Motivation: (Textual) Information Retrieval

- **Information Retrieval** is the “activity of obtaining information resources relevant to an information need from a collection of information resources”.

- To goal is to retrieve relevant (in terms of having the same conceptual topics or meaning) textual documents from databases.

- Example:
  - Prior art (state of the art) searches for patents
  - Searches at google scholar
Lexical Matching is inaccurate because:
  i) There are synonyms
  ii) Words have multiple meanings
so a user’s query may literally match irrelevant documents.

Latent Semantic Indexing (LSI) solved the problem by using conceptual indices rather than individual words.
Consider **car**, **automobile**, **driver** and **elephant** where **car** and **automobile** are synonyms.

- In lexical matching, it is the same for query **automobile** to retrieve documents about cars and documents about elephants, if neither included the term **automobile**.

- In latent semantic indexing, the projection space can reflect interrelationships between terms. **Car** and **automobile** are close to each other because they always occur in similar contexts with words (motor, model, vehicle, engine, etc.).
Latent Semantic Indexing

- An indexing and information retrieval method
- Mathematically based on truncated singular value decomposition (SVD)
- Assumes that words used in the same contexts have similar meanings
- “Called latent semantic indexing because of its ability to correlate semantically related terms that are latent in a collection of text”
Singular Value Decomposition (SVD)

WLOG, assume $m \geq n$ and $\text{rank}(A) = r \leq \min(m, n)$,

$$A_{m \times n} = U_{m \times n} \Sigma_{n \times n} V_{n \times n}^T$$

where $U^T U = V^T V = I$,

$\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_n), \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > \sigma_{r+1} = \cdots = \sigma_n = 0$

- Dyadic decomposition:

$$A = \sum_{i=1}^{r} u_i \sigma_i v_i^T$$

where $U = [u_1 u_2 \cdots u_n], V = [v_1 v_2 \cdots v_n]$
Define

\[ A_k = \sum_{i=1}^{k} u_i \sigma_i v_i^T \]

then

\[
\min_{\text{rank}(B)=k} \| A - B \|_F^2 = \| A - A_k \|_F^2
\]

\( A_k \), which is constructed from the k-largest singular triplets of \( A \), is the closest rank-k matrix to \( A \).
Latent Semantic Indexing

- Construct term-document matrix $A$
- Take SVD decomposition of term-document matrix
- Select $k$ and find $A_k$, the best rank-$k$ approximation
- Queries
- Updating
The term-document matrix

\[ A = [a_{ij}] \]

where \( a_{ij} \) denotes the frequency that term \( i \) occurs in document \( j \). We can write

\[ a_{ij} = L(i,j) \times G(i) \]

where \( L(i,j) \) is the local weighting for term \( i \) in document \( j \), and \( G(i) \) is the global weighting for term \( i \).

- By adding local and global weightings, we can change the importance of terms within or among documents.
<table>
<thead>
<tr>
<th>Label</th>
<th>Medical Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>study of depressed patients after discharge with regard to age of onset and culture</td>
</tr>
<tr>
<td>M2</td>
<td>culture of pleuropneumonia like organisms found in vaginal discharge of patients</td>
</tr>
<tr>
<td>M3</td>
<td>study showed oestrogen production is depressed by ovarian irradiation</td>
</tr>
<tr>
<td>M4</td>
<td>cortisone rapidly depressed the secondary rise in oestrogen output of patients</td>
</tr>
<tr>
<td>M5</td>
<td>boys tend to react to death anxiety by acting out behavior while girls tended to become depressed</td>
</tr>
<tr>
<td>M6</td>
<td>changes in children’s behavior following hospitalization studied a week after discharge</td>
</tr>
<tr>
<td>M7</td>
<td>surgical technique to close ventricular septal defects</td>
</tr>
<tr>
<td>M8</td>
<td>chromosomal abnormalities in blood cultures and bone marrow from leukaemic patients</td>
</tr>
<tr>
<td>M9</td>
<td>study of christmas disease with respect to generation and culture</td>
</tr>
<tr>
<td>M10</td>
<td>insulin not responsible for metabolic abnormalities accompanying a prolonged fast</td>
</tr>
<tr>
<td>M11</td>
<td>close relationship between high blood pressure and vascular disease</td>
</tr>
<tr>
<td>M12</td>
<td>mouse kidneys show a decline with respect to age in the ability to concentrate the urine during a water fast</td>
</tr>
<tr>
<td>M13</td>
<td>fast cell generation in the eye lens epithelium of rats</td>
</tr>
<tr>
<td>M14</td>
<td>fast rise of cerebral oxygen pressure in rats</td>
</tr>
<tr>
<td>Terms</td>
<td>Documents</td>
</tr>
<tr>
<td>---------------</td>
<td>-----------</td>
</tr>
<tr>
<td></td>
<td>M</td>
</tr>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>abnormalities</td>
<td>0</td>
</tr>
<tr>
<td>age</td>
<td>1</td>
</tr>
<tr>
<td>behavior</td>
<td>0</td>
</tr>
<tr>
<td>blood</td>
<td>0</td>
</tr>
<tr>
<td>close</td>
<td>0</td>
</tr>
<tr>
<td>culture</td>
<td>1</td>
</tr>
<tr>
<td>depressed</td>
<td>1</td>
</tr>
<tr>
<td>discharge</td>
<td>1</td>
</tr>
<tr>
<td>disease</td>
<td>0</td>
</tr>
<tr>
<td>fast</td>
<td>0</td>
</tr>
<tr>
<td>generation</td>
<td>0</td>
</tr>
<tr>
<td>oestrogen</td>
<td>0</td>
</tr>
<tr>
<td>patients</td>
<td>1</td>
</tr>
<tr>
<td>pressure</td>
<td>0</td>
</tr>
<tr>
<td>rats</td>
<td>0</td>
</tr>
<tr>
<td>respect</td>
<td>0</td>
</tr>
<tr>
<td>rise</td>
<td>0</td>
</tr>
<tr>
<td>study</td>
<td>1</td>
</tr>
</tbody>
</table>
SVD Decomposition and Best Rank-k Approximation

Take SVD decomposition of term-document matrix $A = U\Sigma V^T$, approximated by $A_k = U_k \Sigma_k V_k^T = \sum_{i=1}^{k} u_i \sigma_i v_i^T$.
Projection Space

- k-dimensional Term Projection Space: \((U_k \Sigma_k)_{m \times k}\)
- k-dimensional Document Projection Space: \((V_k \Sigma_k)_{n \times k}\)

For example, when \(k = 2\), \(A_2 = U_2 \Sigma_2 V_2\)
- 2-dimensional Term Projection Space: \((u_1 \sigma_1, u_2 \sigma_2)\)
- 2-dimensional Document Projection Space: \((v_1 \sigma_1, v_2 \sigma_2)\)
Example: Projection Space
A query is a phrase.

In order to retrieve information, query must be projected in the projection space and compared to all existing documents.

The projected query is

\[ \hat{q} = q^T U_k \Sigma_k^{-1} \]

where \( q \) is the vector of words in the query.

- In terms of similarity, a common measure is the cosine between query and document.
- Generally, all documents exceeding some cosine threshold are returned.
Example: Queries

- We want to identify documents that contain information related to the **age of children with blood abnormalities**.

\[
\begin{pmatrix}
0.1491 \\
-0.1199
\end{pmatrix}
= 
\begin{pmatrix}
1 \\
1 \\
0 \\
1 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0 \\
0
\end{pmatrix}^T \begin{pmatrix}
0.1623 \\
0.2068 \\
0.0597 \\
0.1663 \\
0.0258 \\
0.4534 \\
0.3579 \\
0.2931 \\
0.0690 \\
0.0940 \\
0.0599 \\
0.1560 \\
0.4948 \\
0.0460 \\
0.0369 \\
0.1797 \\
0.1087 \\
0.3814
\end{pmatrix}
\begin{pmatrix}
3.5919 & 0 \\
0 & 2.6471
\end{pmatrix}^{-1}
\]

- All documents with cosine greater than 0.85 is considered relevant.
Example: Queries

Presented by Yiyi Yin

Latent Semantic Indexing

September 11, 2017 17 / 34
Example: Queries

- LSI returned relevant topics M8, M9 and M12
- Lexical-matching returned M1, M8, M10, M11 and M12, but M1 and M10 are not relevant, M9 would be missed.
- In LSI, the most relevant topic M9 is returned.

Query: age of children with blood abnormalities

M9: study of christmas disease with respect to generation and culture (christmas disease is the name of hemophilia in young children)
Suppose an LSI model is already built based on the database, more terms and documents are added. How are we going to update the model fitting?

- Folding-in: essentially the same with query representation, quick and simple
- Recomputing the SVD: repeat the whole procedure, requires more computation time
- SVD-updating: computationally efficient
Folding-in: Document

- $p$ new documents
- folding-in a new document vector, $d$, into the existing LSI projection model, a projection, $\hat{d}$ is computed by

$$\hat{d} = d^T U_k \Sigma_k^{-1}$$
Folding-in: Term

- $q$ new terms
- folding-in a new term vector, $t$, into the existing LSI projection model, a projection, $\hat{t}$ is computed by

$$\hat{t} = t^T V_k \Sigma_k^{-1}$$
<table>
<thead>
<tr>
<th>Label</th>
<th>Medical Topic</th>
</tr>
</thead>
<tbody>
<tr>
<td>M15</td>
<td>behavior of rats after detected rise in oestrogen</td>
</tr>
<tr>
<td>M16</td>
<td>depressed patients who feel the pressure to fast</td>
</tr>
</tbody>
</table>
Example: Folding-in
Simply compute the SVD of a reconstructed term-document matrix, say \( \tilde{A} \). Now the new term-document matrix \( \tilde{A} \) is \( 18 \times 16 \).

Recomputing the SVD

\[
\tilde{A} = \tilde{U} \tilde{\Sigma} \tilde{V}^T
\]

Construct the rank-2 approximation to \( \tilde{A} \) by

\[
\tilde{A}_2 = \tilde{U}_2 \tilde{\Sigma}_2 \tilde{V}_2^T
\]
Example: Recomputing the SVD
- Folding-in didn’t reconstruct the semantic representation in the new database.
- The existing LSI didn’t take the association between **behavior** and **rats** into consideration. Thus it fails to form the cluster **M13, M14** and **M15**.
Let $D_{m \times p}$ denote new document vectors, $B = (A_k | D)$, define $SVD(B) = U_B \Sigma_B V_B^T$. Then

$$U_k^T B \begin{pmatrix} V_k & 0 \\ 0 & I_p \end{pmatrix} = (\Sigma_k | U_k^T D)$$

since $A_k = U_k \Sigma_k V_k^T$. If $F = (\Sigma_k | U_k^T D)$ and $SVD(F) = U_F \Sigma_F V_F^T$, then

$$U_B = U_k U_F, \quad V_B = \begin{pmatrix} V_k & 0 \\ 0 & I_p \end{pmatrix} V_F, \quad \Sigma_F = \Sigma_B$$
Let $T_{q \times n}$ denote new term vectors, $C = \begin{pmatrix} A_k \\ T \end{pmatrix}$, define

$$\text{SVD}(C) = U_C \Sigma_C V_C^T.$$ Then

$$\begin{pmatrix} U_k^T \\ 0 \end{pmatrix} C V_k = \begin{pmatrix} \Sigma_k \\ TV_k \end{pmatrix}$$

If $H = \begin{pmatrix} \Sigma_k \\ TV_k \end{pmatrix}$ and $\text{SVD}(H) = U_H \Sigma_H V_H^T$, then

$$U_C = \begin{pmatrix} U_k \\ 0 \end{pmatrix} U_H, \quad V_C = V_k V_H, \quad \Sigma_H = \Sigma_C$$
For a change of weightings in $j$ terms, let $Y_j$ be an $m \times j$ matrix with rows of zeros or rows of $j$-th order identity matrix $I_j$, let $Z_j$ be $n \times j$ matrix whose columns specify the actual differences between old and new weights for each of the $j$ terms. The correction step is actually to compute the SVD decomposition of $W = A_k + Y_j Z_j^T$. Define $SVD(W) = U_W \Sigma_W V_W^T$. Then

$$U_k^T W V_k = (\Sigma_k + U_k^T Y_j Z_j^T V_k)$$

If $Q = (\Sigma_k + U_k^T Y_j Z_j^T V_k)$ and $SVD(Q) = U_Q \Sigma_Q V_Q^T$, then

$$U_W = U_k U_Q, V_W = V_k V_Q$$
Example: SVD-updating
Orthogonality

One important difference between the folding-in and the SVD-updating is the guarantee of orthogonality.

- The folding-in process corrupts the orthogonality by appending non-orthogonal submatrices.
- The SVD-updating can guarantee the orthogonality.
### Compare Updating Methods

<table>
<thead>
<tr>
<th>Method</th>
<th>Computational Complexity</th>
<th>Orthogonality</th>
</tr>
</thead>
<tbody>
<tr>
<td>Folding-in</td>
<td>*</td>
<td>No</td>
</tr>
<tr>
<td>Recomputing the SVD</td>
<td>***</td>
<td>Yes</td>
</tr>
<tr>
<td>SVD-updating</td>
<td>**</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Presented by Yiyi Yin  
Latent Semantic Indexing  
September 11, 2017
Berry, Michael W., Susan T. Dumais, and Todd A. Letsche., 1995
Computational methods for intelligent information access.

Thanks!