13.1 IR Effectiveness Measures

13.2 Probabilistic IR

13.3 Statistical Language Model

13.4 Latent-Topic Models

  13.4.1 LSI based on SVD
  13.4.2 pLSI and LDA
  13.4.3 Skip-Gram Model

13.5 Learning to Rank

*Not only does God play dice, but He sometimes confuses us by throwing them where they can't be seen.*

-- Stephen Hawking
13.4 Latent Topic Models

- Ranking models like tf*idf, Prob. IR and Statistical LMs do not capture lexical relations between terms in natural language: **synonymy** (e.g. car and automobile), **homonymy** (e.g. java), hyponymy (e.g. SUV and car), meronymy (e.g. wheel and car), etc.

- **Word co-occurrence** and **indirect co-occurrence** can help: car and automobile both occur with fuel, emission, garage, … java occurs with class and method but also with grind and coffee

- **Latent topic models** assume that documents are composed from a number $k$ of **latent (hidden) topics** where $k \ll |V|$ with vocabulary $V$
  → project docs consisting of terms into lower-dimensional space of docs consisting of latent topics
13.4.1 Flashback: SVD

Theorem:
Each real-valued $m \times n$ matrix $A$ with rank $r$ can be decomposed into the form $A = U \times \Delta \times V^T$ with
- an $m \times r$ matrix $U$ with orthonormal column vectors,
- an $r \times r$ diagonal matrix $\Delta$, and
- an $n \times r$ matrix $V$ with orthonormal column vectors.
This decomposition is called singular value decomposition (SVD) and is unique when the elements of $\Delta$ or sorted.

Theorem:
In the singular value decomposition $A = U \times \Delta \times V^T$ of matrix $A$ the matrices $U$, $\Delta$, and $V$ can be derived as follows:
- $\Delta$ consists of the singular values of $A$, i.e. the positive roots of the Eigenvalues of $A^T \times A$,
- the columns of $U$ are the Eigenvectors of $A \times A^T$,
- the columns of $V$ are the Eigenvectors of $A^T \times A$. 
SVD as Low-Rank Approximation (Regression)

Theorem:
Let $A$ be an $m \times n$ matrix with rank $r$, and let $A_k = U_k \times \Delta_k \times V_k^T$, where the $k \times k$ diagonal matrix $\Delta_k$ contains the $k$ largest singular values of $A$ and the $m \times k$ matrix $U_k$ and the $n \times k$ matrix $V_k$ contain the corresponding Eigenvectors from the SVD of $A$.

Among all $m \times n$ matrices $C$ with rank at most $k$ $A_k$ is the matrix that minimizes the Frobenius norm

$$\|A - C\|_F^2 = \sum_{i=1}^{m} \sum_{j=1}^{n} (A_{ij} - C_{ij})^2$$

Example:
$m=2$, $n=8$, $k=1$
projection onto $x'$ axis minimizes „error“ or maximizes „variance“ in $k$-dimensional space
Latent Semantic Indexing (LSI): Applying SVD to Vector Space Model

A is the \(m \times n\) term-document similarity matrix. Then:

- \(U\) and \(U_k\) are the \(m \times r\) and \(m \times k\) term-topic similarity matrices,
- \(V\) and \(V_k\) are the \(n \times r\) and \(n \times k\) document-topic similarity matrices,
- \(A \times A^T\) and \(A_k \times A_k^T\) are the \(m \times m\) term-term similarity matrices,
- \(A^T \times A\) and \(A_k^T \times A_k\) are the \(n \times n\) document-document similarity matrices

Mapping of \(m \times 1\) vectors into latent-topic space:

- \(d_j \mapsto U_k^T \times d_j =: d_j'\)
- \(q \mapsto U_k^T \times q =: q'\)

Scalar-product similarity in latent-topic space: \(d_j' \times q' = ((\Delta_k V_k^T)_{*j})^T \times q'\)
Indexing and Query Processing

• The matrix $\Delta_k V_k^T$ corresponds to a „topic index“ and is stored in a suitable data structure. Instead of $\Delta_k V_k^T$ the simpler index $V_k^T$ could be used.

• Additionally the term-topic mapping $U_k$ must be stored.

• A query $q$ (an $m \times 1$ column vector) in the term vector space is transformed into query $q' = U_k^T \times q$ (a $k \times 1$ column vector) and evaluated in the topic vector space (i.e. $V_k$) (e.g. by scalar-product similarity $V_k^T \times q'$ or cosine similarity)

• A new document $d$ (an $m \times 1$ column vector) is transformed into $d' = U_k^T \times d$ (a $k \times 1$ column vector) and appended to the „index“ $V_k^T$ as an additional column („folding-in“)
Example 1 for Latent Semantic Indexing

\[ m=5 \text{ (interface, library, Java, Kona, blend), } n=7 \]

\[
A = \begin{pmatrix}
1 & 2 & 1 & 5 & 0 & 0 & 0 \\
1 & 2 & 1 & 5 & 0 & 0 & 0 \\
1 & 2 & 1 & 5 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 2 & 3 & 1 \\
0 & 0 & 0 & 0 & 2 & 3 & 1
\end{pmatrix}
= \begin{pmatrix}
0.58 & 0.00 \\
0.58 & 0.00 \\
0.58 & 0.00 \\
0.00 & 0.71 \\
0.00 & 0.71
\end{pmatrix}
\times \begin{pmatrix}
9.64 & 0.00 \\
0.00 & 5.29
\end{pmatrix}
\times \begin{pmatrix}
0.18 & 0.36 & 0.18 & 0.90 & 0.00 & 0.00 & 0.00 \\
0.00 & 0.00 & 0.00 & 0.00 & 0.53 & 0.80 & 0.27
\end{pmatrix}
\]

query \( q = (0\ 0\ 1\ 0\ 0)^T \) is transformed into \( q' = U^T \times q = (0.58 \ 0.00)^T \) and evaluated on \( V^T \)

the new document \( d_8 = (1\ 1\ 0\ 0\ 0)^T \) is transformed into \( d_8' = U^T \times d_8 = (1.16 \ 0.00)^T \) and appended to \( V^T \)
Example 2 for Latent Semantic Indexing

m=6 terms
- t1: bak(e,ing)
- t2: recipe(s)
- t3: bread
- t4: cake
- t5: pastr(y,ies)
- t6: pie

n=5 documents
- d1: How to bake bread without recipes
- d2: The classic art of Viennese Pastry
- d3: Numerical recipes: the art of scientific computing
- d4: Breads, pastries, pies and cakes: quantity baking recipes
- d5: Pastry: a book of best French recipes

\[ A = \begin{pmatrix} 0.5774 & 0.0000 & 0.0000 & 0.4082 & 0.0000 \\ 0.5774 & 0.0000 & 1.0000 & 0.4082 & 0.7071 \\ 0.5774 & 0.0000 & 0.0000 & 0.4082 & 0.0000 \\ 0.0000 & 0.0000 & 0.0000 & 0.4082 & 0.0000 \\ 0.0000 & 1.0000 & 0.0000 & 0.4082 & 0.7071 \\ 0.0000 & 0.0000 & 0.0000 & 0.4082 & 0.0000 \end{pmatrix} \]
Example 2 for Latent Semantic Indexing (2)

\[ A = \begin{pmatrix}
0.2670 & -0.2567 & 0.5308 & -0.2847 \\
0.7479 & -0.3981 & -0.5249 & 0.0816 \\
0.2670 & -0.2567 & 0.5308 & -0.2847 \\
0.1182 & -0.0127 & 0.2774 & 0.6394 \\
0.5198 & 0.8423 & 0.0838 & -0.1158 \\
0.1182 & -0.0127 & 0.2774 & 0.6394 \\
\end{pmatrix} \]

\[ \times \begin{pmatrix}
1.6950 & 0.0000 & 0.0000 & 0.0000 \\
0.0000 & 1.1158 & 0.0000 & 0.0000 \\
0.0000 & 0.0000 & 0.8403 & 0.0000 \\
0.0000 & 0.0000 & 0.0000 & 0.4195 \\
\end{pmatrix} \]

\[ \times \begin{pmatrix}
0.4366 & 0.3067 & 0.4412 & 0.4909 & 0.5288 \\
-0.4717 & 0.7549 & -0.3568 & -0.0346 & 0.2815 \\
0.3688 & 0.0998 & -0.6247 & 0.5711 & -0.3712 \\
-0.6715 & -0.2760 & 0.1945 & 0.6571 & -0.0577 \\
\end{pmatrix} \]
Example 2 for Latent Semantic Indexing (3)

\[
A_3 = \begin{pmatrix}
0.4971 & -0.0330 & 0.0232 & 0.4867 & -0.0069 \\
0.6003 & 0.0094 & 0.9933 & 0.3858 & 0.7091 \\
0.4971 & -0.0330 & 0.0232 & 0.4867 & -0.0069 \\
0.1801 & 0.0740 & -0.0522 & 0.2320 & 0.0155 \\
-0.0326 & 0.9866 & 0.0094 & 0.4402 & 0.7043 \\
0.1801 & 0.0740 & -0.0522 & 0.2320 & 0.0155
\end{pmatrix} = U_3 \times \Delta_3 \times V_3^T
\]

IRDM WS 2015
Example 2 for Latent Semantic Indexing (4)

query \( q \): baking bread
\( q = (1 \ 0 \ 1 \ 0 \ 0 \ 0)^T \)

transformation into topic space with \( k=3 \)
\( q' = U_k^T \times q = (0.5340 \ -0.5134 \ 1.0616)^T \)

scalar product similarity in topic space with \( k=3 \):
\[
\begin{align*}
sim(q, d1) &= V^T_{k*1} \times q' \approx 0.86 \\
sim(q, d2) &= V^T_{k*2} \times q \approx -0.12 \\
sim(q, d3) &= V^T_{k*3} \times q' \approx -0.24 \\
\end{align*}
\]

Folding-in of a new document \( d_6 \):
algorithmic recipes for the computation of pie
\( d_6 = (0 \ 0.7071 \ 0 \ 0 \ 0 \ 0.7071)^T \)

transformation into topic space with \( k=3 \)
\( d_6' = U_k^T \times d_6 \approx (0.5 \ -0.28 \ -0.15) \)

\( d_6' \) is appended to \( V_k^T \) as a new column
Multilingual Retrieval with LSI

- Construct LSI model \((U_k, \Delta_k, V_k^T)\) from training documents that are available in multiple languages:
  - consider all language variants of the same document as a single document and
  - extract all terms or words for all languages.
- Maintain index for further documents by „folding-in“, i.e. mapping into topic space and appending to \(V_k^T\).
- Queries can now be asked in any language, and the query results include documents from all languages.

Example:

\(d1: \textit{How to bake bread without recipes.} \)
\(\textit{Wie man ohne Rezept Brot backen kann.}\)
\(d2: \textit{Pastry: a book of best French recipes.} \)
\(\textit{Gebäck: eine Sammlung der besten französischen Rezepte.}\)

Terms are e.g. bake, bread, recipe, backen, Brot, Rezept, etc.
Documents and terms are mapped into compact topic space.
Connections between LSI and Clustering

LSI can also be seen as an **unsupervised clustering** method (cf. *spectral clustering*):

- simple variant for k clusters
  - map each data point into **k-dimensional space**
  - assign each point to its highest-value dimension: **strongest spectral component**

Conversely, we could compute k clusters for the data points (using any clustering algorithm) and **project data points onto k centroid vectors** („axes“ of k-dim. space) to represent data in LSI-style manner („*concept indexing* (CI)“)
More General Matrix Factorizations

Non-negative Matrix Factorization (NMF)

\[ A_{m \times n} \approx L_{m \times k} \times R_{k \times n} \] to minimize \[ \left\| A - L^T R \right\|_F^2 = \sum_{i=1}^m \sum_{j=1}^n (A_{ij} - L^T R_{ij})^2 \]

with \( L_{ij} \geq 0 \) and \( R_{ij} \geq 0 \)

Matrix Factorization with **L2 Regularizer**

\[ A_{m \times n} \approx L_{m \times k} \times R_{k \times n} \] to minimize \[ \left\| A - L^T R \right\|_F^2 + \left\| L \right\|_F^2 + \left\| R \right\|_F^2 \]

Matrix Factorization with **L1 Regularizer** (favors sparseness)

\[ A_{m \times n} \approx L_{m \times k} \times R_{k \times n} \] to minimize \[ \left\| A - L^T R \right\|_F^2 + \left\| L \right\|_1 + \left\| R \right\|_1 \]

\[ \rightarrow \] numerical methods for non-convex optimization

e.g. iterative **gradient descent**
Power of Non-negative Matrix Factorization (NMF) vs. SVD

SVD of data matrix A

NMF of data matrix A
Application: Recommender Systems

Users $\times$ Items $\rightarrow$ Ratings

<table>
<thead>
<tr>
<th></th>
<th>Alice</th>
<th>Bob</th>
<th>Claire</th>
<th>Don</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>2</td>
<td>?</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>?</td>
<td>4</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
<tr>
<td>5</td>
<td>?</td>
<td>4</td>
<td>4</td>
<td>?</td>
</tr>
<tr>
<td>4</td>
<td>?</td>
<td>5</td>
<td>1</td>
<td>4</td>
</tr>
</tbody>
</table>

Low-rank matrix factorization with regularization:

$$M_{u\times t} \approx L_{u\times k} \times R_{k\times t}$$

such that

$$\sum_{ij} (M_{ij} - (L\times R)_{ij} - b_i - b_j)^2 + \lambda (||L||_2 + ||R||_2) = \text{min!}$$

alternatively:

$$\ldots + \lambda (||L||_1 + ||R||_1) = \text{min!}$$

possibly with constraints: $L_{ij} \geq 0$ and $R_{ij} \geq 0$

plus temporal bias …

plus user-user profile sim …

plus item-item contents sim …
Application: Recommender Systems

Also applicable to social graphs, and co-occurrence graphs from user logs, text mining, etc. For recommending „friends“, communities, bars, songs, etc. (see IRDM Chapter 7) → huge size poses scalability challenge.
+ Elegant well-founded model with automatic consideration of term-term (cor)relations (incl. synonymy/homonymy, morphological variations, cross-lingual)

– **Model Selection**: choice of low rank k not easy

– **Computational and storage cost**:
  term-doc matrix is sparse, SVD factors are dense
  SVD does not scale to Web dimensions (10s of Mio‘s to 100s of Bio‘s))

– Unconvincing results for **IR benchmarks** and Web search
13.4.2 Probabilistic Aspect Models (pLSI, LDA, …)

- each document $d$ is viewed as a mix of (latent) topics (aspects) $z$,
  each with a certain probability (summing up to 1)
- each topic generates words $w$ with topic-specific probabilities

- $P[wdz]$: prob. of word $w$ occurring in doc $d$ about topic $z$
- we postulate: conditional independence of $w$ and $d$ given $z$

$$
P[wdz] = P[wd|z] P[z] = P[w|z] P[d|z] P[z]$$

$$
P[wd] = \sum_z P[w|z] P[d|z] P[z]$$

$$
P[w|d] = \sum_z P[z|d] P[w|z] \quad \text{generative model}$$
Probabilistic LSI (pLSI)

\[ P[w \mid d] = \sum_z P[z \mid d] \cdot P[w \mid z] \]

d and w conditionally independent given z

**Documents** \(d\)  
**Latent concepts** \(z\) (aspects)  
**Terms** \(w\) (words)
Relationship of pLSI to LSI

\[ P[w, d] = \sum_z P[w|z] \cdot P[z] \cdot P[d|z] \]

**Key difference to LSI:**
- non-negative matrix decomposition
- with L1 normalization

**Key difference to LMs:**
- no generative model for docs
- tied to given corpus
Learning and Using the pLSI Model

Parameter estimation:
given \((d,w)\) data and \(\text{#aspects k}\),
estimate \(P[z|d]\) and \(P[w|z]\) by EM
(Expectation Maximization, see Chapter 5: EM Clustering)
or gradient-descent methods for analytically intractable MLE or MAP

Query processing:
\(q = \{w_1…w_n\}\) is „folded in“ (via EM and learned model)
to compute \(P[z|q]\): aspect vector that best explains the query

Ranking of query results:
compare the aspect vectors of query and candidate documents
by Kullback-Leibler divergence or other similarity measure (e.g. cosine)
## Experimental Results: Example

### Concepts (10 of 128) extracted from Science Magazine articles (12K)

<table>
<thead>
<tr>
<th>Category</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>universe</td>
<td>0.0439</td>
</tr>
<tr>
<td>galaxies</td>
<td>0.0375</td>
</tr>
<tr>
<td>clusters</td>
<td>0.0279</td>
</tr>
<tr>
<td>matter</td>
<td>0.0233</td>
</tr>
<tr>
<td>galaxy</td>
<td>0.0232</td>
</tr>
<tr>
<td>cluster</td>
<td>0.0214</td>
</tr>
<tr>
<td>cosmic</td>
<td>0.0137</td>
</tr>
<tr>
<td>dark</td>
<td>0.0131</td>
</tr>
<tr>
<td>light</td>
<td>0.0109</td>
</tr>
<tr>
<td>density</td>
<td>0.01</td>
</tr>
<tr>
<td>bacteria</td>
<td>0.0983</td>
</tr>
<tr>
<td>bacterial</td>
<td>0.0561</td>
</tr>
<tr>
<td>resistance</td>
<td>0.0431</td>
</tr>
<tr>
<td>coli</td>
<td>0.0381</td>
</tr>
<tr>
<td>strains</td>
<td>0.025</td>
</tr>
<tr>
<td>microbiol</td>
<td>0.0214</td>
</tr>
<tr>
<td>microbial</td>
<td>0.0196</td>
</tr>
<tr>
<td>strain</td>
<td>0.0165</td>
</tr>
<tr>
<td>salmonella</td>
<td>0.0163</td>
</tr>
<tr>
<td>resistant</td>
<td>0.0145</td>
</tr>
<tr>
<td>male</td>
<td>0.0558</td>
</tr>
<tr>
<td>females</td>
<td>0.0541</td>
</tr>
<tr>
<td>female</td>
<td>0.0529</td>
</tr>
<tr>
<td>males</td>
<td>0.0477</td>
</tr>
<tr>
<td>sex</td>
<td>0.0339</td>
</tr>
<tr>
<td>reproductive</td>
<td>0.0172</td>
</tr>
<tr>
<td>offspring</td>
<td>0.0168</td>
</tr>
<tr>
<td>sexual</td>
<td>0.0166</td>
</tr>
<tr>
<td>reproduction</td>
<td>0.0143</td>
</tr>
<tr>
<td>eggs</td>
<td>0.0138</td>
</tr>
<tr>
<td>theory</td>
<td>0.0811</td>
</tr>
<tr>
<td>physics</td>
<td>0.0782</td>
</tr>
<tr>
<td>physicists</td>
<td>0.0146</td>
</tr>
<tr>
<td>einstein</td>
<td>0.0142</td>
</tr>
<tr>
<td>university</td>
<td>0.013</td>
</tr>
<tr>
<td>gravity</td>
<td>0.013</td>
</tr>
<tr>
<td>black</td>
<td>0.0127</td>
</tr>
<tr>
<td>theories</td>
<td>0.01</td>
</tr>
<tr>
<td>aps</td>
<td>0.00987</td>
</tr>
<tr>
<td>matter</td>
<td>0.00954</td>
</tr>
<tr>
<td>immune</td>
<td>0.0909</td>
</tr>
<tr>
<td>response</td>
<td>0.0375</td>
</tr>
<tr>
<td>system</td>
<td>0.0358</td>
</tr>
<tr>
<td>responses</td>
<td>0.0322</td>
</tr>
<tr>
<td>antigen</td>
<td>0.0263</td>
</tr>
<tr>
<td>antigens</td>
<td>0.0184</td>
</tr>
<tr>
<td>immunity</td>
<td>0.0176</td>
</tr>
<tr>
<td>immunology</td>
<td>0.0145</td>
</tr>
<tr>
<td>antibody</td>
<td>0.014</td>
</tr>
<tr>
<td>autoimmune</td>
<td>0.0128</td>
</tr>
<tr>
<td>stars</td>
<td>0.0524</td>
</tr>
<tr>
<td>star</td>
<td>0.0458</td>
</tr>
<tr>
<td>astrophys</td>
<td>0.0237</td>
</tr>
<tr>
<td>mass</td>
<td>0.021</td>
</tr>
<tr>
<td>disk</td>
<td>0.0173</td>
</tr>
<tr>
<td>black</td>
<td>0.0161</td>
</tr>
<tr>
<td>gas</td>
<td>0.0149</td>
</tr>
<tr>
<td>stellar</td>
<td>0.0127</td>
</tr>
<tr>
<td>astron</td>
<td>0.0125</td>
</tr>
<tr>
<td>hole</td>
<td>0.00824</td>
</tr>
</tbody>
</table>

Source: Thomas Hofmann, Tutorial at ADFOCS 2004
13.4.3 Latent Dirichlet Allocation (LDA)

- Multiple-cause mixture model
- Documents contain multiple latent topics
- Topics are expressed by (multinomial) word distribution
- LDA is a generative model for such docs (Dirichlet topic mixtures)
LDA Generative Model

for each doc d:

- choose doc length $N$ (# word occurrences) $\sim$ Poisson($\lambda$)
- choose topic-probability params $\beta \sim$ Dirichlet($\alpha$)
- for each of the $N$ word occurrences in d (at position $n$):
  - choose one of $k$ topics $z_n \sim$ multinomial($\beta$, $k$)
  - choose one of $M$ words $w_n$ from per-topic distribution $\sim$ multinomial($\theta$, $M$)
LDA Instance-Level Model

\[ \alpha \]

hypergenerator for topic distribution

doc 1 \[ \beta \]

topics of words

\[ z_1 \quad z_2 \quad \ldots \quad z_N \]

\[ w_1 \quad w_2 \quad \ldots \quad w_N \]

\[ \theta \]

topic 1

\[ \theta \]

topic k

\[ \beta \]

doc D

\[ z_1 \quad z_2 \quad \ldots \quad z_N \]

\[ w_1 \quad w_2 \quad \ldots \quad w_N \]

IRDM WS 2015
Comparison to Other Latent-Topic Models

LDA

Dirichlet ($\alpha$) \rightarrow \text{multinomial ($\beta$, $k$)} \rightarrow \text{topic $z$} \rightarrow \text{word $w$}

pLSI

doc $d$ \rightarrow \text{topic $z$} \rightarrow \text{word $w$}

simple unigram model

word $w$

single-cause mixture of unigrams

discrete univariate distribution

IRDM WS 2015 13-97
LDA Parameter Estimation

for doc \( x \)

(if \( \beta \) were known):

\[
P[x | \beta, \theta] = \prod_{n=1}^{N} \sum_{z_n=1}^{k} P[z_n | \beta] P[x_n | \theta_{z_n}] \\
= \prod_{n=1}^{N} \sum_{z_n=1}^{k} \beta_{z_n} \theta_{z_n}, x_n
\]

with

unknown \( \beta \) :

\[
P[x | \alpha, \theta] = \int_{\beta} P[\beta | \alpha] \left( \prod_{n=1}^{N} \sum_{z_n=1}^{k} \beta_{z_n} \theta_{z_n}, x_n \right) d\beta
\]

\[
\Leftrightarrow
P[x | \alpha, \theta] = \frac{\Gamma(\sum_{y} \alpha_{y})}{\prod_{y} \Gamma(\alpha_{y})} \int_{\beta} \prod_{y=1}^{k} \beta_y^{\alpha_y-1} \left( \prod_{n=1}^{N} \sum_{z_n=1}^{k} \beta_{z_n} \theta_{z_n}, x_n \right) d\beta
\]

log-likelihood function (for corpus of \( D \) docs) is analytically intractable

→ EM algorithm or other variational methods or MCMC sampling
# LDA Experimental Results: Example

<table>
<thead>
<tr>
<th>“Arts”</th>
<th>“Budgets”</th>
<th>“Children”</th>
<th>“Education”</th>
</tr>
</thead>
<tbody>
<tr>
<td>NEW</td>
<td>MILLION</td>
<td>CHILDREN</td>
<td>SCHOOL</td>
</tr>
<tr>
<td>FILM</td>
<td>TAX</td>
<td>WOMEN</td>
<td>STUDENTS</td>
</tr>
<tr>
<td>SHOW</td>
<td>PROGRAM</td>
<td>PEOPLE</td>
<td>SCHOOLS</td>
</tr>
<tr>
<td>MUSIC</td>
<td>BUDGET</td>
<td>CHILD</td>
<td>EDUCATION</td>
</tr>
<tr>
<td>MOVIE</td>
<td>BILLION</td>
<td>YEARS</td>
<td>TEACHERS</td>
</tr>
<tr>
<td>PLAY</td>
<td>FEDERAL</td>
<td>FAMILIES</td>
<td>HIGH</td>
</tr>
<tr>
<td>MUSICAL</td>
<td>YEAR</td>
<td>WORK</td>
<td>PUBLIC</td>
</tr>
<tr>
<td>BEST</td>
<td>SPENDING</td>
<td>PARENTS</td>
<td>TEACHER</td>
</tr>
<tr>
<td>ACTOR</td>
<td>NEW</td>
<td>SAYS</td>
<td>BENNETT</td>
</tr>
<tr>
<td>FIRST</td>
<td>STATE</td>
<td>FAMILY</td>
<td>MANIGAT</td>
</tr>
<tr>
<td>YORK</td>
<td>PLAN</td>
<td>WELFARE</td>
<td>NAMPHY</td>
</tr>
<tr>
<td>OPERA</td>
<td>MONEY</td>
<td>MEN</td>
<td>STATE</td>
</tr>
<tr>
<td>THEATER</td>
<td>PROGRAMS</td>
<td>PERCENT</td>
<td>PRESIDENT</td>
</tr>
<tr>
<td>ACTRESS</td>
<td>GOVERNMENT</td>
<td>CARE</td>
<td>ELEMENTARY</td>
</tr>
<tr>
<td>LOVE</td>
<td>CONGRESS</td>
<td>LIFE</td>
<td>HAITI</td>
</tr>
</tbody>
</table>

Source:
D.M. Blei, A.Y. Ng, M.I. Jordan:

The **William Randolph Hearst Foundation** will give $1.25 million to **Lincoln Center**, Metropolitan Opera Co., New York Philharmonic and Juilliard School. “Our board felt that we had a real opportunity to make a mark on the future of the performing arts with these grants an act every bit as important as our traditional areas of support in health, medical research, education and the social services,” Hearst Foundation President Randolph A. Hearst said Monday in announcing the grants. **Lincoln Center’s share** will be $200,000 for its new building, which will house young artists and provide new public facilities. The Metropolitan Opera Co. and New York Philharmonic will receive $400,000 each. The **Juilliard School**, where music and the performing arts are taught, will get $250,000. The **Hearst Foundation**, a leading supporter of the Lincoln Center Consolidated Corporate Fund, will make its usual **annual $100,000** donation, too.
13.4.4 Word2Vec: Latent Model for Term-Term Similarity

• view **distributional representation** (latent aspects) as a machine learning problem
• focus on **term vectors** for term-term similarity (terms: words, phrases, perhaps paragraphs)

Learn from **text windows C** of Web-scale corpora

Example: once **upon a time in** the west

window C of size 4

Aim to predict

\[ P[w|C] = P[w_t | w_{t-j}, \ldots, w_{t+j} \text{ with } 1 \leq j \leq |C|/2] \]

or

\[ P[C|w] = P[w_{t-j}, \ldots, w_{t+j} \text{ for } 1 \leq j \leq |C|/2 | w_t] \]

CBOW model (continuous bag of words)

**IRDM WS 2015 13-100**

cb:

https://code.google.com/p/word2vec/
Word2Vec: Learning Task

Objective: represent term $w$ as vector $\vec{w}$ such that for training corpus with term sequence $w_1 \ldots w_T$:

$$\frac{1}{T} \sum_{t=1}^{T} \sum_{j \in C(t)} \log \frac{\exp (\vec{w}_j^T \vec{w}_t)}{\sum_{v \in V} \exp (\vec{v}^T \vec{w}_t)} = \max!$$

softmax function based on (shallow) neural network

Approximate solution:
advanced machine learning methods (non-convex optimization)

Output:
distributional vector $\vec{w}$ for each term $w$ (word or phrase or …)
for given term \( w \) with vector \( \vec{w} \) find closest \( \vec{u} \) (e.g using cos) 
\[ \rightarrow u \text{ is interpreted as most related term of } w \]

<table>
<thead>
<tr>
<th>term vector</th>
<th>nearest vectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Redmond</td>
<td>Redmond Washington, Microsoft</td>
</tr>
<tr>
<td>graffiti</td>
<td>spray paint, grafitti, taggers</td>
</tr>
<tr>
<td>San_Francisco</td>
<td>Los_Angeles, Golden_Gate, Oakland, Seattle</td>
</tr>
<tr>
<td>Chinese_river</td>
<td>Yangtze_River, Yangtze, Yangtze_tributary</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2 term vectors</th>
<th>nearest vectors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Czech + currency</td>
<td>koruna, Czech crown, Polish zloty, CTK</td>
</tr>
<tr>
<td>Vietnam + capital</td>
<td>Hanoi, Ho Chi Minh City, Viet Nam, Vietnamese</td>
</tr>
<tr>
<td>German + airlines</td>
<td>airline Lufthansa, carrier Lufthansa</td>
</tr>
<tr>
<td>Russian + river</td>
<td>Moscow, Volga River, upriver, Russia</td>
</tr>
<tr>
<td>French + actress</td>
<td>Juliette Binoche, Charlotte Gainsbourg</td>
</tr>
</tbody>
</table>
Word2Vec: Compositionality

Simply use linear algebra: vector addition and subtraction

\[ \text{vec}(X) - \text{vec}(X') = \text{vec}(Y) - \text{vec}(Y') \]

→ given \( Y, Y', X \), solve for \( X' \) \( : \text{vec}(X') = \text{vec}(Y) - \text{vec}(Y') + \text{vec}(X) \)

<table>
<thead>
<tr>
<th>( Y : Y' )</th>
<th>( X : X' )</th>
</tr>
</thead>
<tbody>
<tr>
<td>France:Paris</td>
<td>Italy:Rome, Japan:Tokyo</td>
</tr>
<tr>
<td>big:bigger</td>
<td>small:larger, cold:colder, quick:quicker</td>
</tr>
<tr>
<td>Einstein:scientist</td>
<td>Messi:midfielder, Mozart:violinist, Picasso:painter</td>
</tr>
<tr>
<td>Microsoft:Windows</td>
<td>Google:Android, IBM:Linux, Apple:iPhone</td>
</tr>
<tr>
<td>Sarkozy:France</td>
<td>Berlusconi:Italy, Merkel:Germany</td>
</tr>
<tr>
<td>Japan:sushi</td>
<td>Germany:bratwurst, France:tapas, USA:pizza</td>
</tr>
</tbody>
</table>

Can also be used to automatically mine linguistic regularities, e.g.:
\[ \text{vec}(\text{woman}) - \text{vec}(\text{man}) = \text{vec}(\text{queen}) - \text{vec}(\text{king}) = \text{vec}(\text{aunt}) \] \( - \text{vec}(\text{uncle}) \)
Summary of Section 13.4

- **Latent-Topic Models** can capture word correlations like synonymy in an implicit manner:
  - docs belong to (mixes of) latent topics, topics create words
- **LSI** is based on *spectral decomposition (SVD)* of term-doc matrix
  - elegant, effective, not scalable to Web size
- **pLSI** and **LDA** use non-negative, probabilistic decomposition
  - parameter estimation and query processing complex & expensive
- Other interesting models: **co-clustering, word2vec**, …
- none of these scales to Bios. of docs and Web workload
- all have a **model selection issue**: # topics (aspects)
Additional Literature for Section 13.4

- T. Hofmann: Matrix Decomposition Techniques in Machine Learning and Information Retrieval, Tutorial Slides, ADFOCS 2004
- D. Blei: Probabilistic Topic Models, CACM 2012
- X. Wei, W.B. Croft: LDA-based Document Models for Ad-hoc Retrieval, SIGIR ‘06
- I.S. Dhillon, S. Mallela, D.S. Modha: Information-theoretic Co-clustering. KDD ‘03
- W. Xu, X. Liu, Y. Gong: Document Clustering based on Non-negative Matrix Factorization, SIGIR 2003
- T. Mikolov et al.: Distributed Representations of Words and Phrases and their Compositionality. NIPS 2013
Outline

13.1 IR Effectiveness Measures
13.2 Probabilistic IR
13.3 Statistical Language Model
13.4 Latent-Topic Models
13.5 Learning to Rank
13.5 Learning to Rank

Why?
• Increasing complexity of combining all feature groups: doc contents, source authority, freshness, geo-location, language style, author’s online behavior, etc. etc.
• High dynamics of contents and user interests

How?
• exploit user feedback on search-result quality
• train a machine-learning predictor: scoring function $f$ (query features, doc features)
• use the learned scoring function (weights) to rank the answers of new queries
• re-train the scoring function periodically
Learning-to-Rank (LTR) Framework

Treat scoring as a **function** of different m **input signals** (feature families) $x_i$ with **weights** (hyper-parameters) $\alpha_i$

$$score(d,q) = f(x_1, \ldots, x_m, \alpha_1, \ldots, \alpha_m)$$

where the weights $\alpha_i$ need to be learned and the $x_i$ are derived from $d$ and $q$ and the context (e.g. tokens and bigrams of $d$ and $q$, last update of $d$, age of $d$‘s Internet domain, user’s preceding query, last clicked doc, etc. etc.)

**Training data**: set of **queries** each with info about docs
- **pointwise**: set of $(q,d)$ points with relevant and irrelevant docs
- **pairwise**: set of $(d,d')$ pairs where $d$ is preferred over $d'$
- **listwise**: list of ranked docs in desc. order of relevance

**Objective function** for learning task varies with setting and quality measure to optimize (e.g. precision, F1, NDCG, …)
Regression for Parameter Fitting
Linear Regression

Estimate $r(x) = E[Y \mid X_1=x_1 \land \ldots \land X_m=x_m]$ using a linear model $Y = r(x) + \varepsilon = \beta_0 + \sum_{i=1}^{m} \beta_i x_i + \varepsilon$ with error $\varepsilon$ with $E[\varepsilon]=0$

given $n$ sample points $(x_1^{(i)}, \ldots, x_m^{(i)}, y^{(i)}), i=1..n$, the least-squares estimator (LSE) minimizes the quadratic error:

$$\sum_{i=1..n} \left( \left( \sum_{k=0..m} \beta_k x_k^{(i)} \right) - y^{(i)} \right)^2 =: E(\beta_0,\ldots,\beta_m) \quad \text{(with } x_0^{(i)}=1\text{)}$$

Solve linear equation system: $\frac{\partial E}{\partial \beta_k} = 0$ for $k=0, \ldots, m$

equivalent to MLE $\hat{\beta} = (X^T X)^{-1} X^T Y$

with $Y = (y^{(1)} \ldots y^{(n)})^T$ and $X = \begin{pmatrix} 1 & x_1^{(1)} & x_2^{(1)} & \ldots & x_m^{(1)} \\ 1 & x_1^{(2)} & x_2^{(2)} & \ldots & x_m^{(2)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(n)} & x_2^{(n)} & \ldots & x_m^{(n)} \end{pmatrix}$
Estimate $r(x) = E[Y \mid X=x]$ for **Bernoulli** $Y$ using a logistic model

$$Y = r(x) + \varepsilon = \frac{e^{\beta_0 + \sum_{i=1}^{m} \beta_i x_i}}{1 + e^{\beta_0 + \sum_{i=1}^{m} \beta_i x_i}} + \varepsilon$$

log-linear

with error $\varepsilon$ with $E[\varepsilon]=0$

→ solution for MLE for $\beta_i$ values

based on **numerical gradient-descent** methods
Pointwise LTR with Linear Regression

given \( n \) samples \((x_1, y_1), (x_2, y_2), \ldots\)
find linear function \( f(x) \) with smallest L2 error \( \sum_i (f(x_i) - y_i)^2 \)
(method of least squares)
solve linear equation system (or SVD) over \((x_i, y_i)\) matrix
generalizes to \( m \)-dimensional input \((x_{i_1}, x_{i_2}, \ldots, x_{i_m}, y_i), \ldots\)

\[
\begin{align*}
f(x_1) &= 0.4 \\
f(x_2) &= 0.6 \\
f(x_3) &= 0.9 \\
f(x_4) &= 0.5 \\
f(x_5) &= 0.8
\end{align*}
\]
Pointwise LTR with Logistic Regression

given n m-dim. samples \((x_{i1}, x_{i2}, \ldots, x_{im}, y_i)\) with \(y_i \in \{0, 1\}\)
find coefficient vector \(\beta\) of logistic function \(f(x)\) with
smallest log-linear error \(~ \sum_{i=1}^{n} (y_i \beta^T x_i - \log (1+e^{\beta^T x_i})) + ||\beta||_1~\)
solve numerically by iterative gradient-descent methods

data error (log-likelihood)
model complexity (regularizer)

this is a binary classifier (cf. Chapter 6)
Pairwise LTR with Ordinal Regression

given $x_1$, $x_2$, $x_3$, … and preferences $x_i \prec_p x_j$ ("$x_i$ is better than $x_j$")
find function $f(x)$ with low violation of preference inequalities
→ minimize ranking loss $\sim \Sigma_{i,j} L(x_i,x_j) + \ldots$ where

$L(x_i,x_j) = 1$ if $x_i \prec_p x_j$ and $f(x_i) > f(x_j)$ or $x_i \succ_p x_j$ and $f(x_i) < f(x_j)$, 0 else
→ advanced optimization methods (e.g. SVM-Rank [T. Joachims et al. 2005])
Additional Literature for Section 13.5

- T. Joachims: Optimizing Search Engines using Clickthrough Data, KDD 2002
- T. Joachims et al.: Accurately Interpreting Clickthrough Data as Implicit Feedback, SIGIR 2005
- C.J.C. Burges et al.: Learning to rank using gradient descent. ICML 2005
Summary of Chapter 13

- **Probabilistic IR** and **Statistical Language Models** are the state-of-the-art ranking methods

- LMs are very versatile and composable

- **Latent Topic Models (LSI, LDA)** are powerful for consideration of term-term (cor)relations, but do not scale to Web

- **Learning-to-Rank** is very powerful and used for Web search, for training hyper-parameters of different feature groups and scoring models