Models for Retrieval and Browsing

- Classical IR Models

Berlin Chen 2004

Reference:
1. Modern Information Retrieval, chapter 2
Index Terms

• Meanings From Two Perspectives
  – In a restricted sense (keyword-based)
    • An index term is a (predefined) keyword (usually a noun) which has some semantic meaning of its own
  – In a more general sense (word-based)
    • An index term is simply any word which appears in the text of a document in the collection
    • Full-text
Index Terms (cont.)

• The semantics (main themes) of the documents and of the user information need should be expressed through sets of index terms
  
  – Semantics is lost when expressed through sets of words
  
  – Match between the documents and user queries is in the (imprecise?) space of index terms
Index Terms (cont.)

• Documents retrieved are frequently irrelevant
  – Since most users have no training in query formation, problem is even worst
    • Not familiar with the underlying IR process
    • E.g: frequent dissatisfaction of Web users
  – Issue of deciding document relevance, i.e. ranking, is critical for IR systems
Documents

Index Term Space

Information Need

Matching and Ranking

Retrieved Documents

1. Doc i
2. Doc j
3. Doc k

IR 2004 – Berlin Chen 5
Ranking Algorithms

• Also called the “information retrieval models”

• Ranking Algorithms
  – Predict which documents are relevant and which are not
  – Attempt to establish a simple ordering of the document retrieved
  – Documents at the top of the ordering are more likely to be relevant
  - The core of information retrieval systems
Ranking Algorithms (cont.)

• A ranking is based on fundamental **premisses** regarding the notion of relevance, such as:
  - Common sets of index terms
  - Sharing of weighted terms
  - Likelihood of relevance
  - Concept/semantic matching

\[
\text{literal-term matching} \quad P(Q|D) \quad \text{or} \quad P(Q,D)\]

• Distinct sets of **premisses** lead to a distinct **IR models**
Taxonomy of Classic IR Models

- References to the text content
  - Boolean Model (Set Theoretic)
    - Documents and queries are represented as sets of index terms
  - Vector (Space) Model (Algebraic)
    - Documents and queries are represented as vectors in a $t$-dimensional space
  - Probabilistic Model (Probabilistic)
    - Document and query are represented based on probability theory

Alternative modeling paradigms will also be extensively studied!
Taxonomy of Classic IR Models (cont.)

• References to the text structure
  – Non-overlapping list
    • A document divided in non-overlapping text regions and represented as multiple lists for chapter, sections, subsection, etc.
  – Proximal Nodes
    • Define a strict hierarchical index over the text which composed of chapters, sections, subsections, paragraphs or lines
Taxonomy of Classic IR Models (cont.)

User Task

- Retrieval: Adhoc Filtering
  - Classic Models
    - boolean
    - vector
    - probabilistic
  - Structured Models
    - Non-Overlapping Lists
    - Proximal Nodes
  - Browsing
    - Flat Structure
    - Guided Hypertext

- Browsing
  - Set Theoretic
  - Fuzzy Extended Boolean
  - Algebraic
    - Generalized Vector
    - Latent Semantic Indexing (LSI)
    - Neural Networks
  - Probabilistic
    - Inference Network
    - Belief Network
    - Hidden Markov Model
    - Language Model
    - Topical Mixture Model
    - Probabilistic LSI

probability-based
Taxonomy of Classic IR Models (cont.)

- Three-dimensional Representation

<table>
<thead>
<tr>
<th>USER TASK</th>
<th>Index Terms</th>
<th>Full Text</th>
<th>Full Text + Structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retrieval</td>
<td>Classic Set Theoretic</td>
<td>Classic Set Theoretic</td>
<td>Structured</td>
</tr>
<tr>
<td></td>
<td>Algebraic</td>
<td>Algebraic</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Probabilistic</td>
<td>Probabilistic</td>
<td></td>
</tr>
<tr>
<td>Browsing</td>
<td>Flat</td>
<td>Flat Hypertext</td>
<td>Structure Guided Hypertext</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- The same IR models can be used with distinct document logical views
Browsing the Text Content

- Flat/Structure Guided/Hypertext
- Example (Spoken Document Retrieval)
Browsing the Text Content (cont.)

• Example (Spoken Document Retrieval)
Browsing the Text Content (cont.)

- Example (Spoken Document Retrieval)
Retrieval: Ad Hoc

- **Ad hoc retrieval**
  - Documents remain relatively static while new queries are submitted to the system
    - The statistics for the entire document collection is obtainable
  - The most common form of user task
Retrieval: Filtering

• Filtering
  – Queries remain relatively static while new documents come into the system (and leave)
  • User Profiles: describe the users’ preferences
  – E.g. news wiring services in the stock market

Do not consider the relations of documents in the streams (only user task)
Filtering & Routing

• **Filtering** task indicates to the user which document might be interested to him
  • Determine which ones are really relevant is fully reserved to the user
    – Documents with a ranking about a given threshold is selected
  • But no ranking information of filtered documents is presented to user

• **Routing**: a variation of filtering
  • Ranking information of the filtered documents is presented to the user
  • The user can examine the Top N documents

• The vector model is preferred
Filtering: User Profile Construction

• Simplistic approach
  – Describe the profile through a set of keywords
  – The user provides the necessary keywords
  – User is not involved too much
  – Drawback: If user not familiar with the service (e.g. the vocabulary of upcoming documents)

• Elaborate approach
  – Collect information from user the about his preferences
  – Initial (primitive) profile description is adjusted by relevance feedback (from relevant/irrelevant information)
    • User intervention
  – Profile is continue changing
A Formal Characterization of IR Models

• The quadruple \(\langle D, Q, F, R(q_i, d_j)\rangle\) definition
  – \(D\): a set composed of logical views (or representations) for the documents in collection
  – \(Q\): a set composed of logical views (or representations) for the user information needs, i.e., “queries”
  – \(F\): a framework for modeling documents representations, queries, and their relationships and operations
  – \(R(q_i, d_j)\): a ranking function which associates a real number with \(q_i \in Q\) and \(d_j \in D\)
A Formal Characterization of IR Models (cont.)

• Classic Boolean model
  – Set of documents
  – Standard operations on sets

• Classic vector model
  – t-dimensional vector space
  – Standard linear algebra operations on vectors

• Classic probabilistic model
  – Sets (relevant/irrelevant document sets)
  – Standard probabilistic operations
    • Mainly the Bayes’ theorem
Classic IR Models - Basic Concepts

• Each document represented by a set of representative keywords or index terms

• An index term is a document word useful for remembering the document main themes

• Usually, index terms are **nouns** because nouns have meaning by themselves
  – Complements: adjectives, adverbs, and connectives

• However, search engines assume that all words are index terms (full text representation)
Classic IR Models - Basic Concepts (cont.)

• Not all terms are equally useful for representing the document contents
  – less frequent terms allow identifying a narrower set of documents
• The importance of the index terms is represented by weights associated to them
  – Let
    • $k_i$ be an index term
    • $d_j$ be a document
    • $w_{ij}$ be a weight associated with $(k_i, d_j)$
    • $\overrightarrow{d_j}=(w_{1,j}, w_{2,j}, ..., w_{t,j})$: an index term vector for the document $d_j$
    • $g_i(\overrightarrow{d_j})= w_{i,j}$

  – The weight $w_{ij}$ quantifies the importance of the index term for describing the document semantic contents
Classic IR Models - Basic Concepts (cont.)

• Correlation of index terms
  – E.g.: computer and network
  – Consideration of such correlation information does not consistently improve the final ranking result
    • Complex and slow operations

• Important Assumption/Simplification
  – Index term weights are mutually independent!
The Boolean Model

• Simple model based on set theory and Boolean algebra

• A query specified as boolean expressions with and, or, not operations
  – Precise semantics, neat formalism and simplicity
  – Terms are either present or absent, i.e., $w_{ij} \in \{0,1\}$

• A query can be expressed as a disjunctive normal form (DNF) composed of conjunctive components
  – $q_{dnf}$: the DNF for a query $q$
  – $q_{cc}$: conjunctive components (binary weighted vectors) of $q_{dnf}$
The Boolean Model (cont.)

- For instance, a query \([q = k_a \land (k_b \lor \neg k_c)]\) can be written as a DNF

\[
\vec{q}_{dnf} = (1,1,1) \lor (1,1,0) \lor (1,0,0)
\]

### Conjunctive Components

(binary weighted vectors)

\[
\begin{align*}
  k_a \land (k_b \lor \neg k_c) \\
  = (k_a \land k_b) \lor (k_a \land \neg k_c) \\
  = (k_a \land k_b \land k_c) \lor (k_a \land k_b \land \neg k_c) \lor (k_a \land \neg k_b \land \neg k_c) \\
  \Rightarrow q_{dnf} = (1,1,1) \lor (1,1,0) \lor (1,0,0)
\end{align*}
\]
The Boolean Model (cont.)

• The similarity of a document \( d_j \) to the query \( q \)

\[
sim(d_j, q) = \begin{cases} 
1: & \exists q_{cc} \mid (q_{cc} \in q_{dnf} \land (\forall k_i, g_i(d_j) = g_i(q_{cc}))) \\
0: & \text{otherwise}
\end{cases}
\]

– \( \sim(d_j, q) = 1 \) means that the document \( d_j \) is relevant to the query \( q \)

– Each document \( d_j \) can be represented as a conjunctive component

A document is represented as a conjunctive normal form
Advantages of the Boolean Model

• Simple queries are easy to understand relatively easy to implement
• Dominant language in commercial (bibliographic) systems until the WWW
Drawbacks of the Boolean Model

• Retrieval based on binary decision criteria with no notion of partial matching (no term weighting)
  – No notation of a partial match to the query condition
  – No ranking (ordering) of the documents is provided (absence of a grading scale)
  – Term frequency counts in documents not considered
  – Much more like a data retrieval model
Drawbacks of the Boolean Model (cont.)

• Information need has to be translated into a Boolean expression which most users find awkward
  – The Boolean queries formulated by the users are most often too simplistic (difficult to specify what is wanted)

• As a consequence, the Boolean model frequently returns either too few or too many documents in response to a user query
The Vector Model

• Also called Vector Space Model

• Some perspectives
  – Use of binary weights is too limiting
  – Non-binary weights provide consideration for partial matches
  – These term weights are used to compute a degree of similarity between a query and each document
  – Ranked set of documents provides for better matching for user information need

SMART system
Cornell U., 1968
The Vector Model (cont.)

- Definition:
  - $w_{ij} \geq 0$ whenever $k_i \in d_j$
  - $w_{iq} \geq 0$ whenever $k_i \in q$
  - document vector $\vec{d}_j = (w_{1j}, w_{2j}, ..., w_{ij})$
  - query vector $\vec{q} = (w_{1q}, w_{2q}, ..., w_{tq})$
  - To each term $k_i$ is associated a unitary (basis) vector $\vec{u}_i$
  - The unitary vectors $\vec{u}_i$ and $\vec{u}_s$ are assumed to be orthonormal (i.e., index terms are assumed to occur independently within the documents)

- The $t$ unitary vectors $\vec{u}_i$ form an orthonormal basis for a $t$-dimensional space
  - Queries and documents are represented as weighted vectors
The Vector Model (cont.)

- How to measure the degree of similarity
  - Distance, angle or projection?

\[ q = 0u_1 + 0u_2 + 3u_3 \]
\[ d_1 = 2u_1 + 4u_2 + 5u_3 \]
\[ d_2 = 3u_1 + 7u_2 + 7u_3 \]
The Vector Model (cont.)

- The similarity of a document $d_j$ to the query $q$

  $sim(d_j, q) = \cosine(\Theta)$
  
  $= \frac{\overrightarrow{d_j} \cdot \overrightarrow{q}}{|\overrightarrow{d_j}| \times |\overrightarrow{q}|}$
  
  $= \frac{\sum_{i=1}^{t} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} \times \sqrt{\sum_{j=1}^{t} w_{i,q}^2}}$

  Document length normalization

  The same for documents, can be discarded

  (if discarded, equivalent to the projection of the query on the document vector)

- Establish a threshold on $sim(d_j, q)$ and retrieve documents with a degree of similarity above the threshold

  Won’t affect the final ranking
The Vector Model (cont.)

- Degree of similarity $\rightarrow$ Relevance
  - Usually, $w_{ij} \geq 0$ & $w_{iq} \geq 0$
    - Cosine measure ranges between 0 and 1
  
  - $\text{sim}(d_j, q) \approx 1$ $\rightarrow$ highly relevant!
  
  - $\text{sim}(d_j, q) \approx 0$ $\rightarrow$ almost irrelevant!
The Vector Model (cont.)

• The role of index terms
  
  - Which index terms (features) better describe the relevant class
    
    • Intra-cluster similarity (\textit{tf}-factor)
    • Inter-cluster dissimilarity (\textit{idf}-factor)

  
  \textbf{Document collection}

\textbf{IR as a binary clustering (relevant/non-relevant) problem}

the ideal answer set
The Vector Model (cont.)

• How to compute the weights $w_{ij}$ and $w_{iq}$?

• A good weight must take into account two effects:
  – Quantification of **intra-document** contents (similarity)
    • $tf$ factor, the **term frequency** within a document
    • High term frequency is needed
  – Quantification of **inter-documents** separation (dissimilarity)
    • Low **document frequency** is preferred
    • $idf$ (**IDF**) factor, the **inverse document frequency**

  $w_{i,j} = tf_{i,j} \times idf_{i}$
The Vector Model (cont.)

• Let,
  – \( N \) be the total number of docs in the collection
  – \( n_i \) be the number of docs which contain \( k_i \)
  – \( freq_{i,j} \) raw frequency of \( k_i \) within \( d_j \)

• A normalized \( tf \) factor is given by

\[
    tf_{i,j} = \frac{freq_{i,j}}{\max_l freq_{l,j}}
\]

  – Where the maximum is computed over all terms which occur
    within the document \( d_j \)
  – \( tf_{i,j} \) will be in the range of 0 to 1
The Vector Model (cont.)

• The \( idf \) factor is computed as

\[
idf_i = \log \frac{N}{n_i}
\]

– the \( \log \) is used to make the values of \( tf \) and \( idf \) comparable. It can also be interpreted as the amount of information associated with the term \( k_i \)

• The best term-weighting schemes use weights which are given by

\[
w_{i,j} = tf_{i,j} \times \log \frac{N}{n_i}
\]

– the strategy is called a \( tf-idf \) weighting scheme
The Vector Model (cont.)

• For the query term weights, a suggestion is

\[ w_{i,q} = (0.5 + \frac{0.5 \text{freq}_{i,q}}{\max_l \text{freq}_{i,q}}) \times \log \frac{N}{n_i} \]

• The vector model with \textit{tf-idf} weights is a good ranking strategy with \textit{general} collections

• The vector model is usually as good as the known ranking alternatives. It is also simple and fast to compute
The Vector Model (cont.)

• Advantages
  - Term-weighting improves quality of the answer set
  - Partial matching allows retrieval of docs that approximate the query conditions
  - Cosine ranking formula sorts documents according to degree of similarity to the query

• Disadvantages
  - Assumes mutual independence of index terms
    • Not clear that this is bad though (??)
The Vector Model (cont.)

• Another *tf-idf* term weighting scheme
  – For query $q$
    \[
    w_{i,q} = \left(1 + \log(freq_{i,q})\right) \cdot \log\left(\frac{N + 1}{n_i}\right)
    \]
    Term Frequency \quad Inverse Document Frequency
  – For document $d_j$
    \[
    w_{i,j} = \left(1 + \log(freq_{i,j})\right)
    \]
The Vector Model (cont.)

- Example

\[
\begin{array}{cccccc}
   & k_1 & k_2 & k_3 & q \cdot d_j & q \cdot d_j / |d| \\
 d_1 & 1 & 0 & 1 & 2 & 2/\sqrt{2} \\
 d_2 & 1 & 0 & 0 & 1 & 1/\sqrt{1} \\
 d_3 & 0 & 1 & 1 & 2 & 2/\sqrt{2} \\
 d_4 & 1 & 0 & 0 & 1 & 1/\sqrt{1} \\
 d_5 & 1 & 1 & 1 & 3 & 3/\sqrt{3} \\
 d_6 & 1 & 1 & 0 & 2 & 2/\sqrt{2} \\
 d_7 & 0 & 1 & 0 & 1 & 1/\sqrt{1} \\
 q & 1 & 1 & 1 & & \\
\end{array}
\]
The Vector Model (cont.)

- Experimental Results on TDT Chinese collections
  - Mandarin Chinese broadcast news
  - Measured in mean Average Precision (mAP)

### Retrieval Results for the Vector Space Model

<table>
<thead>
<tr>
<th>Average Precision</th>
<th>Word-level</th>
<th>Syllable-level</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$S(N), N=1$</td>
<td>$S(N), N=1\sim2$</td>
</tr>
<tr>
<td>TDT-2 (Dev.)</td>
<td>TD</td>
<td>0.5548</td>
</tr>
<tr>
<td></td>
<td>SD</td>
<td>0.5122</td>
</tr>
<tr>
<td>TDT-3 (Eval.)</td>
<td>TD</td>
<td>0.6505</td>
</tr>
<tr>
<td></td>
<td>SD</td>
<td>0.6216</td>
</tr>
</tbody>
</table>

\[
R(q,d) = \sum_{j} w_j \cdot R_j(\tilde{q}_j, \tilde{d}_j),
\]

```plaintext
S(N)
N = 1\sim2
\]
The Probabilistic Model

Roberston & Sparck Jones 1976

• Known as the Binary Independence Retrieval (BIR) model
  – “Binary”: all weights of index terms are binary (0 or 1)
  – “Independence”: index terms are independent!

• Capture the IR problem using a probabilistic framework
  – Bayes’ decision rule
The Probabilistic Model (cont.)

- Retrieval is modeled as a classification process
  - Two classes for each query: the relevant or non-relevant documents

\[
P(R | \tilde{d}_j): \text{the prob. that the doc. } d_j \text{ is relevant to the query } q
\]

\[
P(\bar{R} | \tilde{d}_j): \text{the prob. that the doc. } d_j \text{ is non-relevant to the query } q
\]
The Probabilistic Model (cont.)

• Given a user query, there is an ideal answer set
  – The querying process as specification of the properties of this ideal answer set

• Problem: what are these properties?
  – Only the semantics of index terms can be used to characterize these properties

• **Guess at the beginning** what they could be
  – I.e., an initial guess for the primary probabilistic description of ideal answer set

• **Improve by iterations/iterations**
The Probabilistic Model (cont.)

- Improve the probabilistic description of the ideal answer set
The Probabilistic Model (cont.)

• Given a particular document $d_j$, calculate the probability of belonging to the relevant class, retrieve if greater than probability of belonging to non-relevant class

$$P(R \mid \tilde{d}_j) > P(\overline{R} \mid \tilde{d}_j)$$

Bayes' Decision Rule

• The similarity of a document $d_j$ to the query $q$

$$sim(d_j, q) = \frac{P(R \mid \tilde{d}_j)}{P(\overline{R} \mid \tilde{d}_j)}$$

Likelihood/Odds Ratio Test

Bayes' Theory

$$\approx \frac{P(\tilde{d}_j \mid R)P(R)}{P(\tilde{d}_j \mid \overline{R})P(\overline{R})} \geq \tau$$

if so, retrieved!
The Probabilistic Model (cont.)

• Explanation
  – \( P(R) \): the prob. that a doc randomly selected form the entire collection is relevant
  – \( P(d_j | R) \): the prob. that the doc \( d_j \) is relevant to the query \( q \) (selected from the relevant doc set \( R \))

• Further assume independence of index terms

\[
sim(d_j, q) \approx \frac{P(d_j | R)}{P(d_j | \bar{R})} \approx \prod_{g_i(\bar{d}_j)=1} P(k_i | R) \prod_{g_i(\bar{d}_j)=0} P(k_i | \bar{R})
\]

\[
P(k_i | R) : \text{prob. that } k_i \text{ is present in a doc randomly selected form the set } R
\]
\[
P(\bar{k}_i | R) : \text{prob. that } k_i \text{ is not present in a doc randomly selected form the set } R
\]
\[
P(k_i | R) + P(\bar{k}_i | R) = 1
\]
The Probabilistic Model (cont.)

- Further assume independence of index terms
  - Another representation
    \[
    \text{sim} \left( d_j, q \right) \approx \prod_{i=1}^{t} \left[ P \left( k_i \mid R \right)^{g_i(\tilde{d}_j)} P \left( \bar{k}_i \mid R \right)^{1-g_i(\tilde{d}_j)} \right] \]
  - Take logarithms
    \[
    \text{sim} \left( d_j, q \right) \approx \log \prod_{i=1}^{t} \left[ P \left( k_i \mid R \right)^{g_i(\tilde{d}_j)} P \left( \bar{k}_i \mid R \right)^{1-g_i(\tilde{d}_j)} \right] \]

\[
= \sum_{i=1}^{t} g_i(\tilde{d}_j) \log \frac{P \left( k_i \mid R \right) P \left( \bar{k}_i \mid R \right)}{P \left( k_i \mid R \right) P \left( \bar{k}_i \mid R \right)} + \sum_{i=1}^{t} \log \frac{P \left( k_i \mid R \right)}{P \left( \bar{k}_i \mid R \right)}
\]

The same for all documents!
The Probabilistic Model (cont.)

- Further assume independence of index terms
  - Use term weighting $w_{i,q} \times w_{i,j}$ to replace $g_i(d_j)$

$$
sim(d_j, q) \approx \sum_{i=1}^{t} g_i(d_j) \left[ \log \frac{P(k_i | R)}{1 - P(k_i | R)} + \log \frac{1 - P(k_i | \bar{R})}{P(k_i | \bar{R})} \right]
$$

$$
\approx \sum_{i=1}^{t} w_{i,q} \times w_{i,j} \times \left[ \log \frac{P(k_i | R)}{1 - P(k_i | R)} + \log \frac{1 - P(k_i | \bar{R})}{P(k_i | \bar{R})} \right]
$$

Binary weights (0 or 1) are used here

$R$ is not known at the beginning

How to compute $P(k_i | R)$ and $P(k_i | \bar{R})$
The Probabilistic Model (cont.)

• Initial Assumptions
  
  – \( P(k_i \mid R) = 0.5 \) : is constant for all indexing terms
  
  – \( P(k_i \mid \bar{R}) = \frac{n_i}{N} \) : approx. by distribution of index terms among all doc in the collection, i.e. the document frequency of indexing term \( k_i \) (Suppose that \(|\bar{R}| >> |R|, N \approx |\bar{R}|\))
    
    (\( n_i \): no. of doc that contain \( k_i \). \( N \): the total doc no.)

• Re-estimate the probability distributions
  
  – Use the initially retrieved and ranked Top \( V \) documents

\[
P(k_i \mid R) = \frac{V_i}{V}
\]

\[
P(k_i \mid \bar{R}) = \frac{n_i - V_i}{N - V}
\]

\( V_i \): the no. of documents in \( V \) that contain \( k_i \).
The Probabilistic Model (cont.)

- Handle the problem of “zero” probabilities
  - Add constants as the adjust constant

\[
P(k_i \mid R) = \frac{V_i + 0.5}{V + 1}
\]

\[
P(k_i \mid \overline{R}) = \frac{n_i - V_i + 0.5}{N - V + 1}
\]

- Or use the information of document frequency

\[
P(k_i \mid R) = \frac{V_i + \frac{n_i}{N}}{V + 1}
\]

\[
P(k_i \mid \overline{R}) = \frac{n_i - V_i + \frac{n_i}{N}}{N - V + 1}
\]
The Probabilistic Model (cont.)

• Advantages
  – Documents are ranked in decreasing order of probability of relevance

• Disadvantages
  – Need to guess initial estimates for $P(k_i | R)$
  – All weights are binary: the method does not take into account $tf$ and $idf$ factors
  – Independence assumption of index terms
Brief Comparisons of Classic Models

• Boolean model does not provide for partial matches and is considered to be the weakest classic model

• Salton and Buckley did a series of experiments that indicated that, in general, the vector model outperforms the probabilistic model with general collections