情報検索システム特論
Advanced Information Retrieval Systems
第2回 Lecture #2

2023-04-17

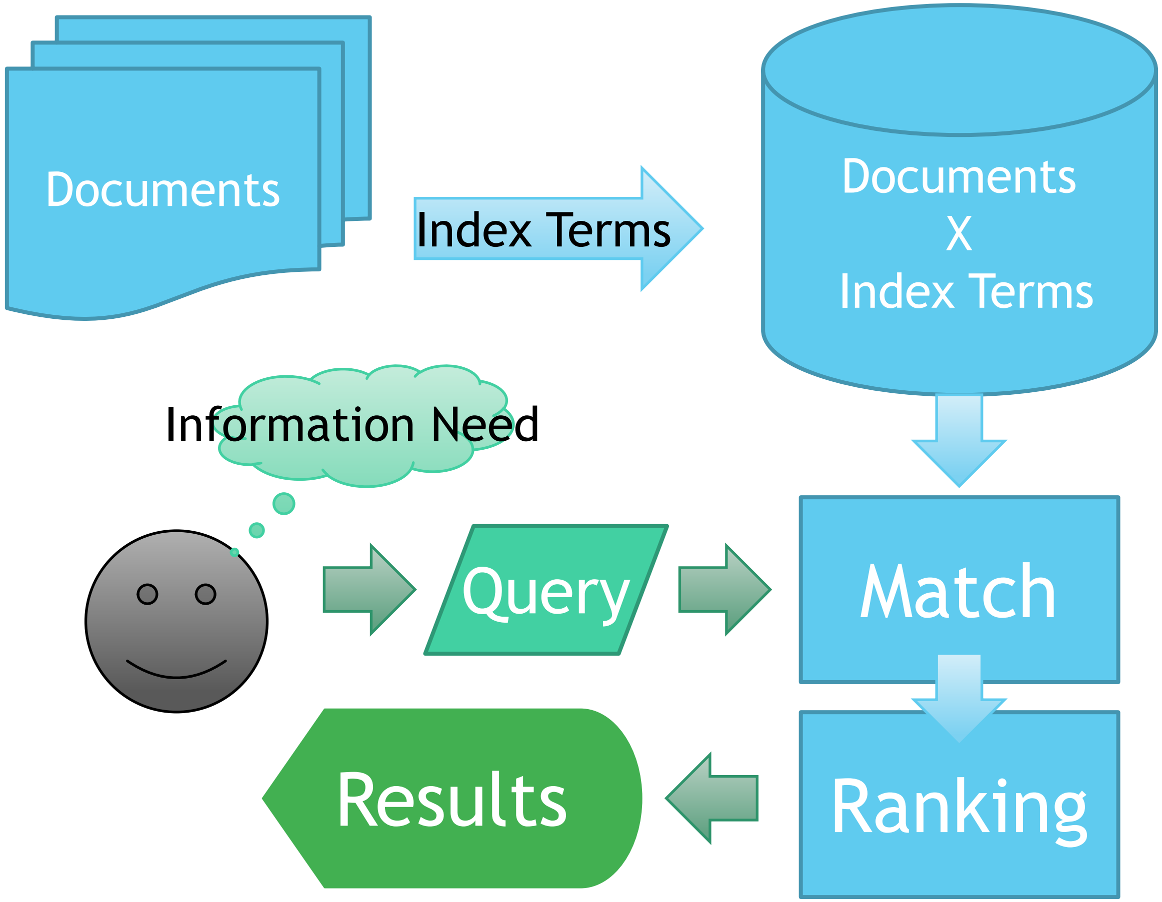
湯川 高志 Takashi Yukawa

Modeling

[Review] Motivation (1)

- IR systems usually adopt index terms to process queries
- Index term:
  - a keyword or group of selected words
  - any word (more general)
  - Stemming might be used
    - connect: connecting, connection, connections
- An inverted file is built for the chosen index terms

[Review] Motivation (2)



[Review] Motivation (3)

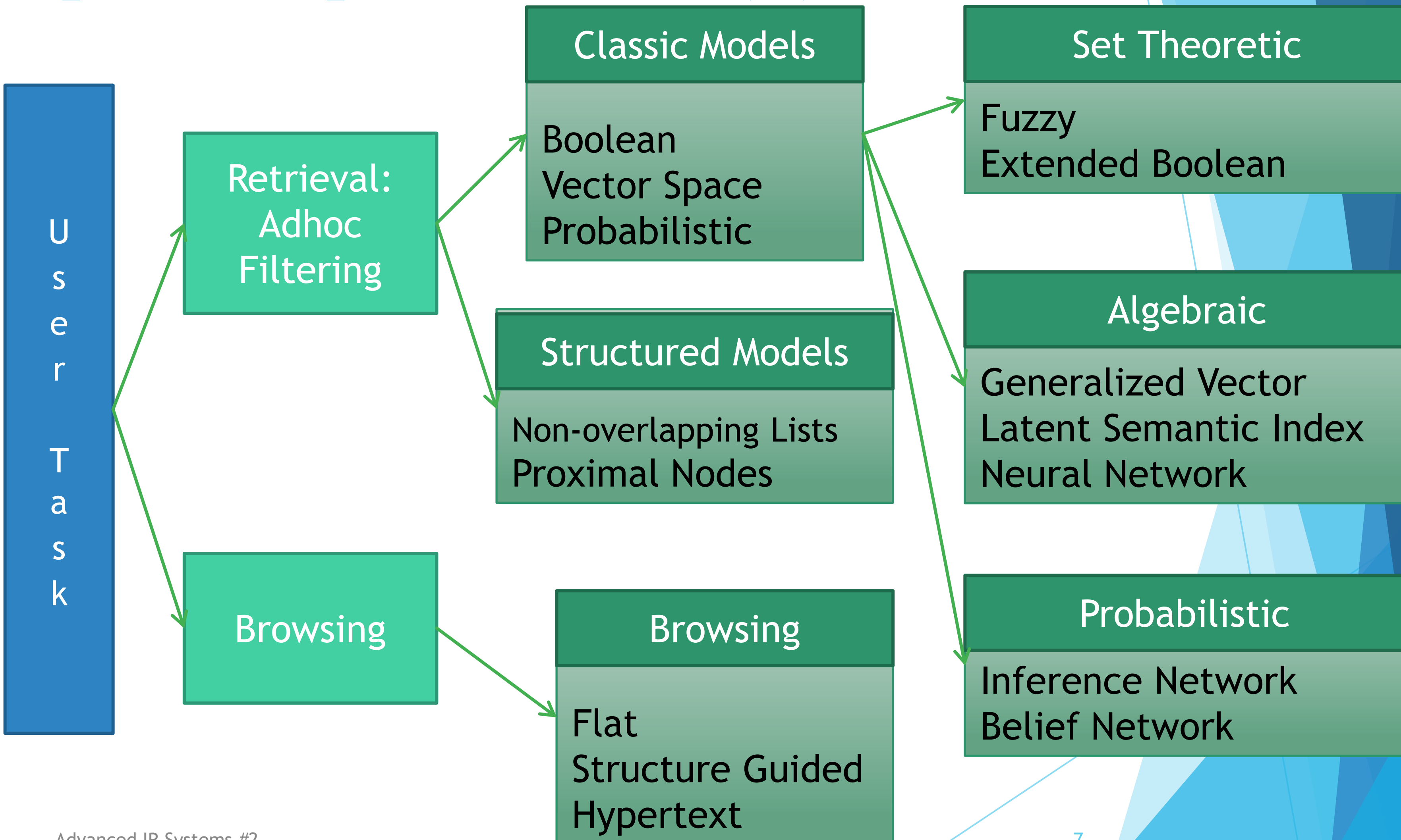
- Matching at index term level is quite imprecise
  → No surprise that users get frequently unsatisfied
- Most users have no training in query formation
  → Frequent dissatisfaction of Web users (problem is even worst)

- "Deciding relevance" is critical for IR systems
  → "ranking"

[Review] Motivation (4)

- A ranking
  - is an ordering of the documents retrieved that (hopefully) reflects the relevance of the documents to the user query
  - is based on fundamental premises regarding the notion of relevance, such as:
    - common sets of index terms
    - sharing of weighted terms
    - likelihood of relevance
- Each set of premises leads to a distinct IR model

[Review] IR models (1)



[Review] IR models (2)

Logical View of Documents

U
s
e
r

T
a
s
k

	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

Classic IR Models

Basic Concepts (1)

- Each document represented by a set of representative keywords or index terms
- An index term is a word useful for remembering the document main themes

- Index terms are nouns
  - because nouns have meaning by themselves

- Search engines assume that all words are index terms
  - full text representation

Basic Concepts (2)

- Not all terms are equally useful for representing the document contents
  - less frequent terms allow identifying a narrower set of documents

- To quantify the importance of an index term, we associate a weight with it

Basic Concepts (3)

- Index terms: $k_1, k_2, \ldots, k_t$
- A document: d_j
- A weight associated with (k_i, d_j), which quantifies the importance of k_i for describing the contents of d_j : $w_{i,j}$ (>0)
  - If a term k_i does not occur within d_j : $w_{i,j}$=0
- A weighted vector associated with d_j :

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

- a reference to the weight w_{ij} :

$$g_i(\vec{d}_j) = w_{i,j}$$

Classic IR Models

- Boolean Model
- Vector Space Model
- Probabilistic Model

The Boolean Model (1)

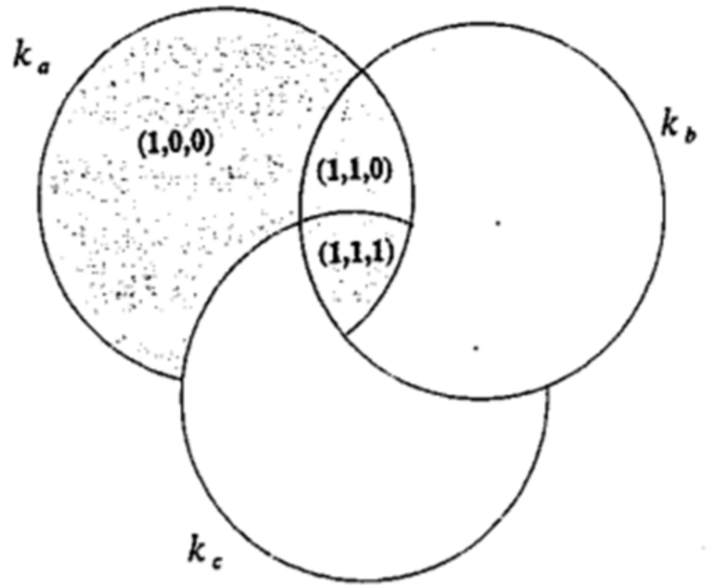
- Simple model based on set theory
- Queries specified as Boolean expressions
  - precise semantics
  - neat formalism
  - $q = k_a \wedge (k_b \vee \neg k_c)$
- Let,
  - w_{iq} : weight associated with pair (k_i, q)
  - $w_{iq} \in \{0,1\}$: terms either present or absent (Boolean)
  - $\overrightarrow{\boldsymbol{d}_q} = (w_{1q}, w_{2q}, \cdots w_{tq})$: weighted vector associated with q

The Boolean Model (2)

- An example of query and DNF
  - $q = k_a \wedge (k_b \vee \neg k_c)$
  - $\overrightarrow{d_q}$: weighted vector associated with q
  - $\mathrm{dnf}(\overrightarrow{d_q})$: disjunctive normal form (加法標準形) for vector $\overrightarrow{d_q}$
- DNF
  - $\mathrm{dnf}(\overrightarrow{d_q}) = (1,1,1) \vee (1,1,0) \vee (1,0,0)$
  - $cc_q \in \mathrm{dnf}(\overrightarrow{d_q})$: a conjunctive component for q

The Boolean Model (3)

- $q = k_a \wedge (k_b \vee \neg k_c)$



- Definition of relevance (similarity)
  - $sim(q, d_j) = 1$, if $\exists cc_q | \forall k_i, g_i(\vec{d_j}) = g_i(cc_q)$
  - $sim(q, d_j) = 0$, otherwise

Drawbacks of the Boolean Model

- Retrieval based on binary decision criteria with no notion of partial matching
- No ranking of the documents is provided (absence of a grading scale)
- 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

- The Boolean model frequently returns either too few or too many documents in response to a user query

The Vector Space Model (1)

- Use of binary weights is too limiting
- Non-binary weights provide consideration for partial matches

⬇

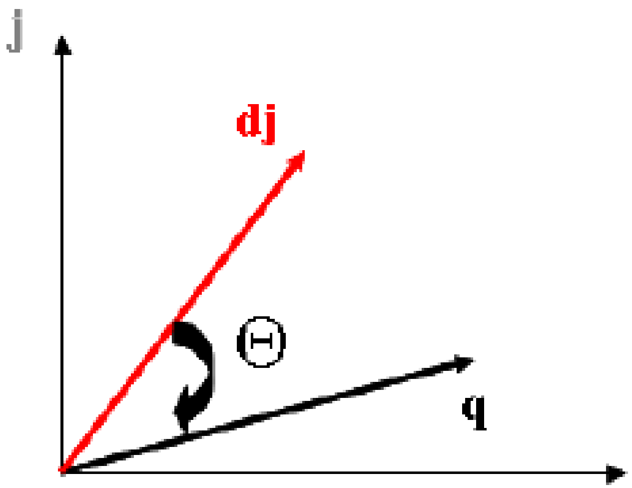
- Term weights are used to compute a degree of similarity between a query and each document
- Ranked set of documents provides for better matching

The Vector Space Model (2)

- Define:
  - $w_{ij} > 0$ whenever $k_i \in d_j$
  - $w_{iq} > 0$ associated with the pair (k_i, q)
  - $\overrightarrow{d_j} = (w_{1j}, w_{2j}, \cdots w_{tj})$
  - $\overrightarrow{d_q} = (w_{1q}, w_{2q}, \cdots w_{tq})$
  - To each term k_i is associated a unitary vector $\vec{\imath}$
  - The unitary vectors $\vec{\imath}$ and $\vec{\jmath}$ are assumed to be orthonormal
    - i.e., index terms are assumed to occur independently within the documents

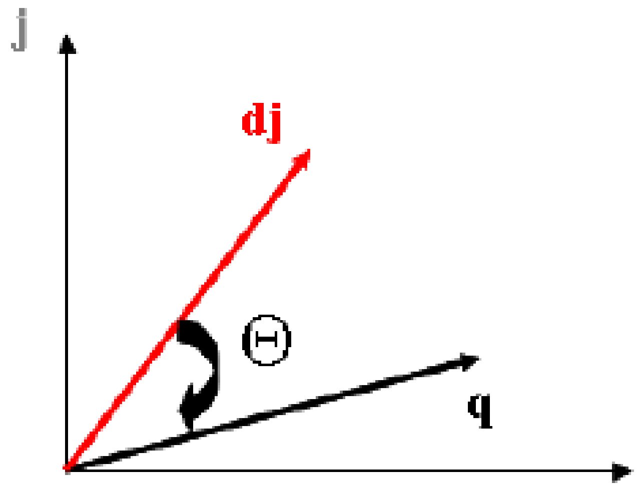
The Vector Space Model (3)

- The t unitary vectors $\vec{i}$ form an orthonormal basis for a t-dimensional space
- In this space, queries and documents are represented as weighted vectors



The Vector Space Model (4)

- Similarity
  - $\mathrm{sim}(d_j, q) = \cos(\theta)$
  - $= \dfrac{\overrightarrow{d_j} \cdot \vec{q}}{|\overrightarrow{d_j}| \times |\vec{q}|}$
  - $= \dfrac{\sum_{i=1}^{t} w_{ij} \times w_{iq}}{\sqrt{\sum_{i=1}^{t} {w_{ij}}^2} \times \sqrt{\sum_{i=1}^{t} {w_{iq}}^2}}$



- $0 \leq \mathrm{sim}(d_j, q) \leq 1$
- A document is retrieved even if it matches the query terms only partially

How to compute the weights (1)

- How should we 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
  - quantification of inter-documents separation (dissimilarity)
    - *idf* factor, the inverse document frequency
  - $w_{ij} = tf_{ij} \times idf_i$

How to compute the weights (2)

- Let, $k_i \in d_j$
  - the total number of documents in the collection: N
  - the number of documents which contain k_i: n_i
  - raw frequency of k_i within d_j: $\mathrm{freq}_{i,j}$
- A normalized *tf* factor
  - $$tf_{i,j} = \frac{freq_{i,j}}{max_l freq_{l,j}}$$
  - where the maximum is computed over all terms which occur within the document d_j

How to compute the weights (3)

- The *idf* factor
  - $idf_i = \log \frac{N}{n_i}$
- 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.

Advantages and Disadvantages

- 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 independence of index terms

solution

Latent Semantic Indexing
(to be explained later)

That's it today

今日はここまで