Information Retrieval
and Web Search

Probabilistic IR and
Alternative IR Models

Rada Mihalcea

(Some of the slides in this slide set come from a lecture by Samer Hassan at U. North Texas)
IR Models

User Task

- Retrieval: Adhoc Filtering
- Structured Models: Non-Overlapping Lists, Proximal Nodes
- Classic Models: boolean vector probabilistic
- Browsing: Flat Structure, Guided Hypertext
- Browsing: Inference Network, Belief Network
- Probabilistic: Generalized Vector, Lat. Semantic Index, Neural Networks
- Algebraic: Extended Boolean
- Set Theoretic

Classic Models

- boolean vector probabilistic

Structured Models

- Non-Overlapping Lists
- Proximal Nodes

User Task

- Retrieval: Adhoc Filtering
- Browsing: Flat Structure, Guided Hypertext
- Browsing: Inference Network, Belief Network
- Probabilistic: Generalized Vector, Lat. Semantic Index, Neural Networks
- Algebraic: Extended Boolean
- Set Theoretic
Probabilistic Model

- Asks the question “what is the probability that the user will see relevant information if they read this document”
  - $P(\text{rel} | D_i)$ – probability of relevance after reading $D_i$
  - How likely is the user to get relevance information from reading this document
  - High probability means more likely to get relevant info.

- Probability ranking principle
  - Rank documents based on decreasing probability of relevance to user
  - Calculate $P(\text{rel} | D_i)$ for each document and rank
Probabilistic Model

• Most probabilistic models are based on combining probabilities of relevance and non-relevance of individual terms
  – Probability that a term will appear in a relevant document
  – Probability that the term will not appear in a non-relevant document

• These probabilities are estimated based on counting term appearances in document descriptions
Example

• Assume we have a collection of 100 documents
  – $N=100$

• 20 of the documents contain the term *IBM*
  – $n_{IBM} = 20$

• Searcher has marked 10 documents as relevant
  – $R=10$

• Of these relevant documents 5 contain the term *IBM*
  – $r_{IBM} = 5$

• How important is the word *IBM* to the searcher?
Probability of Relevance

- From these four numbers we can estimate probability of IBM given relevance information
  - i.e. how important term IBM is to relevant documents

\[
\frac{r_{IBM}}{(R - r_{IBM})}
\]

- \( r_{IBM} \) is number of relevant documents containing IBM (5)
- \( R - r_{IBM} \) is number of relevant documents that do not contain IBM (5)
- Eq. (I) is
  - higher if most relevant documents contain IBM
  - lower if most relevant documents do not contain IBM
  - high value means IBM is important term to user in our example (5/5=1)
Probability of Non-relevance

- Also we can estimate probability of IBM given non-relevance information
  - i.e. how important term IBM is to non-relevant documents

\[
\frac{(n_{IBM} - r_{IBM})}{(N - n_{IBM}) - (R - r_{IBM})}
\]

- \(n_{IBM} - r_{IBM}\): number of non-relevant documents that contain term IBM
- \((N - n_{IBM}) - (R - r_{IBM})\): number of relevant docs that don’t contain IBM
- \# of docs that don’t contain IBM
- \# of non-relevant docs that contain IBM
- \# of relevant docs that don’t contain IBM

Eq(II)
- higher if more documents containing term IBM are non-relevant
- lower if more documents that do not contain IBM are non-relevant
- low value means IBM is important term to user in our example (15/75=0.2)
F4 Reweighting Formula

how important is *IBM* being present in relevant documents

$$\log \left( \frac{r_{IBM} + 0.5}{0.5 + R - r_{IBM}} \right)$$

how important is *IBM* being absent from non-relevant documents

$$\log \left( \frac{0.5 + n_{IBM} - r_{IBM}}{0.5 + N - n_{IBM} - R + r_{IBM}} \right)$$

In the example, weight of *IBM* is $\sim 5 (1/0.205)$
F4 Reweighting Formula

- F4 gives new weights to all terms in collection
  - High weights to important terms
  - Low weights to unimportant terms
  - Replaces *idf*, *tf*, or any other weights
  - Document score is based on sum of query terms in documents

\[
\text{Similarity}(D_j, q) = \sum_{i=1}^{n} F4_{qi}
\]
Probabilistic Model

• Can be also used to rank terms for addition to query
  – Rank terms in *relevant documents by term* reweighting formula
  – I.e. by how good the terms are at retrieving relevant documents
    • Add all terms
    • Add some, e.g. top 4
Probabilistic Model

• Advantages over vector-space
  – Good theoretical basis
  – Based on probability theory

• Disadvantages
  – Needs a starting point (i.e., information on the relevance of a set of documents – can use another IR model for that)
  – Models are often complicated
Extensions of the Vector-Space Model
Explicit/Latent Semantic Analysis

- BOW
  - American politics
    - Democrats, Republicans, abortion, taxes, homosexuality, guns, etc

- Explicit Semantic Analysis
  - Car

- Latent Semantic Analysis
  - Car
    - {car, truck, vehicle}, {tradeshows}, {engine}
Explicit/Latent Semantic Analysis

• Objective
  – Replace indexes that use sets of index terms/docs by indexes that use concepts.

• Approach
  – Map the term vector space into a lower dimensional space, using singular value decomposition.
  – Each dimension in the new space corresponds to an explicit/latent concept in the original data.
Deficiencies with Conventional Automatic Indexing

• Synonymy:
  – Various words and phrases refer to the same concept (lowers recall).

• Polysemy:
  – Individual words have more than one meaning (lowers precision)

• Independence:
  – No significance is given to two terms that frequently appear together

• Explicit/Latent semantic indexing addresses the first of these (synonymy), and the third (dependence)
Technical Memo Example: Titles

- c1 Human machine *interface* for Lab ABC *computer* applications
- c2 A *survey* of *user* opinion of computer *system response time*
- c3 The EPS *user interface* management *system*
- c4 *System* and *human system* engineering testing of EPS
- c5 Relation of *user*-perceived *response time* to error measurement
- m1 The generation of random, binary, unordered *trees*
- m2 The intersection *graph* of paths in *trees*
- m3 *Graph minors* IV: Widths of *trees* and well-quasi-ordering
- m4 *Graph minors*: A survey
# Technical Memo Example: Terms and Documents

<table>
<thead>
<tr>
<th>Terms</th>
<th>c1</th>
<th>c2</th>
<th>c3</th>
<th>c4</th>
<th>c5</th>
<th>m1</th>
<th>m2</th>
<th>m3</th>
<th>m4</th>
</tr>
</thead>
<tbody>
<tr>
<td>human</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>interface</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>computer</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>user</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>system</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>response</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>time</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>EPS</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>survey</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>trees</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>graph</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>minors</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>
Technical Memo Example: Query

• Query: Find documents relevant to "human computer interaction"

• Simple Term Matching:
  – Matches c1, c2, and c4
  – Misses c3 and c5
Latent Semantic Analysis: Mathematical Concepts

• Define X as the term-document matrix, with t rows (number of index terms) and d columns (number of documents).

• Singular Value Decomposition
  – For any matrix $X$, with t rows and d columns, there exist matrices $T_0$, $S_0$ and $D_0'$, such that:
    - $X = T_0 S_0 D_0'$
  – $T_0$ and $D_0$ are the matrices of left and right singular vectors
  – $S_0$ is the diagonal matrix of singular values
Dimensions of Matrices

\[ X = T_0 S_0 D_0' \]

- \( t \times d \) matrix
- \( t \times m \) matrix
- \( m \times m \) matrix
- \( m \times d \) matrix

\( m \) is the rank of \( X \leq \min(t, d) \)
Reduced Rank

• $S_0$ can be chosen so that the diagonal elements are positive and decreasing in magnitude. Keep the first $k$ and set the others to zero.

• Delete the zero rows and columns of $S_0$ and the corresponding rows and columns of $T_0$ and $D_0$. This gives:
  \[ X = TSD' \]

• Interpretation
  – If value of $k$ is selected well, expectation is that $X$ retains the semantic information, but eliminates noise from synonymy and recognizes dependence.
Dimensionality Reduction

\[ X = t \times d \]

\[ X \sim X = TSD' \]

\( k \) is the number of latent concepts (typically 300 ~ 500)
Recombination after Dimensionality Reduction

Calculate similarity between document and query using weights from the new matrix

<table>
<thead>
<tr>
<th></th>
<th>c1</th>
<th>c2</th>
<th>c3</th>
<th>c4</th>
<th>c5</th>
<th>m1</th>
<th>m2</th>
<th>m3</th>
<th>m4</th>
</tr>
</thead>
<tbody>
<tr>
<td>human</td>
<td>0.16</td>
<td>0.40</td>
<td>0.38</td>
<td>0.47</td>
<td>0.18</td>
<td>-0.05</td>
<td>-0.12</td>
<td>-0.16</td>
<td>-0.09</td>
</tr>
<tr>
<td>interface</td>
<td>0.14</td>
<td>0.37</td>
<td>0.33</td>
<td>0.40</td>
<td>0.16</td>
<td>-0.03</td>
<td>-0.07</td>
<td>-0.10</td>
<td>-0.04</td>
</tr>
<tr>
<td>computer</td>
<td>0.15</td>
<td>0.51</td>
<td>0.36</td>
<td>0.41</td>
<td>0.24</td>
<td>0.02</td>
<td>0.06</td>
<td>0.09</td>
<td>0.12</td>
</tr>
<tr>
<td>user</td>
<td>0.26</td>
<td>0.84</td>
<td>0.61</td>
<td>0.70</td>
<td>0.39</td>
<td>0.03</td>
<td>0.08</td>
<td>0.12</td>
<td>0.19</td>
</tr>
<tr>
<td>system</td>
<td>0.45</td>
<td>1.23</td>
<td>1.05</td>
<td>1.27</td>
<td>0.56</td>
<td>-0.07</td>
<td>-0.15</td>
<td>-0.21</td>
<td>-0.05</td>
</tr>
<tr>
<td>response</td>
<td>0.16</td>
<td>0.58</td>
<td>0.38</td>
<td>0.42</td>
<td>0.28</td>
<td>0.06</td>
<td>0.13</td>
<td>0.19</td>
<td>0.22</td>
</tr>
<tr>
<td>time</td>
<td>0.16</td>
<td>0.58</td>
<td>0.38</td>
<td>0.42</td>
<td>0.28</td>
<td>0.06</td>
<td>0.13</td>
<td>0.19</td>
<td>0.22</td>
</tr>
<tr>
<td>EPS</td>
<td>0.22</td>
<td>0.55</td>
<td>0.51</td>
<td>0.63</td>
<td>0.24</td>
<td>-0.07</td>
<td>-0.14</td>
<td>-0.20</td>
<td>-0.11</td>
</tr>
<tr>
<td>survey</td>
<td>0.10</td>
<td>0.53</td>
<td>0.23</td>
<td>0.21</td>
<td>0.27</td>
<td>0.14</td>
<td>0.31</td>
<td>0.44</td>
<td>0.42</td>
</tr>
<tr>
<td>trees</td>
<td>-0.06</td>
<td>0.23</td>
<td>-0.14</td>
<td>-0.27</td>
<td>0.14</td>
<td>0.24</td>
<td>0.55</td>
<td>0.77</td>
<td>0.66</td>
</tr>
<tr>
<td>graph</td>
<td>-0.06</td>
<td>0.34</td>
<td>-0.15</td>
<td>-0.30</td>
<td>0.20</td>
<td>0.31</td>
<td>0.69</td>
<td>0.98</td>
<td>0.85</td>
</tr>
<tr>
<td>minors</td>
<td>-0.04</td>
<td>0.25</td>
<td>-0.10</td>
<td>-0.21</td>
<td>0.15</td>
<td>0.22</td>
<td>0.50</td>
<td>0.71</td>
<td>0.62</td>
</tr>
</tbody>
</table>
Explicit Semantic Analysis

• Determine the extent to which each word is associated with every concept (article) of Wikipedia via term frequency or some other method.

• For a text, sum up the associated concept vectors for a composite text concept vector.

• Compare the texts using a standard cosine similarity or other vector similarity measure.
Explicit Semantic Analysis Example

- Text1: The dog caught the red ball.
- Text2: A Labrador played in the park.

<table>
<thead>
<tr>
<th></th>
<th>Glossary of cue sports terms</th>
<th>American Football Strategy</th>
<th>Baseball</th>
<th>Boston Red Sox</th>
</tr>
</thead>
<tbody>
<tr>
<td>Text1:</td>
<td>271</td>
<td>40</td>
<td>48</td>
<td>52</td>
</tr>
<tr>
<td>Text2:</td>
<td>10</td>
<td>17</td>
<td>10</td>
<td>7</td>
</tr>
</tbody>
</table>

- Can also be adapted to cross-language information retrieval
Extensions of the Boolean Model
Extended Boolean Model

• Boolean model is simple and elegant.
• But, no provision for a ranking
• Extend the Boolean model with the notions of partial matching and term weighting
• Combine characteristics of the vector-model with properties of Boolean algebra
• The extended Boolean model (introduced by Salton, Fox, and Wu, 1983) is based on a critique of a basic assumption in Boolean algebra
An Example

- Let,
  - $Q = K_x \land K_y$
  - Use weights associated with $K_x \land K_y$
  - In boolean model: $w_x = w_y = 1$; all other documents are irrelevant
  - In extended boolean model: use tf.idf or other weighting schemes
An Example
Extended Boolean Model: OR

- For query $Q=K_x$ or $K_y$, $(0,0)$ is the point we try to avoid. Thus, to rank documents we can use

$$sim(Q_{or}, d) = \sqrt{\frac{W_{Kx}^2 + W_{Ky}^2}{2}}$$

- Larger values are better
Extended Boolean Model: AND

- For query $Q=K_x$ and $K_y$, $(1,1)$ is the most desirable point.
- We rank documents with

$$sim(Q_{\text{and}}, d) = 1 - \sqrt{\frac{(1-w_{K_x})^2 + (1-w_{K_y})^2}{2}}$$

- Larger values are better
Fuzzy Set Model

• Queries and docs represented by sets of index terms: matching is *approximate* from the start

• This *vagueness* can be modeled using a fuzzy framework, as follows:
  – with each term is associated a *fuzzy set*
  – each doc has a degree of membership in this fuzzy set

• This interpretation provides the foundation for many models for IR based on fuzzy theory

• In here, the model proposed by Ogawa, Morita, and Kobayashi (1991)
Fuzzy Information Retrieval

• Fuzzy sets are modeled based on a thesaurus

• This thesaurus is built as follows:
  – Let vec(c) be a term-term correlation matrix
  – Let c(i,l) be a normalized correlation factor for (Ki,Kl):

\[
c(i,l) = \frac{n(i,l)}{ni + nl - n(i,l)}
\]

  - ni: number of documents that contain Ki
  - nl: number of documents that contain Kl
  - n(i,l): number of documents that contain both Ki and Kl

• We now have the notion of proximity among index terms
Exercise

• Assume the following counts are collected from a collection of documents:

  • orange: 100
  • banana: 300
  • computer: 500
  • orange-banana: 50
  • orange-computer: 10
  • banana-computer: 20

• Calculate the correlations for all three pairs of words
• Which two words have the highest correlation?
Fuzzy Set Theory

- Framework for representing classes whose boundaries are not well defined
- Key idea is to introduce the notion of a degree of membership associated with the elements of a set
- This degree of membership varies from 0 to 1 and allows modeling the notion of marginal membership
- Thus, membership is now a gradual notion, contrary to the notion enforced by classic Boolean logic
Fuzzy Set Theory

• Definition
  – A fuzzy subset $A$ of $U$ is characterized by a membership function
    \[
    \mu(A,u) : U \rightarrow [0,1]
    \]
    which associates with each element $u$ of $U$ a number $\mu(u)$ in the interval $[0,1]$.

• Definition
  – Let $A$ and $B$ be two fuzzy subsets of $U$. Also, let $\neg A$ be the complement of $A$. Then,
    - $\mu(\neg A, u) = 1 - \mu(A, u)$
    - $\mu(A \cup B, u) = \max(\mu(A, u), \mu(B, u))$
    - $\mu(A \cap B, u) = \min(\mu(A, u), \mu(B, u))$
Fuzzy Information Retrieval

• The correlation factor $c(i,l)$ can be used to define fuzzy set membership for a document $D_j$ as follows:
  $$
  \mu(i,j) = 1 - \prod_{K_l \in D_j} (1 - c(i,l))
  $$
  
  - $\mu(i,j)$: membership of doc $D_j$ in fuzzy subset associated with $K_i$

• The above expression computes an algebraic sum over all terms in the doc $D_j$

• A doc $D_j$ belongs to the fuzzy set for $K_i$, if its own terms are associated with $K_i$

• If doc $D_j$ contains a term $K_l$ which is closely related to $K_i$, we have
  - $c(i,l) \sim 1$
  - $\mu(i,j) \sim 1$
Fuzzy Information Retrieval

• Disjunctions and conjunctions

• Disjunctive set: algebraic sum
  \[ \mu(K_1 | K_2 | K_3, j) = 1 - \Pi (1 - \mu(K_i, j)) \]

• Conjunctive set: algebraic product
  \[ \mu(K_1 \land K_2 \land K_3, j) = \Pi (\mu(K_i, j)) \]
An Example

\[ Q = K_a \land (K_b \lor \neg K_c) \]

\[ D_a \quad c c_3 \quad D_b \]

\[ D_c \quad c c_1 \quad D_q = c c_1 + c c_2 + c c_3 \]

\[ (1,1,1) + (1,1,0) + (1,0,0) \]

\[ \text{vec}(c c_1) + \text{vec}(c c_2) + \text{vec}(c c_3) \]
An Example

• \( Q = K_a \land (K_b \lor \neg K_c) \)

• \( \text{Vec}(Q_{\text{dnf}}) = (1,1,1) + (1,1,0) + (1,0,0) \)
  \( = \text{vec}(cc_1) + \text{vec}(cc_2) + \text{vec}(cc_3) \)

• \( \mu(Q,D_j) = \mu(cc_1 | cc_2 | cc_3, j) \)
  \( = 1 - \prod (1 - \mu(cc_i, j)) \)
  \( = 1 - \left( (1 - \mu(K_{a,j}) \mu(K_{b,j}) \mu(K_{c,j})) \right) \times \left( (1 - \mu(K_{a,j}) \mu(K_{b,j}) (1 - \mu(K_{c,j}))) \right) \times \left( (1 - \mu(K_{a,j}) (1 - \mu(K_{b,j})) (1 - \mu(K_{c,j}))) \right) \)