Information Retrieval

WS 2012 / 2013

Lecture 8, Wednesday December 12th, 2012
(Synonyms, Latent Semantic Indexing)

Prof. Dr. Hannah Bast
Chair of Algorithms and Data Structures
Department of Computer Science
University of Freiburg

Overview of this lecture

- Organizational
  - Your experiences with ES#7 (cookies, UTF-8)
  - Date for the exam !
- Synonyms
  - Automatic approach: latent semantic indexing (LSI)
  - Based on singular value decomposition (SVD) of the term-document matrix
  - Effectively compute pairs of related terms
  - Exercise Sheet 8: create term-document matrix from our example collection, and get related terms via LSI

Experiences with ES#7 (cookies, UTF-8)

- Summary / excerpts   last checked December 12, 15:54
  - Confusion in Java, some file readers autocorrect the UTF-8
  - Some problems with Cookies and file://...
  - Subscribe to the **Announcements** subforum, then you will get an email when something is posted there

    Actually: best to subscribe to **all** subforums (one per ES)
  - "UTF-8 is a great topic ... lost my fear of text encoding issues"
  - Web stuff nice to look at, but not so nice to actually build it
  - Master solution for the chocolate chip cookies please
  - Last sheets took much longer than 8 hours, please reduce

    I will do my best ... but please don't forget that most of the overtime is due to lack of programming skills / experience

Results for ES#6+7 (web apps)

- Let's look at some of the web apps
  - Suggestions also for multiple keywords
  - Result snippets
  - Nice layout
  - Postponed to next lecture …

Synonyms 1/4

- Problem: another source of word variation
  - We have already seen prefix search

    Type uni ... find university
  - And error-tolerant search

    Type uniwercity ... find university
  - But sometimes there are simply totally different words expressing more or less the same thing

    Type university ... find college

    Type bringdienst ... find lieferservice

    Type cookie ... find biscuit

Synonyms 2/4

- Solution 1: Maintain a thesaurus
  - Hand-maintain a thesaurus of synonyms

    university: uni, academy, college, ...

    bringdienst: lieferservice, heimservice, pizzaservice, ...

    cookie: biscuit, confection, wafer, ...
  - **Problem 1:** laborious, yet notoriously out of date
  - **Problem 2:** it depends on the context, which synonyms are appropriate

    university award ≠ academy award

    http cookie ≠ http biscuit
  - Anyway, that's not the topic of today's lecture ...

Synonyms 3/4

- Solution 2: Track user behaviour
  - Investigate not just individual searches but whole **search sessions** (tracked using, guess what, cookies):
    - The initial query
    - The subsequent queries
    - What the user eventually clicked on
  - Interesting, but not the topic of today's lecture either ...

Synonyms 4/4

- Solution 3: Automatic methods
  - The text itself also tells us which words are related
  - For example: pizza delivery webpages

    they have similar contents (and style)

    some use the word Bringdienst

    some use the word Lieferservice
  - **Latent Semantic Indexing** (LSI) tries to find such relations, based on similar context, automatically
  - This is the topic of today's lecture !

Latent Semantic Indexing 1/9

- An example term-document matrix

	D1	D2	D3	D4	D5	D6	Query
internet	0	1	1	0	1	0	0
web	1	0	1	0	1	0	1
surfing	1	1	1	0	1	1	1
beach	0	0	0	1	1	1	0

same sim.
to query

4 x 6 matrix

Latent Semantic Indexing 2/9

- Assume our matrix is a product of these two

$$\begin{array}{r}\text{internet}\\ \text{web}\\ \text{surfing}\\ \text{beach}\end{array}\underbrace{\begin{bmatrix}1 & 0\\ 1 & 0\\ 1 & 1\\ 0 & 1\end{bmatrix}}_{4\times 2}\cdot\underbrace{\begin{bmatrix}1 & 1 & 1 & 0 & 1 & 0\\ 0 & 0 & 0 & 1 & 1 & 1\end{bmatrix}}_{2\times 6} = \underbrace{\begin{bmatrix}1 & 1 & 1 & 0 & 1 & 0\\ 1 & 1 & 1 & 0 & 1 & 0\\ 1 & 1 & 1 & 1 & 2 & 1\\ 0 & 0 & 0 & 1 & 1 & 1\end{bmatrix}}_{4\times 6}$$

- This is a matrix with column rank 2
  - column rank = all columns can be written as a linear combination of that many "base" columns, but not less
  - row rank = defined analogously
  - Theorem: column rank = row rank

Latent Semantic Indexing 3/9

- If we change only few entries in that matrix
  - we obtain a full-rank matrix again ... check in Octave
  - Let us assume that the matrix came from a rank-2 matrix by changing only a few entries ... which it did
  - Then it's not hard to guess that rank-2 matrix here
  - LSI does this recovering automatically

$$
\begin{array}{r}
\text{internet} \\ \text{web} \\ \text{surfing} \\ \text{beach}
\end{array}
\underbrace{\begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 1 & 1 \\ 0 & 1 \end{bmatrix}}_{4 \times 2}
\cdot
\underbrace{\begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}}_{2 \times 6}
=
\underbrace{\begin{bmatrix} \cancel{1}^{0} & 1 & 1 & 0 & 1 & 0 \\ 1 & \cancel{1}^{0} & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & \cancel{1}^{0} & \cancel{2}^{1} & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}}_{4 \times 6}
$$

Latent Semantic Indexing 4/9

- Definition of Latent Semantic Indexing (LSI)
  - Given an $m \times n$ term-document matrix A
  - And a rank k, typically $<< \min(m, n)$

    Note that the maximal rank is $\min(m, n)$ ... why?
  - Then LSI computes $\mathbf{argmin}_{A_k,\ \mathrm{rank}(A_k) = k}\ \| A - A_k \|$

    that is, the rank-k matrix A_k with minimal distance to A
  - Here $\| . \|$ is the Frobenius norm:

    For a matrix $A = [a_{ij}]$ defined as $\| A \| := \mathrm{sqrt}(\Sigma a_{ij}^2)$
  - **How to compute this miraculous matrix ?**

Latent Semantic Indexing 5/9

- Eigenvector decomposition (EVD)
  - For an $m \times m$ matrix A, and an $m \times 1$ vector x
    we say that x is an eigenvector of A if $\mathbf{A \cdot x = \lambda \cdot x}$
    λ is called an Eigenvalue of A
  - If A is symmetric, A has m linear independent eigenvectors, which hence form a basis of the R^m
  - Then A can be written as $U \cdot D \cdot U^T$
    where D is diagonal, containing the Eigenvalues
    and U is unitarian, that is, $U \cdot U^T = U^T \cdot U = I$
  - This is called the Eigenvector decomposition of A
    sometimes also called Schur decomposition

$$A = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$$

$$\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 3 \\ 3 \end{pmatrix} \quad EV = 3$$

$$\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} 1 \\ -1 \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \end{pmatrix} \quad EV = 1$$

$$U = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad ; \quad UU^T = \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \frac{1}{2}\begin{pmatrix} 2 & 0 \\ 0 & 2 \end{pmatrix}$$

$$= I_2$$

can write A as

$$\begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix} = \frac{1}{2}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} 3 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

Latent Semantic Indexing 6/9

- Singular Value Decomposition (SVD)
  - Let A be an **arbitrary** rectangular $m \times n$ matrix A
  - Then A can be written as $U \cdot \Sigma \cdot V^T$

    where U is $m \times k$, Σ is $k \times k$, and V is $n \times k$ $k = \mathrm{rank}(A)$

    and $U^T \cdot U = I$ and $V^T \cdot V = I$ (but not vice versa !)

    and Σ is a diagonal matrix with the so-called singular values on its diagonal
  - Let's look at an example in Octave ...

Latent Semantic Indexing 7/9

- How to compute the SVD
  - Easy to compute from the EVD ... see below
  - In pratice, use the more direct Lanczos method
  - Which has complexity O(k · nnz), where k is the rank and nnz is the number of non-zero values in the matrix
  - Note that for term-document matrices nnz << n · m

$(A \cdot A^T)^T = A^{T^T} \cdot A^T = A \cdot A^T$

$\underset{m\times n}{A} \cdot \underset{n\times m}{A^T}$ and $\underset{n\times m}{A^T} \cdot \underset{m\times n}{A}$ are symmetric.

$A = U \cdot \Sigma \cdot V^T$ SVD, $U^T U = I$, $V^T V = I$

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

$A \cdot A^T = U \cdot \Sigma \cdot \underbrace{V^T \cdot V}_{} \cdot \Sigma \cdot U^T = U \cdot \Sigma^2 \cdot U^T$

$A^T \cdot A = V \cdot \Sigma \cdot \underbrace{U^T \cdot U}_{=I} \cdot \Sigma \cdot V^T = V \cdot \Sigma^2 \cdot V^T$

Latent Semantic Indexing 8/9

- With the SVD, rank-k approximation becomes easy
  - For a given $m \times n$ matrix A, compute SVD $A = U \cdot \Sigma \cdot V^T$
  - Let U_k = the first k columns of U
  - Let Σ_k = the upper $k \times k$ part of Σ
  - Let V_k = the first k columns of V
  - Then $AA = U_k \cdot \Sigma_k \cdot V_k^T$ is the desired approximation

    that is, that rank-k matrix A_k which minimizes $\mathbf{\| A - A_k \|}$
  - Let's look at our example in Octave ...

Latent Semantic Indexing 9/9

- LSI can be viewed as **document expansion**
  - LSI "replaces" $A = U \cdot \Sigma \cdot V^T$ by $AA = U_k \cdot \Sigma_k \cdot V_k^T$
  - Observe: $\underset{m \times k}{U_k} \cdot \underset{k \times m}{U_k^T} \cdot \underset{m \times m}{U} = \underset{m \times m}{[U_k\ 0]}$ ... let's check in Octave

    U_k with zeroes padded on the right
  - Hence $AA = T \cdot A$, where $T = U_k \cdot U_k^T$ (m x m matrix)
  - Exercise Sheet 9: on our Wikipedia collection, see which term pairs get a high value in T (for various values of k)

$$\begin{array}{l} \\ \text{internet} \\ \text{web} \\ \vdots \\ \\ \end{array} \begin{array}{c} \text{internet} \quad \text{web} \qquad\quad \\ \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \end{array} \cdot \begin{pmatrix} 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}$$

example T

this entry says: if there is "web" add "internet"

Octave 1/5

- Script language for numerical computation
  - GNU's open source version of the proprietary Matlab
  - Makes numerical computations easy, which would otherwise be a pain to use in Java / C++

    In particular: comp. involving matrices and vectors
  - Also comes with an interactive shell ... see next slilde
  - Language has C-like commands (printf, fopen, ...)
  - Still it's a **script language**, and correspondingly slow
  - The built-in functions (like svd) are fast though
  - Download and Doc.: http://www.gnu.org/software/octave

Octave 2/5

- Use the Octave shell pretty much like a Bash shell
  - Arrow ↑ : previous command
  - Arrow ↓ : next command
  - CTRL+R : search in history
  - CTRL+A : go to beginning of line
  - CTRL+E : go to end of line
  - CTRL+K : delete from cursor position to end of line

Octave 3/5

- Here are some commands useful for ES#8
  - Create a vector or matrix

    ```
    A = [1 1 1 0 0; 0 0 1 2 0; 1 0 0 1 1];   // 3 x 5 matrix.
    ```

  - Compute part of SVD pertaining to k top singular values

    ```
    [U, S, V] = svd(A);         // For dense matrices, k = rank(A)
    [U, S, V] = svds(A, k);     // For sparse matrices, must spec. k
    ```

  - Get a portion of a matrix or vector

    ```
    UU = U(:, 1:k);   // First k columns of U.
    ```

  - Multiply a matrix with its transpose

    ```
    T = UU * UU';
    ```

  - Note: if you omit the semicolon or write a comma, the result will be printed on the screen

Octave 4/5

- Sparse matrices
  - Our term-document matrices are very sparse, that is

    nnz << #rows · #cols where nnz = #non-zero values
  - Therefore write in following format, one entry per line

    <row-index> <column-index> <value>
  - Read such a sparse matrix into Octave with

```
tmp = load("A.matrix"));
A = spconvert(tmp);
clear tmp;
```

Octave 5/5

- Vectors of strings
  - Read file with one string per line into Octave like this

```
A = {};
file = fopen("words.txt");
i = 1;
while true
  line = fgetl(file);
  if line == -1, break; endif;
  A(1, i) = line;
  i++;
endwhile
```

  - With Octave version ≥ 3.4, easier with textread ...

References

- Further reading
  - Textbook Chapter 18: Matrix decompositions & LSI
    http://nlp.stanford.edu/IR-book/pdf/18lsi.pdf
  - Deerwester, Dumais, Landauer, Furnas, Harshman
    Indexing by Latent Semantic Analysis, JASIS 41(6), 1990
- Wikipedia
  - http://en.wikipedia.org/wiki/Latent_semantic_indexing
  - http://en.wikipedia.org/wiki/Singular_value_decomposition
  - http://www.gnu.org/software/octave
  - http://en.wikipedia.org/wiki/GNU_Octave