Models for Retrieval and Browsing

- Classical IR Models

Berlin Chen 2003

Reference:

1. Modern Information Retrieval, chapter 2

Index Terms

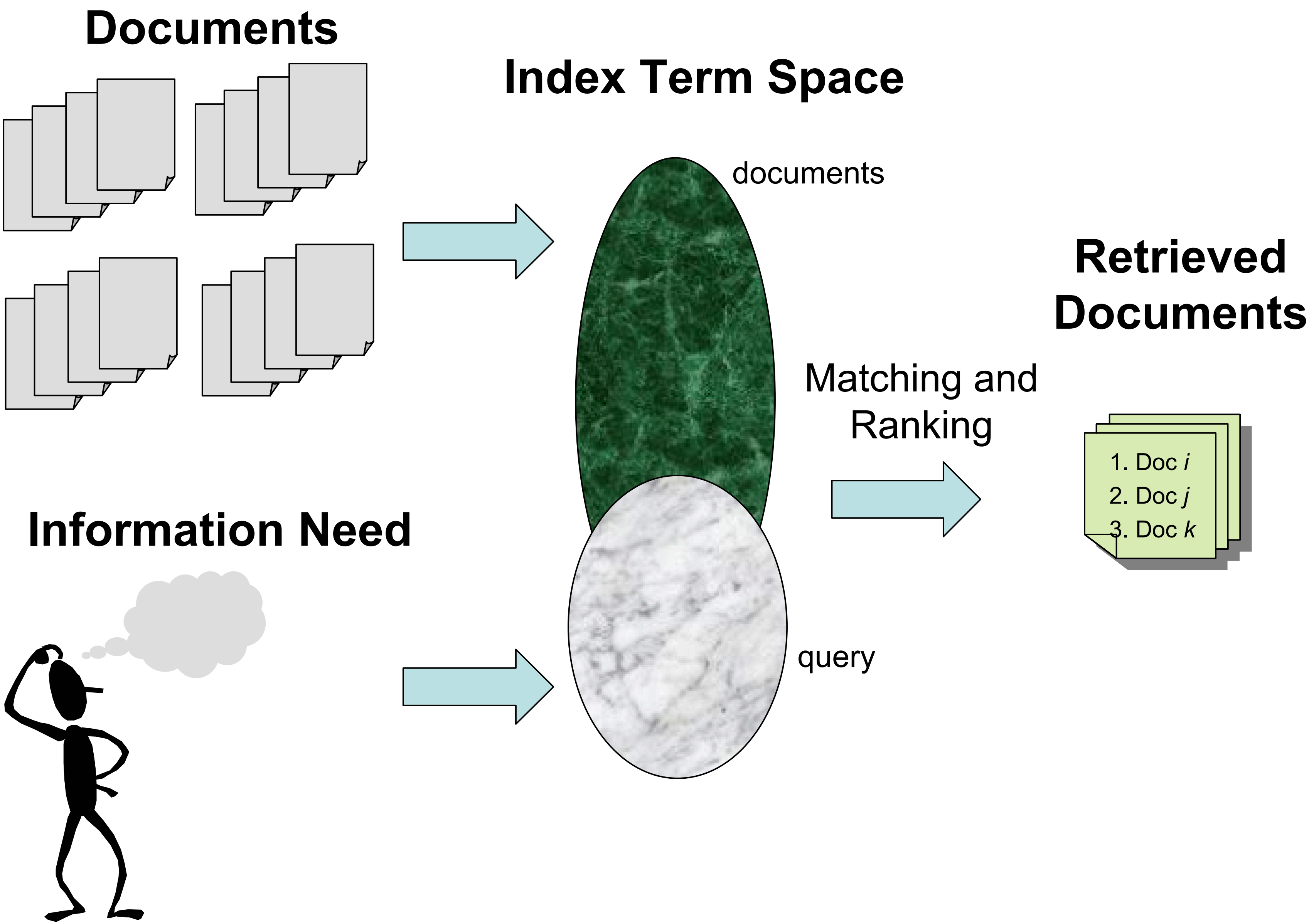
- Meanings From Two Perspectives
  - In a *restricted* sense (keyword-based)
    - An index term is a (predefined) keyword (usually a noun) which has some semantic meaning of its own
  - In a *more general* sense (word-based)
    - An index term is simply any word which appears in the text of a document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

Index Terms

- 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

- Documents retrieved are flrequently irrelevant
  - Since most users have no training in query formation, problem is even worst
  - E.g: frequent dissatisfaction of Web users
  - Issue of deciding document relevance, i.e. *ranking*, is critical for IR systems


Documents
Index Term Space
documents
Retrieved Documents
Matching and Ranking
1. Doc i
2. Doc j
3. Doc k
Information Need
query

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

- 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

- Distinct sets of premisses lead to a distinct IR models

Taxonomy of Classic IR Models

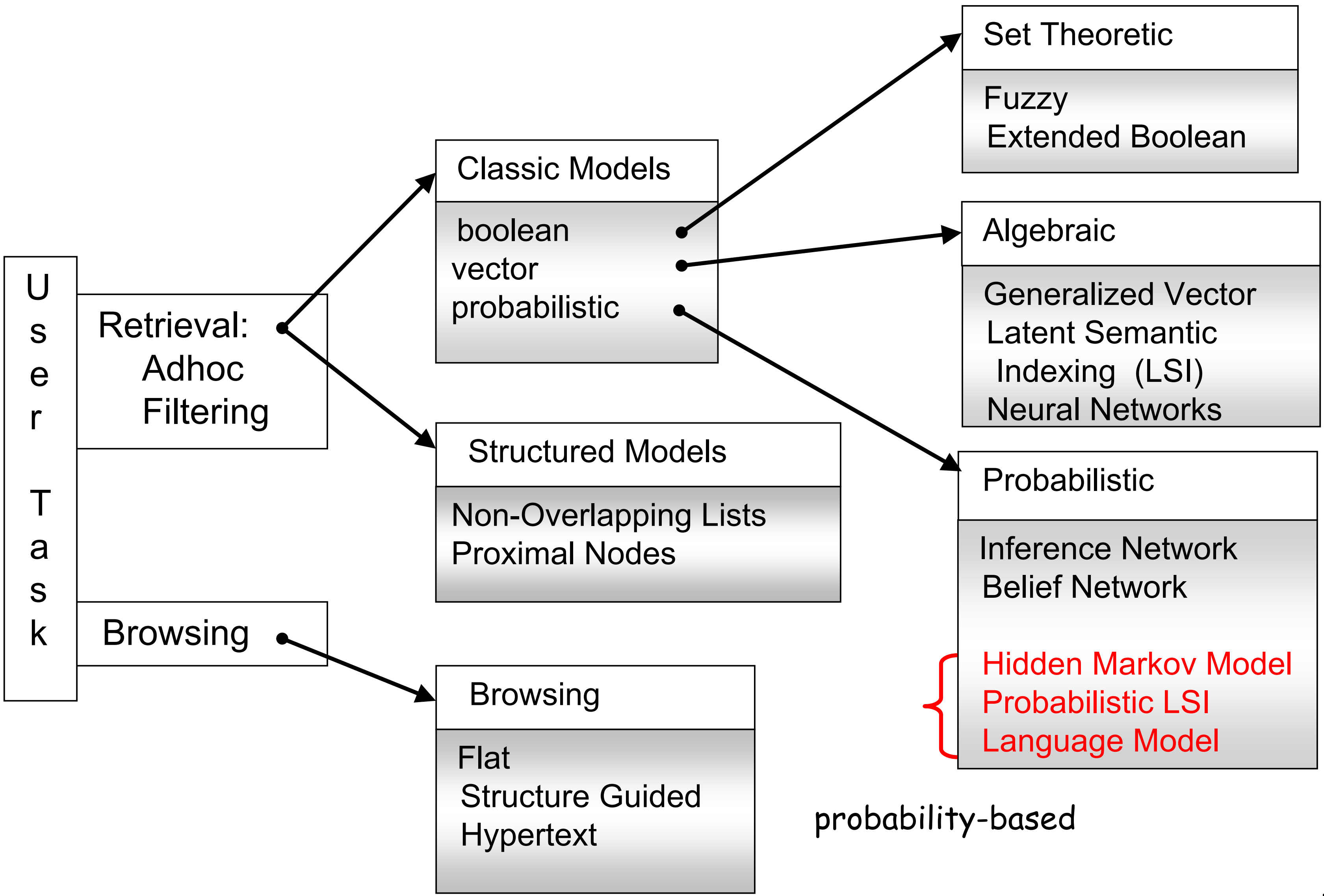
- References to the text content
  - Boolean Model (*Set Theoretic*)
    - Documents and queries are represented as sets of index terms
  - Vector (Space) Model (*Algebraic*)
    - Documents and queries are represented as vectors in a *t*-dimensional space
  - Probabilistic Model (*Probabilistic*)
 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

Alternative modeling paradigms will also be extensively studied !

Taxonomy of Classic IR Models

- 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



Taxonomy of Classic IR Models

LOGICAL VIEW OF DOCUMENTS

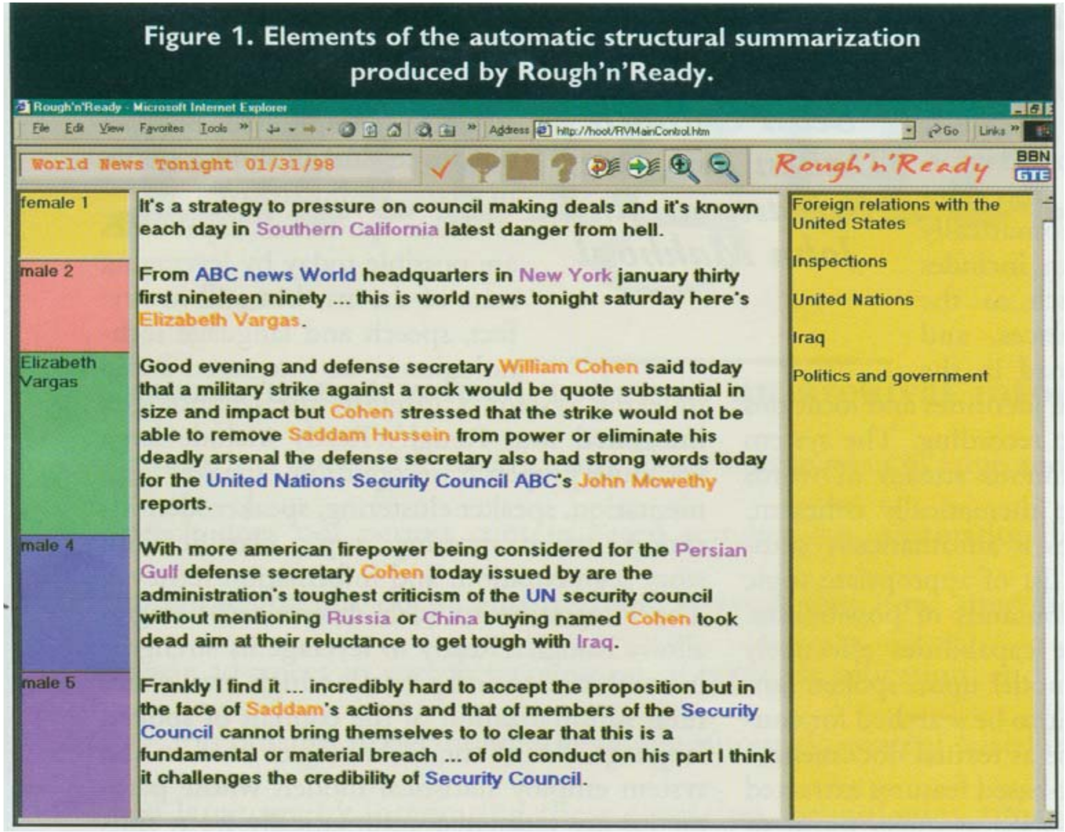
USER TASK

	Index Terms	**Full Text**	**Full Text + Structure**
Retrieval	Classic Set Theoretic Algebraic Probabilistic	Classic Set Theoretic Algebraic Probabilistic	Structured
Browsing	Flat	Flat Hypertext	Structure Guided Hypertext

- The same IR models can be used with distinct document logical views

Browsing the Text Content

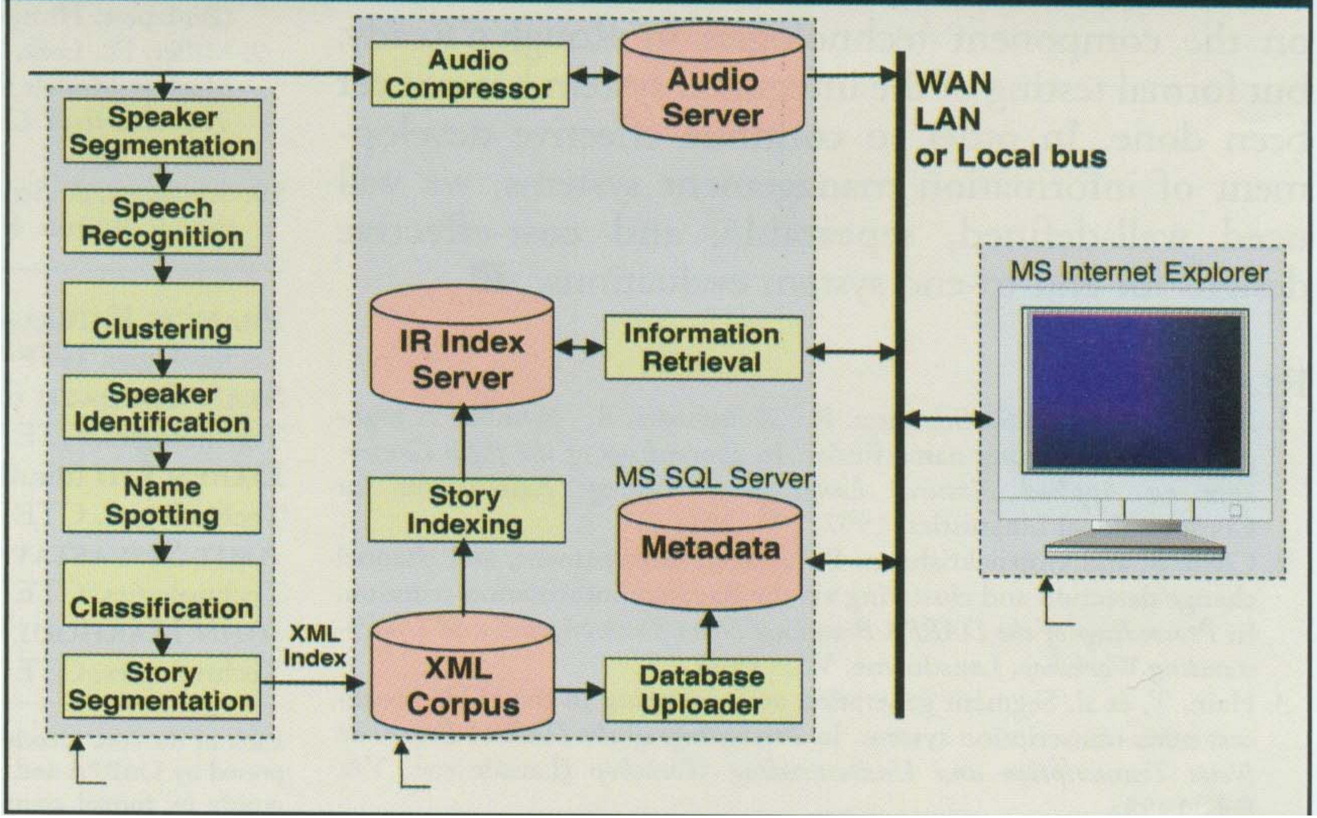
- Flat/Structure Guided/Hypertext
- Example (Spoken Document Retrieval)



Browsing the Text Content

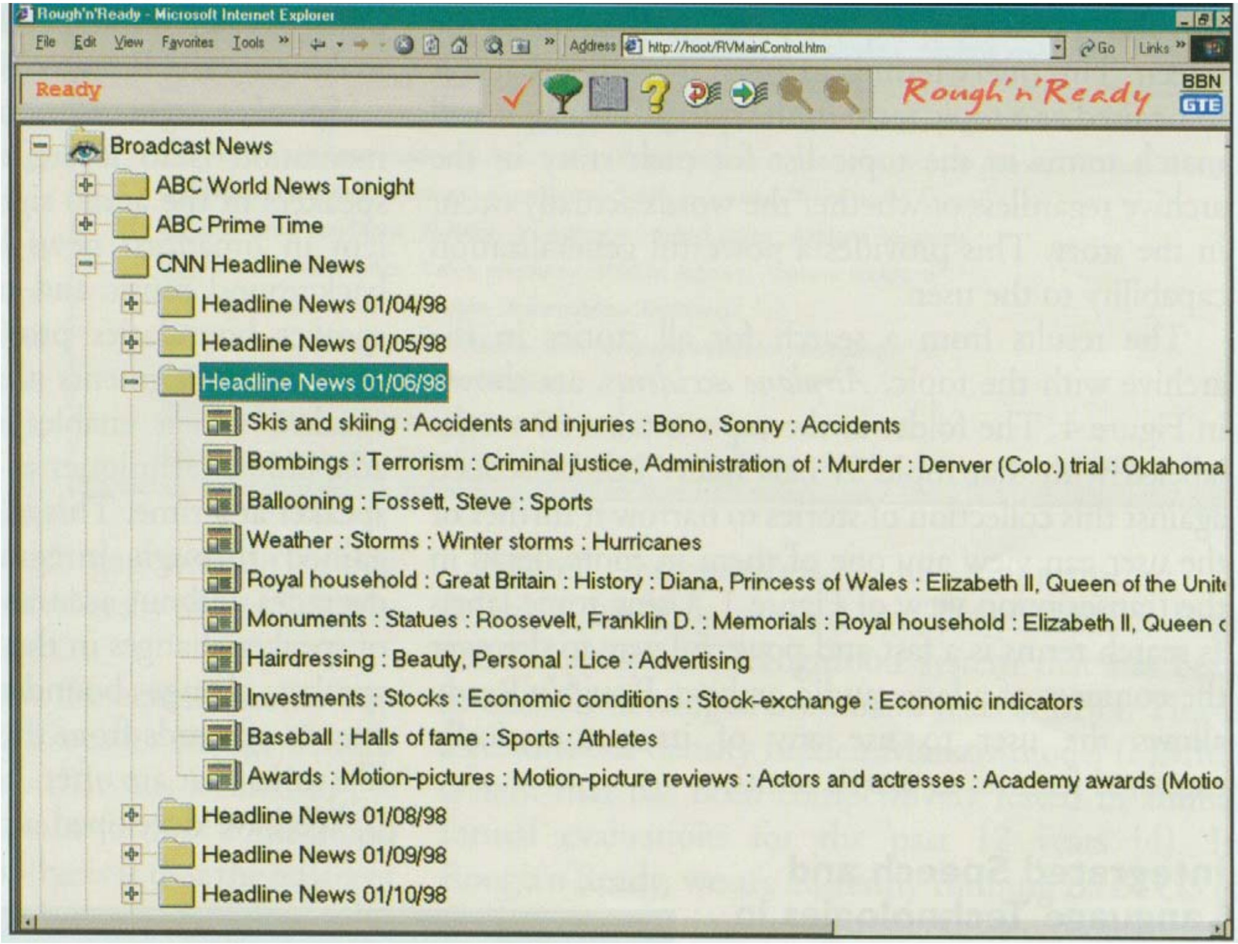
- Example (Spoken Document Retrieval)

Figure 5. Distinguished architecture of the Rough'n'Ready audio indexing system.



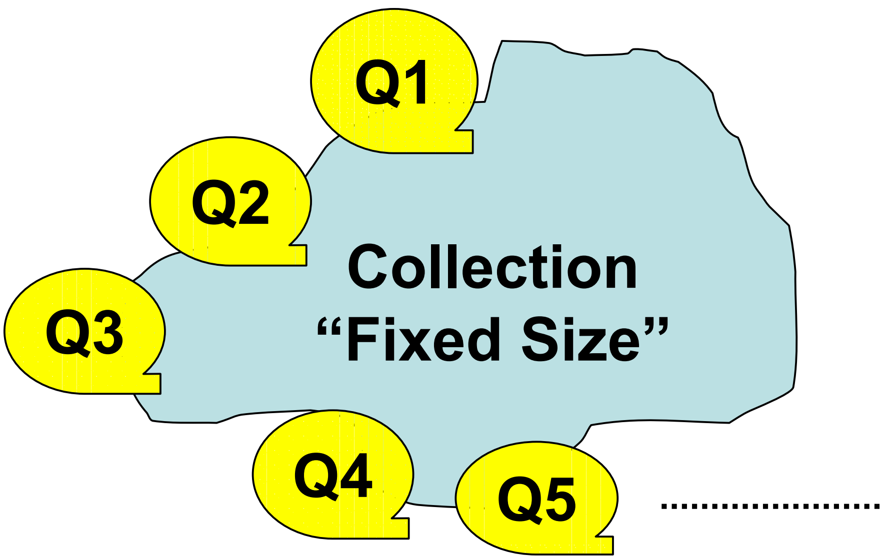
Browsing the Text Content

- Example (Spoken Document Retrieval)



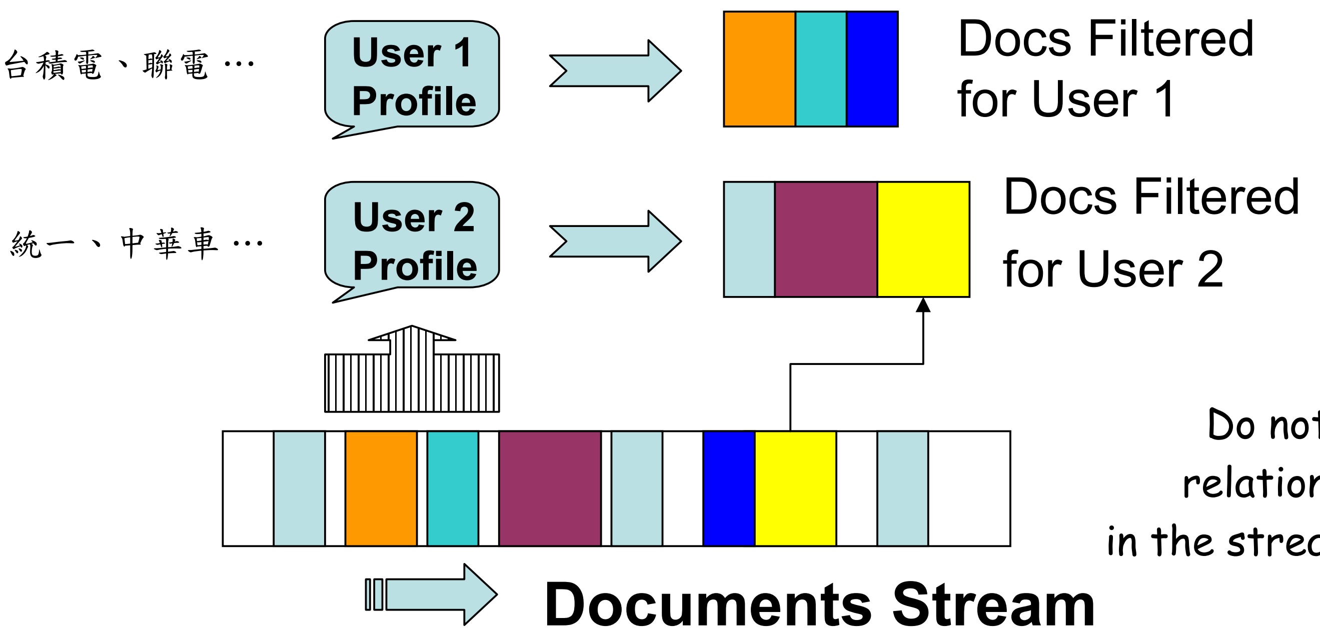
Retrieval: Ad Hoc and Filtering

- *Ad hoc* retrieval
  - Documents remain relatively static while new queries are submitted the system
  - The most common form of user task



Retrieval: Ad Hoc and 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)
  - Profile is continue changing

A Formal Characterization of IR Models

- The quadruple /**D**, **Q**, *F*, $R(q_i, d_j)$/ 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 associations a real number with $q_i \in \mathbf{Q}$ and $d_j \in \mathbf{D}$

A Formal Characterization of IR Models

- 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, amd connectives
- However, search engines assume that all words are index terms (full text representation)

Classic IR Models - Basic Concepts

- 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)
    - $\vec{d}_j = (w_{1,j}, w_{2,j}, \ldots, w_{t,j})$: an index term vector for the document d_j
    - $g_i(\vec{d}_j) = w_{i,j}$
  - The weight w_{ij} quantifies the importance of the index term for describing the document semantic contents

Classic IR Models - Basic Concepts

- 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
- A query specified as boolean expressions with and, or, not operations
  - Precise semantics and neat formalism
  - 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
  - $\vec{q}_{dnf}$: the DNF for a query q
  - $\vec{q}_{cc}$: conjunctive components (binary weighted vectors) of $\vec{q}_{dnf}$

The Boolean Model

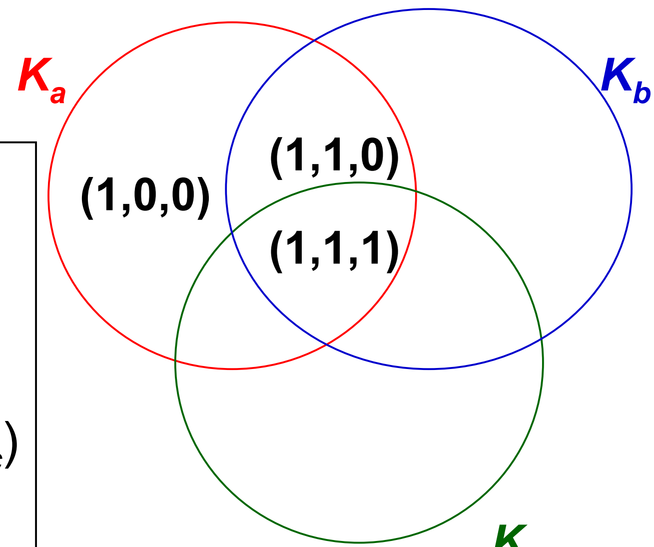
- For intance, a query [$q = k_a \wedge (k_b \vee \neg k_c)$] can be written as a DNF

$$\vec{q}_{dnf} = (1,1,1) \vee (1,1,0) \vee (1,0,0)$$

a canonical representation

conjunctive components

$k_a \wedge (k_b \vee \neg k_c)$
$= (k_a \wedge k_b) \vee (k_a \wedge \neg k_c)$
$= (k_a \wedge k_b \wedge k_c) \vee (k_a \wedge k_b \wedge \neg k_c)$
$\vee (k_a \wedge k_b \wedge \neg k_c) \vee (k_a \wedge \neg k_b \wedge \neg k_c)$
$= (k_a \wedge k_b \wedge k_c) \vee (k_a \wedge k_b \wedge \neg k_c) \vee (k_a \wedge \neg k_b \wedge \neg k_c)$
$\Rightarrow \vec{q}_{dnf} = (1,1,1) \vee (1,1,0) \vee (1,0,0)$



The Boolean Model

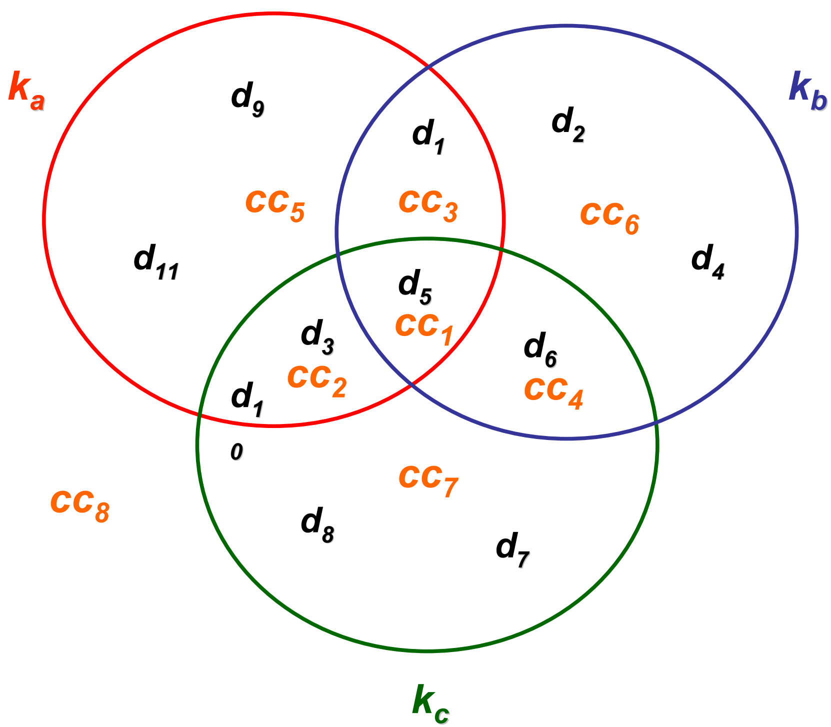
- The similarity of a document d_j to the query q

$$sim(d_j,q)=\begin{cases} 1: \text{if } \exists \vec{q}_{cc} \mid (\vec{q}_{cc} \in \vec{q}_{dnf} \wedge (\forall k_i, g_i(\vec{d}_j)=g_i(\vec{q}_{cc})) \\ 0: \text{otherwise} \end{cases}$$

  - $sim(d_j,q)=1$ means that the document d_j is relevant to the query q
  - Each document d_j can be represented as a conjunctive component

Advantages of the Boolean Model

- Simple queries are easy to understand relatively easy to implement
- Dominant language in commercial systems until the WWW



$cc_1 = k_a \, k_b \, k_c$
$cc_2 = k_a \, \overline{k_b} \, k_c$
$cc_3 = k_a \, k_b \, \overline{k_c}$
$cc_4 = \overline{k_a} \, k_b \, k_c$
$cc_5 = k_a \, \overline{k_b} \, \overline{k_c}$
$cc_6 = \overline{k_a} \, k_b \, \overline{k_c}$
$cc_7 = \overline{k_a} \, \overline{k_b} \, k_c$
$cc_8 = \overline{k_a} \, \overline{k_b} \, \overline{k_c}$

Drawbacks of the Boolean Model

- Retrieval based on *binary decision criteria* with no notion of partial matching (*no term weighting*)
  - *No ranking* (ordering) of the documents is provided (absence of a *grading* scale)
  - Term freqency counts in documents not considered

- 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)

Drawbacks of the Boolean Model

- 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*

SMART system
Cornell U., 1968

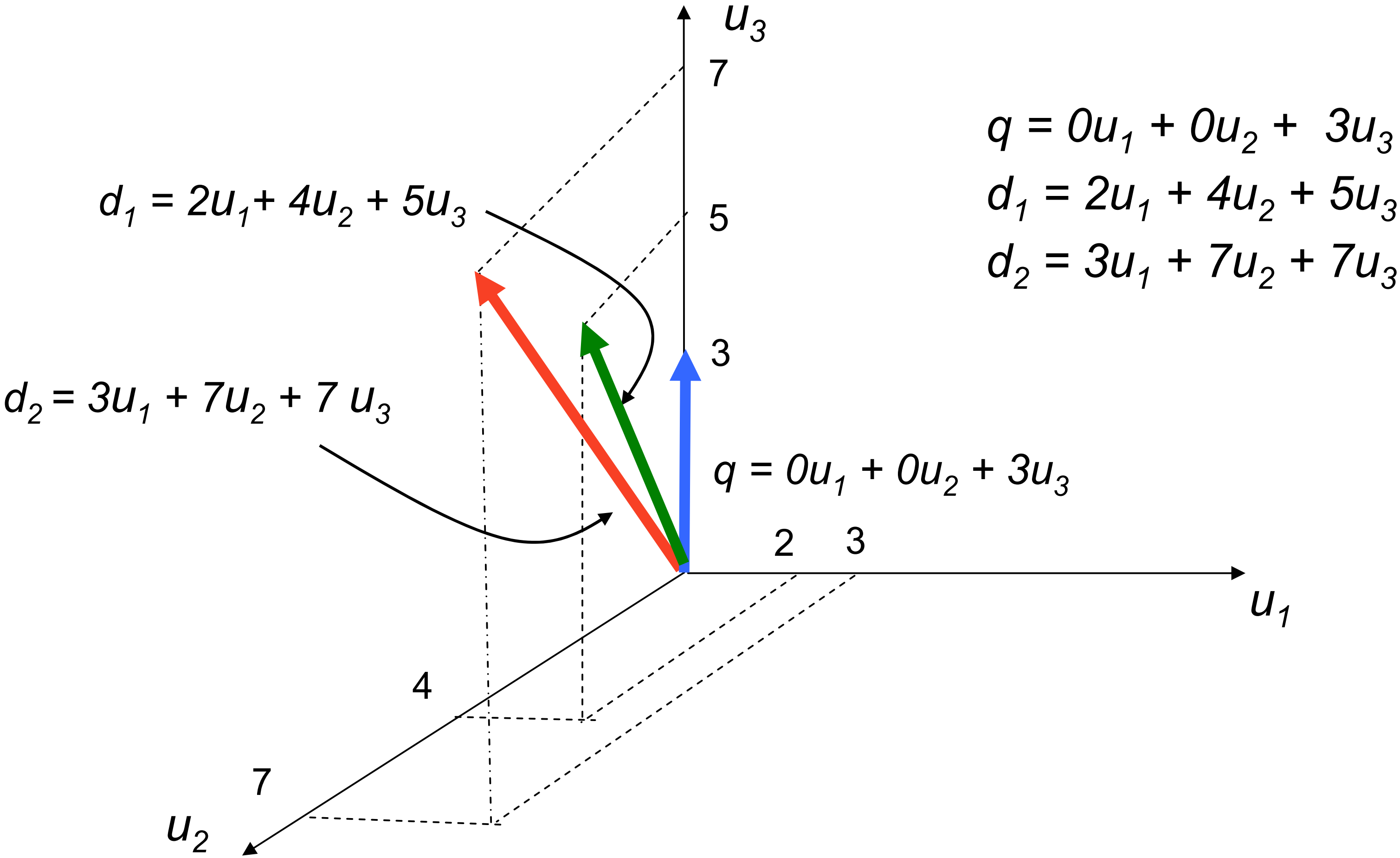
- Some perspectives
  - Use of *binary weights* is too limiting
  - *Non-binary weights* provide consideration for partial matches
  - These term weights are used to compute a *degree of similarity* between a query and each document
  - Ranked set of documents provides for better matching for user information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

The Vector Model

- Definition:
  - w_{ij} > =0 whenever $k_i \in d_j$
  - w_{iq} >= 0 whenever $k_i \in q$
  - document vector $\vec{d_j} = (w_{1j}, w_{2j}, ..., w_{tj})$
  - query vector $\vec{q} = (w_{1q}, w_{2q}, ..., w_{tq})$
  - To each term k_i is associated a unitary 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

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



The Vector Model

- The similarity of a document d_j to the query q

$$sim(d_j, q)$$

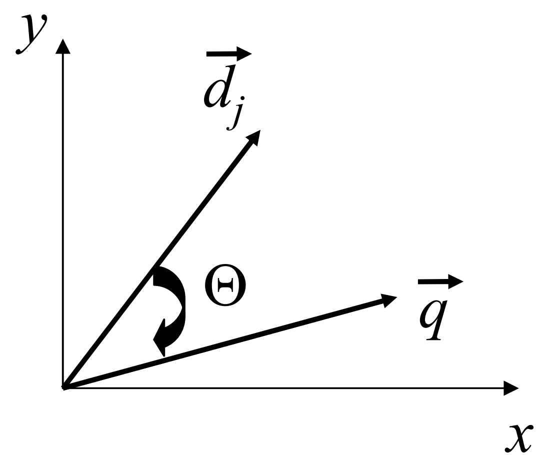
$$= \text{cosine}(\Theta)$$

$$= \frac{\vec{d}_j \bullet \vec{q}}{|\vec{d}_j| \times |\vec{q}|}$$

$$= \frac{\sum_{i=1}^{t} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} \times \sqrt{\sum_{j=1}^{t} w_{i,q}^2}}$$

Document length normalization

The same for documents, can be discarded



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

The Vector Model

- 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 (dissi-milarity)
    - Low **document frequency** is preferred
    - *idf* (*IDF*) factor, the **inverse document frequency**
  - $w_{i,j} = tf_{i,j} * idf_i$

The Vector Model

- 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

The Vector Model

Sparck Jones

- The *idf* factor is computed as

$$idf_i = \log \frac{N}{n_i}$$

Document frequency of term k_i = $\frac{n_i}{N}$

  - 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 give 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

- For the query term weights, a suggestion is

$$w_{i,q} = (0.5 + \frac{0.5\, freq_{i,q}}{\max_l freq_{i,q}}) \times \log \frac{N}{n_i}$$

Salton & Buckley

- The vector model with *tf-idf* weights is a good ranking strategy with *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

- 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

- Another *tf-idf* term weighting scheme
  - For query *q*

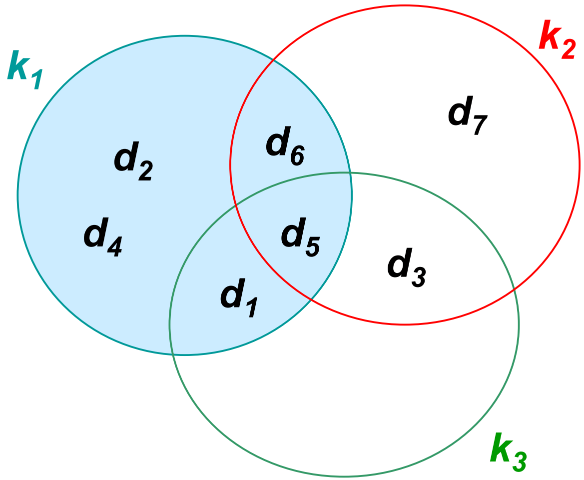
$$w_{i,q} = \underbrace{(1+\log(freq_{i,q}))}_{\text{Term Frequency}} \cdot \underbrace{\log((N+1)/n_i)}_{\text{Inverse Document Frequency}}$$

  - For document d_j

$$w_{i,j} = (1+\log(freq_{i,j}))$$

The Vector Model

- Example



	k_1	k_2	k_3	$q \bullet d_j$	$q \bullet d_j/\vert d\vert$
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		

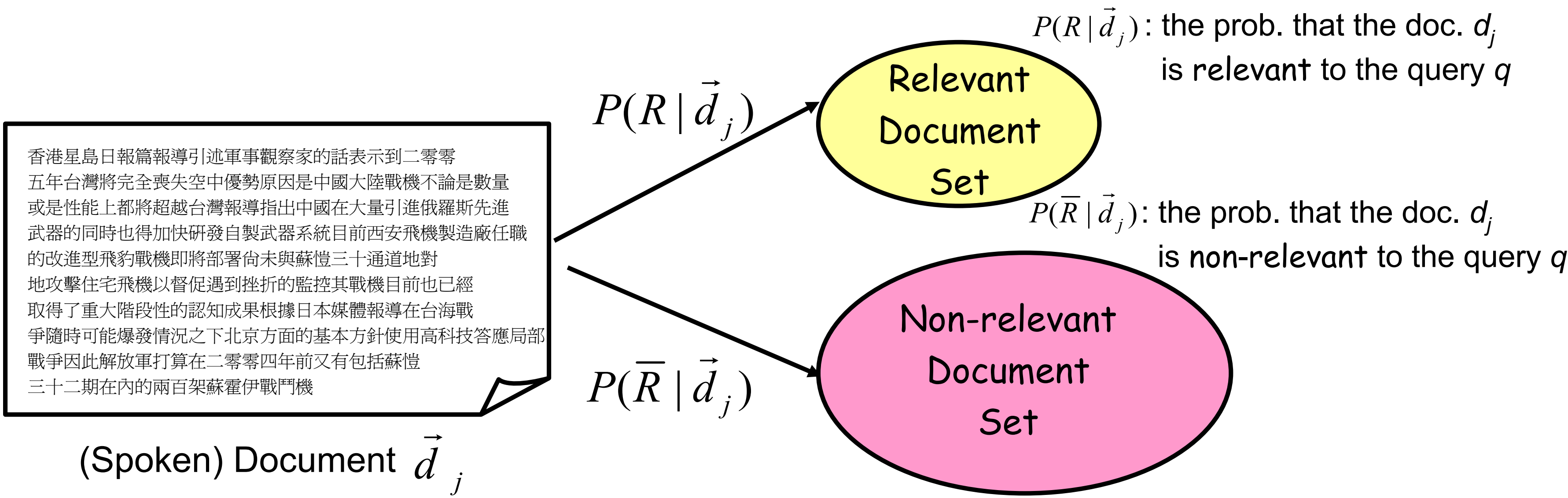
The Probabilistic Model

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

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

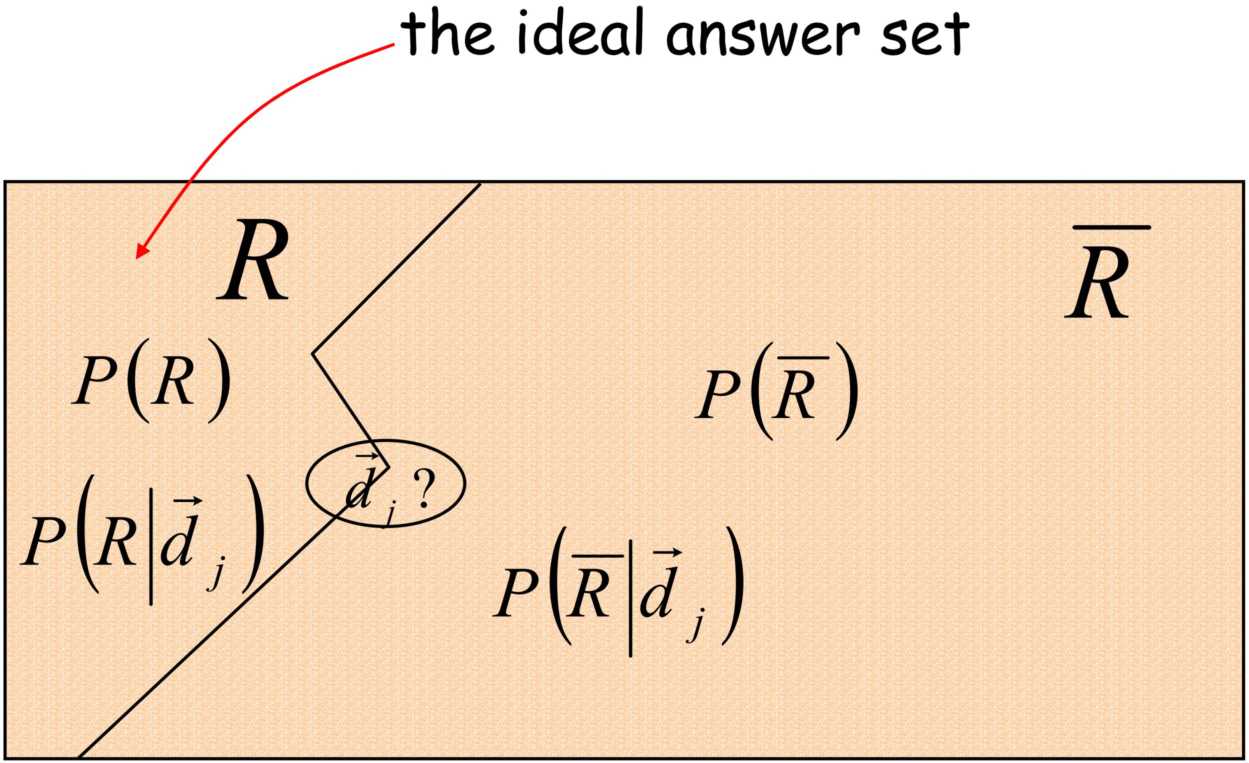


The Probabilistic Model

- 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 primimary probabilistis description of ideal answer set
- **Improve by iterations/interations**

The Probabilistic Model

- Improve the probabilistic description of the ideal answer set



Document Collection

The Probabilistic Model

- 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

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

Bayes' Decision Rule

- The similarity of a document d_j to the query q

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

Likelihood/Odds Ratio Test

Bayes' Theory

The same for all documents

$$= \frac{P(\vec{d}_j \mid R) P(R)}{P(\vec{d}_j \mid \overline{R}) P(\overline{R})} \approx \frac{P(\vec{d}_j \mid R)}{P(\vec{d}_j \mid \overline{R})}$$

$\geq \tau$?

if so, retrieved !

The Probabilistic Model

- Explanation
  - $P(R)$: the prob. that a doc randomly selected form the entire collection is relevant
  - $P(\vec{d}_j \mid R)$: 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

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

$P(k_i \mid R)$: prob. that k_i is present in a doc randomly selected form the set R

$P(\overline{k}_i \mid R)$: prob. that k_i is not present in a doc randomly selected form the set R

$P(k_i \mid R) + P(\overline{k}_i \mid R) = 1$

$$\approx \frac{\left[\prod_{g_i(\vec{d}_j)=1} P(k_i \mid R)\right]\left[\prod_{g_i(\vec{d}_j)=0} P(\overline{k}_i \mid R)\right]}{\left[\prod_{g_i(\vec{d}_j)=1} P(k_i \mid \overline{R})\right]\left[\prod_{g_i(\vec{d}_j)=0} P(\overline{k}_i \mid \overline{R})\right]}$$

The Probabilistic Model

- Further assume *independence* of index terms
  - Another representation

$$sim(d_j, q) \approx \frac{\prod_{i=1}^{t}\left[P(k_i \mid R)^{g_i(\vec{d}_j)} P(\bar{k}_i \mid R)^{1-g_i(\vec{d}_j)}\right]}{\prod_{i=1}^{t}\left[P(k_i \mid \bar{R})^{g_i(\vec{d}_j)} P(\bar{k}_i \mid \bar{R})^{1-g_i(\vec{d}_j)}\right]}$$

  - Take logarithms

$$sim(d_j, q) \approx \log \frac{\prod_{i=1}^{t}\left[P(k_i \mid R)^{g_i(\vec{d}_j)} P(\bar{k}_i \mid R)^{1-g_i(\vec{d}_j)}\right]}{\prod_{i=1}^{t}\left[P(k_i \mid \bar{R})^{g_i(\vec{d}_j)} \left(P(\bar{k}_i \mid \bar{R})\right)^{1-g_i(\vec{d}_j)}\right]}$$

The same for all documents!

$$P(k_i \mid R) + P(\bar{k}_i \mid R) = 1$$
$$P(k_i \mid \bar{R}) + P(\bar{k}_i \mid \bar{R}) = 1$$

$$= \sum_{i=1}^{t} g_i(\vec{d}_j) \log \frac{P(k_i \mid R) P(\bar{k}_i \mid \bar{R})}{P(k_i \mid \bar{R}) P(\bar{k}_i \mid R)} + \sum_{i=1}^{t} \log \frac{P(\bar{k}_i \mid R))}{P(\bar{k}_i \mid \bar{R})}$$

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

The Probabilistic Model

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

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

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

Binary weights (0 or 1) are used here

R is not known at the beginning

⇨ How to compute $P(k_i \mid R)$ and $P(k_i \mid \bar{R})$

The Probabilistic Model

- Initial Assumptions
  - $P(k_i \mid R) = 0.5$:is constant for all indexing terms
  - $P(k_i \mid \overline{R}) = \frac{n_i}{N}$:approx. by distribution of index terms among all doc in the collection, i.e. the **document frequency** of indexing term k_i (Suppose that $|\overline{R}| >> |R|$, $N \approx |\overline{R}|$)

  (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

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

$$P(k_i \mid \overline{R}) = \frac{n_i - V_i}{N - V}$$

> V_i : the no. of documents in V that contain k_i

The Probabilistic Model

- 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

- Advantages
  - Documents are ranked in decreasing order of probability of relevance
- Disadvantages
  - Need to guess initial estimates for $P(k_i \mid R)$
  - All weights are binary: the method does not take into account *tf* and *idf* factors
  - Independence assumption of index terms

Brief Comparison 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*