Modeling in Information Retrieval

- Classical Models

Berlin Chen
Department of Computer Science & Information Engineering
National Taiwan Normal University

References:
1. *Modern Information Retrieval*, Chapter 3 & Teaching material
2. *Language Modeling for Information Retrieval*, Chapter 3

Modeling

- Produce a ranking function that assigns scores to documents with regard to a given query
  - Ranking is likely the most important process of an IR system

- This process consists of two main tasks
  - The conception of a logical framework for representing documents and queries
    - Sets, vectors, probability distributions, etc.
  - The definition of a ranking function (or retrieval model) that computes a rank (e.g., a real number) for each document in response to a given query

Index Terms

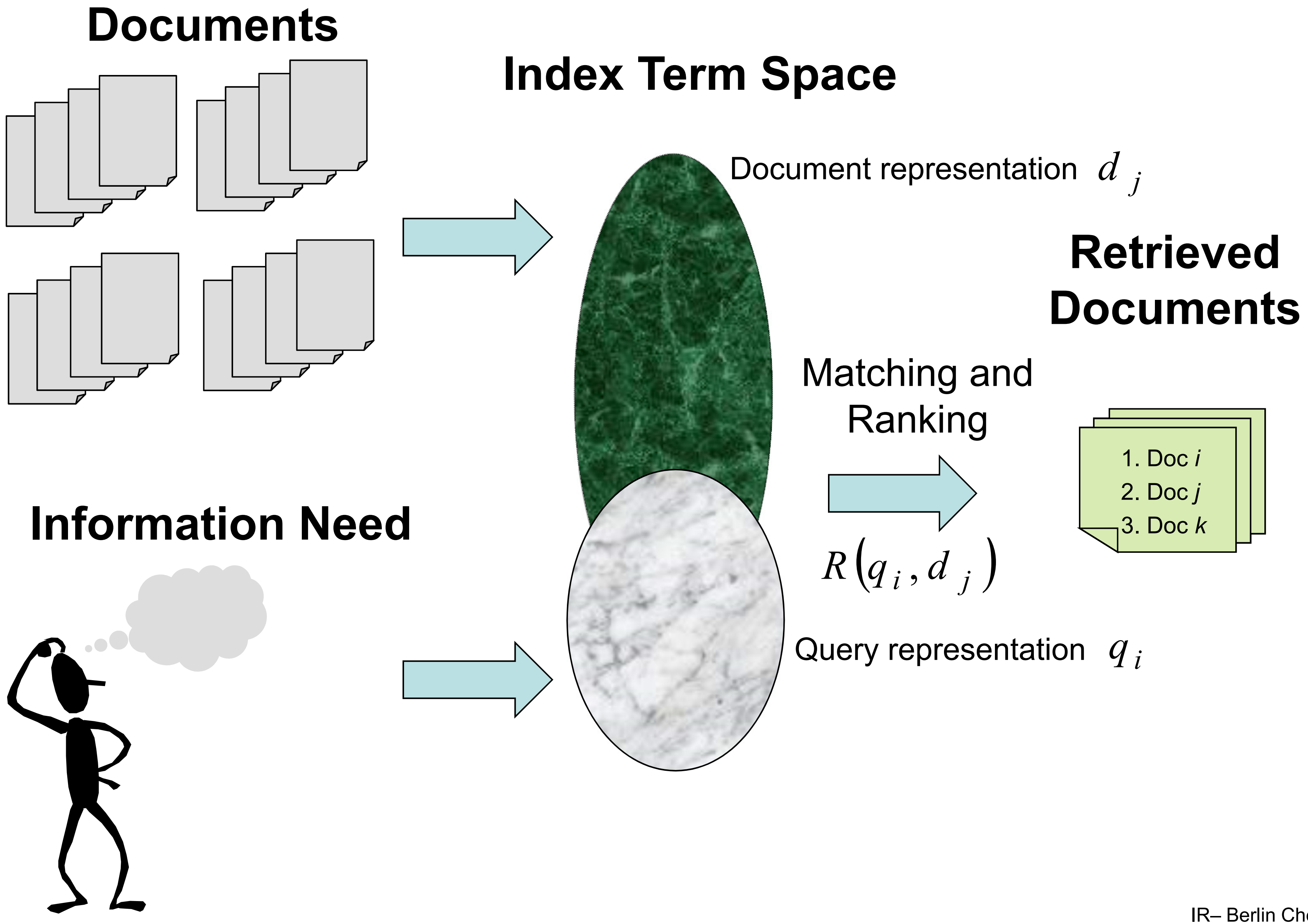
- Meanings From Two Perspectives
  - In a *restricted* sense (keyword-based)
    - An index term is a (predefined) keyword (usually a noun) which has some semantic meaning of its own
  - In a *more general* sense (word-based)
    - An index term is simply any word which appears in the text of a document 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 often lost when expressed through sets of words (e.g., *possible, probable, likely*)
    - Expressing query intent (information need) using a few words restricts the semantics of what can be expressed
  - Match between the documents and user queries is in the (*imprecise?*) space of index terms

Index Terms (cont.)

- Documents retrieved are flrequently irrelevant
  - Since most users have no training in query formation, problem is even worst
    - Not familar 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
    - A ranking algorithm predicts which documents the users will find relevant and which ones they will find irrelevant
      - Establish a simple ordering of the document retrieved; documents appearing on the top of this ordering are considered to be more likely to be relevant
    - However, two users might disagree what is relevant and what is not
      - Hopefully, the ranking algorithm can approximate the opinions of a large fraction of the users on the relevance of answers to a large fraction of queries


Documents
Index Term Space
Document representation d_j
Retrieved Documents
Matching and Ranking
1. Doc i
2. Doc j
3. Doc k
Information Need
$R(q_i, d_j)$
Query representation q_i

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 premises regarding the notion of document relevance, such as:
  - Common sets of index terms
  - Sharing of weighted terms
  - Likelihood of relevance

  literal-term matching

  $$P(Q|D) \text{ or } P(Q,D) \;?$$

  - Sharing of same aspects/concepts

  Concept/semantic matching

- Distinct sets of premises lead to a distinct IR models

Ranking Algorithms (cont.)

- Concept Matching vs. Literal Matching

Spoken Query

relevant ?

Transcript of Spoken Document

香港星島日報篇報導引述軍事觀察家的話表
示，到二零零五年台灣將完全喪失空中優勢，
原因是中國大陸戰機不論是數量或是性能上
都將超越台灣，報導指出中國在大量引進俄
羅斯先進武器的同時也得加快研發自製武器
系統，目前西安飛機製造廠任職的改進型飛
豹戰機即將部署尚未與蘇愷三十通道地對地
攻擊住宅飛機，以督促遇到挫折的監控其戰
機目前也已經取得了重大階段性的認知成果。
根據日本媒體報導在台海戰爭隨時可能爆發
情況之下北京方面的基本方針，使用高科技
答應局部戰爭。因此，解放軍打算在二零零
四年前又有包括蘇愷三十二期在內的兩百架
蘇霍伊戰鬥機。

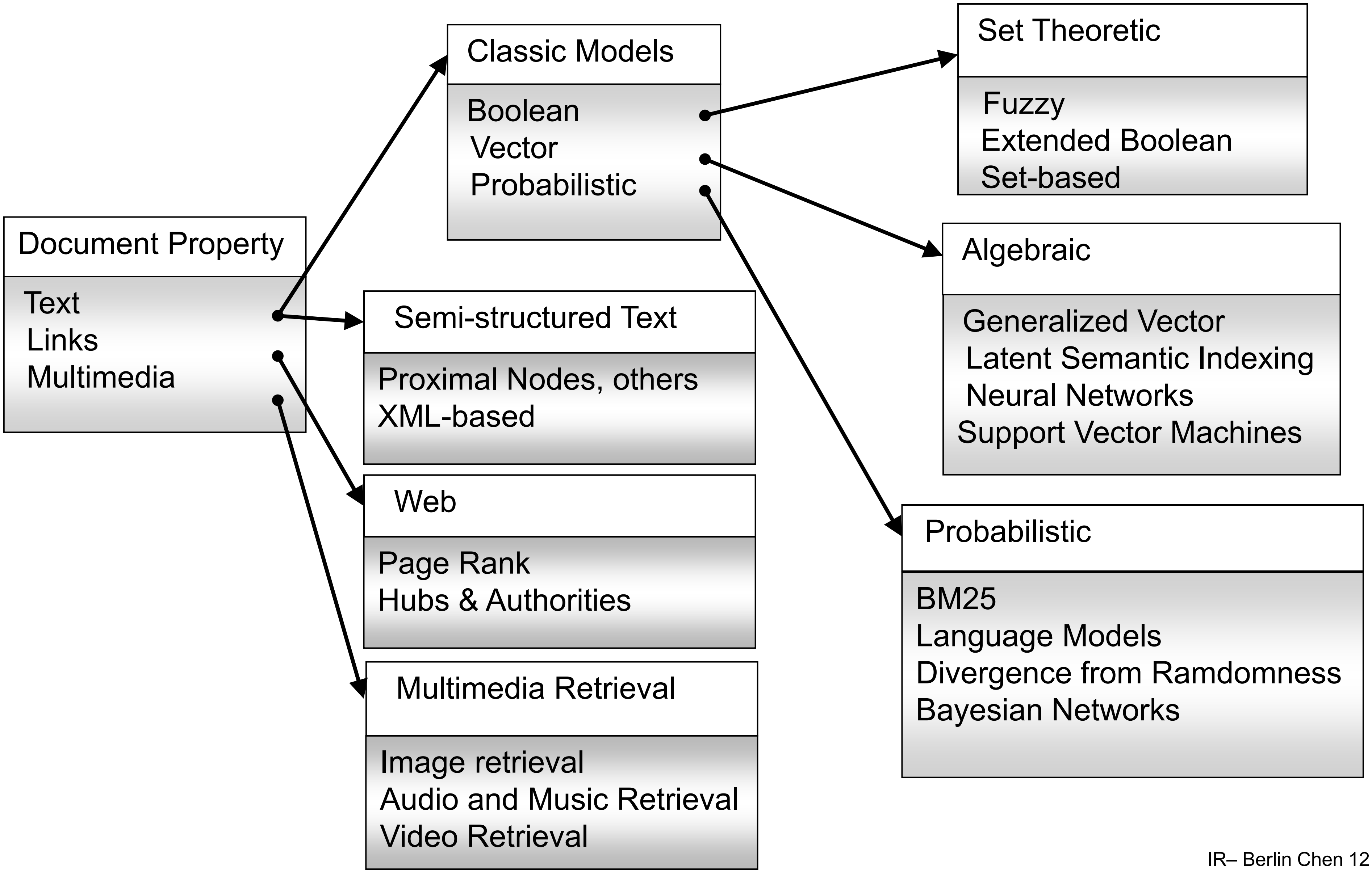
Taxonomy of Classic IR Models

- Refer to the text content
  - Unstructured
    - 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
  - Semi-structured (Chapter 13)
    - Take into account the structure components of the text like titles, sections, subsections, paragraphs
    - Also include unstructured text

Taxonomy of Classic IR Models (cont.)

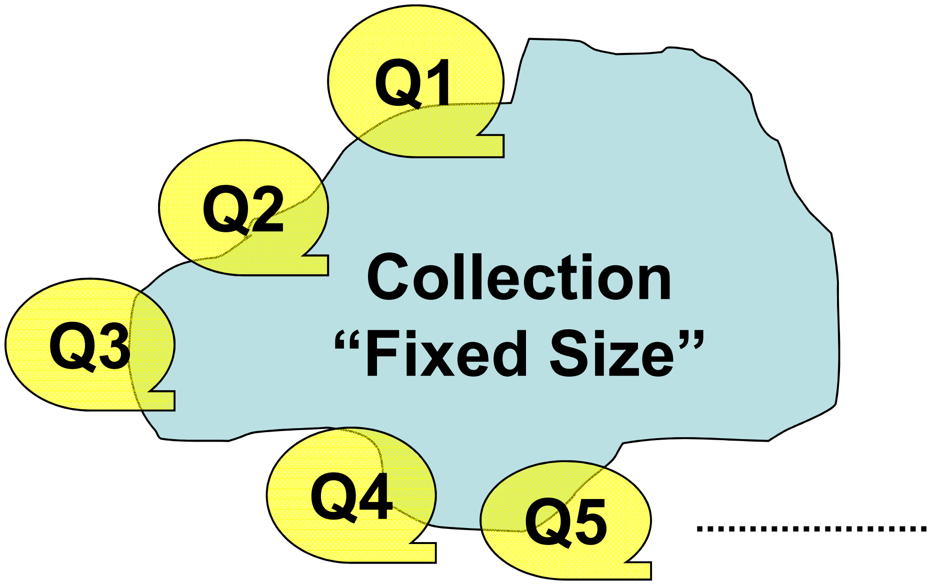
- Refer to the link structure of the Web (Chapter 11)
  - Consider the links among Web pages as an integral part of the model
- Refer to the content of multimedia objects (Chapter 14)
  - Images, video objects, audio objects

Taxonomy of Classic IR Models (cont.)



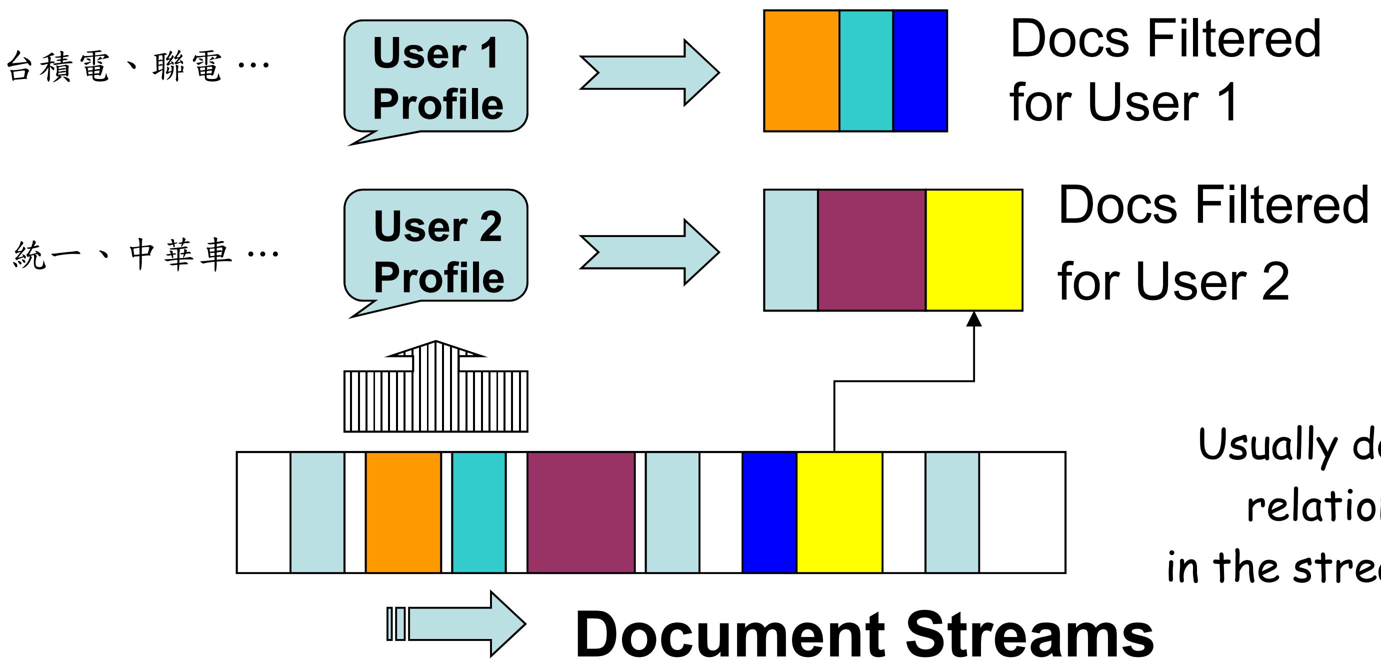
Retrieval: Ad Hoc

- *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



Usually 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 (for simplicity!)
    - **For filtering/routing, the crucial step is not ranking but the construction of user profiles**

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 continuously 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 associates a real number with $q_i \in \mathbf{Q}$ and $d_j \in \mathbf{D}$
    - Define an ordering among the documents d_j with regard to the query q_i

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

Basic Concepts

- Each document represented by a set of representative keywords or index terms
- An index term is a **word** or **group of consecutive words** in a document whose semantics is useful for remembering (summarizing) the *document main themes*
- Usually, index terms are *nouns* because nouns have meaning by themselves
  - Adjectives, adverbs, and connectives mainly work as complements
- However, search engines assume that all words are index terms (full text representation)

Basic Concepts (cont.)

- Let,
  - t be the number of index terms in the document collection
  - k_i be a generic index term
- Then,
  - The **vocabulary** $V = \{k_1, . . . , k_t\}$ is the set of all distinct index terms in the collection

$$V = \boxed{k_1 \quad k_2 \quad k_3 \quad \cdots \quad k_t}$$

vocabulary of *t*
index terms

Basic Concepts (cont.)

- Documents and queries can be represented by **patterns of term co-occurrences**

$V =$	k_1	k_2	k_3	...	k_t
	1	0	0	...	0	pattern that represents documents (and queries) with the term k_1 and no other
	⋮					
	1	1	1	...	1	pattern that represents documents (and queries) with all index terms

- Each of these patterns of term co-occurrence is called a **term conjunctive component**
- For each document d_j (or query q) we associate a unique term conjunctive component $c(d_j)$ (or $c(q)$)

The Term-Document Matrix

- The occurrence of a term k_i in a document d_j establishes a relation between k_i and d_j
- A **term-document relation** between k_i and d_j can be quantified by the frequency of the term in the document
- In matrix form, this can written as

$$
\begin{array}{cc}
 & \begin{array}{cc} d_1 & d_2 \end{array} \\
\begin{array}{c} k_1 \\ k_2 \\ k_3 \end{array} &
\left[\begin{array}{cc} f_{1,1} & f_{1,2} \\ f_{2,1} & f_{2,2} \\ f_{3,1} & f_{3,2} \end{array}\right]
\end{array}
$$

  – where each $f_{i,j}$ element stands for the frequency of term k_i in document d_j

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)
    - $\vec{d_j}=(w_{1,j}, w_{2,j}, \ldots, w_{t,j})$: 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 (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 ! (bag-of-words modeling)
  - However, the appearance of one word often attracts the appearance of the other (e.g., "Computer" and "Network")

The Boolean Model

- Simple model based on set theory and Boolean algebra
- A query is specified as boolean expressions with and, or, not operations (connectives)
  - 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
  - $\vec{q}_{dnf}$: the DNF for a query q
  - $\vec{q}_{cc}$: conjunctive components (binary weighted vectors) of $\vec{q}_{dnf}$

The Boolean Model (cont.)

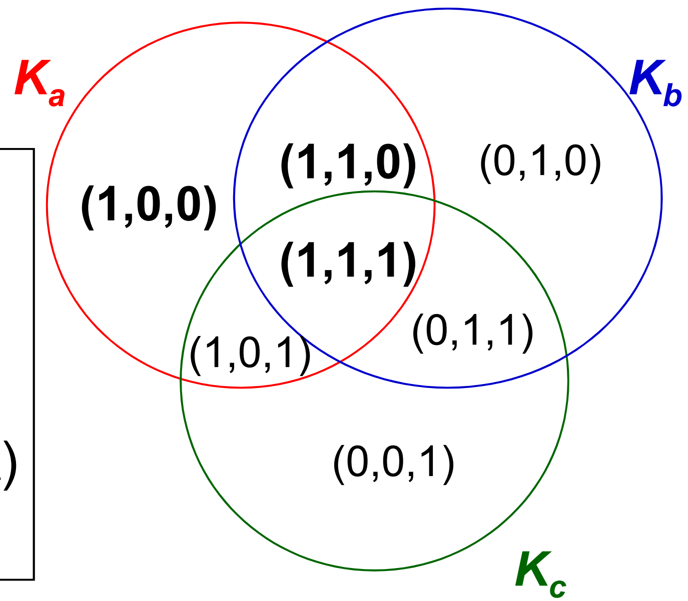
- For intance, a query [$q = k_a \wedge (k_b \vee \neg k_c)$] can be written as a DNF

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

a canonical representation

conjunctive components
(binary weighted vectors)

Does $d = k_a \wedge k_b \wedge \neg k_c$ satisfy q ?



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

The Boolean Model (cont.)

- The similarity of a document d_j to the query q (i.e., premise of relevance)

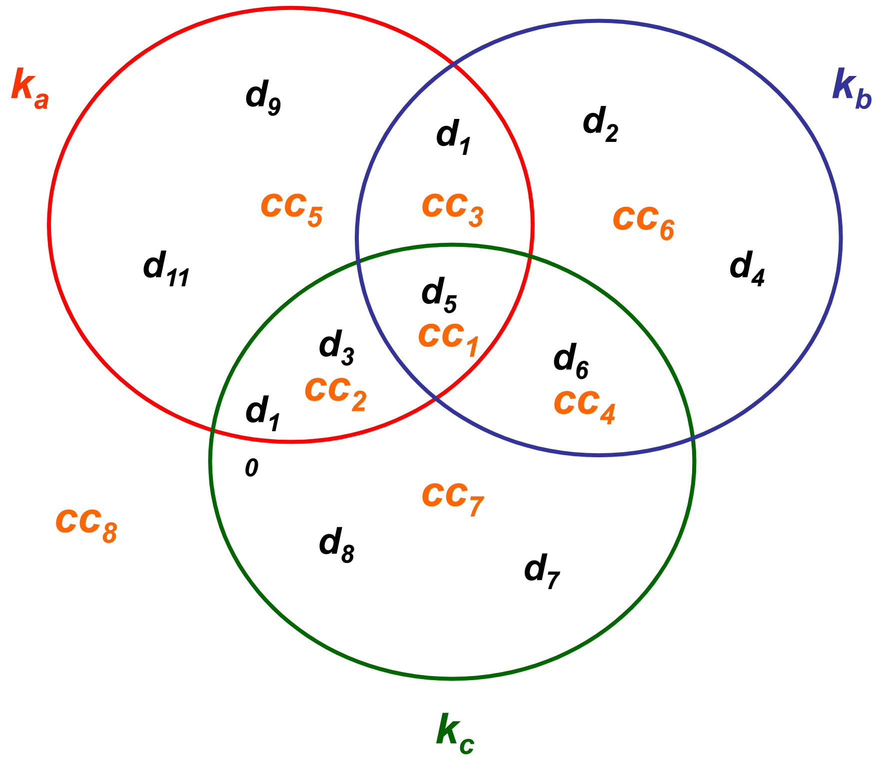
$$sim(d_j,q)=\begin{cases} 1: \text{if } \exists \vec{q}_{cc} \mid (\vec{q}_{cc} \in \vec{q}_{dnf} \wedge (\forall k_i,\ g_i(\vec{d}_j)=g_i(\vec{q}_{cc})) \\ 0: \text{otherwise} \end{cases}$$

A document is represented as
a conjunctive normal form

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

Advantages of the Boolean Model

- Simple queries are easy to understand and relatively easy to implement (simplicity and neat model formulation)
- The dominant language (model) in commercial (bibliographic) systems until the WWW



$cc_1 = k_a\, k_b\, k_c$

$cc_2 = k_a\, \bar{k}_b\, k_c$

$cc_3 = k_a\, k_b\, \bar{k}_c$

$cc_4 = \bar{k}_a\, k_b\, k_c$

$cc_5 = k_a\, \bar{k}_b\, \bar{k}_c$

$cc_6 = \bar{k}_a\, k_b\, \bar{k}_c$

$cc_7 = \bar{k}_a\, \bar{k}_b\, k_c$

$cc_8 = \bar{k}_a\, \bar{k}_b\, \bar{k}_c$

Drawbacks of the Boolean Model

- Retrieval based on binary decision criteria with no notion of partial matching (no term weighting)
  – No noton of a partial match to the query condition
  – No ranking (ordering) of the documents is provided (absence of a grading scale)
  – Term freqency counts in documents are 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
- However, the Boolean model is still dominant model with commercial document database systems

Term Weighting

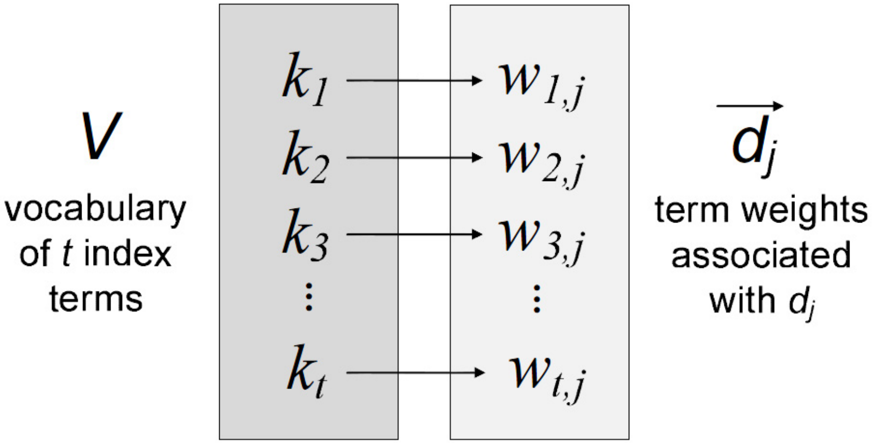
- The terms of a document are not equally useful for describing the document contents
- In fact, there are index terms which are simply vaguer than others
- There are (occurrence) properties of an index term which are useful for evaluating the importance of the term in a document
- For instance, a word which appears in all documents of a collection is completely useless for retrieval tasks
  - However, deciding on the importance of a term for summarizing the contents of a document is not a trivial issue

Term Weighting (cont.)

- To characterize term importance, we associate a weight $w_{i,j} > 0$ with each term k_i that occurs in the document d_j
  - If k_i that does not appear in the document d_j , then $w_{i,j} = 0$
- The weight $w_{i,j}$ quantifies the importance of the index term k_i for describing the contents of document d_j
- These weights are useful to compute a rank for each document in the collection with regard to a given query

Term Weighting (cont.)

- Let,
  - k_i be an index term and dj be a document
  - $V = \{k_1, k_2, ..., k_t\}$ be the set of all index terms
  - $w_{i,j} > 0$ be the weight associated with (k_i, d_j)
- Then we define $\vec{d_j} = (w_{1,j}, w_{2,j}, ..., w_{t,j})$ as **a weighted vector** that contains the weight $w_{i,j}$ of each term $k_i \in V$ in the document d_j



Term Weighting (cont.)

- The weights $w_{i,j}$ can be computed using the **frequencies of occurrence** of the terms within documents

- Let $f_{i,j}$ be the frequency of occurrence of index term k_i in the document d_j

- The **total frequency of occurrence** F_i of term k_i in the collection is defined as

$$F_i = \sum_{j=1}^{N} f_{i,j}$$

  – where N is the number of documents in the collection

Term Weighting (cont.)

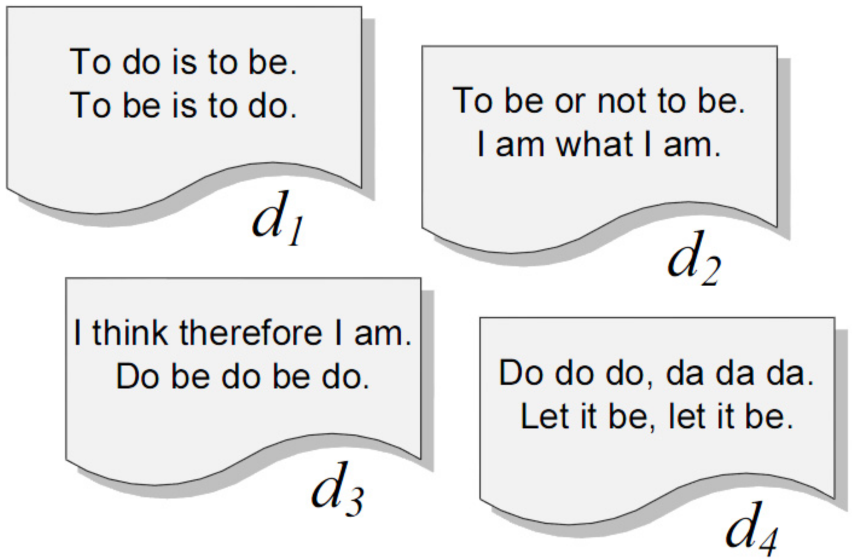
- The **document frequency** n_i of a term k_i is the number of documents in which it occurs
  - Notice that $n_i \leq F_i$

- For instance, in the document collection below, the values $f_{i,j}$, F_i and n_i associated with the term "*do*" are

$$f(do, d_1) = 2$$
$$f(do, d_2) = 0$$
$$f(do, d_3) = 3$$
$$f(do, d_4) = 3$$

$$F(do) = 8$$

$$n(do) = 3$$



Term-Term Correlation Matrix

- For classic information retrieval models, the index term weights are assumed to be **mutually independent**
  - This means that $w_{i,j}$ tells us nothing about $w_{i+1,j}$
- This is clearly a simplification because occurrences of index terms in a document are not uncorrelated
- For instance, the terms **computer** and **network** tend to appear together in a document about **computer networks**
  - In this document, the appearance of one of these terms attracts the appearance of the other
  - Thus, they are correlated and their weights should reflect this correlation

Term-Term Correlation Matrix (cont.)

- To take into account term-term correlations, we can compute a correlation matrix
- Let $\vec{M} = [m_{ij}]$ be a term-document matrix $t \times N$ where $m_{ij} = w_{i,j}$
- The matrix $\vec{C} = \vec{M} \cdot \vec{M}^t$ is a term-term correlation matrix
- Each element $c_{u,v} \in C$ expresses a correlation between terms k_u and k_v, given by

$$c_{uv} = \sum_{d_j} w_{u,j} \times w_{v,j}$$

  – Higher the number of documents in which the terms k_u and k_v co-occur, stronger is this correlation

Term-Term Correlation Matrix (cont.)

- Term-term correlation matrix for a sample collection

$$
\underbrace{
\underset{\mathbf{M}}{
\begin{array}{c} \\ k_1 \\ k_2 \\ k_3 \end{array}
\begin{array}{c}
\begin{array}{cc} d_1 & d_2 \end{array} \\
\left[\begin{array}{cc} w_{1,1} & w_{1,2} \\ w_{2,1} & w_{2,2} \\ w_{3,1} & w_{3,2} \end{array}\right]
\end{array}
}
\quad \times \quad
\underset{\mathbf{M}^T}{
\begin{array}{c} \\ d_1 \\ d_2 \end{array}
\begin{array}{c}
\begin{array}{ccc} k_1 & k_2 & k_3 \end{array} \\
\left[\begin{array}{ccc} w_{1,1} & w_{2,1} & w_{3,1} \\ w_{1,2} & w_{2,2} & w_{3,2} \end{array}\right]
\end{array}
}
}
$$

$$\Downarrow$$

$$
\begin{array}{c} \\ k_1 \\ k_2 \\ k_3 \end{array}
\begin{array}{c}
\begin{array}{ccc} k_1 & k_2 & k_3 \end{array} \\
\left[\begin{array}{ccc}
w_{1,1}w_{1,1} + w_{1,2}w_{1,2} & w_{1,1}w_{2,1} + w_{1,2}w_{2,2} & w_{1,1}w_{3,1} + w_{1,2}w_{3,2} \\
w_{2,1}w_{1,1} + w_{2,2}w_{1,2} & w_{2,1}w_{2,1} + w_{2,2}w_{2,2} & w_{2,1}w_{3,1} + w_{2,2}w_{3,2} \\
w_{3,1}w_{1,1} + w_{3,2}w_{1,2} & w_{3,1}w_{2,1} + w_{3,2}w_{2,2} & w_{3,1}w_{3,1} + w_{3,2}w_{3,2}
\end{array}\right]
\end{array}
$$

  - Further, we can take advantage of factors such as term-term distances inside documents to improve the estimates of term-term correlations (see Chapter 5)

TF-IDF Weights

- Term frequency (TF)
- Inverse document frequency (IDF)

They are foundations (building blocks) of the most popular term weighting scheme in IR, called **TF-IDF**

Term Frequency (TF) Weights

- The simplest formulation is

$$tf_{i,j} = f_{i,j}$$

The frequency of occurrence of index term k_i im the document d_j

- A variant of *tf* weight used in the literature is

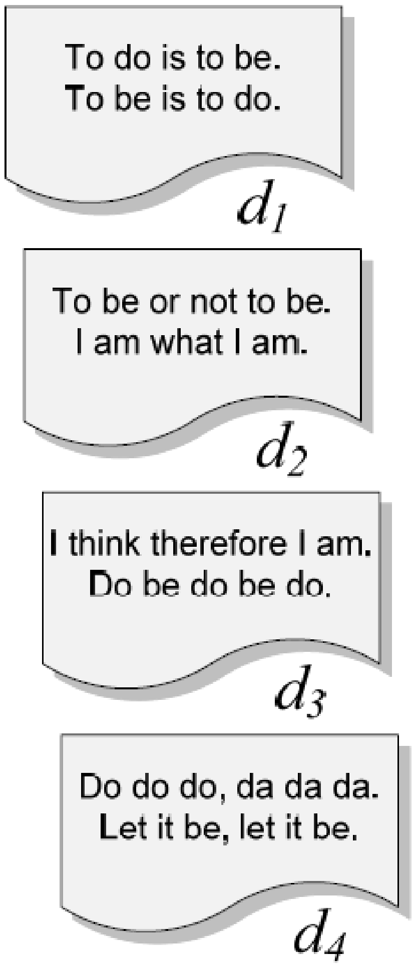
$$tf_{i,j} = \begin{cases} 1 + \log f_{i,j} & \text{if } f_{i,j} > 0 \\ 0 & \text{otherwise} \end{cases}$$

  – Where the log is taken in base 2

- The log expression is a the preferred form because it makes them directly comparable to *idf* weights, as we later discuss

Term Frequency (TF) Weights: An Example

- Log *tf* weights $tf_{i,j}$ for the example collection



Vocabulary		$tf_{i,1}$	$tf_{i,2}$	$tf_{i,3}$	$tf_{i,4}$
1	to	3	2	-	-
2	do	2	-	2.585	2.585
3	is	2	-	-	-
4	be	2	2	2	2
5	or	-	1	-	-
6	not	-	1	-	-
7	I	-	2	2	-
8	am	-	2	1	-
9	what	-	1	-	-
10	think	-	-	1	-
11	therefore	-	-	1	-
12	da	-	-	-	2.585
13	let	-	-	-	2
14	it	-	-	-	2

Inverse Document Frequency

- We call **document exhaustivity** the number of index terms assigned to a document
- The more index terms are assigned to a document, the higher is the probability of retrieval for that document
  - If too many terms are assigned to a document, it will be retrieved by queries for which it is not relevant
- **Optimal exhaustivity**: we can circumvent this problem by optimizing the number of terms per document
- Another approach is by weighting the terms differently, by exploring the notion of term specificity

Inverse Document Frequency (cont.)

- **Specificity** is a property of the term semantics
  - Term is more or less specific depending on its meaning
  - To exemplify, the term **beverage** is less specific than the terms **tea** and **beer**
  - We could expect that the term **beverage** occurs in more documents than the terms **tea** and **beer**
- Term specificity should be interpreted as a statistical rather than semantic property of the term
- **Statistical term specificity**: the inverse of the number of documents in which the term occurs

Inverse Document Frequency : Derivation

- Terms are distributed in a text according to **Zipf's Law**
- Thus, if we sort the vocabulary terms in decreasing order of document frequencies we have

$$n(r) \propto r^{-\alpha}$$

There is an inverse relationship between $n(r)$ and r

  – Where $n(r)$ refers to the r-th largest **document frequency** and α is an empirical constant

- That is, the **document frequency** of term k_i is an exponential function of its rank

$$n(r) = Cr^{-\alpha}$$

  – where C is a second empirical constant

Inverse Document Frequency : Derivation

- Setting α = 1 (simple approximation for English collections) and taking logs we have

$$\log n(r) = \log C - \log r$$

- For r = 1, we have C = n(1), i.e., the value of C is the largest document frequency
  - This value (i.e., C's value) works as a normalization constant
- An alternative is to do the normalization assuming C = N, where N is the number of documents in the collection

$$\log r \approx \log N - \log n(r) = \log \frac{N}{n(r)}$$

Inverse Document Frequency : Derivation

- Let k_i be the term with the r-th largest document frequency, i.e., $n(r) = n_i$. Then,

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

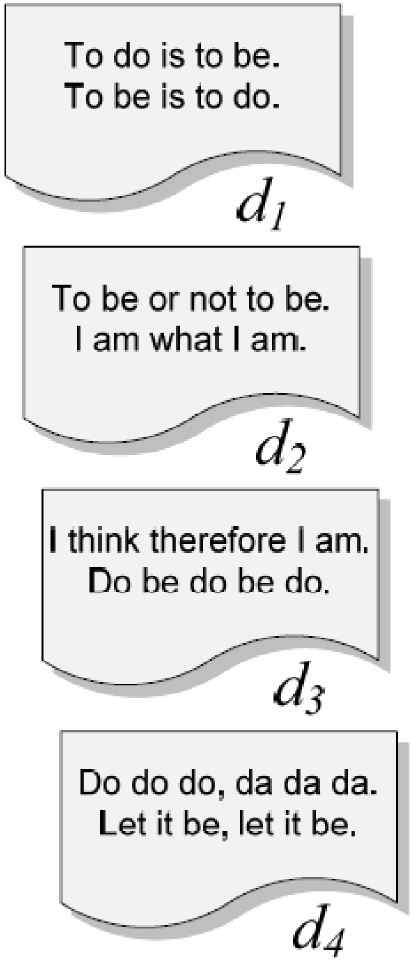
Sparck Jones, 1972

  - where idf_i is called the inverse document frequency of term k_i

- IDF provides a foundation for modern term weighting schemes and is used for ranking in almost all IR systems

Inverse Document Frequency : An Example

- IDF values for example collection



	term	n_i	$idf_i = \log(N/n_i)$
1	to	2	1
2	do	3	0.415
3	is	1	2
4	be	4	0
5	or	1	2
6	not	1	2
7	I	2	1
8	am	2	1
9	what	1	2
10	think	1	2
11	therefore	1	2
12	da	1	2
13	let	1	2
14	it	1	2

More on Inverse Document Frequency

- In a large real collection, we expect the most selective terms to be nouns or noun groups (a noun composed of various words)
- The least selective terms are usually article, conjunctions, and prepositions which are frequently referred to as **stop words**
- IDF weights provide a foundation for modern term weighting schemes and are used by almost any modern IR system

TF-IDF weighting scheme

Salton and Yang, 1973

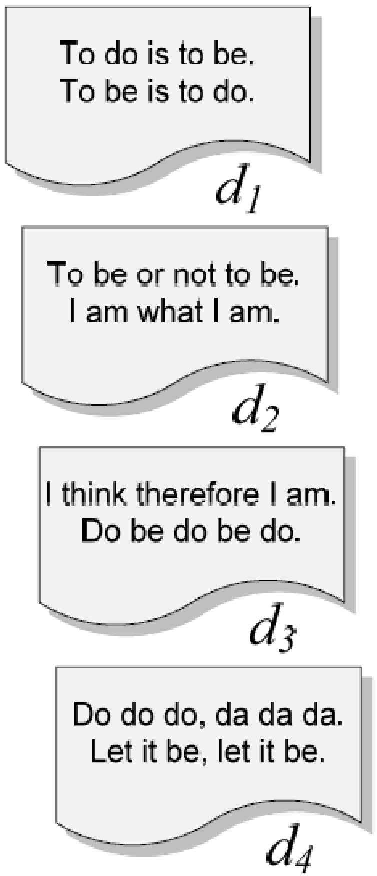
- The best known term weighting schemes use weights that combine **IDF factors** with **term frequencies**
- Let $w_{i,j}$ be the term weight associated with the term k_i and the document d_j
- Then, we define

$$w_{i,j} = \begin{cases} \left(1 + \log f_{i,j}\right) \times \log \frac{N}{n_i} & \text{if } f_{i,j} > 0 \\ 0 & \text{otherwise} \end{cases}$$

  – Which is referred to as a **TF-IDF weighting scheme**

TF-IDF weighting scheme: An Example

- TF-IDF weights of all terms present in our example document collection



		d_1	d_2	d_3	d_4
1	to	3	2	-	-
2	do	0.830	-	1.073	1.073
3	is	4	-	-	-
4	be	-	-	-	-
5	or	-	2	-	-
6	not	-	2	-	-
7	I	-	2	2	-
8	am	-	2	1	-
9	what	-	2	-	-
10	think	-	-	2	-
11	therefore	-	-	2	-
12	da	-	-	-	5.170
13	let	-	-	-	4
14	it	-	-	-	4

Variants of TF-IDF

- Several variations of the above expression for TF-IDF weights are described in the literature
- For **TF weights**, five distinct variants are illustrated below

	tf weight
binary	{0,1}
raw frequency	$f_{i,j}$
log normalization	$1 + \log f_{i,j}$
double normalization 0.5	$0.5 + 0.5 \frac{f_{i,j}}{max_i f_{i,j}}$
double normalization K	$K + (1 - K) \frac{f_{i,j}}{max_i f_{i,j}}$

Variants of TF-IDF (Cont.)

- Five distinct variants of **IDF weights**

	idf weight
unary	1
inverse frequency	$\log \frac{N}{n_i}$
inv frequency smooth	$\log(1 + \frac{N}{n_i})$
inv frequeny max	$\log(1 + \frac{max_i n_i}{n_i})$
probabilistic inv frequency	$\log \frac{N - n_i}{n_i}$

  – The probabilistic inverse frequency variant arises from the classic probabilistic model, as discussed later on

Variants of TF-IDF (Cont.)

- Distinct combinations of TF variants and IDF variants yield various forms of TF-IDF weights
  - Recommended TF-IDF weighting schemes

weighting scheme	document term weight	query term weight
1	$f_{i,j} * \log \frac{N}{n_i}$	$(0.5 + 0.5 \frac{f_{i,q}}{max_i \ f_{i,q}}) * \log \frac{N}{n_i}$
2	$1 + \log f_{i,j}$	$\log(1 + \frac{N}{n_i})$
3	$(1 + \log f_{i,j}) * \log \frac{N}{n_i}$	$(1 + \log f_{i,q}) * \log \frac{N}{n_i}$

Salton & Buckley

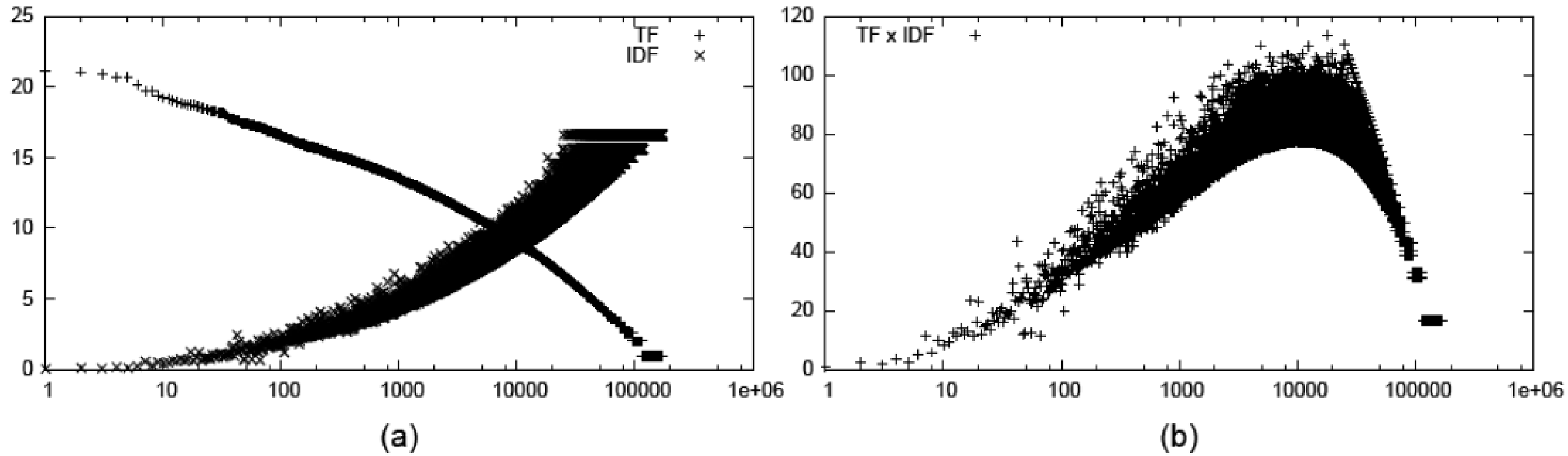
TF-IDF Properties

- Consider the TF, IDF, and TF-IDF weights for the Wall Street Journal reference collection
- To study their behavior, we would like to plot them together
- While IDF is computed over all the collection, TF is computed on a per document basis
  - Thus, we need a representation of TF based on all the collection, which is provided by the **term collection frequency** *F_i*
- This reasoning leads to the following TF and IDF term weight

$$TF_i = 1 + \sum_{j=1}^{N} f_{i,j} \qquad IDF_i = \frac{N}{n_i}$$

TF-IDF Properties (Cont.)

- Plotting TF and IDF in logarithmic scale yields



  - Statistics are gathered from the *Wall Street Journal* collection
  - The horizontal axis corresponding the rank of each term according to TF
- We observe that TF and IDF weights present power-law behaviors that balance each other
- The terms of **intermediate IDF values** display maximum TF-IDF weights and are most interesting for ranking

TF-IDF Properties (Cont.)

- Common terms (such as stopwords) and rare terms (such as foreign words or misspellings) are not of great value for ranking

Document Length Normalization

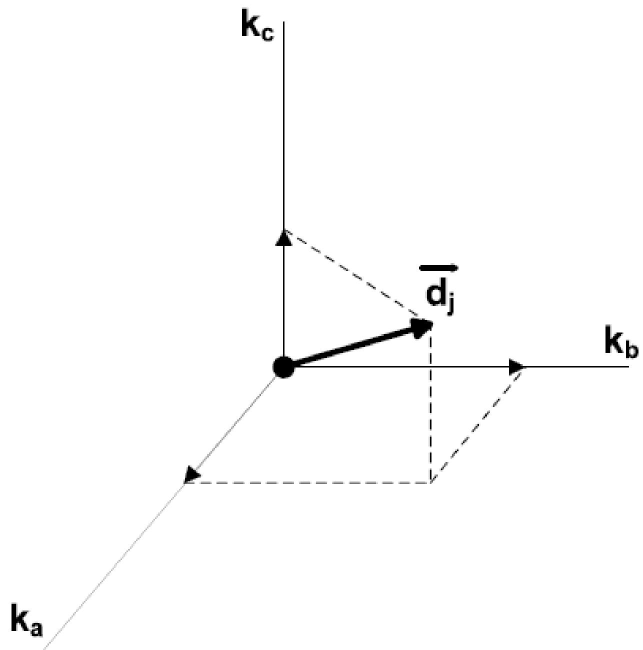
- Document sizes might vary widely
- This is a problem because longer documents are more likely to be retrieved by a given query
- To compensate for this undesired effect, we can divide the rank of each document by its length
- This procedure consistently leads to better ranking, and it is called **document length normalization**

Document Length Normalization (cont.)

- Methods of document length normalization depend on the representation adopted for the documents:
  - **Size in bytes**: consider that each document is represented simply as a stream of bytes
  - **Number of words**: each document is represented as a single string, and the document length is the number of words in it
  - **Vector norms**: documents are represented as vectors of weighted terms

Document Length Normalization (cont.)

- Documents represented as vectors of weighted terms
  - Each term of a collection is associated with **an orthonormal unit vector** $\vec{k_i}$ in a t-dimensional space
  - For each term k_i of a document d_j is associated the term vector
  - component $w_{i,j} \times \vec{k_i}$



Document Length Normalization (cont.)

- The document representation $\vec{d_j}$ is a vector composed of all its term vector components

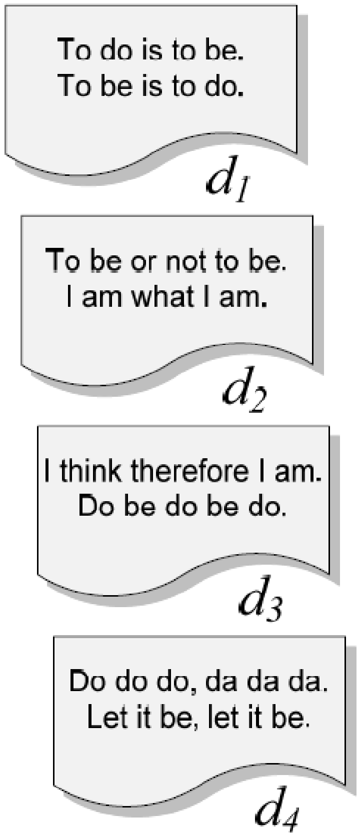
$$\vec{d}_j = \left(w_{1,j}, w_{2,j}, \ldots, w_{t,j} \right)$$

- The document length is given by the norm of this vector, which is computed as follows

$$\left| \vec{d}_j \right| = \sqrt{\sum_{i=1}^{t} w_{i,j}^2}$$

Document Length Normalization (cont.)

- Three variants of document lengths for the example
- collection



	d_1	d_2	d_3	d_4
size in bytes	34	37	41	43
number of words	10	11	10	12
vector norm	5.068	4.899	3.762	7.738

The Vector Model

SMART system
Cornell U., 1968

- Also called *Vector Space Model (VSM)*
- 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 better matching for user information need

The Vector Model (cont.)

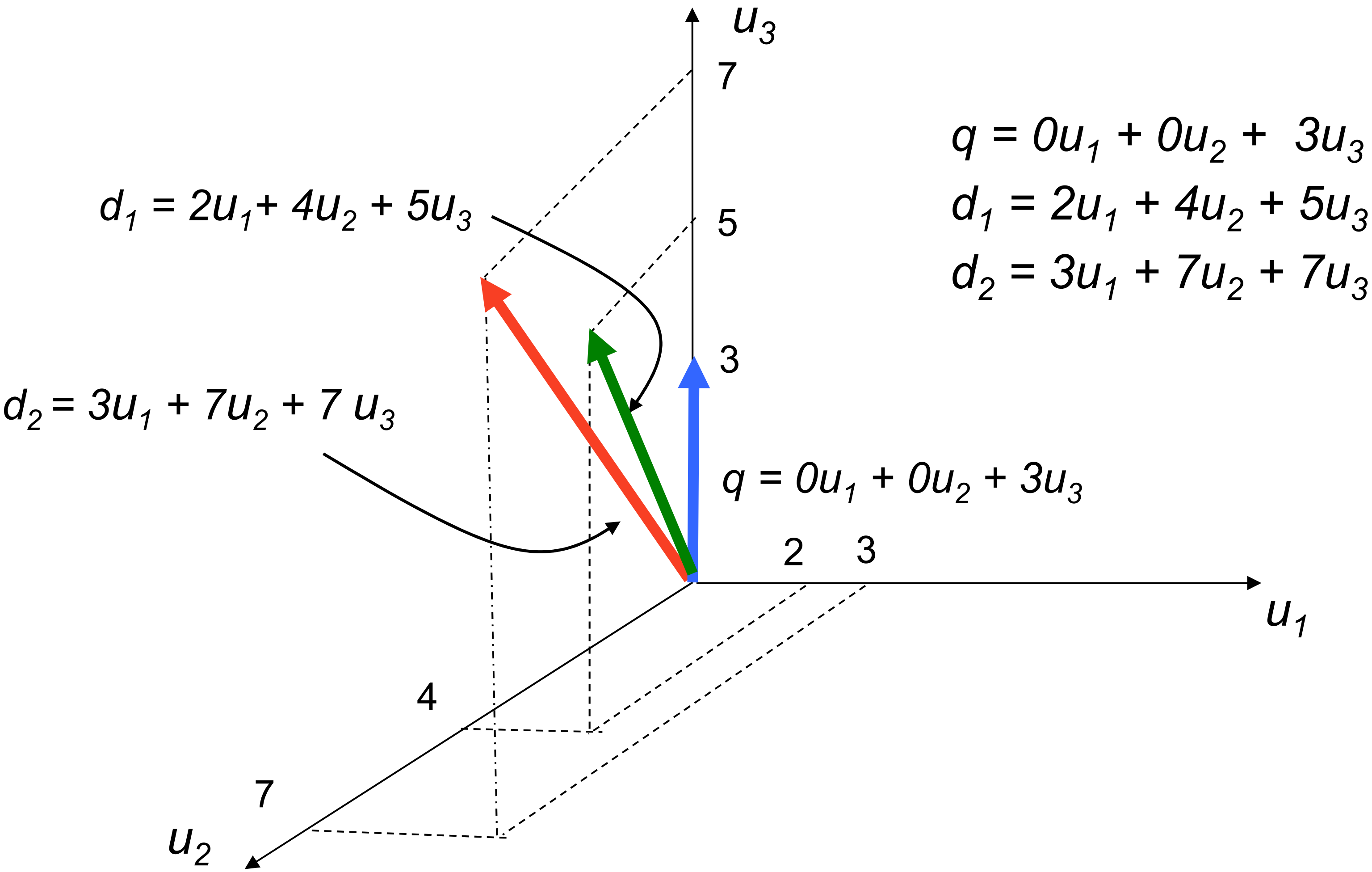
- Definition:
  - $w_{ij} >= 0$ whenever $k_i \in d_j$
  - $w_{iq} >= 0$ whenever $k_i \in q$
  - document vector $\vec{d_j} = (w_{1j}, w_{2j}, ..., w_{tj})$
  - query vector $\vec{q} = (w_{1q}, w_{2q}, ..., w_{tq})$
  - To each term k_i is associated a unitary (basis) vector $\vec{u_i}$
  - The unitary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

totally t terms in the vocabulary

- 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.)

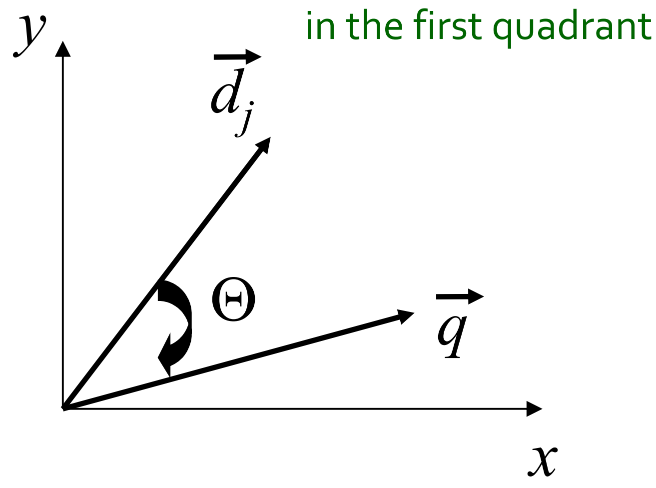
- How to measure the degree of similarity
  - Distance, angle or projection?



The Vector Model (cont.)

- The similarity of a document d_j to the query q

$$
\begin{aligned}
& sim(d_j, q) \\
& = \text{cosine}(\Theta) \\
& = \frac{\vec{d}_j \bullet \vec{q}}{|\vec{d}_j| \times |\vec{q}|} \\
& = \frac{\sum_{i=1}^{t} w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^{t} w_{i,j}^2} \times \sqrt{\sum_{i=1}^{t} w_{i,q}^2}}
\end{aligned}
$$



Document length normalization

The same for documents, can be discarded ⇨ Won't affect the final ranking

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

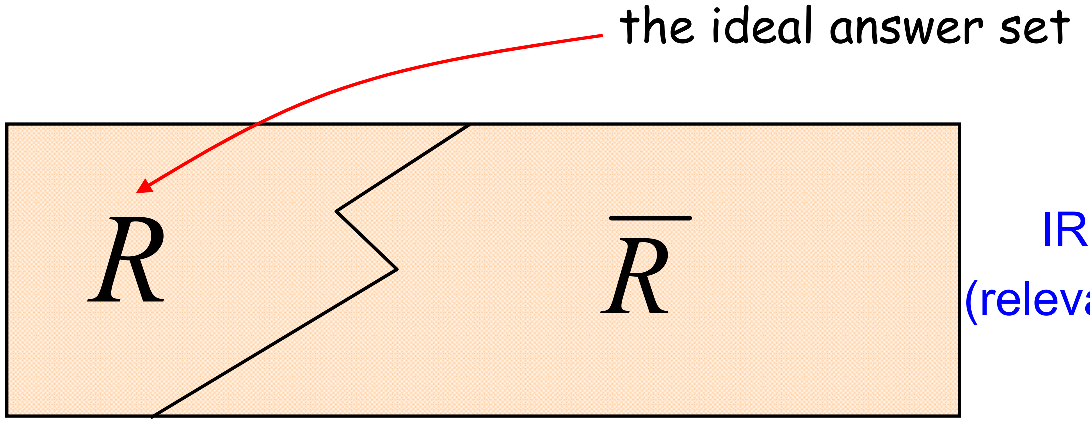
- Establish a threshold on $sim(d_j,q)$ and retrieve documents with a degree of similarity above the threshold

The Vector Model (cont.)

- Degree of similarity ⇨ Relevance
  - Usually, w_{ij} > =0 & w_{iq} >= 0
    - Cosine measure ranges between 0 and 1
  - $sim(d_j, q) \approx 1$ ⇨ highly relevant !
  - $sim(d_j, q) \approx 0$ ⇨ almost irrelevant !

The Vector Model (cont.)

- The role of index terms



Document collection

IR as a binary clustering (relevant/non-relevant) problem

  - Which index terms (features) better describe the relevant class
    - Intra-cluster similarity (TF-factor)
    - Inter-cluster dissimilarity (IDF-factor)

balance between these two factors

The Vector Model (cont.)

- The vector model with **TF-IDF weights** is a good ranking strategy with *general* collections, for example

$$w_{i,q} = \left(1 + \log f_{i,q}\right) \times \log\left(\frac{N}{n_i}\right)$$

$$w_{i,j} = \left(1 + \log f_{i,j}\right) \times \log\left(\frac{N}{n_i}\right)$$

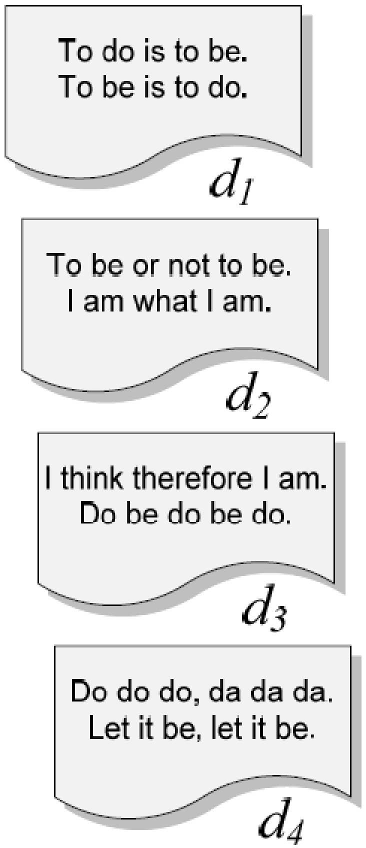
  – These equations should only be applied for values of term frequency greater than zero

  – If the term frequency is zero, the respective weight is also zero

- The vector model is usually as good as the known ranking alternatives. It is also simple and fast to compute

The Vector Model (cont.)

- Document ranks computed by the Vector model for the
- query "**to do**" (see TF-IDF weight values in Slide 49)



doc	rank computation	rank
d_1	$\frac{1*3+0.415*0.830}{5.068}$	0.660
d_2	$\frac{1*2+0.415*0}{4.899}$	0.408
d_3	$\frac{1*0+0.415*1.073}{3.762}$	0.118
d_4	$\frac{1*0+0.415*1.073}{7.738}$	0.058

The Vector Model (cont.)

- Experimental Results on TDT Chinese collections
  - Mandarin Chinese broadcast news
  - Measured in *mean* Average Precision (*m*AP)
  - ACM TALIP (2004)

Retrieval Results for the Vector Space Model

Average Precision		Word-level		Syllable-level	
		S(N), *N*=1	*S(N)*, *N*=1~2	*S(N)*, *N*=1	*S(N)*, *N*=1~2
TDT-2 (Dev.)	TD	0.5548	0.5623	0.3412	0.5254
	SD	0.5122	0.5225	0.3306	0.5077
TDT-3 (Eval.)	TD	0.6505	0.6531	0.3963	0.6502
	SD	0.6216	0.6233	0.3708	0.6353

$$R(q,d) = \sum_{j} w_j \cdot R_j(\vec{q}_j, \vec{d}_j),$$

types of index terms

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
  - Document normalization is naturally built-in into the ranking

- Disadvantages
  - Assumes mutual independence of index terms
    - Not clear that this is bad though (??): leveraging term dependencies is challenging and might lead to poor results, if not done appropriately

The Probabilistic Model

Roberston & Sparck Jones 1976

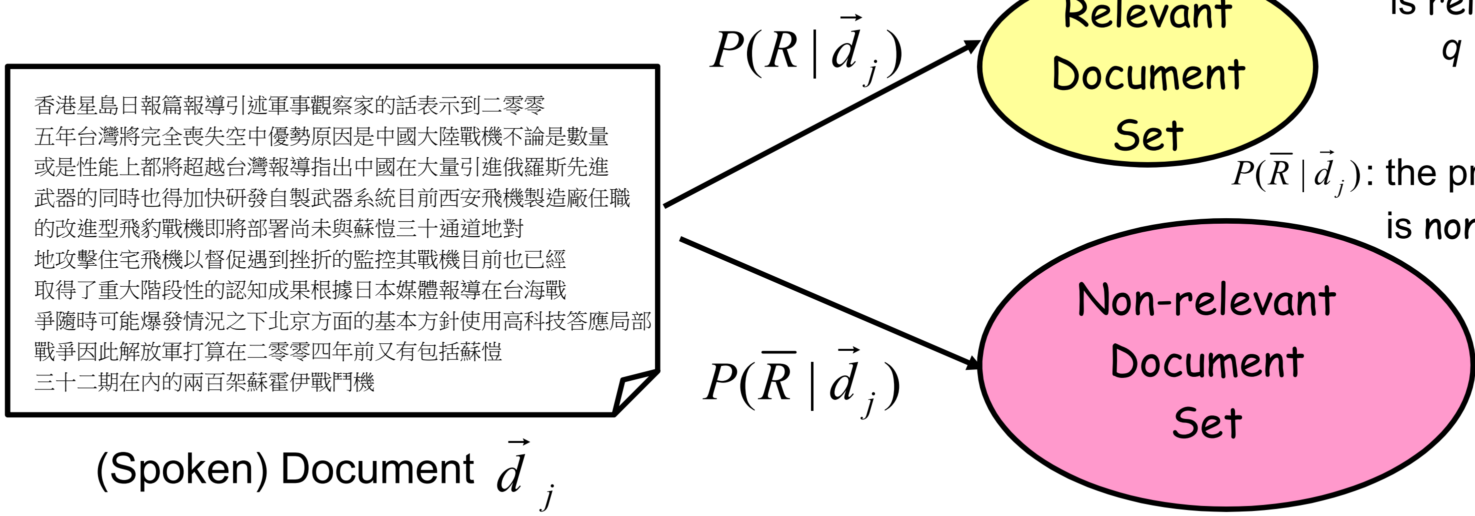
- Known as the Binary Independence Retrieval (BIR) model
  - "**Binary**": all weights of index terms are binary (0 or 1)
  - "**Independence**": index terms are independent !
- Capture the IR problem using a probabilistic framework
  - Bayes' decision rule

The Probabilistic Model (cont.)

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

$P(R \mid \vec{d}_j)$: the prob. that the doc. d_j is relevant to the query q

$P(\overline{R} \mid \vec{d}_j)$: the prob. that the doc. d_j is non-relevant to the query q



The Probabilistic Model (cont.)

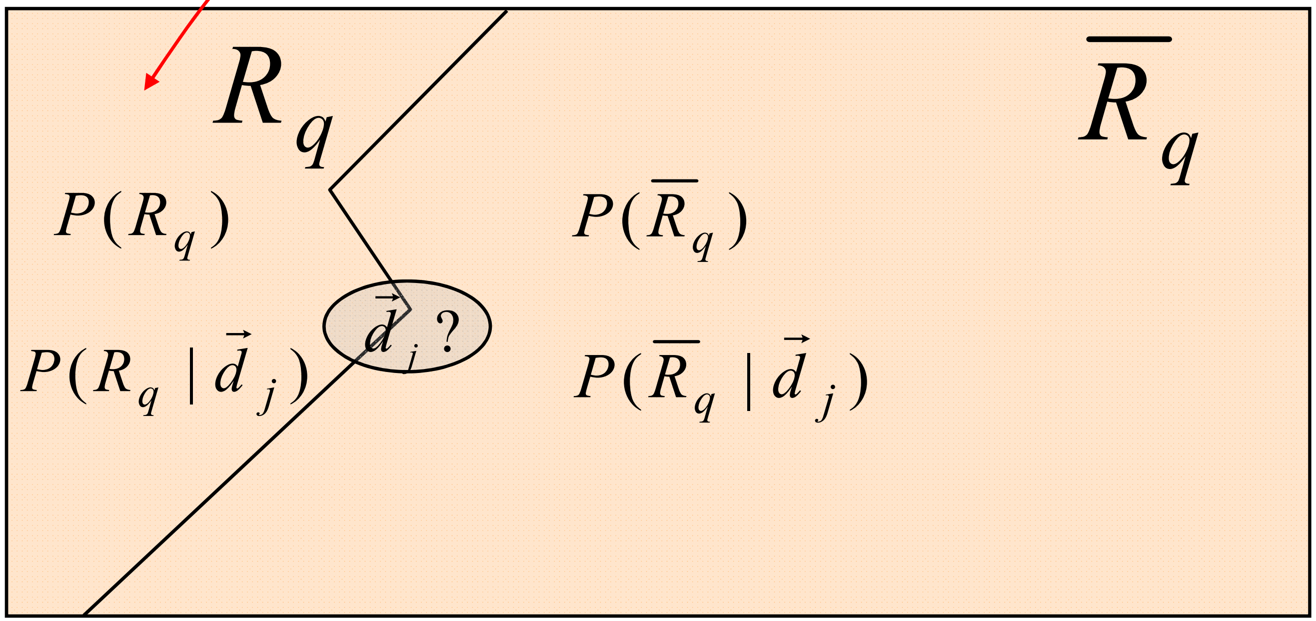
- Given a user query, there is an *ideal answer set*
  - Contain exactly the relevant documents and no others
  - The querying process as a specification of the properties of this ideal answer set (R_q)
- Problem: what are these properties?
  - Only the *semantics of* *index terms* can be used to characterize these properties
- ***Guess at the beginning*** what they could be
  - I.e., an initial guess for the preliminary probabilistis description of ideal answer set
- ***Improve/Refine the probabilistic description of the answer set by iterations/interations***
  - Without (or with) the assistance from a human subject

The Probabilistic Model (cont.)

- How to improve the probabilistic description of the ideal answer set ?

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

the ideal answer set



Document Collection

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

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

Bayes' Decision Rule

- The similarity of a document d_j to the query q

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

Likelihood/Odds Ratio Test

Bayes' Theory

The same for all documents

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

$\geq \tau$?

if so, retrieved !

The Probabilistic Model (cont.)

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

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

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

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

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

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

The Probabilistic Model (cont.)

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

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

  - Take logarithms

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

The same for all documents!

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

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

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

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

The Probabilistic Model (cont.)

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

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

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

Binary weights (0 or 1) are used here

R_q is not known at the beginning

⇨ How to compute $P(k_i \mid R_q)$ and $P(k_i \mid \overline{R}_q)$

The Probabilistic Model (cont.)

- Initial Assumptions
  - $P(k_i \mid R_q) = 0.5$:is constant for all indexing terms
  - $P(k_i \mid \overline{R}_q) = \frac{n_i}{N}$: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 *D* documents

$$P(k_i \mid R_q) = \frac{D_i}{D}$$

$$P(k_i \mid \overline{R}_q) = \frac{n_i - D_i}{N - D}$$

D_i : the no. of documents in *D* that contain k_i

The Probabilistic Model (cont.)

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

$$P(k_i \mid R_q) = \frac{D_i + 0.5}{D + 1}$$

$$P(k_i \mid \overline{R}_q) = \frac{n_i - D_i + 0.5}{N - D + 1}$$

  - Or use the information of document frequency

$$P(k_i \mid R_q) = \frac{D_i + \frac{n_i}{N}}{D + 1}$$

$$P(k_i \mid \overline{R}_q) = \frac{n_i - D_i + \frac{n_i}{N}}{N - D + 1}$$

The Probabilistic Model (cont.)

- Advantages
  - Documents are ranked in decreasing order of probability of relevance (optimality)

- Disadvantages
  - Need to guess initial estimates for $P(k_i \mid R)$
  - Estimate the characteristics of the relevant class/set R through user-identified examples of relevant docs (without true training data)
  - All weights are binary: the method does not take into account *tf* and *idf* factors
  - Independence assumption of index terms
  - The lack of document length normalization

More advanced variations of the probabilistic models, such as the BM-25 model, correct these deficiencies to yield improved retrieval.

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*
  - This also seems to be the dominant thought among researchers and practitioners of IR
  - The vector model, whose weighting scheme is firmly grounded on information theory, provides a simple yet effective ranking formula for general collections