	8.5	14.5 (+70.6)
90	8.2	12.6 (+53.7)
100	6.3	10.1 (+60.3)
average	23.7	30.4 (+28.3)

Table 3: Comparison of LSI model (rank 100) and the thesaurus-based model for CISI.

	Precision (% change) CRANFIELD	
Recall	Original	LSI
10	54.7	70.7 (+29.3)
20	49.6	67.5 (+36.1)
30	47.3	62.4 (+31.9)
40	46.1	59.2 (+28.4)
50	45.0	55.1 (+22.4)
60	36.2	48.4 (+33.7)
70	33.2	43.1 (+29.8)
80	25.4	28.8 (+13.4)
90	15.7	23.2 (+7.5)
100	13.1	19.1 +(45.8)
average	36.6	47.8 (+30.6)

Table 4: Comparison of LSI (rank 100) model and the thesaurus-based model for CRANFIELD.

	Precision (% change) ADINUL	
Recall	Original	LSI
10	34.0	37.1 (+9.1)
20	33.5	36.5 (+9.0)
30	33.6	35.1 (+4.5)
40	31.3	33.1 (+5.8)
50	31.2	33.0 (+5.8)
60	16.4	23.4 (+42.7)
70	13.0	16.2 (+24.6)
80	12.5	14.2 (+13.6)
90	9.9	12.7 (+28.3)
100	9.4	9.9 (+5.4)
average	22.5	25.1 (+11.6)

Table 5: Comparison of LSI model (rank 70) and the thesaurus-based model for ADINUL.

	Precision (% change) CACM	
Recall	VSM	Neural Network
10	66.9	71.0 (+6.1)
20	58.7	62.2 (+6.0)
30	55.2	54.8 (−4.0)
40	43.5	47.3 (+8.7)
50	33.4	39.4 (+18.0)
60	31.2	35.2 (+12.8)
70	28.3	30.8 (+8.8)
80	22.7	26.6 (+17.2)
90	13.0	19.1 (+46.9)
100	9.2	12.0 (+30.4)
average	36.2	39.8 (+9.9)

Table 6: Comparison of LSI neural network model (rank 150) and LSI vector space model (rank 200) for CACM.

	Precision (% change) CISI	
Recall	VSM	Neural Network
10	58.0	62.4 (+9.3)
20	48.5	57.7 (+18.4)
30	38.2	41.1 (+16.1)
40	26.1	31.7 (+28.3)
50	20.1	28.1 (+41.1)
60	15.7	25.2 (+63.6)
70	13.9	20.8 (+51.8)
80	12.1	14.5 (+38.1)
90	10.6	12.6 (+37.0)
100	9.5	10.1 (+12.2)
average	25.3	30.4 (+20.2)

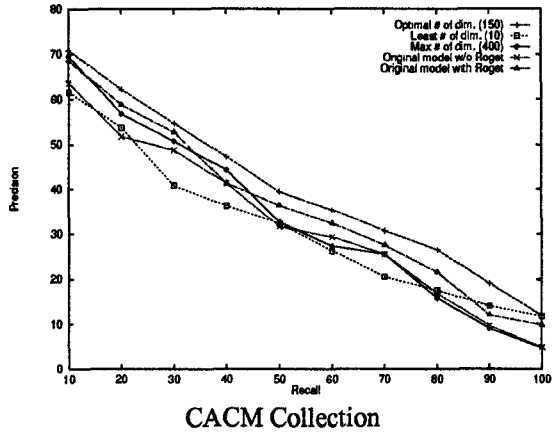
Table 7: Comparison of LSI neural network model (rank 100) and LSI vector space model (rank 200) for CISI.

	Precision (% change) CRANFIELD	
Recall	VSM	Neural Network
10	65.1	70.7 (+8.6)
20	59.6	67.5 (+13.3)
30	49.3	62.4 (+26.6)
40	46.2	59.2 (+28.1)
50	44.3	55.1 (+24.4)
60	37.2	48.4 (+30.1)
70	33.6	43.1 (+43.2)
80	24.4	28.8 (+18.0)
90	15.9	23.2 (+45.9)
100	15.1	19.1 +(26.5)
average	39.9	47.8 (+19.8)

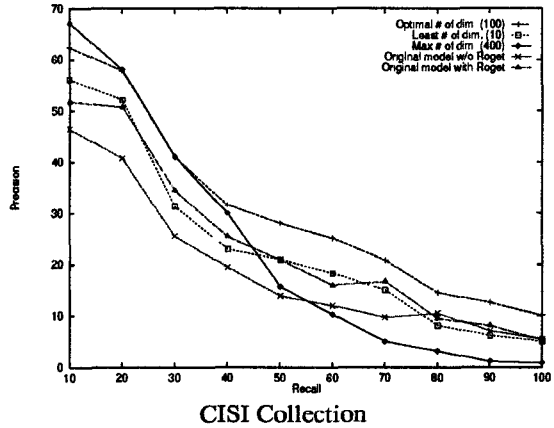
Table 8: Comparison of LSI neural network model (rank 100) and LSI vector space model (rank 100) for CRANFIELD.

	Precision (% change) ADINUL	
Recall	VSM	Neural Network
10	35.0	37.1 (+6.0)
20	32.5	36.5 (+12.3)
30	32.5	35.1 (+8.0)
40	31.0	33.1 (+6.8)
50	31.2	33.0 (+5.8)
60	18.4	23.4 (+27.2)
70	15.0	16.2 (+8.0)
80	12.1	14.2 (+17.4)
90	9.5	12.7 (+33.7)
100	9.2	9.9 (+7.6)
average	22.6	25.1 (+11.1)

Table 9: Comparison of LSI neural network model (rank 70) and LSI vector space model (rank 70) for ADINUL.

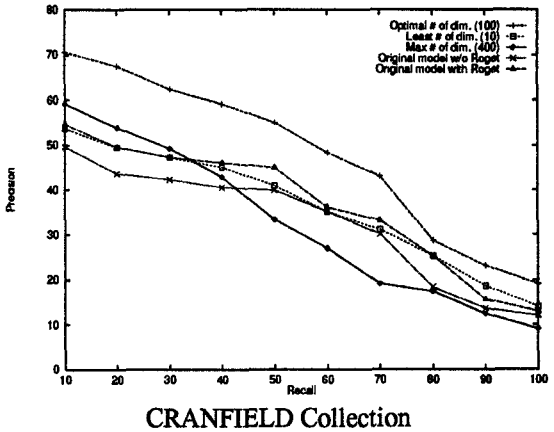


CACM Collection

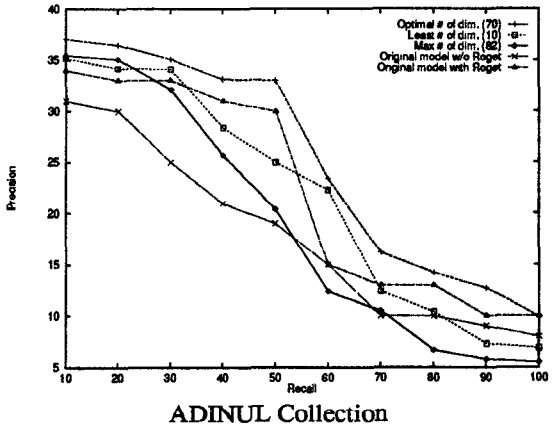


CISI Collection

Figure 3: Recall-precision graphs.



CRANFIELD Collection



ADINUL Collection

Figure 4: Recall-precision graphs.

are reported in Tables 6 to 9. The results show that our LSI neural network model performs better than the LSI vector space model.

4.3 Time Analysis

The experiment for each collection includes three procedures: SVD, preprocessing, and query evaluation. For each collection, the SVD procedure was performed once; the preprocessing and query evalution procedures were executed once for each k value.

The formulas to estimate the time required for processing a collection with n documents and m unique index terms are as follows:

- SVD

$$n \times cost(U) + m \times cost(V^T) + n \times cost(W) \quad (11)$$

- Preprocessing

$$cost(A_k) + m \times n \times cost(s) \quad (12)$$

- Query Evaluation (for each query)

$$t_{avg} \times cost(M) + NI \times (TFO \times cost(d_i) + DFO \times cost(\tau_j)) \quad (13)$$

A brief summary of the symbols used in the formulas is given in Table 10. For the sake of brievity, we only describe the comparison between the estimated time and the actual time for the CACM and CISI collections. The result is shown in Table 11. The computation detail of the cost for each procedure is not listed.

Based on Table 11, we note that the actual time for preprocessing and query evaluation is on average 6% higher than the corresponding estimated time for both CACM and CISI. However, there is a 10% and 11% discrepancy between the actual SVD processing time and estimated time for CACM and CISI, respectively. The discrepancy can be attributed to the system overhead and the inaccuracy in estimating parameters TFO, DFO, t_{avg}, etc.

5 Conclusion

In this paper, we incorporated the Latent Semantic Indexing (LSI) technique into a competition-based neural network model for information retrieval. The LSI technique provides an automated procedure that captures the semantic associations between the documents and the index terms, without using a thesaurus. Our experiments using four document collections demonstrated that the LSI-based model using optimal rank values outperforms the thesaurus-based model in retrieval effectiveness. Also, since the LSI model uses a smaller network, it usually requires less memory space and query evaluation time. Therefore, our LSI-based neural network model has the potential to handle large document collections such as those outlined in the TREC conferences. Furthermore, since neural network computations are inherently parallel, our LSI model has the potential for efficient parallel implementations.

Symbol	Definition
$cost(U)$	cost for calculating the matrix U (see Eq. 1)
$cost(V^T)$	cost for calculating the matrix V^T (see Eq. 1)
$cost(W)$	cost for calculating the matrix W (see Eq. 1)
$cost(A_k)$	cost for calculating the reduced matrix A_k (see Eq. 2)
$cost(s)$	cost for initializing a connection strength (see Eq. 5)
t_{avg}	average number of marked index terms
$cost(M)$	cost for marking an index term
NI	average number of iterations to reach equilibrium (22 for both CACM and CISI)
TFO	average number of the most connections linked to an index term node
DFO	average number of the most connections linked to a document node
$cost(\tau_i)$	cost for updating an index term node activation (see Eq. 6)
$cost(d_i)$	cost for updating a document node activation (see Eq. 9)

Table 10: Symbols used in formulas for estimating collection processing time.

Procedure	CACM		CISI	
	Estimated	Actual	Estimated	Actual
SVD	65200	72000	52000	57600
Preprocessing	482	512	210	224
Query Evaluation (entire collection)	195	208	309	324

Table 11: Comparison (in seconds) between the estimated time and actual time for collection processing.

References

[Bar94] Bartell, B. T. Optimizing Ranking Functions: A Connectionist Approach to Adaptive Information Retrieval. *PhD thesis, Computer and Information Sciense Dept., Univ. of California, San Diego, CA.*, 1994.

[Bas94] Basilevsky, A. *Statistical Factor Analysis and Related Methods: Theory and Applications.* John Wiley & Sons, Inc., New York, 1994.

[BCB92] Bartell, B. T., G. W. Cottrell, and R. K Belew. Latent Semantic Indexing is an Optimal Special Case of Multidimensional Scaling. *Proc. of the 15th International Conference on Research and Development in Information Retrieval*, 161-167, 1992.

[BD94] Berry, M. W. and S. T. Dumais. Using Linear Algebra for Information Retrieval. *Technical Report CS-94-270, Department of Computer Science, University of Tennessee*, 1994.

[CW85] Cullum, J. K. and R. A. Willoughby. *Lanczos Algorithms for Large Symmetric Eigenvalue Computations - Vol 1, Theory (Chapter 5: Real Rectangular Matrices).* Brikhauser, Boston, 1985.

[DDFL90] Deerwester, S., S. T. Dumais, G. W. Furnas, and T. K. Landauer. Indexing by Latent Semantic Analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

[DN92] Dumais, S. T. and J. Nielsen. Automating the Assignment of Submitted Manuscripts to Reviewers. *Proc. of the 15th International Conference on Research and Development in Information Retrieval*, 233-244, 1992.

[Dum91] Dumais, S. T. Improving the Retrieval of Information from External Sources. *Behavior Research Methods, Instruments and Computers*, 23(1):229–236, 1991.

[Dum92] Dumais, S. T. LSI Meets TREC: A Status Report. *The First Text REtrieval Conference (TREC-1), NIST Special Publication 500-207*, 137-152, 1992.

[Dum94] Dumais, S. T. Lantent Semantic Indexing and TREC-2. *The Second Text REtrieval Conference (TREC-2), NIST Special Publication 500-215*, 105-115, 1994.

[Dum95] Dumais, S. T. Lantent Semantic Indexing (LSI). *The Third Text REtrieval Conference (TREC-3), NIST Special Publication 500-225*, 219-230, 1995.

[FLGD87] Furnas, G. W., T. K. Landauer, L. M. Gomez, and S. T. Dumais. The Vocabulary Problem in Human-System Communications. *Communications of the ACM*, 30:964–971, 1987.

[Fol90] Foltz, P. W. Using Latent Semantic Indexing for Information Filtering. *Proc. of ACM Conference on Office Information Systems*, 40-47, 1990.

[Gol89] Golub, J. K. *Matrix Computations, 2nd ed.* 1989.

[HM86] Humphrey, S. and B. Melloni. *Databases: A Primer for Retrieveing Information by Computer.* Englewood Cliffs, Prentice Hall, New Jersey, 1986.

[Hul94] Hull, D. Improving Text Retrival for the Routing Problem Using Latent Semantic Indexing. *Proc. of the 17th International Conference on Research and Development in Information Retrieval*, 282-291, 1994.

[KMS89] Kahaner, D., C. Moler, and Nash. S. *Numerical Methods and Software.* Prentice Hall, New Jersey, 1989.

[Ort87] Ortega, J. M. *Matrix Theory.* Plenum, New York and London, 1987.

[PR89] Peng, Y. and J. A. Reggia. A Connectionist Model for Diagnostic Problem Solving. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(2):285–298, 1989.

[PTVF92] Press, W. H., S .A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C, Second Edition.* Cambridge University Press, 1992.

[Sal89] Salton, G. *Automatic Text Processing.* Addison-Wesley. New York, 1989.

[SL94a] Syu, I. and S. D. Lang. A Competition-Based Connectionist Model for Information Retrieval Using a Merged Thesaurus. *Proc. of the Third International Conference on Information and Knowledge Management*, 248-265, 1994.

[SL94b] Syu, I. and S. D. Lang. Heuristic Information Retrieval: A Competition-Based Connectionist Model. *Proc. of Intelligent Multimedia Information Retrieval Systems and Management*, 248-265, 1994.

[SL96] Syu, I. and S. D. Lang. Adapting a Diagnostic Problem Solving Model to Information Retrieval. *to appear in the Journal of Information Processing & Management*, 1996.

[TC91] Turtle, H. and W. B. Croft. Evaluation of an Inference Network-Based Retrieval Model. *ACM Transaction on Information Systems*, 9(3):187–222, 1991.

[WPW95] Wiener, E., J. O. Pedersen, and A. S. Weigend. A Neural Network Approach to Topic Spotting. *Proc. of the Symposium on Document Analysis and Information Retrieval*, 317-332, 1995.