Latent semantic indexing

Traditional search

Term-document matrix

$$\begin{matrix} & & \mathbf{d}_j & \\ & & \downarrow & \\ \mathbf{t}_i^T \rightarrow & \begin{bmatrix} x_{1,1} & \dots & x_{1,n} \\ \vdots & \ddots & \vdots \\ x_{m,1} & \dots & x_{m,n} \end{bmatrix} \end{matrix}$$

Now a row in this matrix will be a vector corresponding to a term, giving its relation to each document:

$$\mathbf{t}_i^T = \begin{bmatrix} x_{i,1} & \dots & x_{i,n} \end{bmatrix}$$

Likewise, a column in this matrix will be a vector corresponding to a document, giving its relation to each term:

$$\mathbf{d}_j = \begin{bmatrix} x_{1,j} \\ \vdots \\ x_{m,j} \end{bmatrix}$$

- Let A be the term-document matrix. We then form a query vector and compare it with the document vector.
- Matrix multiplication of Transpose of A and query vector gives what we want.
- Multiplication of (n*m)matrix and (m*1) query vector gives (n*1) result vector.

Use of cosine angle

- Cosine(theta)=<d,q>/|d||q|
- Note that it involves division by the length (euclidean norm)
- Near 1 means the document and query vector are close to each other while near 0 means the are not close.

- We usually use cosine angle to compare the two document vector (or query vector) about how close they are.
- The reason of using cosine angle is to eliminate the effect of :

1: Too many terms in the document vector (e.g. encyclopedia).

2: Too many terms in the query vector.

LSI

Now SVD comes in:

$$
\begin{matrix}
 & X & & & U & & \Sigma & & V^T \\
 & (\mathbf{d}_j) & & & & & & & (\hat{\mathbf{d}}_j) \\
 & \downarrow & & & & & & & \downarrow \\
(\mathbf{t}_i^T) \rightarrow &
\begin{bmatrix}
x_{1,1} & \dots & x_{1,n} \\
\vdots & \ddots & \vdots \\
x_{m,1} & \dots & x_{m,n}
\end{bmatrix}
& = &
(\hat{\mathbf{t}}_i^T) \rightarrow &
\begin{bmatrix}
\begin{bmatrix} \, \\ \, \\ \mathbf{u}_1 \\ \, \\ \, \end{bmatrix}
\dots
\begin{bmatrix} \, \\ \, \\ \mathbf{u}_l \\ \, \\ \, \end{bmatrix}
\end{bmatrix}
& \cdot &
\begin{bmatrix}
\sigma_1 & \dots & 0 \\
\vdots & \ddots & \vdots \\
0 & \dots & \sigma_l
\end{bmatrix}
& \cdot &
\begin{bmatrix}
\begin{bmatrix} & & \mathbf{v}_1 & & \end{bmatrix} \\
\vdots \\
\begin{bmatrix} & & \mathbf{v}_l & & \end{bmatrix}
\end{bmatrix}
\end{matrix}
$$

We keep the first t singular values only. Note that U and V are not square matrix anymore; while “singular matrix” becomes square matrix.

- The term vectors are the rows of U_(t) while the column vectors are now the columns of transpose of V_(t).
- They are pseudo are they are represented in lower dimension space than before and they are shorter.

Computation of pseudo vectors:

$$\hat{\mathbf{d}}_j = \Sigma_k^{-1} U_k^T \mathbf{d}_j$$

$$\hat{\mathbf{q}} = \Sigma_k^{-1} U_k^T \mathbf{q}$$

Effect of dimension reduction

{(car), (truck), (flower)} --> {(1.3452 * car + 0.2828 * truck), (flower)}

2 terms are combined in the document vector and query vector.

Example

- The query is *gold silver truck* and the "collection" consists of just three "documents":

- d1: *Shipment of gold damaged in a fire.*
- d2: *Delivery of silver arrived in a silver truck.*
- d3: *Shipment of gold arrived in a truck.*

Terms	d1	d2	d3	q
a	1	1	1	0
arrived	0	1	1	0
damaged	1	0	0	0
delivery	0	1	0	0
fire	1	0	0	0
gold	1	0	1	1
in	1	1	1	0
of	1	1	1	0
shipment	1	0	1	0
silver	0	2	0	1
truck	0	1	1	1

$$A = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 2 & 0 \\ 0 & 1 & 1 \end{bmatrix} \qquad q = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

SVD results

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

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

Dimensionality reduction (figure 4)

k = 2

$$U \approx U_k = \begin{bmatrix} -0.4201 & 0.0748 \\ -0.2995 & -0.2001 \\ -0.1206 & 0.2749 \\ -0.1576 & -0.3046 \\ -0.1206 & 0.2749 \\ -0.2626 & 0.3794 \\ -0.4201 & 0.0748 \\ -0.4201 & 0.0748 \\ -0.2626 & 0.3794 \\ -0.3151 & -0.6093 \\ -0.2995 & -0.2001 \end{bmatrix}$$

$$S \approx S_k = \begin{bmatrix} 4.0989 & 0.0000 \\ 0.0000 & 2.3616 \end{bmatrix}$$

$$V \approx V_k = \begin{bmatrix} -0.4945 & 0.6492 \\ -0.6458 & -0.7194 \\ -0.5817 & 0.2469 \end{bmatrix}$$

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

pseudo

$$\mathbf{d} = \mathbf{d}^T \mathbf{U}_k \mathbf{S}_k^{-1}$$

$$\mathbf{q} = \mathbf{q}^T \mathbf{U}_k \mathbf{S}_k^{-1}$$

$$\mathrm{sim}(\mathbf{q}, \mathbf{d}) = \mathrm{sim}(\mathbf{q}^T \mathbf{U}_k \mathbf{S}_k^{-1}, \mathbf{d}^T \mathbf{U}_k \mathbf{S}_k^{-1})$$

Pseudo query vector:

Reduced query

$$q = q^T U_k S_k^{-1} \qquad k = 2$$

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

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

Pseudo document vector:

d1(-0.4945, 0.6492)
d2(-0.6458, -0.7194)
d3(-0.5817, 0.2469)

Cosine similarities in reduced space

$$sim(q, d) = \frac{q \bullet d}{|q||d|}$$

$$sim(q, d_1) = \frac{(-0.2140)(-0.4945) + (-0.1821)(0.6492)}{\sqrt{(-0.2140)^2 + (-0.1821)^2}\sqrt{(-0.4945)^2 + (0.6492)^2}} = -0.0541$$

$$sim(q, d_2) = \frac{(-0.2140)(-0.6458) + (-0.1821)(-0.7194)}{\sqrt{(-0.2140)^2 + (-0.1821)^2}\sqrt{(-0.6458)^2 + (-0.7194)^2}} = 0.9910$$

$$sim(q, d_3) = \frac{(-0.2140)(-0.5817) + (-0.1821)(0.2469)}{\sqrt{(-0.2140)^2 + (-0.1821)^2}\sqrt{(-0.5817)^2 + (0.2469)^2}} = 0.4478$$

Ranking documents in descending order

$$d_2 > d_3 > d_1$$

Advantages of LSI

- Traditional method cannot effectively find documents on the same topic but with synonyms. LSI is able to do that.

Drawback of LSI

While LSI can do this:

{(car), (truck), (flower)} --> {(1.3452 * car + 0.2828 * truck), (flower)} where (1.3452 * car + 0.2828 * truck) component could be interpreted as "vehicle".

However:

It is very likely that cases close to{(car), (bottle), (flower)} --> {(1.3452 * car + 0.2828 * **bottle**), (flower)} will also occur.

Reference

- www.miislita.com
- Barbara Rosario, Latent Semantic Indexing: An overview(2000)
- Wikipedia: Latent Semantic analysis