2.6 Latent Semantic Indexing

Matrix computation is used as a basis for information retrieval in the retrieval strategy called Latent Semantic Indexing [Deerwester et al., 1990]. The premise is that more conventional retrieval strategies (i.e., vector space, probabilistic and extended Boolean) all have problems because they match directly on keywords. Since the same concept can be described using many different keywords, this type of matching is prone to failure. The authors cite a study in which two people used the same word for the same concept only twenty percent of the time.

Searching for something that is closer to representing the underlying semantics of a document is not a new goal. Canonical forms were proposed for natural language processing since the early 1970's [Winograd, 1983, Schank, 1975]. Applied here, the idea is not to find a canonical knowledge representation, but to use matrix computation, in particular Singular Value Decomposition (SVD). This filters out the noise found in a document, such that two documents that have the same semantics (whether or not they have matching terms) will be located close to one another in a multi-dimensional space.

The process is relatively straightforward. A term-document matrix A is constructed such that location (i, j) indicates the number of times term i appears in document j. A SVD of this matrix results in matrices $U \sum V^T$ such that $\sum$ is a diagonal matrix. A is a matrix that represents each term in a row. Each column of A represents documents. The values in $\sum$ are referred to as the singular values. The singular values can then be sorted by magnitude and the top k values are selected as a means of developing a "latent semantic" representation of the A matrix. The remaining singular values are then set to 0. Only the first k columns are kept in U_k; only the first k rows are recorded in V_k^T. After setting the results to 0, a new A' matrix is generated to approximate $A = U \sum V^T$.

Comparison of two terms is done via an inner product of the two corresponding rows in U_k. Comparison of two documents is done as an inner product of two corresponding rows in V_k^T.

A query-document similarity coefficient treats the query as a document and computes the SVD. However, the SVD is computationally expensive; so, it is not recommended that this be done as a solution. Techniques that approximate Σ and avoid the overhead of the SVD exist. For an infrequently updated document collection, it is often pragmatic to periodically compute the SVD.

2.6.1 LSI Example

To demonstrate Latent Semantic Indexing, we once again use our previous query and document example:

Q

1, Sym

Q: "gold silver truck"
D$_1$ "Shipment of gold damaged in a fire."
D$_2$: "Delivery of silver arrived in a silver truck."
D$_3$: "Shipment of gold arrived in a truck."

0.99

0.9

The *A* matrix is obtained from the numeric columns in the term-document table given below:

A =

A = SVD of

	D_1	D_2	D_3
a	1	1	1
arrived	0	1	1
damaged	1	0	0
delivery	0	1	0
fire	1	0	0
gold	1	0	1
in	1	1	1
of	1	1	1
shipment	1	0	1
silver	0	2	0
truck	0	1	1

This step computes the singular value decompositions (SVD) on *A*. This results in an expression of *A* as the product of $U \sum V^T$. In our example, *A* is equal to the product of:

U Σ singular value V^T

$$\begin{bmatrix} -0.4201 & 0.0748 & -0.0460 \\ -0.2995 & -0.2001 & 0.4078 \\ -0.1206 & 0.2749 & -0.4538 \\ -0.1576 & -0.3046 & -0.2006 \\ -0.1206 & 0.2749 & -0.4538 \\ -0.2626 & 0.3794 & 0.1547 \\ -0.4201 & 0.0748 & -0.0460 \\ -0.4201 & 0.0748 & -0.0460 \\ -0.2626 & 0.3794 & 0.1547 \\ -0.3151 & -0.6093 & -0.4013 \\ -0.2995 & -0.2001 & 0.4078 \end{bmatrix} \begin{bmatrix} 4.0989 & 0 & 0 \\ 0 & 2.3616 & 0 \\ 0 & 0 & 1.2737 \end{bmatrix} \begin{bmatrix} -0.4945 & -0.6458 & -0.5817 \\ 0.6492 & -0.7194 & -0.2469 \\ -0.5780 & -0.2556 & 0.7750 \end{bmatrix}$$

3

However, it is not the intent to reproduce *A* exactly. What is desired, is to find the best rank *k* approximation of *A*. We only want the largest *k* singular values ($k < 3$). The choice of *k* and the number of singular values in Σ to use is somewhat arbitrary. For our example, we choose $k = 2$. We now have $A_2 = U_2 \sum_2 V_2^T$. Essentially, we take only the first two columns of *U* and the first two rows of $\sum$ and V^T.

$\boxed{K} = 2$ $k = 2$

This new product is:

$$\begin{bmatrix} -0.4201 & 0.0748 \\ -0.2995 & -0.2001 \\ -0.1206 & 0.2749 \\ -0.1576 & -0.3046 \\ -0.1206 & 0.2749 \\ -0.2626 & 0.3794 \\ -0.4201 & 0.0748 \\ -0.4201 & 0.0748 \\ -0.2626 & 0.3794 \\ -0.3151 & -0.6093 \\ -0.2995 & -0.2001 \end{bmatrix} \overset{\Sigma^{-1}}{\begin{bmatrix} 4.0989 & 0 \\ 0 & 2.3616 \end{bmatrix}} \overset{V_2^T}{\begin{bmatrix} -0.4945 & -0.6458 & -0.5817 \\ 0.6492 & -0.7194 & -0.2469 \end{bmatrix}}$$

inverse

D1 D2 D2

To obtain a $k \times 1$ dimensional array, we now incorporate the query. The query vector q^T is constructed in the same manner as the original A matrix. The query vector is now mapped into a 2-space by the transformation $q^T U_2 \Sigma_2^{-1}$.

$$\begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}^T \begin{bmatrix} -0.4201 & 0.0748 \\ -0.2995 & -0.2001 \\ -0.1206 & 0.2749 \\ -0.1576 & -0.3046 \\ -0.1206 & 0.2749 \\ -0.2626 & 0.3794 \\ -0.4201 & 0.0748 \\ -0.4201 & 0.0748 \\ -0.2626 & 0.3794 \\ -0.3151 & -0.6093 \\ -0.2995 & -0.2001 \end{bmatrix} \begin{bmatrix} 0.2440 & 0 \\ 0 & 0.4234 \end{bmatrix} = \begin{bmatrix} -0.2140 & -0.1821 \end{bmatrix}$$

gold
silver
truck
11×1

We could use the same transformation to map our document vectors into 2-space, but the rows of V_2 contain the co-ordinates of the documents. Therefore:

$D_1 = (-0.4945 \quad -0.0688)$
$D_2 = (-0.6458 \quad 0.9417)$
$D_3 = (-0.5817 \quad 1.2976)$

Finally, we are ready to compute our relevance value using the cosine similarity coefficient. This yields the following:

$$D_1 = \frac{(-0.2140)(-0.4945) + (-0.1821)(0.6492)}{\sqrt{(-0.2140)^2 + (-0.1821)^2)}\sqrt{(-0.4945)^2 + (0.6492)^2)}} = -0.0541$$

$$D_2 = \frac{(-0.2140)(-0.6458) + (-0.1821)(-0.7194)}{\sqrt{(-0.2140)^2 + (-0.1821)^2)}\sqrt{(-0.6458)^2 + (-0.7194)^2)}} = 0.9910$$

$$D_3 = \frac{(-0.2140)(-0.5817) + (-0.1821)(-0.2469)}{\sqrt{(-0.2140)^2 + (-0.1821)^2)}\sqrt{(-0.5817)^2 + (-0.2469)^2)}} = 0.9543$$

9

2.6.2 Choosing a Good Value of k

The value k is the number of columns kept after the SVD, and it is determined via experimentation. Using the MED database of only 1,033 documents and thirty queries, the average precision over nine levels of recall was plotted for different values of k. Starting at twenty, the precision increases dramatically up to values of around 100, and then it starts to level off.

2.6.3 Comparison to Other Retrieval Strategies

A comparison is given between Latent Semantic Indexing (LSI) with a factor of 100 to both the basic *tf-idf* vector space retrieval strategy and the extended Boolean retrieval strategy. For the MED collection, LSI had thirteen percent higher average precision than both strategies. For the CISI collection of scientific abstracts, LSI did not have higher precision. Upon review, the authors found that the term selection for the LSI and *tf-idf* experiments was very different. The LSI approach did not use stemming or stop words. When the same terms were used for both methods, LSI was comparable to *tf-idf*. More work was done with LSI on the TIPSTER collection [Dumais, 1994]. In this work, LSI was shown to perform slightly better than the conventional vector space model, yielding a 0.24 average precision as compared to 0.22 average precision.