# Variable Latent Semantic Indexing

Anirban Dasgupta
Department of Computer Science
Cornell University
Ithaca, NY 14853.
adg@cs.cornell.edu

Ravi Kumar
IBM Almaden Research Center
650 Harry Road
San Jose, CA 95120.
ravi@almaden.ibm.com

Prabhakar Raghavan*
Yahoo!, Research Labs
701 First Avenue
Sunnyvale, CA 94089.
pragh@yahoo-inc.com

Andrew Tomkins
IBM Almaden Research Center
650 Harry Road
San Jose, CA 95120.
tomkins@us.ibm.com

## ABSTRACT

Latent Semantic Indexing is a classical method to produce optimal low-rank approximations of a term-document matrix. However, in the context of a particular query distribution, the approximation thus produced need not be optimal. We propose VLSI, a new query-dependent (or "variable") low-rank approximation that minimizes approximation error for any specified query distribution. With this tool, it is possible to tailor the LSI technique to particular settings, often resulting in vastly improved approximations at much lower dimensionality. We validate this method via a series of experiments on classical corpora, showing that VLSI typically performs similarly to LSI with an order of magnitude fewer dimensions.

## Categories and Subject Descriptors

G.1.3 [**Numerical Analysis**]: Numerical Linear Algebra—*Singular value decomposition*; G.1.3 [**Numerical Analysis**]: Numerical Linear Algebra—*Sparse, structured, and very large systems (direct and iterative methods)*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Miscellaneous*

## General Terms

Algorithms, Experimentation, Measurement, Theory

## Keywords

Linear algebra, SVD, Matrix approximation, LSI, VLSI

*Work done at Verity, Inc.

## 1. INTRODUCTION

### 1.1 Overview

Dimensionality reduction is a classic technique in data analysis and mining. Here one has a number of *entities*, each of which is a vector in a space of *features*. For instance in text retrieval the entities are documents and the features (axes of the vector space) are usually terms occurring in the documents. We can then view the set of entities as a matrix $A$ in the space of features, where each feature is a column of $A$. In text analysis for instance, the number of axes in this vector space can thus be in the tens of thousands, corresponding to tens of thousands of terms in the lexicon. An entry in the matrix connotes some measure of the strength of a term in a document, usually derived from occurrence statistics (e.g., the frequency of the term in the document). Other examples of entities studied in this form include images (where the features include color, hue etc.), audio, face recognition, OCR, human fingerprints, etc.

Dimensionality reduction recognizes that despite the large number of axes in the space of features, most data sets arising in practical applications result in the matrix $A$ having a good *low-dimensional approximation* $A'$: a matrix $A'$ with the same number of rows/columns as $A$, but with rank considerably smaller than the number of axes in the vector space. Intuitively, $A'$ captures the most salient features of $A$ but enjoys a representation in a subspace of very low dimension. In text analysis, for instance, it is widely reported that good approximations of rank 200-300 exist for typical document collections. Computationally such approximations are typically found using the linear-algebraic technique of *singular value decompositions (SVD's)*, a method rooted in statistical analysis [13]. SVD's have become a "workhorse in machine learning, data mining, signal processing, computer vision, ..." [10]. Eckart and Young [8] proved that in a specific technical sense (made precise below), the SVD yields the best possible approximation to any matrix $A$, given any target rank for the approximation $A'$. As a result of the SVD, each document can be viewed as a vector in a low-dimensional space of a few hundred dimensions; the axes in the new space do not in general correspond to terms in the lexicon.

The classic application of SVD's to text analysis stems from work of Dumais et al. [3, 7]. The authors adapted SVD to the term-document matrix, characterizing their method as *latent semantic indexing (LSI)*. The principal application of LSI was to respond to text queries: following standard practice in text retrieval, each query (expressed as a set of terms) is viewed as a unit vector in the space in which each document is represented (whether the original set of terms as in $A$, or in the low-dimensional approximate space). Given a query, the system would identify the documents with the highest cosine similarity to the query (in the primary or approximate space) and return these as the best matches to the query.

Experimentally, these and subsequent papers [4, 5, 6] showed that latent semantic indexing was effective not only in that it found a good approximation $A'$ of rank about 300, but further that for retrieval on queries the approximation $A'$ often yielded *better* (rather than almost as good) results than $A$. Qualitatively, this was explained by Dumais et al. and others by the following intuition: the approximation $A'$, in being forced to "squeeze" the primary vector space (spanned by $A$), collapses synonymous terms (axes) such as `car` and `automobile`. Additionally, it was argued that LSI separated multiple meanings of a single term (such as `charge`), into different axes of $A'$ based on co-occurrences of `charge` with disparate groups of other terms (say, `electron` and `proton`, as opposed to `brigade` and `cannon`).

Somewhat surprisingly, the entire premise of LSI – the computation of the approximation $A'$ as well as its empirical success in retrieval – are oblivious to the characteristics of queries. Given that the motivation of Dumais et al. was to respond to queries, it is surprising that the approximation pays no attention to the types of queries. For a simple example, consider querying a collection of news stories. The approximation constructed by LSI would faithfully represent all the topics in the news. Suppose now that the distribution of queries is, however, focused heavily on the subject of Finance. Could it be that there are better low-rank approximations to the matrix $A$ that are especially tuned to queries focused on Finance, ignoring terms (axes) that epitomize other subjects?

One approach might be to remove terms commonplace in non-Finance categories from the matrix $A$, then perform LSI on the resulting matrix with fewer axes to begin with. This raises the question: is there a principled way to compute such a query-dependent LSI, and establish its optimality via an analog of the Eckart-Young theorem? Could such a query-dependent LSI significantly outperform query-oblivious LSI in retrieval performance? We answer these questions in the affirmative: we devise a novel form of LSI that takes the query distribution into account, prove its optimality, and establish experimentally that it dramatically outperforms LSI on retrieval, for any target rank for the approximation.

Our results are more general than the particular application to text. Indeed, one could apply the same technique to querying images, fingerprints or other entities represented in a vector space. From a pragmatic standpoint, the query distribution could be "learned" over time, so that one could periodically recompute an approximation tuned to the current query distribution. More generally, dimensionality reduction for data analysis is never performed in a vacuum; rather, it is performed with a *context* in mind. To the extent that this context can be formulated as a certain type of co-occurrence matrix (made precise below), our technique is applicable.

### 1.2 Other related prior work

In addition to the research mentioned above, SVD's have been applied to a variety of settings in data analysis including face recognition [18], collaborative filtering [12], denoising [17] and object analysis [14]. All of these applications are – in the sense outlined above – query-oblivious. Weighted generalizations of SVD have been considered before [23, 25]; however these minimize a weighted Frobenius norm instead of the usual norm and are not applicable to our case. Probabilistic latent semantic indexing [11] and its cousins from statistics [24] use a generative probabilistic model for the *entries* of the matrix $A$, rather than for the query distribution.

### 1.3 Our results

Section 2 gives the mathematical development of our new approximation method, and proves its optimality. By characterizing the queries likely to arise through a probability distribution (in fact, we use a general model based on where there is a co-occurrence matrix on pairs of query terms), we derive a form of query-dependent, or *variable* LSI, which we denote VLSI. A nice feature of our approximation is that it reduces to the standard LSI approximation for the special case when the co-occurrence matrix is the scaled identity matrix.

Section 3 details experiments on a collection of documents. We study various query distributions, including ones that we tailor to be topic-focused (in the sense of the Finance example from Section 1.1 above). We study the retrieval effectiveness as a function of the number of dimensions in the low-dimensional approximation $A'$. In all cases, we find that VLSI dramatically outperforms LSI on retrieval effectiveness for any given number of dimensions in the low-dimensional approximation. An alternative way of viewing these results: for any quantitative level of retrieval effectiveness, the number of dimensions in the low-rank approximation is dramatically lower for VLSI than for LSI. As an example, whereas LSI on text corpora appears to require hundreds of dimensions in the approximation, a few tens of dimensions often suffice for VLSI.

## 2. ALGORITHM

### 2.1 Preliminaries and background

Let $A \in \Re^{m \times n}$ be the term–document matrix over $m$ terms (the rows) and $n$ documents (the columns); in this section we do not address the issue of how this matrix is constructed. The *singular value decomposition (SVD)* of a matrix is the most commonly used orthogonal decomposition of the matrix, expressing it as a product of two orthogonal matrices and a diagonal matrix. For a matrix $A \in \Re^{m \times n}$, the singular value decomposition of $A$ is written as

$$A = U \Sigma V^T, \tag{1}$$

where $U = [u_1, \ldots, u_n]$ and $V = [v_1, \ldots, v_n]$ are column orthogonal matrices, and $\Sigma = (\sigma_1, \ldots, \sigma_n)$ is a diagonal matrix of nonnegative entries. The columns of $U$ and $V$ are referred to as the left and right singular vectors of $A$ and the diagonal entries in $\Sigma$ as the singular values of $A$. It is well known that every real matrix has an SVD decomposition and if in addition the matrix is symmetric and positive semidefinite, then it has a decomposition (the eigenvalue decomposition) of the form $Y \Lambda Y^T$ in which all entries of $\Lambda$ are non-negative.

**Notation.** For a matrix $A$, we use $\mathrm{rk}(A)$ to denote its *rank* and $\mathrm{Tr}(A)$ to denote its *trace*, i.e., the sum of its diagonal entries. We use $\|A\|_F^2$ to denote the *Frobenius norm*, where $\|A\|_F^2 = \sum_{ij} A_{ij}^2$. An alternate expression for the Frobenius norm is $\|A\|_F^2 = \mathrm{Tr}\left(A^T A\right)$.

DEFINITION 1 (SVD RANK-$k$ APPROXIMATION). *If $A = U\Sigma V^T$ is the singular value decomposition of $A$, then the* SVD rank-$k$ approximation *of $A$ is defined as*

$$A_k = \sum_{i=1}^{k} \sigma_i \cdot u_i \cdot v_i^T.$$

We write $A_k = U_k \Sigma_k V_k^T$, where $U_k = [u_1, \ldots, u_k]$, $\Sigma_k = (\sigma_1, \ldots, \sigma_k)$ and $V_k = [v_1, \ldots, v_k]$. Note also that $A_k = AV_kV_k^T$.

A well-known property of the SVD is that it optimizes the Frobenius norm (cf. [9]).

THEOREM 2. *For any matrix $A \in \Re^{m \times n}$,*

$$\min_{X|\mathrm{rk}(X) \le k} \|A - X\|_F^2 = \|A - A_k\|_F^2.$$

We now show that the SVD rank-$k$ approximation of $A$ can also be interpreted as the rank-$k$ matrix that best approximates the average distortion that $A$ imparts to a unit random vector. Suppose that $q = (q_1, \ldots, q_n)$ is a random (query) vector such that $\mathbf{E}[q_i] = 0$ for all coordinates $i = 1, \ldots, n$. Consider the average distortion that occurs on multiplication of $q$ by $A$. The rank-$k$ matrix that best approximates the average distortion is given by

$$\mathrm{argmin}_{X|\mathrm{rk}(X) \le k} \mathbf{E}\left[\|q^T(A - X)\|_2^2\right].$$

We now motivate our approach by showing a relationship between the SVD of $A$ and optimizing this average distortion.

DEFINITION 3 (CO-OCCURRENCE MATRIX). *Let $\mathcal{Q}$ be any distribution on $\Re^m$. The* co-occurrence matrix $C_{\mathcal{Q}} \in \Re^{m \times m}$ *is defined to be $C_{\mathcal{Q}} = \mathbf{E}_{q \sim Q}[qq^T]$.*

Sometimes it is useful to think of $\mathcal{Q}$ as the probability distribution from which queries are drawn: the $i$-th coordinate being 1 corresponds to the $i$-th term appearing in the query. Note that if $\mathcal{Q}$ is the product distribution obtained by taking, say, the Gaussian $N(0, \sigma^2)$ distribution in each coordinate, then $C_{\mathcal{Q}} = \sigma^2 I$. In general, there is a useful connection between optimizing the average distortion and the SVD when the co-occurrence matrix is the scaled identity matrix.

LEMMA 4. *If for a distribution $Q$ over $\Re^m$, the co-occurrence matrix is $C_{\mathcal{Q}} = \sigma^2 I$ for some $\sigma$, then*

$$\min_{X|\mathrm{rk}(X) \le k} \mathbf{E}_{q \sim Q}\left[\|q^T(A - X)\|_2^2\right] = \mathbf{E}_{q \sim Q}\left[\|q^T(A - A_k)\|_2^2\right].$$

PROOF. To prove this correspondence we just need to simplify the given expression. Using the fact that for two vectors $u$ and $v$, $u^T v = \mathrm{Tr}\left(vu^T\right)$, we have

$$\begin{aligned}
&\mathbf{E}\left[\|q^T(A - X)\|_2^2\right] \\
&= \mathbf{E}\left[q^T(A - X)(A^T - X^T)q\right] \\
&= \mathbf{E}\left[\mathrm{Tr}\left((A^T - X^T)qq^T(A - X)\right)\right] \\
&= \mathrm{Tr}\left((A^T - X^T)C_{\mathcal{Q}}(A - X)\right) \\
&= \sigma^2 \mathrm{Tr}\left((A^T - X^T)(A - X)\right) \\
&= \sigma^2 \|A - X\|_F^2.
\end{aligned}$$

Thus optimizing the expected distortion can be done by taking $X$ to be $A_k$, the SVD rank-$k$ approximation of the matrix $A$. □

## 2.2 Main result

We now extend the above result to compute the best rank-$k$ approximation to a given matrix when the distribution $\mathcal{Q}$ is arbitrary. A random query generated from this distribution $\mathcal{Q}$ is again denoted by $q$.

First note that $C_{\mathcal{Q}}$ is positive semidefinite, as for any vector $v$,

$$v^T C_{\mathcal{Q}} v = \mathbf{E}\left[(v^T q)^2\right] \ge 0.$$

This also means that any such $C_{\mathcal{Q}}$ has an eigenvalue decomposition $C_{\mathcal{Q}} = Y \Lambda Y^T$ where $\Lambda = (\lambda_1, \ldots, \lambda_n)$ with $\lambda_1 \ge \cdots \ge \lambda_n \ge 0$. If $\mathrm{rk}(C_{\mathcal{Q}}) = r$, we write this eigenvalue decomposition as $C_{\mathcal{Q}} = Y_r \Lambda_r Y_r^T$.

Motivated by Lemma 4, the natural generalization of SVD to arbitrary query distributions $\mathcal{Q}$ is to find a rank-$k$ approximation $A_{Q,k}$ to $A$ such that

$$A_{Q,k} = \mathrm{argmin}_{X|\mathrm{rk}(X) \le k} \mathbf{E}_{q \sim Q}\left[\|q^T(A - X)\|_2^2\right]. \qquad (2)$$

DEFINITION 5 (SQUARE ROOT AND PSEUDOINVERSE). *For a distribution $Q$, let $C_{\mathcal{Q}} = Y_r \Lambda_r Y_r^T$ be the co-occurrence matrix of rank $r$. The* square-root *of $C_{\mathcal{Q}}$ is defined to be $C_{\mathcal{Q}}^{1/2} = Y_r \Lambda_r^{1/2} Y_r^T$ and the* pseudoinverse *of $C_{\mathcal{Q}}^{1/2}$ is defined to be $C_{\mathcal{Q}}^{-1/2} = Y_r \Lambda_r^{-1/2} Y_r^T$.*

We show the following.

THEOREM 6. *Suppose $V_k \in \Re^{n\times k}$ contains the top $k$ right singular vectors of $C_{\mathcal{Q}}^{1/2}A$ in its columns. Then the matrix $A_{Q,k}$ minimizing (2) is the matrix $A_{Q,k} = AV_kV_k^T$.*

PROOF. Suppose for now $C_{\mathcal{Q}}$ is a full-rank matrix. Then, $C_{\mathcal{Q}}^{1/2}C_{\mathcal{Q}}^{-1/2} = YY^T = I$ as $Y$ is an $n \times n$ orthogonal matrix. Define the random vector $w = C_{\mathcal{Q}}^{-1/2}q$. Then,

$$\mathbf{E}\left[ww^T\right] = \mathbf{E}\left[C_{\mathcal{Q}}^{-1/2}qq^TC_{\mathcal{Q}}^{-1/2}\right] = I.$$

Using this,

$$\begin{aligned}\mathbf{E}\left[\|q^T(A-X)\|_2^2\right] &= \mathbf{E}\left[\|w^TC_{\mathcal{Q}}^{1/2}(A-X)\|_2^2\right] \\ &= \mathbf{E}\left[\|w^T(C_{\mathcal{Q}}^{1/2}A - C_{\mathcal{Q}}^{1/2}X)\|_2^2\right].\end{aligned}$$

Since the co-occurrence matrix of $w$ is $I$, we can apply Lemma 4. Thus, the above expression is minimized when $C_{\mathcal{Q}}^{1/2}X$ is the rank-$k$ approximation of the matrix $C_{\mathcal{Q}}^{1/2}A$. If $V_k$ are the top-$k$ right singular vectors of $C_{\mathcal{Q}}^{1/2}A$, then the rank-$k$ approximation of $C_{\mathcal{Q}}^{1/2}A$ is given by $(C_{\mathcal{Q}}^{1/2}A)V_kV_k^T$. Therefore, we need

$$\begin{aligned}C_{\mathcal{Q}}^{1/2}X &= C_{\mathcal{Q}}^{1/2}AV_kV_k^T \\ X &= AV_kV_k^T\end{aligned}$$

as $C_{\mathcal{Q}}$ is a full rank matrix.

If $C_{\mathcal{Q}}$ is not full rank, then $C_{\mathcal{Q}}^{1/2}C_{\mathcal{Q}}^{-1/2} = Y_rY_r^T$, the projection onto the space spanned by the query vectors. Intuitively, the only change that we need to handle this case is to work in the $r$-dimensional space spanned by the query distribution. Below we work out the details for a rigorous proof. Define $w = Y_r^T(C_{\mathcal{Q}}^{-1/2}q) = \Lambda_r^{-1/2}Y_r^Tq$. Note that $w$ is in $\Re^r$ where $r$ is the rank of the query distribution. Also,

$$\mathbf{E}\left[ww^T\right] = I_r.$$

It follows that

$$\mathbf{E}\left[\|q^T(A-X)\|_2^2\right] = \mathbf{E}\left[\|q^TY_rY_r^T(A-X)\|_2^2\right].$$

Substituting the value of $q^TY_rY_r^T = w^TY_r^TC_{\mathcal{Q}}^{1/2}$ in the above expression, we have

$$\begin{aligned}&\mathbf{E}\left[\|q^T(A-X)\|_2^2\right] \\ &= \mathbf{E}\left[\|q^TY_rY_r^T(A-X)\|_2^2\right] \\ &= \mathbf{E}\left[\|w^TY_r^TC_{\mathcal{Q}}^{1/2}(A-X)\|_2^2\right] \\ &= \mathbf{E}\left[\|w^T(Y_r^TC_{\mathcal{Q}}^{1/2}A - Y_r^TC_{\mathcal{Q}}^{1/2}X)\|_2^2\right]. \qquad (3)\end{aligned}$$

So, arguing as before, we need to compute the rank-$k$ approximation to the matrix $Y_r^TC_{\mathcal{Q}}^{1/2}A$. Now, for any vector $v$,

$$\begin{aligned}\|(C_{\mathcal{Q}}^{1/2}A)v\| &= \|(Y_r\Lambda_rY_r^TA)v\| \\ &= \|(Y_rY_r^TY_r\Lambda_rY_r^TA)v\| \\ &= \|Y_r(Y_r^TC_{\mathcal{Q}}^{1/2}A)v\| \\ &= \|(Y_r^TC_{\mathcal{Q}}^{1/2}A)v\|,\end{aligned}$$

where the last equality follows from the column orthogonality of $Y_r$. Thus the right singular vectors of $Y_r^TC_{\mathcal{Q}}^{1/2}A$ are the same as those of $C_{\mathcal{Q}}^{1/2}A$. So if $V_k$ are the top $k$ right singular vectors of $C_{\mathcal{Q}}^{1/2}A$, then the rank-$k$ approximation of $Y_r^TC_{\mathcal{Q}}^{1/2}A$ is $(Y_r^TC_{\mathcal{Q}}^{1/2}A)V_kV_k^T$. Thus applying Theorem 2, (3) is minimized when

$$Y_r^TC_{\mathcal{Q}}^{1/2}X = (Y_r^TC_{\mathcal{Q}}^{1/2}A)V_kV_k^T,$$

and it suffices to choose $X = AV_kV_k^T$.

And so, in both the cases, $A_{Q,k} = AV_kV_k^T$ is the minimizing choice for (2). □

Note that the above approximation collapses to the usual SVD when $\mathcal{Q}$ is the uniform distribution on unit vectors. In fact, it does so even for isotropic single term query distributions, i.e., distributions where the probability of any term appearing is $p$, say for all terms, and no two terms appear together in the query, i.e., $\mathbf{E}[q_iq_j] = 0$. Then the co-occurrence matrix of such distributions is $C_{\mathcal{Q}} = pI$. Thus for this case, our optimization is the same as that of Lemma (4), and therefore, just computing the SVD of $A$ would serve our goals.

Finally, the computational requirements of VLSI are similar to that of standard LSI. One benefit of the rank-$k$ approximation is that instead of storing the matrix $A$ that potentially requires $O(mn)$ space, one could store the rank-$k$ decomposition $(U_k, \Sigma_k, V_k)$ that needs only $O(mk+nk)$ space. In our case, it is sufficient to store the matrices $(AV_k, V_k^T)$, which needs $O(mk + nk)$ space as well.

# 3. EXPERIMENTS

We selected the Reuters-21578 document set Distribution 1.0 [16] for our experiments. This corpus has been widely used in machine learning because it contains detailed topic labels for each document. We chose the corpus because we may employ these topic labels in order to be able to generate controlled query distributions. The corpus contains 21,578 documents that appeared in the Reuters news feed in 1987. Documents are labeled with membership in five sets of categories, one of which is TOPICS, which represent a broad range of 135 economic subject categories such as "gold" or (the commodity) "coconuts." Figure 1 shows the distribution (in words) of the sizes of the documents in the collection. There are 112,356 distinct words in the corpus, with an average document length of 134 words.

The lexicon is built as follows. First, the text is extracted from the XML files in which the Reuters-21578 corpus is delivered. Next, tokens are extracted by splitting at whitespace boundaries. As is commonplace in text processing, tokens are then Porter-stemmed and case-folded, punctuation is removed, and a standard 416-word stopword list is employed to remove stopwords. Finally, certain html formations and tokens that appear only once in the entire corpus are removed. This results in a final cleaned dictionary of 33,749 terms.

Based on this dictionary, the corpus is scanned to produce a term–document matrix of counts, $A^c$, for which $A_{ij}^c$ is the

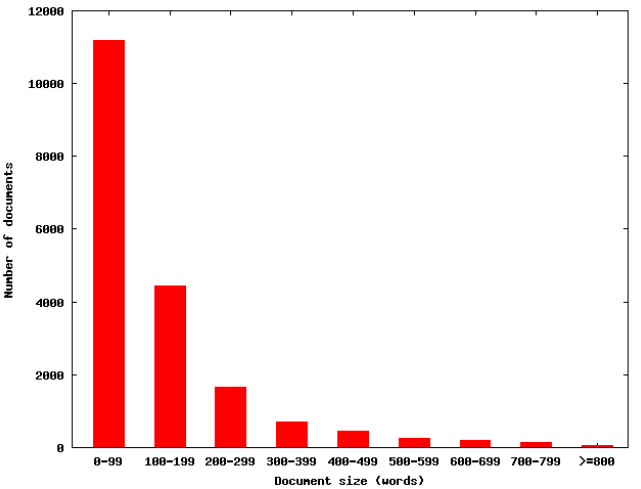


**Figure 1: Histogram of document size for Reuters-21578 collection.**

number of occurrences of term $i$ in document $j$. From this matrix, we derive two other matrices on which we perform our experiments. First, the Boolean matrix $A^b$ is produced by setting $A^b_{ij}$ to 1 if $A^c_{ij}$ is non-zero, and to zero otherwise.

In addition, we study a weighted version of the term–document matrix. In classic text analysis, various techniques are known for the derivation of the weight of a term in a document based on term and corpus statistics. Of the many such mappings (from term/corpus statistics to weights) known, one that has proven particularly successful in text retrieval is the so-called *Okapi* weighting, defined for a particular query $Q$ and a particular document $D$ as follows:

$$\sum_{t \in Q \cap D} \ln \frac{N - \mathrm{df} + 0.5}{\mathrm{df} + 0.5} \cdot \frac{(k_1+1)\mathrm{tf}}{k_1(1 - b + b(\mathrm{dl}/\mathrm{adl})) + \mathrm{tf}} \cdot \frac{(k_3+1)\mathrm{qtf}}{k_3 + \mathrm{qtf}},$$

Here, $N$ is the total number of documents, df is the number of documents containing term $t$, tf is the number of occurrences of $t$ in document $D$, dl is the length of $D$, adl is the average length of documents in the corpus, and qtf is the number of occurrences of $t$ in the query.

Accordingly,the Okapi weighting matrix $A^O$ is produced by applying the standard Okapi term weighting algorithm to the entries of $A^c$; we note in passing that we use the Okapi formula parameters $k_1 = 1.2$, $b = 0.75$, and $k_3 = 7$ common in prior text analysis [21, 19].

We then used an external memory version of SVDPACKC 1.0 package [2] to perform a 1000-dimensional SVD on the Boolean and Okapi matrices $A^b$ and $A^O$ as a baseline. We also apply VLSI to each of these matrices with a family of query distributions defined below. We show results comparing the two low-rank approximations using two different metrics, which we now describe.

## 3.1 Evaluation metrics

Our evaluation is always in the context of a particular query distribution $Q$, where a query in the distribution is a $T$-element vector of real numbers representing the weighting of each of the $T$ terms. For most of our experiments, we consider single-word queries.

We compare results using two metrics. For any particular query vector $q$, the score assigned by some matrix $A$ (taken generically to represent $A^b$ or $A^O$) to each document is simply $q^T A$. We will write $\mathrm{LSI}(A, k)$ to represent the rank-$k$ approximation of $A$ produced by LSI, and $\mathrm{VLSI}(A, Q, k)$ to represent the rank-$k$ approximation of $A$ produced by VLSI with respect to query distribution $Q$. The $L_2$ error of an approximation $\tilde{A}$ to $A$ for a query $q$ is simply $||q^T(A - \tilde{A})||_2$, and the error of $\tilde{A}$ with respect to distribution $Q$ is

$$\mathbf{E}_{q \sim Q}[\|q^T(A - \tilde{A})\|_2^2].$$

Second, we also consider an evaluation metric based on information retrieval that compares the rankings induced by $A$ versus $\tilde{A}$. Following the standard vector space ranking algorithm, for query vector $q$, consider ranking the columns of $M$ by their score in $q^T M$ (breaking ties by ranking the smaller index first). Let $S$ be the top $k$ documents as ranked by $\tilde{A}$, the approximation to $A$. The *competitive precision at* $d$ [22] is defined as $1/d$ times the number of those documents that appear in the top $d$ documents as ranked by $A$. Thus, a competitive precision approaching 1 means that most of the highly-ranked documents in our approximation would also have been ranked highly by the original matrix. We consider competitive precision at 10 unless specified otherwise. For brevity, we name one minus competitive precision as *competitive error (CE)*.

## 3.2 Query distributions

There is evidence to suggest that query distributions follow a power law: the probability that a particular query occurs $x$ times is proportional to $x^{-\alpha}$ for some $\alpha$. We build on this evidence to generate some of our query distributions, in a manner described below. We found three explicit references to particular power law exponents over query distributions in the literature. Baeza-Yates [1] reports an exponent of 1.7 in the context of a Chilean search engine; Lempel and Moran [15] give 2.4 for a log of 7M queries submitted to the Altavista search engine; and Saraiva et al. [20] report 2.7 for a Brazilian search engine. Given this variation in exponents, we adopt a power law exponent of 2.4 as being a middle ground.

We consider the following query distributions. Consider the lexicon of all terms in the analysis (corresponding to all rows in the matrix $A$). Now, rank this lexicon by total number of occurrences of each term across the entire corpus. Let $t_i$ be the $i$-th term in this ranking, and let $p_i$ be the occurrence probability of term $t_i$. We consider three distributions over single-term queries. The first distribution simply mirrors the distribution of terms in the corpus, while the other three follow a power law. We then consider two cases of planted power laws, all with exponent 2.4. First, the power law is placed over the terms in their order of frequency in the corpus; second, the power law is placed over terms in random order. More formally, the three query distributions are as follows:

**D1** : The probability of $t_i$ is $p_i$.

**D2** : The probability of $t_i$ is $ci^{0.714}$.

**D3** : The probability of $t_i$ is $c(\sigma(i))^{0.714}$ where $\sigma$ is a random permutation of the terms.[1]

In addition to these three distributions, we also consider distributions D1 and D2 in which the terms $t_i$ are ranked, not according to their frequency in the corpus, but according to their frequency in the subset of documents that discuss a particular topic family. We consider two such topic families: `money`, and `commodities`, which cover 2615 and 1849 documents respectively. Both topics contain about 12,000 unique terms. Terms that do not appear in the documents covering the topic have zero probability in the query distribution, and all other terms have probability determined by scheme D1–D3. This yields a further six query distributions, which we will refer to as "money.(D1,D2,D3)" and "commodity.(D1,D2,D3)," in addition to the original 3.

## 3.3 Results

In all our result graphs, we use two measures for the quality of the approximations: (1) $L_2$ error normalized so that the error with a single dimension in the approximation is 1, and (2) competitive error, i.e., one minus competitive precision. Note that both measures are query-dependent, unlike the Frobenius norm in the theory of SVD.

**Comparing results on query distributions D1–D3.** Figure 2 shows the results for query distributions D1–D3 for the matrix $A^O$ using approximations of rank 1 through 1000. For query distribution D1, which mirrors the term distribution in the original corpus, the trends of LSI and VLSI are quite similar but at different magnitudes of error. VLSI shows a 10% improvement at 10 dimensions, a 27% improvement at 50 dimensions, a 50% improvement at 125 dimensions, and an 80% improvement at 1000 dimensions. Stated alternatively, VLSI with just 40 dimensions is about equivalent in performance to LSI with 250 dimensions.

When a power law is introduced to the query log in query distribution D2, matching the significant skew found in real-world queries, the results are more significant. Error rates drop more rapidly for both LSI and VLSI. At 50 dimensions, the error of LSI is about 60% of the rank-1 error, while the error of VLSI plunges to about 7%. At 125 dimensions, LSI remains at 56% the initial error while VLSI has dropped to under 3%. Viewed alternatively, the error rate of LSI with 250 dimensions is matched by the error of VLSI using only 10 dimensions.

As expected, when the power law is planted over a random permutation of the terms in the corpus, the results are more dramatic. At 50 dimensions, the error of LSI has dropped by barely 5% from a rank-1 approximation. At 22 dimensions, the error of VLSI is only 7% as great (i.e., down by 93%), and by 50 dimensions, it is between 1 and 2%. Comparing

[1]In both D2 and D3, the power law with exponent 2.4 is equivalent to a Zipf exponent of 0.714 on the rank. $c$ is a normalizing constant.

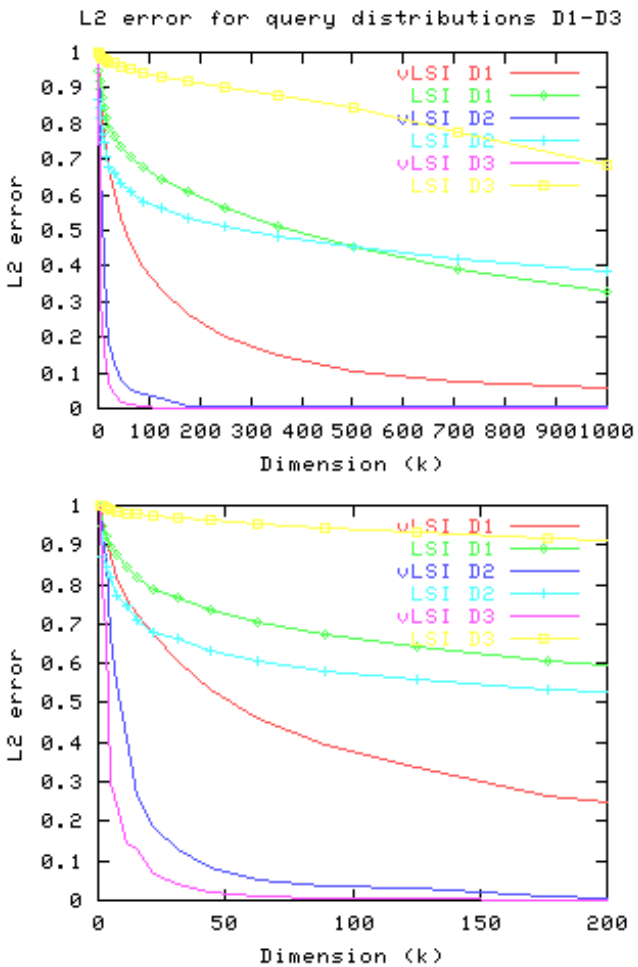


**Figure 2: LSI and VLSI for query distributions D1–D3. The first plot shows results for 1000 dimensions while the second plot show a more detailed view of the first 200 dimensions.**

results at 1000 dimensions, the error of VLSI is four orders of magnitude smaller. This is an extreme instance, and not representative of actual query logs.

**Boolean versus Okapi weighting.** We may ask whether this significant improvement in approximation for a specific query distribution is an artifact of the Okapi weighting introduced into the matrix. Fixing upon distribution D2, which contains a planted power law over the terms with the same rank order as shown in the corpus distribution of term frequencies, we compare the Okapi-weighted matrix $A^O$ to the Boolean matrix $A^b$. Figure 3 shows the results. Dropoffs are much faster for the Boolean matrix for both LSI and VLSI. The error rate for VLSI begins at about half that of LSI, and by 100-150 dimensions, it has dropped to about 10% of the error rate of LSI for a similar number of dimensions.

**Topic-specific query distributions.** Turning to topic-specific keyword distributions, we would expect that such

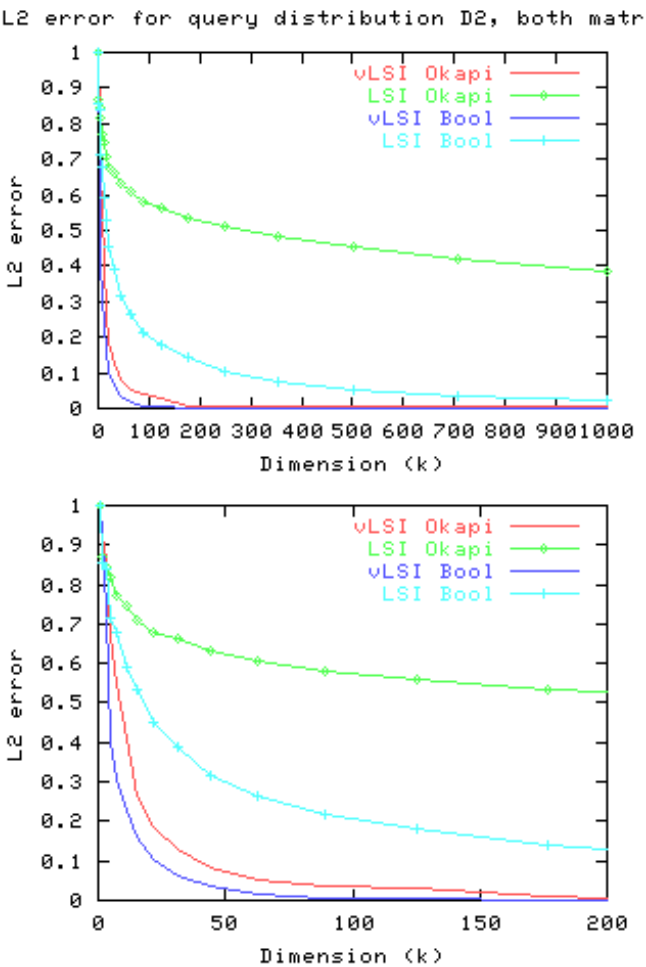


**Figure 3: LSI and VLSI for query distribution D2 over both Okapi-weighted matrix and Boolean matrix. The first plot shows results for 1000 dimensions while the second plot show a more detailed view of the first 200 dimensions.**

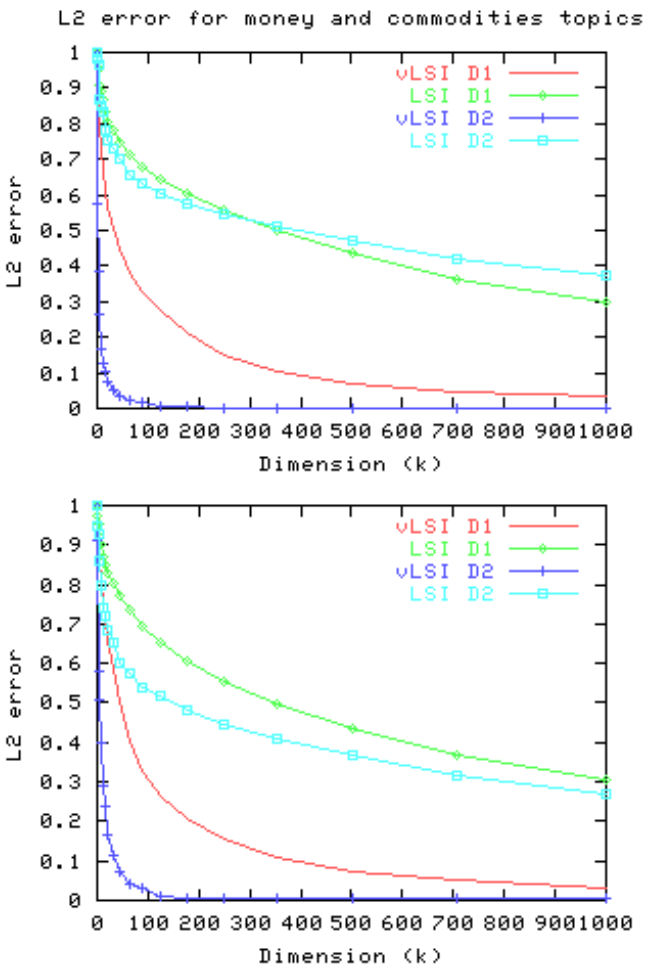


**Figure 4: LSI and VLSI over 1000 dimensions. Top figure shows query distributions money.D1 and money.D2; bottom figure shows commodity.D1 and commodity.D2.**

distributions focus on specific types of keywords and are thus an appropriate arena for VLSI. Figure 4 shows the results for the `money` and `commodity` topics. The combination of topic focus and planted power law results in low-rank highly-accurate approximations in the VLSI case: the approximation shows only 10% of the initial error at as few as 15 dimensions, and only 1% by 100 dimensions. LSI's performance at 250 dimensions is approximately similar to that of VLSI at 25 dimensions. In money.D1, in which the query distribution follows the corpus distribution within the money topic, the improvements are slower. VLSI with 25 dimensions again is comparable to LSI with 250 dimensions, but in this case LSI with 1000 dimensions (and VLSI with 100 dimensions) still show 30% of the original error.

**Competitive precision.** So far, we have considered the $L_2$ error of the approximation. However, we may also consider competitive precision, which is our rank-oriented measure: how do various matrix approximations modify the rank order in which documents are returned? Figure 5 shows the results for query distributions D1 and D2, reporting competitive error, i.e., one minus competitive precision. The competitive error measure does not tend quickly to zero as tiny differences in the approximation may have significant impact on the rank order. The queries we study tend to match a relatively large number of documents, and the ordering of those documents in the original and approximate matrices is largely random until the number of dimensions becomes extremely high. The two cases show different behaviors: D2 drops off extremely quickly due to the skewed nature of the query distribution, but for each competitive error parameter, flattens to a particular value dependent on the size of the result set for different queries weighted by the query probability. In D1, VLSI at 100 dimensions attains a competitive error comparable to LSI at 1000 dimensions. The same condition holds in D2, but in this case it is more instructive to note that both schemes have flattened by around 100 dimensions, and the difference in competitive error is about a factor of 2 in favor of VLSI.

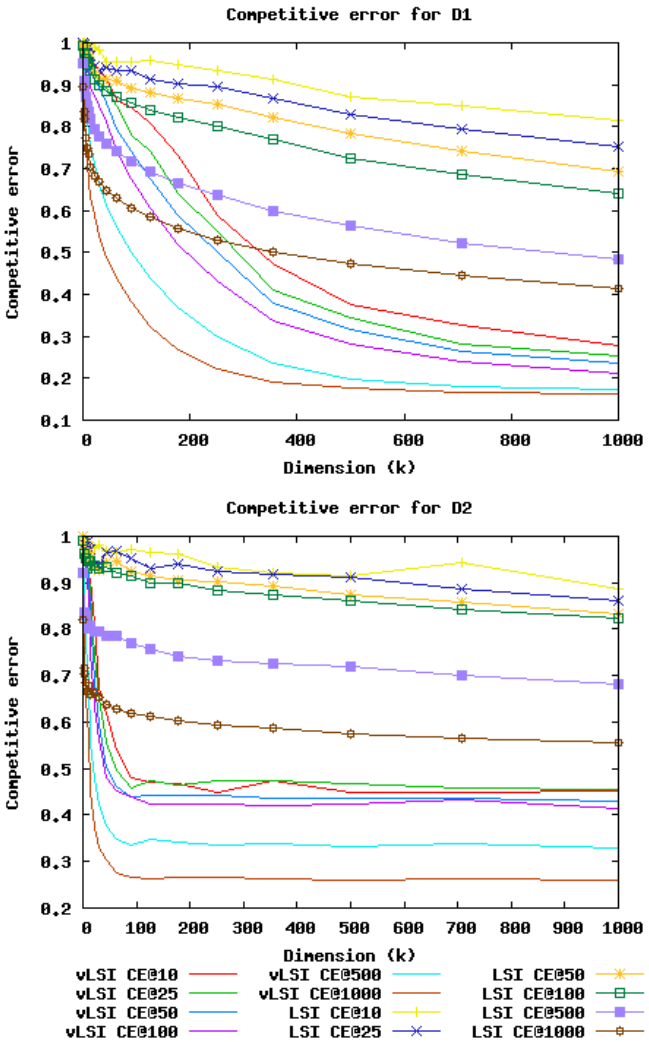


**Figure 5: LSI and VLSI over 1000 dimensions with query distribution D1 (above) and D2 (below) measured by competitive error at 10 through 1000 documents. Plots with points correspond to LSI; those without points correspond to VLSI.**

**Uniform measurements.** The results so far make heavy use of the significant skew encountered in query logs. However, VLSI also shows an improvement when measured in the following manner. Imagine drawing 100 queries from the query distribution without replacement, thus significantly reducing the skew. The results for distribution D2 are shown in Figure 6. The quality of approximation in either metric for LSI at 1000 dimensions is about equivalent to that of VLSI at 100 or fewer dimensions.

**Multi-word queries.** All experiments so far have been performed on single-term query distributions. In this case, the formulation in Section 2 has a clean form and is straightforward to implement. However the formulation is more general, and allows the query distribution to be over arbitrary vectors. Therefore, we consider a small experiment over two-word queries. The distribution is computed as follows.

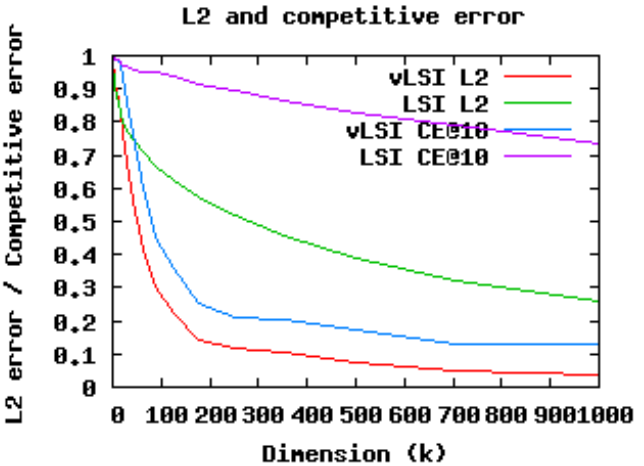


**Figure 6: LSI and VLSI over 1000 dimensions with query distribution D2 measured via 100 queries drawn from the distribution without replacement.**

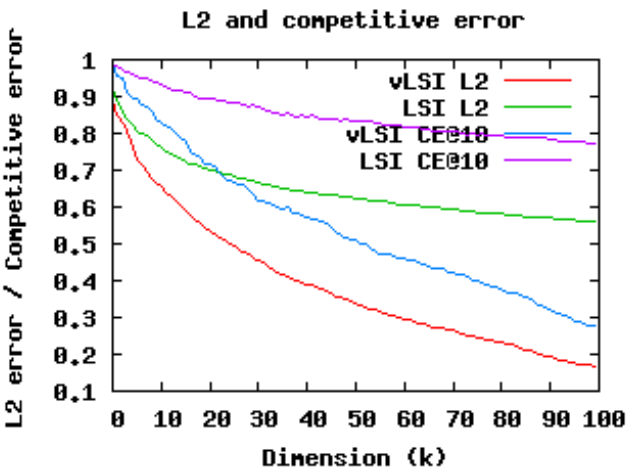


**Figure 7: LSI and VLSI over 100 dimensions with two-word query distribution.**

First, the documents for the `commodity` topic are scanned, and the counts of all bigrams are kept. The top 25 bigrams are dropped as they are very frequent without being meaningful queries. The remaining bigrams are ranked according to frequency, and a power law with exponent 2.4 is planted on this ranking, as we did in the single-term case for query distribution D2. The results of this experiment are shown in Figure 7 for both $L_2$ distance and competitive error at 10. Due to computational constraints, we completed the experiment for 100 dimensions only. About ten dimensions of VLSI produced the same $L_2$ error and competitive error as 100 dimensions of LSI.

## 4. CONCLUSIONS

Beginning with the observation that latent semantic indexing (and its precursors) are data-dependent but query-oblivious, we developed a new form of low-rank approximation that is query-dependent. Experiments with our new approximation

are extremely encouraging, suggesting an order of magnitude improvement in the quality of approximation *for the purposes of serving queries*. A number of further directions arise:

- How do we combine our technique with an algorithm that learns/adapts to a query distribution over time?
- Our formal development suggests a particular optimal low-rank approximation; the computational requirements for this requirement are fairly high. Could one compute, quickly, an approximation to this approximation that is nearly as good for retrieval effectiveness on the given query distribution?
- We have established empirically a dramatic improvement in the domain of text documents. Can one replicate this experimental success in other domains?

## 5. REFERENCES


[1] R. Baeza-Yates. Web usage mining in search engines. In A. Scime, editor, *Web Mining: Applications and Techniques*, chapter XIV. Idea Group, 2004.

[2] M. Berry, T. Do, G. O'Brien, V. Krishna, and S. Varadhan. SVDPACKC (version 1.0) user's guide. Technical Report CS-93-194, University of Tennessee, 1993.

[3] S. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *Journal of the Society for Information Science*, 41(6):391–407, 1990.

[4] S. T. Dumais. LSI meets TREC: A status report. In *The First Text REtrieval Conference (TREC1), National Institute of Standards and Technology*, pages 137–152, 1992.

[5] S. T. Dumais. Latent semantic indexing (LSI) and TREC-2. In *The Second Text REtrieval Conference (TREC2), National Institute of Standards and Technology*, pages 105–116, 1993.

[6] S. T. Dumais. Latent semantic indexing (LSI): TREC-3 report. In *The Third Text REtrieval Conference (TREC3), National Institute of Standards and Technology*, pages 105–115, 1994.

[7] S. T. Dumais, G. Furnas, T. Landauer, and S. Deerwester. Using latent semantic analysis to improve information retrieval. In *Proceedings of ACM Conference on Computer Human Interaction (CHI)*, pages 281–285, 1988.

[8] C. Eckart and G. Young. The approximation of a matrix by another of lower rank. *Psychometrika*, 1:211–218, 1936.

[9] G. H. Golub and C. F. V. Loan. *Matrix Computation*. John Hopkins University Press, 1991. Second Edition.

[10] T. Hoffmann. Matrix decomposition techniques in machine learning and information retrieval. `http://www.mpi-sb.mpg.de/~adfocs/adfocs04.slides-hofmann.pdf`, 2004.

[11] T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 50–57, 1999.

[12] T. Hofmann. Latent semantic models for collaborative filtering. *ACM Transactions on Information Systems*, 22(1):89–115, 2004.

[13] I. T. Jolliffe. *Principal Component Analysis*. Springer, 2002.

[14] D. D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401:788–791, 1999.

[15] R. Lempel and S. Moran. Predictive caching and prefetching of query results in search engines. In *Proceedings of the 12th International World Wide Web Conference (WWW)*, pages 19–28, 2003.

[16] D. D. Lewis. `http://www.daviddlewis.com/resources/testcollections/reuters21578/`.

[17] S. Mika, B. Schlkopf, A. Smola, K.-R. Mller, M. Scholz, and G. Rtsch. Kernel PCA and de-noising in feature spaces. In *Advances in Neural Information Processing Systems 11 (NIPS)*, pages 536–542, 1998.

[18] B. Moghaddam and A. Pentland. Face recognition using view-based and modular eigenspaces. In *Automatic Systems for the Identification and Inspection of Humans, SPIE,*, volume 2277, pages 12–21, 1994.

[19] S. E. Robertson and S. Walker. Okapi/Keenbow at TREC-8. In *The Eighth Text REtrieval Conference (TREC8), National Institute of Standards and Technology*, 1999.

[20] P. C. Saraiva, E. S. de Moura, N. Ziviani, W. Meira, R. Fonseca, and B. Riberio-Neto. Rank-preserving two-level caching for scalable search engines. In *Proceedings of the 24th ACM Conference on Research and Development in Information Retrieval (SIGIR)*, pages 51–58, 2001.

[21] A. Singhal. Modern information retrieval: A brief overview. *IEEE Data Engineering Bulletin*, 24(4):35–43, 2001.

[22] P. K. C. Singitham, M. S. Mahabhashyam, and P. Raghavan. Efficiency-quality tradeoffs for vector score aggregation. In *Proceedings of the 30th International Conference on Very Large Data Bases (VLDB)*, pages 624–635, 2004.

[23] N. Srebro and T. Jaakkola. Weighted low-rank approximations. In *Proceedings of the 20th International Conference on Machine Learning (ICML)*, pages 720–727, 2003.

[24] M. Tipping and C. Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society, Series B*, 61(3):611–622, 1999.

[25] J. Ye. Generalized low-rank approximations of matrices. In *Proceedings of the 21st International Conference on Machine Learning (ICML)*, 2004.