Approximate Dimension Equalization in Vector-based Information Retrieval

Fan Jiang
Department of Computer Science, Duke University, Durham, NC 27708 USA

Michael L. Littman
AT&T Labs Research, Florham Park, NJ 07932-0971 USA
Department of Computer Science, Duke University, Durham, NC 27708 USA

Abstract
Vector-based information retrieval methods such as the vector space model (VSM), latent semantic indexing (LSI), and the generalized vector space model (GVSM) represent both queries and documents by high-dimensional vectors learned from analyzing a training collection of text. VSM scales well to large collections, but cannot represent term-term correlations, which prevents it from being used in cross-language retrieval. GVSM and LSI can represent term-term correlations, but do not scale well to large collections. We point out a deep mathematical similarity between VSM, LSI, and GVSM, and use this to derive a novel method we call approximate dimension equalization (ADE) that performs well on large collections, scales well computationally, and can represent term-term correlations. We compare the performance of ADE to the other methods on both large and small collections with both monolingual and cross-language queries. ADE outperforms all other methods on large cross-language collections, and is close to the best in all other cases.

This is especially important in cross-language IR applications. While LSI has shown impressive performance in many applications, its retrieval performance on extremely large and diverse text collections has lagged behind that of VSM. We present some evidence that this is because of the large number of dimensions needed for accurate retrieval in large collections. GVSM also models term-term associations, and it scales much more easily than LSI. Unfortunately, its performance also lags substantially behind that of VSM and often LSI. We argue that this is because, although GVSM uses more dimensions than LSI, it puts tremendous weight on the largest ones, resulting in an effective dimensionality that is substantially smaller.

We show how VSM, LSI, and GVSM can be viewed as variations of a general algorithm that projects documents and queries into a vector space defined by the singular vectors (principal dimensions) of the matrix defined by the training collection. The algorithms differ in the weights they assign to the dimensions of the vector space. We introduce a new method in this same class we call approximate dimension equalization (ADE) that scales well and supports term-term associations. On small retrieval collections, ADE performs best or close to the best. On large monolingual collections, ADE’s performance approaches that of VSM. On large cross-language collections, to which VSM cannot be applied, its performance is a substantial improvement over both GVSM and LSI.

1. Introduction
Three basic vector-space methods for information retrieval (IR) are the vector space model (VSM), latent semantic indexing (LSI), and the generalized vector space model (GVSM). Each has its own unique strengths. VSM is simple, scales extremely well, and gives excellent performance on large text retrieval collections. LSI extends VSM by learning a reduced dimensional representation that models term-term associations, thus allowing a query to have a positive similarity to a document with which it shares no terms.

2. Vector-Based Information Retrieval
Information retrieval (IR) is the task of locating information items (documents) from a retrieval collection, which are relevant to a user query. Cross-language (or translanguag) information retrieval is the problem of selecting useful documents from a retrieval collection in one language by a query in another.
Vector-based information retrieval methods represent both queries and documents by high-dimensional vectors and compute their similarity by the vector inner product. When the vectors are normalized to unit length, then the inner product is equal to measuring the cosine of the angle between the two vectors in the vector space. The components of the vectors can be term weights, as in VSM, developed by Salton et al. (1975), or they can be transformed into another space whose dimensions bear some new meanings, as in, for example, GVSM (Wong et al., 1985) and LSI (Deerwester et al., 1990).

Mathematically, vector-based IR methods can be formulated in the following way:

$$\text{Sim}_P(d,q) = (P^T \hat{d}) \cdot (P^T \hat{q}) = \hat{d}^T \hat{q} = \hat{d}^T \hat{q} \quad (1)$$

where $\text{Sim}_P(d,q)$ represents the similarity between a document and a query, $\hat{d}$ is the vector representation of document $d$, $\hat{q}$ is the vector representation of document $q$, and $P$ is a transformation matrix.

The vector $\hat{d}$ in Equation 1 is usually obtained this way: a training collection is indexed into a term-by-document matrix $D = [d_{ij}] \in \mathbb{R}^{m \times n}$, where $d_{ij}$ is the weight of the $i$-th term in the $j$-th document, $m$ is the size of the vocabulary and $n$ is the number of documents in the training collection. Then, the $j$-th column of the matrix $D$ is an $m$-dimensional vector of term weights. A user query can be represented in a similar way. In the conventional VSM, the transformation matrix $P$ is simply the $m \times m$ identity matrix:

$$\text{Sim}_{VSM}(d,q) = (I^T \hat{d}) \cdot (I^T \hat{q}) = \hat{d}^T \hat{q}$$

A criticism of VSM is that it treats terms as unrelated in that they occupy orthogonal dimensions in the vector space. A nonzero similarity score between two vectors will result only if they both have nonzero values in at least one dimension, i.e., their original documents contain at least one word in common. This is far from ideal because, for example, for a query that contains the search word “computer” but not “digital,” a document that uses exclusively the word “digital” will be deemed no more relevant than a document that talks about gardening. It is also obvious that the exact term matching scheme of VSM is not suited for cross-language information retrieval.

Techniques that address this problem often use a training collection to derive statistical information about term-term correlations and related terms can be added to the original query to enrich retrieval. Ideally, the training collection would be a thesaurus that lists for every word entry all the related words and how strong the relationships are. At the other extreme, this “thesaurus” would only contain word entries followed by no related words at all, and this is exactly analogous to conventional VSM. In cross-language information retrieval, a bilingual training collection can be used for learning term-term associations, where corresponding documents in the two collections are translations of each other or are on the same or related subjects.

The generalized vector space model (GVSM) (Wong et al., 1985), also known as “the dual space” approach (Sheridan & Ballerini, 1996), is a method that captures term-term correlations from the documents (or matching document pairs, in the case of cross-language retrieval) they co-occur in. In monolingual retrieval, let $A \in \mathbb{R}^{m \times n}$ be the term-document matrix of the training collection (the training matrix). Matching the vector elements in $\hat{d}$ and rows of $A$ by the terms they represent, $A^T \hat{d}$ transforms $\hat{d}$ into a new vector whose elements correspond to the $n$ documents in the training collection. The query vector $\hat{q}$ can also be transformed by $A$ in the same way. Then, the query-document similarity is measured between the transformed vectors:

$$\text{Sim}_{GVSM-ML}(d,q) = (A^T \hat{d}) \cdot (A^T \hat{q}) = \hat{d}^T A A^T \hat{q} \quad (2)$$

where the $m \times m$ matrix $AA^T$ has a nonzero value in its row $i$ and column $j$ if and only if there is a document in $A$ that contains both the $i$-th and $j$-th terms.

The extension of GVSM to cross-language IR was proposed by Yang et al. (1998). Using a bilingual collection for training, two matrices $A$ and $B$ are formed, where $A$ is a term-document training matrix in the language of the retrieval documents, and $B$ is a parallel term-document training matrix in the language of the queries. While the number of unique terms in the two languages are different, the number of documents in the training collection is the same, and are represented by the corresponding columns of $A$ and $B$. Thus, when document $\hat{d}$ is transformed by $A$ and query $\hat{q}$ by $B$, we can compute their inner product:

$$\text{Sim}_{GVSM-CL}(d,q) = (A^T \hat{d}) \cdot (B^T \hat{q}) = \hat{d}^T A B^T \hat{q} \quad (3)$$

Latent semantic indexing (LSI) (Deerwester et al., 1990) is another vector-based IR method that uses a training collection. Given a term-document training matrix $A$, it uses the singular value decomposition (SVD) (Golub & Van Loan, 1989) to factor it into three parts:

$$A = U \Sigma V^T = U \text{diag}(\sigma_1, \sigma_2, \ldots, \sigma_r) V^T$$

\textsuperscript{1} Query expansion techniques such as global analysis and local feedback can be used to deal with the word mismatch problem in VSM (Xu & Croft, 1996).
where $U$ and $V$ are unitary matrices (i.e., $U^T U = I$ and $V^T V = I$) whose columns are the left and the right singular vectors of $A$, $\Sigma$ is a diagonal matrix whose diagonal elements are non-negative and arranged in descending order, and $r$ is the rank of $A$ (see Figure 1). The values $\sigma_1, \ldots, \sigma_r$ are known as the singular values of $A$, and are the square roots of the eigenvalues of $A^T A$ and $A A^T$. The first $k$ ($k \leq r$) columns of $U$ that correspond to the $k$ largest singular values of $A$ together are the transformation matrix for LSI:

$$\text{Sim}_{\text{LSI}}(d, q) = (I_k U^T \tilde{d}) \cdot (I_k U^T \tilde{q}) = \tilde{d}^T U I_k U^T \tilde{q},$$

where $I_k$ is like the identity matrix $I$ except that only its first $k$ diagonal elements are nonzero (one). When $k < r$, we say we performed dimension reduction on $U$ (or $A$). The use of the columns of $U$ as the transformation matrix amounts to an orthogonal projection of queries and documents into the row space of the training matrix. In this way, LSI learns the structure of the training collection, and uses this in making new judgments of query-document relatedness.

Computationally, a $k$-dimensional SVD of $A$ returns $U_k$, $\Sigma_k$, and $V_k$, which are the first $k$ columns of $U$, $\Sigma$, and $V$. Using these, computing Equation 4 requires only two $(1 \times n) \times (n \times k)$ matrix multiplications and a dot product between two $k \times 1$ vectors; no $n^2$ or $mn$ computations need be performed.

3. Dimension Equalization

Equation 1 provides a general form for vector-based information retrieval using a general transformation matrix $P$. For each of the methods described in Section 2, however, the matrix $P$ can be written in a more specific form in terms of the singular value decomposition of the term-document training matrix $A = U \Sigma V^T$. In particular, for all three methods, $P = U \Phi V^T$ where $\Phi$ is a diagonal matrix of dimension weights. So, for this class of retrieval methods, we can write

$$\text{Sim}_\Phi(d, q) = (V \Phi U^T \tilde{d}) \cdot (V \Phi U^T \tilde{q}) = \tilde{d}^T U \Phi^2 U^T \tilde{q},$$

since $V^T V = I$. So, VSM, GVSM, and LSI all project queries and documents into the row space of the training matrix ($U$), but differ in how they weight the dimensions of this space ($\Phi$).

To show this, we consider each retrieval method in turn. For GVSM, from Equation 2, we have $P = A = U \Sigma V^T$. Thus, GVSM can be expressed as Equation 5 with $\Phi = \Sigma$. For LSI, from Equation 4, we have $P = U I_k$. Thus, LSI can be expressed as Equation 5 with $\Phi = I_k$. For VSM, note what happens if we set $\Phi = I$. Equation 5 becomes $\tilde{d}^T U U^T \tilde{q}$, which is equivalent to VSM’s similarity of $d^T \tilde{q}$ if either the training matrix has more documents than terms (since $U^T U = I$ in that case) or if $\tilde{d}$ is a document from the training set, since its projection onto the column space of $A$ is $d$ itself ($U U^T d = \tilde{d}$). Since VSM generally uses the retrieval collection for training, this second case is almost always applicable.

Derivations for the cross-language versions are similar, where the similarities are expressed in terms of the singular value decompositions of the training matrices $A$ and $B$.

Table 1 summarizes the observations of the previous paragraphs, providing the dimension weights $\Phi$ for each of the retrieval methods discussed. We call LSI and VSM dimension equalization methods, since they assign equal weights (1) to all dimensions included in the similarity computation.

Note that the formulation of cross-language LSI in Table 1 is different from the original formulation by Landauer and Littman (1990). Their cross-language LSI formula includes a single SVD of the matrix that combines aligned documents from both training collections into single documents:

$$\begin{bmatrix} A \\ B \end{bmatrix} = U_{AB} \Sigma_{AB} V_{AB}^T.$$

The vectors $\tilde{d}$ and $\tilde{q}$ can be extended (with all zeros) to cover terms in the other language. Compared to this traditional cross-language LSI, the approach suggested in Table 1 computes two separate SVDs of smaller matrices instead. This comes in handy when the combined matrix becomes too large to analyze via SVD. When $k = n$, the number of dimensions used in LSI is equal to the number of documents in the training collection, this new cross-language LSI is a special form of the Procrustes method (Littman et al., 1998).
Table 1. Basic vector-based information retrieval methods in terms of the weights they assign to the dimensions of the training matrix $A$. $B$ represents a parallel training matrix in the cross-language case. $U$ and $\Sigma$ are the left singular vectors and singular values of $A$ and $U_B$ and $\Sigma_B$ are the left singular vectors and singular values of $B$.

<table>
<thead>
<tr>
<th>query-document similarity</th>
<th>Monolingual $(\Phi U^T d) \cdot (\Phi U^T q)$</th>
<th>Cross-Language $(\Phi_1 U^T d) \cdot (\Phi_2 U^T q)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>GVSM</td>
<td>$\Phi = \Sigma$</td>
<td>$\Phi_1 = \Sigma, \Phi_2 = \Sigma_B$</td>
</tr>
<tr>
<td>VSM</td>
<td>$\Phi = I$</td>
<td>N/A</td>
</tr>
<tr>
<td>LSI</td>
<td>$\Phi = I_k$</td>
<td>$\Phi_1 = I_k, \Phi_2 = I_k$</td>
</tr>
</tbody>
</table>

A natural question to ask at this point is, what is the “right” value of $\Phi$ to use in Equation 5? In some sense, the answer is whichever gives the best empirical results. But, we can make a motivated prediction based on linear algebra. Ideally, our transformation matrix should project queries and documents into the row space of the training matrix, since this is our source of information on term–term relatedness. However, the least significant dimensions of the row space (those corresponding to the smallest singular values) are likely to be modeling “noise”—variability in term usage that does not correspond to a significant change in meaning. Thus, we might want to project documents and queries into all but the least significant dimensions of the row space. This is achieved by setting $\Phi = I_k$, where $k^*$ is the optimal number of dimensions to include. The experiments in Section 6 suggest that this is close to the full rank of $A$.

4. Comparison of Singular Values

If the singular values of the training matrix $A$ were to somehow be all the same, then the formulae for GVSM and VSM in Table 1 would give the same values. Similarly, LSI and VSM are the same if the dimensionality $k$ is chosen equal to the rank $r$ of $A$. Therefore, the differences we observe between VSM, GVSM, and LSI in practice will depend on the distribution of the singular values of the training matrix.

Figure 2 shows the singular values of the small 1,121 UNICEF English test documents used by Yang et al. (1998). The collection is indexed with the SMART mtc weighting scheme (Yang et al., 1998) and a complete set of 1,103 singular values is easily found from the training matrix. The characteristic of this plot is that it possesses the so-called low-rank-plus-shift structure, reported by Zha and Zhang (1999) in their study of LSI: the singular values decrease sharply at first, level off noticeably, and dip abruptly at the end. We have plotted the singular values of dozens of collections, varying size, language and indexing scheme, and all seem to have this special property.

![Figure 2](image)

Figure 2. Plot of all singular values of the UNICEF English collection of 1,121 documents.

When the training matrix size gets too large to compute the complete set of singular values using current software, we still see the trend of initial dropping and leveling off of singular values. Plotted in Figure 3 are the first 800 singular values of the TREC AP 1990 collection of 78,321 documents with Okapi term weighting (Robertson et al., 1995). We hypothesize that the singular values eventually do drop to near zero.

Given that the singular values take this basic shape, Figure 4 depicts the dimension weights used by each of the vector-based IR methods listed in Table 1 (and a fourth, explained in the next section).

As argued in Section 3, we'd predict that the ideal dimension weights would be flat up to some value $k^*$ less than the full rank of the training matrix $A$, but perhaps much larger than what can be computed using current software. This matrix $I_k^*$ is depicted by the dotted line in Figure 4.

While none of VSM, GVSM, and LSI match the idealized dimension weights, based on Figure 4 we'd expect
VSM to produce the most accurate similarity scores, followed by LSI, which matches the ideal weights for the dimensions with the largest singular values. However, VSM cannot be used directly for cross-language retrieval. In the next section, we develop a novel approximation to the idealized dimension weights.

5. Approximate Dimension Equalization

For a matrix $A$ with singular values $\Sigma$ and a number $k \leq r$, define

$$\hat{I}_k = I_k + \frac{1}{\sigma_k} \Sigma - \frac{1}{\sigma_k} \Sigma_k.$$ 

This diagonal matrix is illustrated graphically in Figure 5.

The matrix $\hat{I}_k$ takes advantage of the special shape of the singular value plots of all the term-document matrices we have seen to approximate $I_k$; it flattens out the first $k$ very large singular values and attaches the rest of the real singular values, which are a relatively flat and long middle portion with a small dipping tail. Using $\hat{I}_k$ as the dimension weights $\Phi$ in Equation 5, we obtain relatively equalized dimensions of the training matrix until close to the rank of $A$. We call this approach approximate dimension equalization (ADE). The dimension weights of the ADE method are also plotted in Figure 4; it is predicted that ADE approximates the ideal singular values better than either LSI or GVSM. From one perspective, ADE is trying to extend the limited ability of LSI to compute the singular vectors and values of a large training matrix by implicitly adding additional ones with relatively equal weights (“extrapolating” the singular values, in a way). From another, ADE makes cross-language VSM possible, obtaining most if not all the dimensions of both training matrices with equalization. From a third, it uses GVSM’s approach to scalably capturing term-term correlations, modified to prevent overemphasizing the first handful of dimensions.

It remains to show that Equation 5 using $\hat{I}_k$ can be computed efficiently and to evaluate its performance on real collections.

Computationally, ADE only needs to compute the first $k$ singular values and left singular vectors of the training matrix, much like $k$-dimensional LSI. It uses them...
to compute similarities in the following way:

$$\text{Sim}_{\text{ADE-ML}}(d,q) = \frac{1}{\sigma_k} \Sigma - \frac{1}{\sigma_k} \Sigma_k^2 U^T q$$

$$= \frac{1}{\sigma_k} \Sigma_k U_k^T q - \frac{1}{\sigma_k} \Sigma_k U_k^T q$$

$$= \text{Sim}_{\text{LSI-ML}}(d,q) + \frac{1}{\sigma_k} \text{Sim}_{\text{GVSML-ML}}(d,q)$$

$$- \frac{1}{\sigma_k} ((\Sigma_k U_k^T q) - (\Sigma_k U_k^T q)),$$

Thus, the ADE similarity is a weighted combination of LSI and GVSML, subtracting off the extra weight due to GVSML. All of these are efficiently computable.

Similarly, ADE for cross-language retrieval is:

$$\text{Sim}_{\text{ADE-CL}}(d,q) =$$

$$\frac{1}{\sigma_k} \Sigma - \frac{1}{\sigma_k} \Sigma_k^2 U^T q$$

$$- \frac{1}{\sigma_k} ((\Sigma_k U_k^T q) - (\Sigma_k U_k^T q)),$$

where $\Sigma_B$ and $U_B$ are the singular values and left singular vectors of the parallel training matrix $B$.

6. Results

In our experiments, we tested all four methods on various training collections of different size and in different languages. We have used both the traditional $tf \times idf$ and the more recent Olapi weighting schemes, and have presented our results for whichever weighting scheme works better with the conventional VSM. We tested both small and large collections and ran monolingual and cross-language comparisons.

6.1 Monolingual

Table 2 shows the results for monolingual retrieval on two different collections: Cranfield and TREC AP 1990. The numbers in the table are the non-interpolated average precision score over a set of standard test queries and precision at document cutoff of 10; both are standard measures in IR with larger values preferred. The Cranfield collection consists of 1,400 documents, 3,763 unique terms, and 225 test queries. A complete SVD of the training matrix takes about 20 minutes to compute on an AlphaStation with one 266Mhz processor using a standard sparse SVD package. The Cranfield collection demonstrates the ability of LSI to exploit term-term correlations. LSI has an average precision of 0.4121 at 300 dimensions, with the full rank being 1,400, while ADE’s near best score of 0.4115 is obtained at 75 dimensions. To obtain these best retrieval scores and to examine the effects of dimension reduction, we ran retrieval with LSI and ADE varying the number of singular values from 50 to 1,400. A plot of 11-point average precision vs. number of dimensions used for these two methods is shown in Figure 6, with the score of VSM included for comparison. Here, ADE’s plot looks like a shift-forward image of that of LSI, confirming that it approximates LSI with fewer singular values, “extrapolating” based on the values computed.

Although Figure 6 shows that the performance of LSI and ADE on small collections is sensitive to the number of singular values used, we have not found this to be true for large collections. The 78,321-document TREC AP 1990 collection is a subset of the TREC AP collection. This size is too large to compute a complete SVD of its term-document training matrix. For this collection, 800 singular values with correspond-
ing vectors were calculated in about 21 hours on an SGI machine with four MIPS R10000 2.5 processors; all were used in LSI and ADE. This is only about 1% of the rank and we find that LSI scores substantially less than VSM. But, ADE partially bridges the gap between LSI and VSM, achieving results close to VSM with the same number of computed singular values as LSI. As we have predicted, GVSM behaves like a very low dimensional LSI on both collections. We also included the average precision vs. dimension plot for this collection. In Figure 7, we see that the results for LSI and ADE are strictly increasing, as they are still trying to “catch up” with the performance of VSM using a small number of singular values.

6.2 Monolingual and Cross-Language

Our second set of results, listed in Table 3, shows the vector-based methods on both monolingual (English) and cross-language IR on a small collection. The experiments used the UNICEF collection created and described by Yang et al. (1998). There are 1,134 training documents in addition to 1,121 test documents, each in both English and Spanish, and these documents are aligned in both sets. Queries are available in both languages. In Table 3, $A = D$ means the training is done on the retrieval collection itself (generally possible only in monolingual retrieval), and $A \neq D$ means a separate training collection was used. As in the case of the Cranfield collection, we can compute the complete SVD of the training matrices (in about 18 minutes on the same SGI machine mentioned earlier). We varied the number of singular values ($k$) and tested LSI and ADE to find the best possible results. While LSI gets its best retrieval scores using 180, 900, and 900 singular values (for the three average precision scores in the row for LSI), ADE achieves similar results with fewer singular values (80, 800, and 400). This suggests, again, that ADE is able to “extrapolate” based on the singular values it is given. It is also interesting to note that the number of singular values needed in both methods is smallest when training and retrieval use the same matrix $(A = D)$. We currently do not have an explanation of this.

Table 4 summarizes results for the various retrieval methods on a much larger cross-language collection, the TREC French and German collections of 185,082 and 141,643 documents, respectively. Queries are available in both languages. The training documents are a subset of 40,000 pairs on approximately the same topics as used by Rehler et al. (1997). We computed and used 1,700 dimensions (which is still only a little more than 4% of the full rank), and found both LSI and ADE lagging behind VSM in monolingual retrieval, with ADE improving upon LSI. But, while VSM cannot be applied to cross-language retrieval, ADE obtains very strong results in that territory; ADE outperforms the best of LSI and GVSM on 19 out of 21 queries ($p < .0002$ by a sign test). In fact, ADE’s results compare favorably with the very best results obtained for this collection (Franz et al., 1999), which required passage-aligned text to train a statistical machine translation system.

7. Conclusion

This paper shows that three traditional vector-based methods of information retrieval, VSM, LSI, and

<table>
<thead>
<tr>
<th></th>
<th>Monolingual</th>
<th>Cross-Language</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$A = D$</td>
<td>$A \neq D$</td>
</tr>
<tr>
<td></td>
<td>$A \neq D$</td>
<td>$A \neq D$</td>
</tr>
<tr>
<td>GVSM AvgP</td>
<td>0.4165</td>
<td>0.3719</td>
</tr>
<tr>
<td>P10</td>
<td>0.4172</td>
<td>0.4009</td>
</tr>
<tr>
<td>VSM AvgP</td>
<td>0.4562</td>
<td>0.4468</td>
</tr>
<tr>
<td>P10</td>
<td>0.3931</td>
<td>0.4000</td>
</tr>
<tr>
<td>LSI AvgP</td>
<td>0.4697</td>
<td>0.4519</td>
</tr>
<tr>
<td>P10</td>
<td>0.4276</td>
<td>0.4207</td>
</tr>
<tr>
<td>ADE AvgP</td>
<td>0.4697</td>
<td>0.4596</td>
</tr>
<tr>
<td>P10</td>
<td>0.4207</td>
<td>0.4172</td>
</tr>
</tbody>
</table>
Table 4. Monolingual and cross-language average precision results for TREC French-German collections with TREC-6 cross-language topics. Top scores are shown in boldface.

<table>
<thead>
<tr>
<th></th>
<th>Monolingual</th>
<th>Cross-Language</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Pre</td>
<td>Ger</td>
<td>G → F</td>
</tr>
<tr>
<td>GVSM</td>
<td>AvgP</td>
<td>0.0509</td>
<td>0.0101</td>
</tr>
<tr>
<td></td>
<td>P10</td>
<td>0.1333</td>
<td>0.0174</td>
</tr>
<tr>
<td>VSM</td>
<td>AvgP</td>
<td>0.3080</td>
<td>0.2794</td>
</tr>
<tr>
<td></td>
<td>P10</td>
<td>0.4619</td>
<td>0.3870</td>
</tr>
<tr>
<td>LSI</td>
<td>AvgP</td>
<td>0.1711</td>
<td>0.1025</td>
</tr>
<tr>
<td></td>
<td>P10</td>
<td>0.2190</td>
<td>0.1609</td>
</tr>
<tr>
<td>ADE</td>
<td>AvgP</td>
<td>0.2004</td>
<td>0.1635</td>
</tr>
<tr>
<td></td>
<td>P10</td>
<td>0.3095</td>
<td>0.2304</td>
</tr>
</tbody>
</table>

GVSM, can be viewed as belonging to a family of algorithms that differ only in how they weight the dimensions of the training matrix when making similarity computations. Ignoring computational concerns, LSI is the most desirable method to use because, like VSM, it projects queries and documents into the row space of the training collection, and, unlike VSM, it takes advantage of term-term correlations. But, as the training collection size increases, so does the number of dimensions required by LSI, and computational concerns dominate. With approximate dimension equalization (ADE), a new method that combines ideas from VSM, LSI, and GVSM, we can approximate the ideal dimension weights more accurately using available computational resources. Our results show that ADE can improve upon LSI in both monolingual and cross-language information retrieval.

References


