Geometrical Models for Lexical Semantics: Machine Learning for Information Retrieval

R. Basili

Web Mining e *Retrieval*
a.a. 2021-22

May 8, 2022

Change of Basis

Change of Basis

Given two alternative basis $B = \{\underline{b}_1, \ldots, \underline{b}_n\}$ and $B' = \{\underline{b}'_1, \ldots, \underline{b}'_n\}$, such that the square matrix $\mathbf{C} = (c_i k)$ describe the change of the basis, i.e.

$$\underline{b}'_k = c_{1k}\underline{b}_1 + c_{2k}\underline{b}_2 + \ldots c_{nk}\underline{b}_n \qquad \forall k = 1, \ldots, n$$

Matrix and Change of Basis

Matrix and Change of Basis

The effect of the matrix $\mathbf{C}$ on a generic vector $\underline{x}$ allows to compute the change of basis according only to the involved basis B and B'. For every $\underline{x} = \sum_{k=1}^{n} x_k \underline{b}_k$ such that in the new basis B', $\underline{x}$ can be expressed by $\underline{x} = \sum_{k=1}^{n} x'_k \underline{b}'_k$, then it follows that:

$$\underline{x} = \sum_{k=1}^{n} x'_k \underline{b}'_k = \sum_{k} x'_k \left(\sum_{i} c_{ik} \underline{b}_i \right) = \sum_{i,k=1}^{n} x'_k c_{ik} \underline{b}_i$$

from which it follows that:

$$x_i = \sum_{k=1}^{n} x'_k c_{ik} \qquad \forall i = 1, \ldots, n$$

Matrix and Change of Basis

Matrix and Change of Basis

The effect of the matrix $\mathbf{C}$ on a generic vector $\underline{x}$ allows to compute the change of basis according only to the involved basis B and B'. For every $\underline{x} = \sum_{k=1}^{n} x_k \underline{b}_k$ such that in the new basis B', $\underline{x}$ can be expressed by $\underline{x} = \sum_{k=1}^{n} x'_k \underline{b}'_k$, then it follows that:

$$\underline{x} = \sum_{k=1}^{n} x'_k \underline{b}'_k = \sum_k x'_k \left(\sum_i c_{ik} \underline{b}_i \right) = \sum_{i,k=1}^{n} x'_k c_{ik} \underline{b}_i$$

from which it follows that:

$$x_i = \sum_{k=1}^{n} x'_k c_{ik} \qquad \forall i = 1, \dots, n$$

The above condition suggests that $\mathbf{C}$ is sufficient to describe any change of basis through the matrix vector mutliplication operations:

$$\underline{x} = \mathbf{C}\underline{x}'$$

Matrix and Change of Basis

Matrix and Change of Basis

The effect of the matrix $\mathbf{C}$ on a matrix $\mathbf{A}$ can be seen by studying the case where $\underline{x}, \underline{y}$ are the expression of two vectors in a base B while their counterpart on B' are $\underline{x}', \underline{y}'$, respectively. Now if $\mathbf{A}$ and $\mathbf{B}$ are such that $\underline{y} = \mathbf{A}\underline{x}$ and $\underline{y}' = \mathbf{B}\underline{x}'$, then it follows that:

$$\underline{y} = \mathbf{C}\underline{y}' = \mathbf{A}\underline{x} = \mathbf{A}(\mathbf{C}\underline{x}') = \mathbf{A}\mathbf{C}\underline{x}'$$

$$\text{(this means that)}$$

$$\underline{y}' = \mathbf{C}^{-1}\mathbf{A}\mathbf{C}\underline{x}'$$

from which it follows that:

$$\mathbf{B} = \mathbf{C}^{-1}\mathbf{A}\mathbf{C}$$

The transformation of basis $\mathbf{C}$ is a *similarity transformation* and matrices $\mathbf{A}$ and $\mathbf{B}$ are said *similar*.

From EigenVectors and Matrix Decomposition to Topic Models

Matrix Eigendecomposition

Let us create a matrix $\mathbf{S}$ with columns the n eigenvectors of a matrix $\mathbf{A}$. We have that

$$\mathbf{AS} = \mathbf{A}[\underline{x}_1, \dots, \underline{x}_n] =$$

$$= \mathbf{A}\underline{x}_1 + \dots + \mathbf{A}\underline{x}_n =$$

$$= \lambda_1 \underline{x}_1 + \dots + \lambda_n \underline{x}_n = [\underline{x}_1, \dots, \underline{x}_n]\Lambda$$

where Λ is the diagonal matrix with the eigenvalues of $\mathbf{A}$ along its diagonal:

$$\Lambda = \begin{pmatrix} \lambda_1 & 0 & \dots & 0 \\ 0 & \lambda_2 & \dots & 0 \\ 0 & 0 & \ddots & \vdots \\ \vdots & \vdots & \dots & \lambda_n \end{pmatrix}$$

EigenVectors of symmetric matrices

Now suppose that the above n eigenvectors are linearly independent. This is true when the matrix has n distinct eigenvalues. Then matrix $\mathbf{S}$ is invertible and it holds: $\mathbf{AS} = \mathbf{S}\Lambda$ so that

$$\mathbf{A} = \mathbf{S}\Lambda\mathbf{S}^{-1}$$

Towards SVD

EigenDecomposition of Symmetric matrices

Now let $\mathbf{A}$ be an $m \times n$ matrix with entries being real numbers and $m > n$.
Let us consider the $n \times n$ square matrix $\mathbf{B} = \mathbf{A}^T\mathbf{A}$.
It is easy to verify that B is symmetric, as $\mathbf{B}^T = (\mathbf{A}^T\mathbf{A})^T = \mathbf{A}^T(\mathbf{A}^T)^T = \mathbf{A}^T\mathbf{A} = \mathbf{B}$.
It has been shown that the eigenvalues of such matrices $(\mathbf{A}^T\mathbf{A})$ are real non-negative numbers. Since they are non-negative we can write them in decreasing order as squares of non-negative real numbers:

$$\sigma_1^2 \geq \sigma_2^2 \geq \ldots \geq \sigma_n^2.$$

For some index r (possibly n) the first r numbers $\sigma_1, \ldots, \sigma_r$ are positive whereas the rest are zero. For the above eigenvalues, we know that the corresponding eigenvectors $\underline{x}_1, \ldots, \underline{x}_r$ are perpendicular. Furthermore, we normalize them to have length 1. Let

$$\mathbf{S}_1 = [\underline{x}_1, \ldots, \underline{x}_r]$$

Towards SVD (2)

From the set of r orthonormal eigenvectors we can create the following vectors

$$\underline{y}_1 = \frac{1}{\sigma_1}\mathbf{A}\underline{x}_1, \ldots, \underline{y}_r = \frac{1}{\sigma_r}\mathbf{A}\underline{x}_r$$

These are perpendicular m-dimensional vectors of length 1 (orthonormal vectors) as:

$$\underline{y}_i^T \underline{y}_j = \left(\tfrac{1}{\sigma_i}\mathbf{A}\underline{x}_i\right)^T \tfrac{1}{\sigma_j}\mathbf{A}\underline{x}_j =$$

$$= \frac{1}{\sigma_i\sigma_j}\underline{x}_i^T \mathbf{A}^T \mathbf{A}\underline{x}_j = \frac{1}{\sigma_i\sigma_j}\underline{x}_i^T \mathbf{B}\underline{x}_j = \frac{1}{\sigma_i\sigma_j}\underline{x}_i^T \sigma_j^2 \underline{x}_j = \frac{\sigma_j}{\sigma_i}\underline{x}_i^T \underline{x}_j$$

Now this is 0 when $i \neq j$ and 1 when $i = j$
(as $\underline{x}_i^T \underline{x}_j = 0$ when $i \neq j$ and $\underline{x}_i^T \underline{x}_i$=1 $\forall i$)

Towards SVD (3)

Moreover, given

$$\mathbf{S}_2 = [\underline{y}_1, \ldots, \underline{y}_r]$$

we have

$$\underline{y}_j^T \mathbf{A} \underline{x}_i = \underline{y}_j^T (\sigma_i \underline{x}_i) = \sigma_i \underline{y}_j^T \underline{x}_i$$

which is 0 if $i \neq j$, and σ_i if $i = j$.
It follows thus that:

$$\mathbf{S}_2^T \mathbf{A} \mathbf{S}_1 = \Sigma$$

where Σ is the diagonal $r \times r$ matrix with $\sigma_1, \ldots, \sigma_r$ along the diagonal.

The SVD

Observe that $\mathbf{S}_2^T$ is $r \times m$, $\mathbf{A}$ is $m \times n$, and $\mathbf{S}_1$ is $n \times r$, and thus the above matrix multiplication is well defined.
Since $\mathbf{S}_2$ and $\mathbf{S}_1$ have orthonormal columns, $\mathbf{S}_2\mathbf{S}_2^T = \mathbf{I}_{m\times m}$ and $\mathbf{S}_1\mathbf{S}_1^T = \mathbf{I}_{n\times n}$ (where $\mathbf{I}_{m\times m}$ and $\mathbf{I}_{n\times n}$ are the $m \times m$ and $n \times n$ identity matrices.
Thus, by multiplying the equality

$$\mathbf{S}_2^T\mathbf{A}\mathbf{S}_1 = \Sigma$$

by $\mathbf{S}_2$ on the left and $\mathbf{S}_1^T$ on the right, we have

$$\mathbf{A} = \mathbf{S}_2\Sigma\mathbf{S}_1^T$$

Summing-up the SVD definition

Reiterating, matrix Σ is diagonal and the values along the diagonal are $\sigma_1, \ldots, \sigma_r$ which are called *singular values*.
They are the square roots of the eigenvalues of $\mathbf{A}^T\mathbf{A}$ and thus completely determined by $\mathbf{A}$.

SVD

The above decomposition of $\mathbf{A}$ into

$$\mathbf{S}_2\Sigma\mathbf{S}_1^T$$

is called *singular value decomposition*.
For the ease of notation, let us denote $\mathbf{S}_2$ by $\mathbf{V}$ and $\mathbf{S}_1$ by $\mathbf{U}$ (getting thus rid of the subscripts). Then

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T$$

Overview

- From SVD to document spaces
- The *Singular Value Decomposition*
  - Definition
  - Examples
  - Tasks and Dimensionality Reduction
- *Latent Semantic Analysis* and SVD
  - SVD: Interpretation
  - *Latent Semantic Indexing*
- LSA applications: *term* and *document clustering*
- *Latent Semantic kernels*

Principal Component Analysis

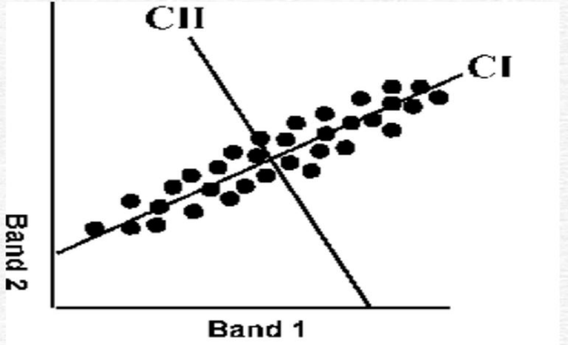


Figure 1 : The Principal Components Transform
[after Lillesand and Kiefer, 1994, 573]

LSA and Singular Value Decomposition

Given $\mathscr{V}$ the term vocabulary ($|\mathscr{V}| = m$) and
$\mathscr{T}$ the collection of texts ($|\mathscr{T}| = n$), we can apply the *Singular Value Decomposition* (Golub and Kahan, 1965) (as seen in the previous section) to the matrix W of m terms (rows) by n documents (columns) :

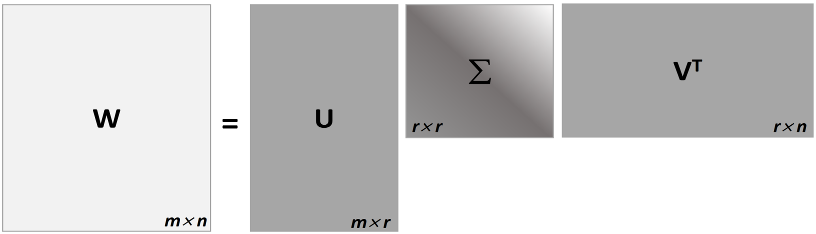
$$W = U\Sigma V^T$$

where:

- U $(m \times r)$ with the m row vectors u_i which are singular (i.e. $UU^T = I$)
- Σ $(r \times r)$ is diagonal, with σ_{ij} such that $\sigma_{ij} = 0 \quad \forall i = 1, \ldots, r$ and the singular values $\sigma_i = \sigma_{ii}$ in the main diagonal and $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_r > 0$
- V $(n \times r)$ with n row vectors v_i that are singular ($VV^T = I$)

(Notice that r is the *rank* of the matrix W).

What is SVD doing to term-to-document matrix W



Σ is diagonal with the singular values of WW^T or W^TW.
In general, $r = rank(W) \leq min(m,n)$.

SVD properties

SVD maps the source matrix W in:

$$W = U\Sigma V^T$$

where:

- U and V are the *left* and *right* singular vector matrices of W (i.e. they are made of the *eigenvectors* of WW^T and W^TW, respectively)

SVD properties

SVD maps the source matrix W in:

$$W = U\Sigma V^T$$

where:

- U and V are the *left* and *right* singular vector matrices of W (i.e. they are made of the *eigenvectors* of WW^T and $W^T W$, respectively)
- the columns of U and the rows of V define an *orthonormal* space, i.e. $UU^T = I$ and $VV^T = I$

SVD properties

SVD maps the source matrix W in:

$$W = U\Sigma V^T$$

where:

- U and V are the *left* and *right* singular vector matrices of W (i.e. they are made of the *eigenvectors* of WW^T and W^TW, respectively)
- the columns of U and the rows of V define an *orthonormal* space, i.e. $UU^T = I$ and $VV^T = I$
- Σ is the diagonal matrix of singular values of W. Singular values σ_i are the roots of the eigenvalues λ_i of WW^T (or W^TW: they are in fact identical)

We get two linear transofrmations (as we will see hereafter): WV and W^TU.

Latent Semantic Analysis and the properties of SVD

The SVD

$$W = U \Sigma V^T$$

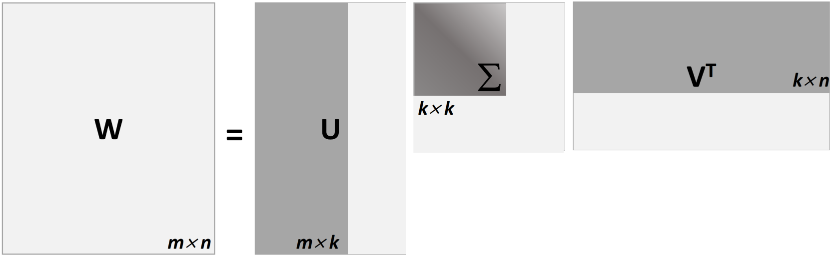
can be approximated by:

$$W \sim W' = U_k \Sigma_k V_k^T$$

by neglecting the linear transformations with the order higher than k with $k << r$ so that:

- U_k $(M \times r)$ with the M row vectors u_i which are singular and orthonormal (i.e. $U_k U_k^T = I$)
- Σ_k $(k \times r)$ is diagonal, with σ_{ij} such that $\sigma_{ij} = 0 \quad \forall i = 1, \ldots, k$ and the singular values $\sigma_i = \sigma_{ii}$ in the main diagonal and $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_k > 0$
- V_k $(N \times r)$ with N row vectors v_i that are singular $(V_k V_k^T = I)$

Rank reduction



k is the number of singular values chosen to represent the (latent) concepts in the document set.
In general, $k << r$ (original rank of W).

Latent Semantic Analysis and the properties of SVD

The SVD

$$W \sim W' = U_k \Sigma_k V_k^T$$

has a number of properties

- the matrix Σ_k is unique (although U and V are not)
- by definition, W' is the matrix obtained with an SVD of order k **closest** to W (according to the Frobenius norm)
- σ_i are the root values $\sigma_i = \sqrt{\lambda_i}$ of the k largest eigenvalues λ_i of WW^T
- the *principal components* of the task (i.e. characterizing the collection) are expressed by U_k and V_k

LSA: semantic interpretation

We can say that $W = U\Sigma V^T$, Σ captures the *latent semantic structure* of the source space where W is defined, $\mathscr{V} \times \mathscr{T}$ and that the approximation to W' has no significant effect on this propoerty.
to se why:

LSA: semantic interpretation

We can say that $W = U\Sigma V^T$, Σ captures the *latent semantic structure* of the source space where W is defined, $\mathscr{V} \times \mathscr{T}$ and that the approximation to W' has no significant effect on this propoerty.
to se why:

- eigenvectors are data specific direction of the original space $\mathscr{V} \times \mathscr{T}$, characterized by the linear trasformation (from terms to documents) W.

LSA: semantic interpretation

We can say that $W = U\Sigma V^T$, Σ captures the *latent semantic structure* of the source space where W is defined, $\mathscr{V} \times \mathscr{T}$ and that the approximation to W' has no significant effect on this propoerty.
to se why:

- eigenvectors are data specific direction of the original space $\mathscr{V} \times \mathscr{T}$, characterized by the linear trasformation (from terms to documents) W.
- $U\Sigma$ is derived from $W = U\Sigma V^T$: infact, $WV = U\Sigma V^T V = U\Sigma$, that is for every i-th row (i.e. term) in W (or U), $u_i\Sigma = w_i V$.

LSA: semantic interpretation

We can say that $W = U\Sigma V^T$, Σ captures the *latent semantic structure* of the source space where W is defined, $\mathscr{V} \times \mathscr{T}$ and that the approximation to W' has no significant effect on this propoerty.
to se why:

- eigenvectors are data specific direction of the original space $\mathscr{V} \times \mathscr{T}$, characterized by the linear trasformation (from terms to documents) W.
- $U\Sigma$ is derived from $W = U\Sigma V^T$: infact, $WV = U\Sigma V^T V = U\Sigma$, that is for every i-th row (i.e. term) in W (or U), $u_i\Sigma = w_i V$.
- CONSEQUENCE: representing term vectors (i.e. rows w_i in W) through $u_i\Sigma$, MEANS: combining linearly through Σ the elements (i.e. the correlations with all documents, v_j) from the orthonormal basis defined by V (after truncation at k)

LSA: semantic interpretation(cont'd)

Moreover, (for the **documents**):

- $V\Sigma$ is obtained from $W = (U\Sigma V^T)$: infact, $W^T = (U\Sigma V^T)^T = V\Sigma U^T$, from which it follows that $W^T U = V\Sigma$. The columns (documents) w_j of W (or rows in V) are such that $v_j\Sigma = w_j U$.

LSA: semantic interpretation(cont'd)

Moreover, (for the **documents**):

- $V\Sigma$ is obtained from $W = (U\Sigma V^T)$: infact, $W^T = (U\Sigma V^T)^T = V\Sigma U^T$, from which it follows that $W^T U = V\Sigma$. The columns (documents) w_j of W (or rows in V) are such that $v_j\Sigma = w_j U$.
- CONSEQUENCE: representing document vectors (i.e. columns in W) through $v_j\Sigma$ MEANS combining linearly (through Σ) the rows (i.e. the correlations wth terms u_i) of the orthonormal basis expressed by U

LSA: semantic interpretation(cont'd)

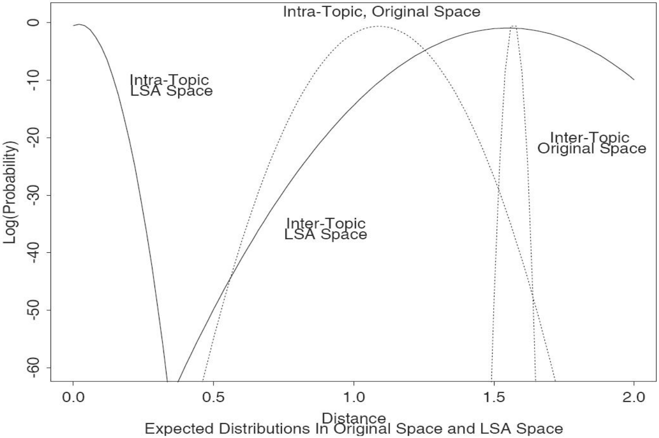
Moreover, (for the **documents**):

- $V\Sigma$ is obtained from $W = (U\Sigma V^T)$: infact, $W^T = (U\Sigma V^T)^T = V\Sigma U^T$, from which it follows that $W^T U = V\Sigma$. The columns (documents) w_j of W (or rows in V) are such that $v_j\Sigma = w_j U$.
- CONSEQUENCE: representing document vectors (i.e. columns in W) through $v_j\Sigma$ MEANS combining linearly (through Σ) the rows (i.e. the correlations wth terms u_i) of the orthonormal basis expressed by U
- $U\Sigma$ and $V\Sigma$ express two mappings from terms in $\mathscr{V}$ and documents in $\mathscr{T}$ into the k-dimensional space generated by the SVD

LSA: semantic interpretation (cont'd)

- Trasnformation are linear combinations towards the k dimensions corresponding to concepts (i.e. latent topics). Infact:
- Matrices U and V are orthonormal basis for the k-dimensional Latent Semantic space. Dimensions here correspond to priviledged directions of the linera transformation defined by W and are lineari combinations in WW^T (or W^TW): in other owrds they corresponds to concepts (or discussion topics) determined by the systematic occurrences of some terms with some documents (and viceversa).
- Term vectors w_i are represented in such space through WV that is the (linear) combination of source documents, equivalent to compute $U\Sigma$
- Analogously, documents w_j through $W^TU=V\Sigma$

LSA: an example of SVD over an artificially derived term to document distribution



LSA: an example

$$
\begin{array}{l}
\text{Terms}\\
\text{a}\\
\text{arrived}\\
\text{damaged}\\
\text{delivery}\\
\text{fire}\\
\text{gold}\\
\text{in}\\
\text{of}\\
\text{shipment}\\
\text{silver}\\
\text{truck}
\end{array}
\qquad
\mathbf{W} =
\begin{array}{c}
\begin{array}{ccc} d1 & d2 & d3 \end{array}\\
\begin{bmatrix}
1 & 1 & 1\\
0 & 1 & 1\\
1 & 0 & 0\\
0 & 1 & 0\\
1 & 0 & 0\\
1 & 0 & 1\\
1 & 1 & 1\\
1 & 1 & 1\\
1 & 0 & 1\\
0 & 2 & 0\\
0 & 1 & 1
\end{bmatrix}
\end{array}
\qquad
\mathbf{q} =
\begin{array}{c}
q\\
\begin{bmatrix}
0\\ 0\\ 0\\ 0\\ 0\\ 1\\ 0\\ 0\\ 0\\ 1\\ 1
\end{bmatrix}
\end{array}
$$

LSA: ... computing $U\Sigma V^T$

$$\mathbf{U} = \begin{bmatrix} -0.4201 & 0.0748 & -0.0460 \\ -0.2995 & -0.2001 & 0.4078 \\ -0.1206 & 0.2749 & -0.4538 \\ -0.1576 & -0.3046 & -0.2006 \\ -0.1206 & 0.2749 & -0.4538 \\ -0.2626 & 0.3794 & 0.1547 \\ -0.4201 & 0.0748 & -0.0460 \\ -0.4201 & 0.0748 & -0.0460 \\ -0.2626 & 0.3794 & 0.1547 \\ -0.3151 & -0.6093 & -0.4013 \\ -0.2995 & -0.2001 & 0.4078 \end{bmatrix} \qquad \mathbf{S} = \begin{bmatrix} 4.0989 & 0.0000 & 0.0000 \\ 0.0000 & 2.3616 & 0.0000 \\ 0.0000 & 0.0000 & 1.2737 \end{bmatrix}$$

$$\mathbf{V} = \begin{bmatrix} -0.4945 & 0.6492 & -0.5780 \\ -0.6458 & -0.7194 & -0.2556 \\ -0.5817 & 0.2469 & 0.7750 \end{bmatrix} \qquad \mathbf{V}^{\mathbf{T}} = \begin{bmatrix} -0.4945 & -0.6458 & -0.5817 \\ 0.6492 & -0.7194 & 0.2469 \\ -0.5780 & -0.2556 & 0.7750 \end{bmatrix}$$

LSA: Rank reduction, $k = 2$

$$U = \begin{bmatrix} -0.4201 & 0.0748 \\ -0.2995 & -0.2001 \\ -0.1206 & 0.2749 \\ -0.1576 & -0.3046 \\ -0.1206 & 0.2749 \\ -0.2626 & 0.3794 \\ -0.4201 & 0.0748 \\ -0.4201 & 0.0748 \\ -0.2626 & 0.3794 \\ -0.3151 & -0.6093 \\ -0.2995 & -0.2001 \end{bmatrix} \qquad S = \begin{bmatrix} 4.0989 & 0.0000 \\ 0.0000 & 2.3616 \end{bmatrix}$$

$$V = \begin{bmatrix} -0.4945 & 0.6492 \\ -0.6458 & -0.7194 \\ -0.5817 & 0.2469 \end{bmatrix} \qquad V^T = \begin{bmatrix} -0.4945 & -0.6458 & -0.5817 \\ 0.6492 & -0.7194 & 0.2469 \end{bmatrix}$$

LSA: exploiting SVD in ad hoc IR

Computing a Query-Doc similarity score

- For n documents, matrix V encodes n rows, one for each component of the document d_i projected in the LSA space
- A query q can be processed as a pseudo-document and projected in the LSA space by the same transformation

LSA: exploiting SVD in ad hoc IR (2)

Use of the SVD

If $W = U\Sigma V^T$ the it follows that:

- $V = W^T U \Sigma^{-1}$
  (in fact $W = U\Sigma V^T$ that is $W^T = (U\Sigma V^T)^T = V(U\Sigma)^T = V\Sigma U^T$
  so that
  $V\Sigma U^T = W^T$ that implies $V\Sigma = W^T U$ so that :
  $V = W^T U \Sigma^{-1}$)

LSA: exploiting SVD in ad hoc IR (2)

Use of the SVD

If $W = U\Sigma V^T$ the it follows that:

- $V = W^T U \Sigma^{-1}$
  (in fact $W = U\Sigma V^T$ that is $W^T = (U\Sigma V^T)^T = V(U\Sigma)^T = V\Sigma U^T$
  so that
  $V\Sigma U^T = W^T$ that implies $V\Sigma = W^T U$ so that :
  $V = W^T U \Sigma^{-1}$)
- $d = d^T U \Sigma^{-1}$

LSA: exploiting SVD in ad hoc IR (2)

Use of the SVD

If $W = U\Sigma V^T$ the it follows that:

- $V = W^T U \Sigma^{-1}$
  (in fact $W = U\Sigma V^T$ that is $W^T = (U\Sigma V^T)^T = V(U\Sigma)^T = V\Sigma U^T$
  so that
  $V\Sigma U^T = W^T$ that implies $V\Sigma = W^T U$ so that :
  $V = W^T U \Sigma^{-1}$)
- $d = d^T U \Sigma^{-1}$
- $q = q^T U \Sigma^{-1}$ (pseudo document)

LSA: exploiting SVD in ad hoc IR (2)

Use of the SVD

If $W = U\Sigma V^T$ the it follows that:

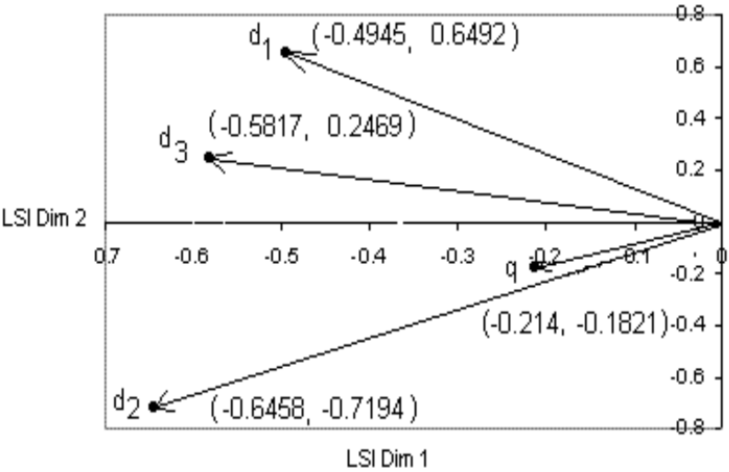
- $V = W^T U \Sigma^{-1}$
  (in fact $W = U\Sigma V^T$ that is $W^T = (U\Sigma V^T)^T = V(U\Sigma)^T = V\Sigma U^T$
  so that
  $V\Sigma U^T = W^T$ that implies $V\Sigma = W^T U$ so that :
  $V = W^T U \Sigma^{-1}$)
- $d = d^T U \Sigma^{-1}$
- $q = q^T U \Sigma^{-1}$ (pseudo document)

After the k-order dimensionality reduction step:

- $d = d^T U_k \Sigma_k^{-1}$
- $q = q^T U_k \Sigma_k^{-1}$ (pseudo document)

As a consequence: $sim(q,d) = sim(q^T U_k \Sigma_k^{-1}, d^T U_k \Sigma_k^{-1})$

LSA: query and document vectors



LSA: ... computing the query vector

$$q = q^T U S^{-1}$$

$$q = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{bmatrix} -0.4201 & 0.0748 \\ -0.2995 & -0.2001 \\ -0.1206 & 0.2749 \\ -0.1576 & -0.3046 \\ -0.1206 & 0.2749 \\ -0.2626 & 0.3794 \\ -0.4201 & 0.0748 \\ -0.4201 & 0.0748 \\ -0.2626 & 0.3794 \\ -0.3151 & -0.6093 \\ -0.2995 & -0.2001 \end{bmatrix} \begin{bmatrix} \frac{1}{4.0989} & 0.0000 \\ 0.0000 & \frac{1}{2.3616} \end{bmatrix}$$

$$q = \begin{bmatrix} -0.2140 & -0.1821 \end{bmatrix}$$

LSA: Second order relations

LSA and word meaning

- The LSA representation of terms depends on **all** the co-occurrences in the different documents (i.e. different discourse contexts) expressed by the initial matrix W
- The term t representation in LSA is no longer the versor $\vec{t}$ orthogonal to (and thus independent from) all the other versors

LSA: Second order relations

LSA and word meaning

- The LSA representation of terms depends on **all** the co-occurrences in the different documents (i.e. different discourse contexts) expressed by the initial matrix W
- The term t representation in LSA is no longer the versor $\vec{t}$ orthogonal to (and thus independent from) all the other versors
- Similarity between two terms t_i and t_j depends on the transfromation $U\Sigma$ and inherits information from all the shared co-occurrences with other terms t_k (with $t_k \neq t_i, t_j$). These dependences characterize *second order* relations.

LSA: SVD and term clustering

M =

	d_1	d_2	d_3	d_4	d_5	d_6
shuttle	1	0	1	0	0	0
astronaut	0	1	0	0	0	0
moon	1	1	0	0	0	0
car	1	0	0	1	1	0
truck	0	0	0	1	0	1

$$\mathbf{M} = \mathbf{K}_{t\times s}\mathbf{S}_{s\times s}\mathbf{D}^{T}_{s\times N}$$

$$= \boxed{txs} \times \boxed{sxs} \times \boxed{sxN}$$

K =

	dim_1	dim_2	dim_3	dim_4	dim_5
shuttle	-0.44	-0.30	0.57	0.58	0.25
astronaut	-0.13	-0.33	-0.59	0.00	0.73
moon	-0.48	-0.51	-0.37	0.00	-0.61
car	-0.70	0.35	0.15	-0.58	0.16
truck	-0.26	0.65	-0.41	0.58	-0.09

S =

2.16	0.00	0.00	0.00	0.00
0.00	1.59	0.00	0.00	0.00
0.00	0.00	1.28	0.00	0.00
0.00	0.00	0.00	1.00	0.00
0.00	0.00	0.00	0.00	0.39

LSA: SVD and term clustering

M =

	d_1	d_2	d_3	d_4	d_5	d_6
shuttle	1	0	1	0	0	0
astronaut	0	1	0	0	0	0
moon	1	1	0	0	0	0
car	1	0	0	1	1	0
truck	0	0	0	1	0	1

$$\mathbf{M} = \mathbf{K}_{t \times s} \mathbf{S}_{s \times s} \mathbf{D}^T_{s \times N}$$

= txs X sxs X sxN

K =

	dim_1	dim_2	dim_3	dim_4	dim_5
shuttle	-0.44	-0.30	0.57	0.58	0.25
astronaut	-0.13	-0.33	-0.59	0.00	0.73
moon	-0.48	-0.51	-0.37	0.00	-0.61
car	-0.70	0.35	0.15	-0.58	0.16
truck	-0.26	0.65	-0.41	0.58	-0.09

S =

2.16	0.00	0.00	0.00	0.00
0.00	1.59	0.00	0.00	0.00
0.00	0.00	1.28	0.00	0.00
0.00	0.00	0.00	1.00	0.00
0.00	0.00	0.00	0.00	0.39

LSA: SVD and term clustering

M =

	d_1	d_2	d_3	d_4	d_5	d_6
shuttle	1	0	1	0	0	0
astronaut	0	1	0	0	0	0
moon	1	1	0	0	0	0
car	1	0	0	1	1	0
truck	0	0	0	1	0	1

$$\mathbf{M} = \mathbf{K}_{t\times s}\mathbf{S}_{s\times s}\mathbf{D}^T_{s\times N}$$

$$= \boxed{\text{txs}} \; \text{X} \; \boxed{\text{sxs}} \; \text{X} \; \boxed{\text{sxN}}$$

K =

	dim_1	dim_2	dim_3	dim_4	dim_5
shuttle	-0.44	-0.30	0.57	0.58	0.25
astronaut	-0.13	-0.33	-0.59	0.00	0.73
moon	-0.48	-0.51	-0.37	0.00	-0.61
car	-0.70	0.35	0.15	-0.58	0.16
truck	-.0.26	0.65	-0.41	0.58	-0.09

S =

2.16	0.00	0.00	0.00	0.00
0.00	1.59	0.00	0.00	0.00
0.00	0.00	1.28	0.00	0.00
0.00	0.00	0.00	1.00	0.00
0.00	0.00	0.00	0.00	0.39

LSA: Weighting policies

For obtaining useful LSA spaces different weightiing models for the matrix W can be used to improve the search for better possible SVD and linear transformations

- Frequency. c_{ij} (or its normalized variants $\frac{c_{ij}}{|d_j|}$, $\frac{c_{ij}}{max_{lk} c_{lk}}$)
- (Landauer) $w_{ij} = \frac{log(c_{ij}+1)}{\frac{1+\sum_{j=1}^{N} \frac{c_{ij}}{t_i} log \frac{c_{ij}}{t_i}}{logN}} = \frac{log(c_{ij}+1)}{\frac{1+\sum_{j=1}^{N} P_{ij} log P_{ij}}{logN}}$
- (Bellegarda, Language modeling) $w_{ij} = (1-\varepsilon_i)\frac{c_{ij}}{n_j}$ con

$$\varepsilon_i = -\frac{1}{log_2 N}\sum_{j=1}^{N} \frac{c_{ij}}{t_i} log \frac{c_{ij}}{t_i}$$

LSA: Term similarity metrics

The LSA term similarity is defined by:

$$WW^T$$

computed by:

$$U\Sigma V^T(U\Sigma V^T)^T = (U\Sigma V^T)(V\Sigma^T U^T) = U\Sigma\Sigma^T U^T = U\Sigma(U\Sigma)^T$$

Applications: *Document Indexing* (representation of docs through the LSA terms), *Word/term clustering* (Clustering of terms in topics or sinonimy classes).

LSA: Word Clustering

Cluster 1
Andy, antique, antiques, art, artist, artist's, artists, artworks, auctioneers, Christie's, collector, drawings, gallery, Gogh, fetched, hysteria, masterpiece, museums, painter, painting, paintings, Picasso, Pollock, reproduction, Sotheby's, van, Vincent, Warhol

Cluster 2
appeal, appeals, attorney, attorney's, counts, court, court's, courts, condemned, convictions, criminal, decision, defend, defendant, dismisses, dismissed, hearing, here, indicted, indictment, indictments, judge, judicial, judiciary, jury, juries, lawsuit, leniency, overturned, plaintiffs, prosecute, prosecution, prosecutions, prosecutors, ruled, ruling, sentenced, sentencing, suing, suit, suits, witness

LSA: Metrics for document similarity

Lsa document similairty is definec by:

$$W^T W$$

computed through:

$$(U\Sigma V^T)^T U\Sigma V^T = (V\Sigma^T U^T)(U\Sigma V^T) = V\Sigma\Sigma V^T = V\Sigma(V\Sigma)^T$$

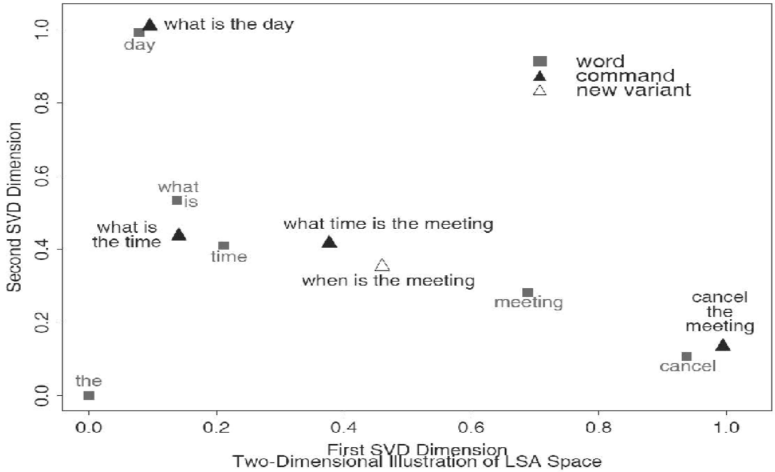
Applications: *Document clustering* and *Text Classification*

LSA: Other Applications

Semantic Inferece in *Automatic Call Routing*

- The *task*: mapping questions (to a *Call Center*) into a procedure for *replying* (e.g. in a dialogue).
- Training Questions (4 classes, T1-T4):
  (T1) *What is the time*, (T2) *What is the day*, (T3) *What time is the meeting*, (T4) *Cancel the meeting*
- *the* is irrelevant, *time* is ambiguous (between class T1 and T3)
- Input (i.e. test) question: *when is the meeting* (expcted label: **T3**)

Automatic Call Routing in the LSA space



LSA vs. Machine Learning

What is the relation between LSA and *learning* paradigms?

Induction in LSA

- LSA provides a general method for extracting a **similarity estimate** from data and every example-driven algorithm can use it for guide the generalization (i.e. optimizing the model such as in selecting the separating hyperplane)

LSA vs. Machine Learning

What is the relation between LSA and *learning* paradigms?

Induction in LSA

- LSA provides a general method for extracting a **similarity estimate** from data and every example-driven algorithm can use it for guide the generalization (i.e. optimizing the model such as in selecting the separating hyperplane)
- **LSA provides a data-driven metric** suitable to the application of every algebraic approach to learning (e.g. *pretraining* for a neural network)

LSA vs. Machine Learning

What is the relation between LSA and *learning* paradigms?

Induction in LSA

- LSA provides a general method for extracting a **similarity estimate** from data and every example-driven algorithm can use it for guide the generalization (i.e. optimizing the model such as in selecting the separating hyperplane)
- **LSA provides a data-driven metric** suitable to the application of every algebraic approach to learning (e.g. *pretraining* for a neural network)
- As it can be applied to unlabeled data LSA extends the generalization power of *supervised* method by exploiting data properties independent from the task (labeled data). This is fully complementary to the task itself

LSA: Machine Learning tasks for IR

LSA applications in Relevance Feedback

Relevance feedback is a technique to refine the user query definition by extending (or reweighting) it according to the IR system output (i.e. ge results against the currentl available collection). LSa can be used in this scenario for:

- Automatic Global Analysis, i.e. the a priori costruction of a lexicon of similar terms.
- Estimating relevance *before* query expansion.

LSA: Machine Learning tasks in NLP

LSA and language semantics

- SVD in lexical semantic analysis: semantic spaces for distributional analysis; automatic compilation of word spaces from corpora (see (Pado and Lapata, 2007))
- Word Sense Discrimination as clustering in LSA-like spaces (see Schutze, 1998)
- Word Sense Disambiguation in LSA spaces (see (Gliozzo et al., 2005), (Basili et al., 2006)
- Framenet predicate induction (see (Basili et al., 2008), (Pennacchiotti et al., 2008))

LSA and Kernel methods for Machine Learning

Kernel functions $K(\vec{o}_i, \vec{o})$ can be used as similarity estimates of term or document pairs, o_i e o, in complex spaces in order to train kernel machines (e.g. SVMs) according to:

$$h(\vec{x}) = sgn\left(\sum_{i=1}^{l} \alpha_i K(o_i, o) + b\right)$$

where $\vec{x} = \phi(o)$, and l depends on the learning set.
It is natural to adopt the inner product in LSA spaces, as a definition of $K(o_i, o)$. This approach has been applied to tasks such as:

- Word Sense Disambiguation (automatic classification of a word w occurrences in texts into one of w sense definitions)
- Text Categorization (document classification)

LSA-based Domain Kernels: Applications to Lexicons

Basic Assumptions

- Let o_i to represent a "*term*" $\vec{t}_i$
- Let a standard Vecor Space Model to be used for representing $\vec{t}_i$ (e.g. applying the $tf \times idf$ weighting)
- The source matrix T is thus terms by documents
- By applying SVD we get a lexical vector for each term in the LSA space

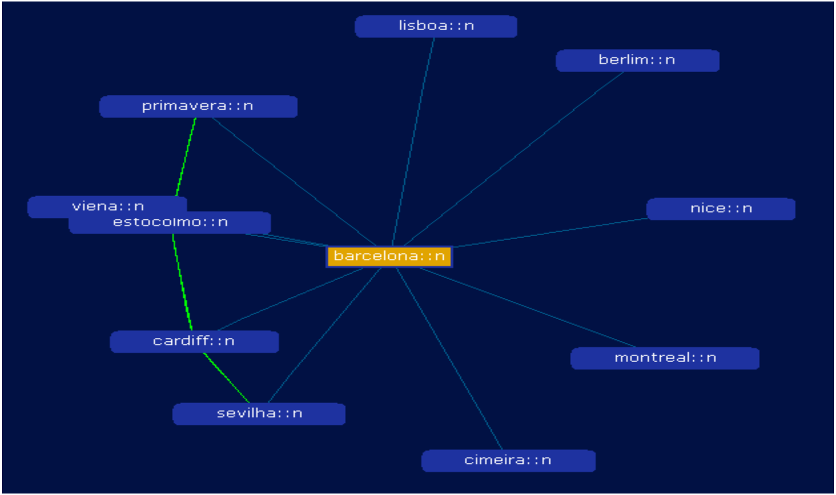
LSA-based Domain Kernels (2)

Process:

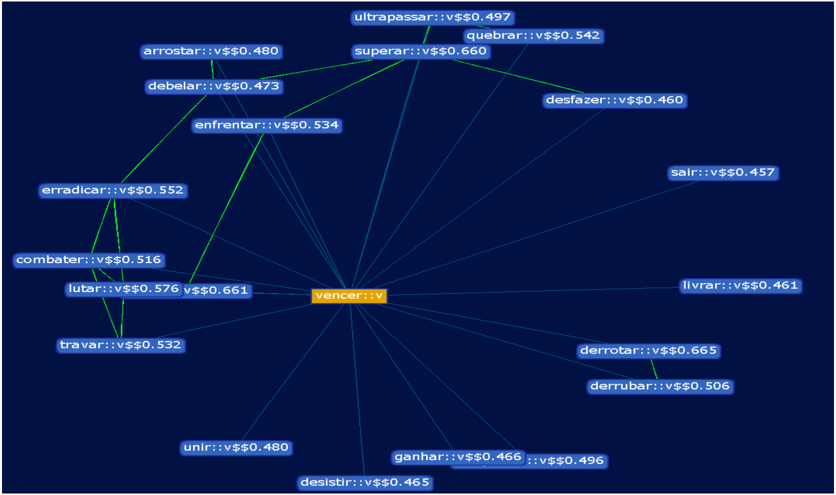
- First, apply the LSA trasformation of dimension k, $o_i \leftarrow \vec{t}_i$ (Note: $\vec{t}_i$ are the LSA vectors of o_i)
- Use the term similarity metrics between $\vec{t}_i$ pairs as the estimation of the object o_i similarity (estimation in the *Latent Semantic Space*)
- Train a supervised classifier (a semantic labeling system) through the kernel $K(.,.)$ defined as follows:
  $K(\vec{t}_i,\vec{t}) = K(\phi^{-1}(o_i),\phi^{-1}(o)) \doteq K_{LSA}(o,o_i)$ where:
  $$K_{LSA}(o,o_i) = o_i \otimes_{LSA} o$$

Note: $\otimes_{LSA}$ stands for the inner product between pairs of o_i objects (i.e. terms) as computed in the LSA space.

LSA space for terms in a foreign language (portuguese)



LSA space for terms in a foreign language (portuguese)



LSA-based Domain Kernels (3)

LSA determines the SVD transformation and the order k approximation.
k is the dimension of the LSA space, i.e. the number of principal components of the original problem (i.e. the standard VSM)
We can interpret these notions as *domains*:

- o_i corresponds to the description of the i-th term $\vec{t}_i$ in the different domains
- similar terms according to $K_{LSA}(o_i, o)$ share a large number of domains
- the resulting kernel $K_{LSA}(o_i, o)$ is called *Latent Semantic Kernel* (Cristianini&Shawe-Taylor,2004) or *domain kernel* (Gliozzo & Strapparava,2005).

LSA-based Domain Kernels:

Application to *Text Categorization*

- a document x_i (like a term) can be mapped into an LSA space, $o_i \leftarrow \vec{x}_i$
- the k components of o_i stand for the description of x_i in the new space
- the resulting kernel $K_{LSA}(o_i, o)$ captures the similarity according to the domain description of $\vec{x}_i$
- a linear supervised classifier training (e.g. an SVM) determines the hyperplane equation in the LSA space, such as:

$$f(\vec{x}) = \left(\sum_{i=1}^{l} \alpha_i K_{LSA}(o_i, o) + b \right)$$

Note: LSA can be computed on an unlabeled document collection and is thus *external* to the training data set.

References

SVD and LSA

- Susan T. Dumais, Michael Berry, Using Linear Algebra for Intelligent Information Retrieval, SIAM Review, 1995, 37, 573–595
- G. W. Furnas, S. Deerwester, S. T. Dumais, T. K. Landauer, R. A. Harshman, L. A. Streeter, K. E. Lochbaum, Information retrieval using a singular value decomposition model of latent semantic structure, SIGIR '88: Proc. of the ACM SIGIR conference on Research and development in Information Retrieval, 1988

Non linear embeddings

- S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. Science, 2000.
- B. J. Tenenbaum, V. Silva, and J. Langford. A Global Geometric Framework for Nonlinear Dimensionality Reduction. Science, pp.2319-2323, 2000
- Xiaofei He, Partha Niyogi, Locality Preserving Projections. Proceedings of Advances in Neural Information Processing Systems, Vancouver, Canada, 2003.

References

LSA in language learning

- Hinrich Schutze, Automatic word sense discrimination, Computational Linguistics, 24(1), 1998.
- Beate Dorow and Dominic Widdows, Discovering Corpus-Specific Word Senses. EACL 2003, Budapest, Hungary. Conference, pages 79-82
- Basili Roberto, Marco Cammisa, Alfio Gliozzo, Integrating Domain and Paradigmatic Similarity for Unsupervised Sense Tagging, 17th European Conference on Artificial Intelligence (ECAI06), Riva del Garda, Italy, 2006.
- Sebastian Pado and Mirella Lapata. Dependency-based Construction of Semantic Space Models. In Computational Linquistics, Vol. 33(2):161-199, 2007.
- Basili R. Pennacchiotti M., Proceedings of GEMS "*Geometrical Models of Natural Language Semantics*", 2009 (http://www.aclweb.org/anthology/W/W09/#0200), 2010 (http://aclweb.org/anthology-new/W/W10/#2800)