


Multimedia Information Systems

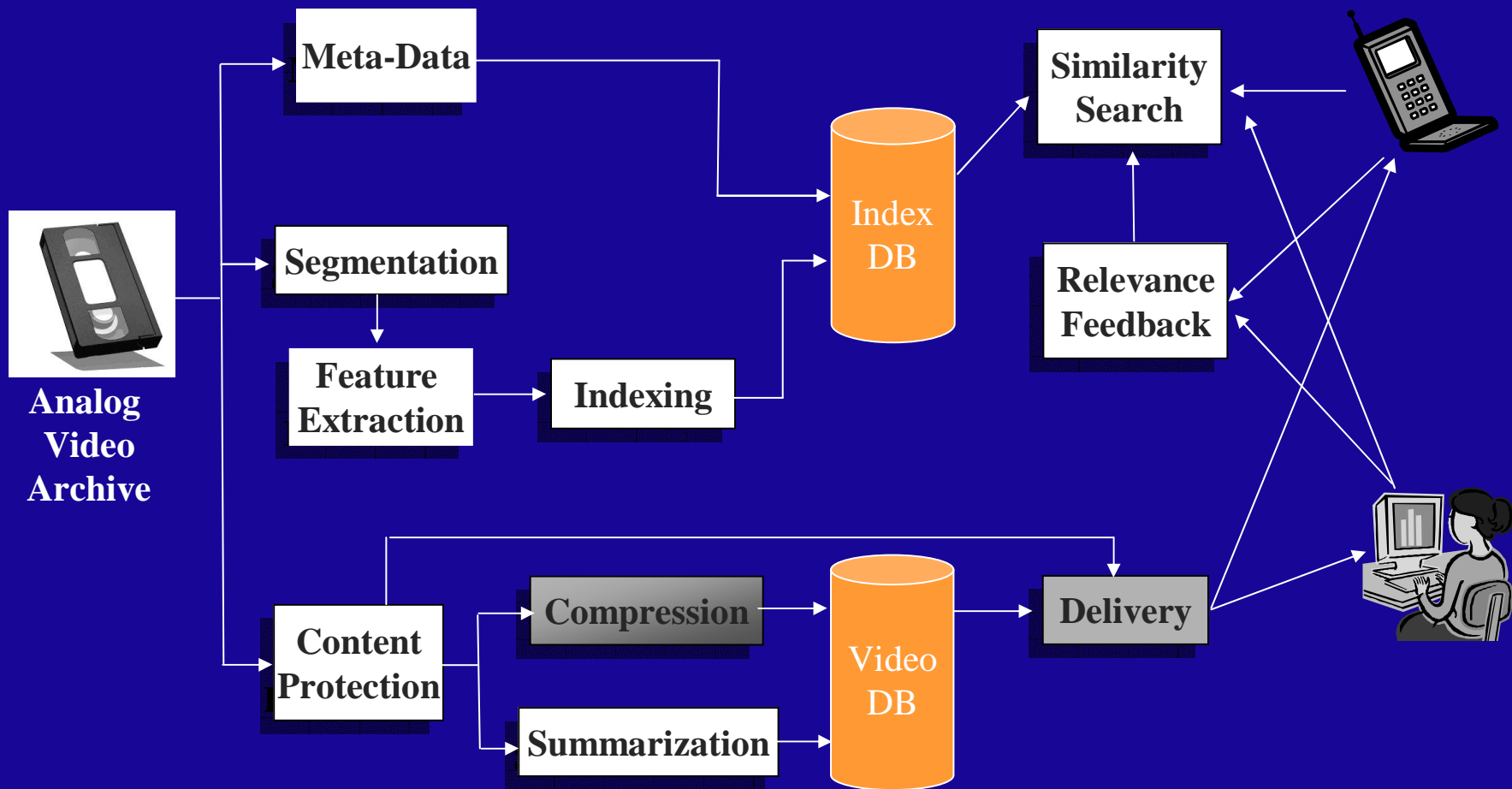


Samson Cheung

EE 639, Fall 2004

Lecture 6: Text Information Retrieval

Digital Video Library



Text Retrieval

- **Why?**
 - Longest history
 - Fundamentals of Information Retrieval

- **Where?**
 - Online library catalogs (Infokat)
 - Web search engines
 - Specialized systems
 - MEDLINE (medical)
 - Lexis-Nexis (legal, business, academic, ...)
 - Westlaw (legal articles)
 - Dialog (business information)

Typical Text Retrieval

- **Given:**
 - A corpus of textual natural-language documents
 - A user query in the form of a textual string

- **Find:**
 - A **ranked** set of documents that are **relevant** to the query.

Manual vs. Automatic Indexing

- **Pros**

- **Human judgments are most reliable**
- **Search controlled vocabularies is more efficient**

- **Cons**

- **Time consuming**
- **Require experts**
- **Incoherent classification**

Relevance

- **Relevance is a subjective judgment and may include:**
 - **Being on the proper subject**
 - **Being timely**
 - **Being authoritative**
 - **Satisfying the goals of the user and his/her intended use of the information**
- **Evaluation needs ground-truth**
 - **Small dataset**
 - **Pooling – first by multiple automatic methods, followed by manual examination**

Performance Evaluation

Given a query : •

■ Precision

relevant retrieved

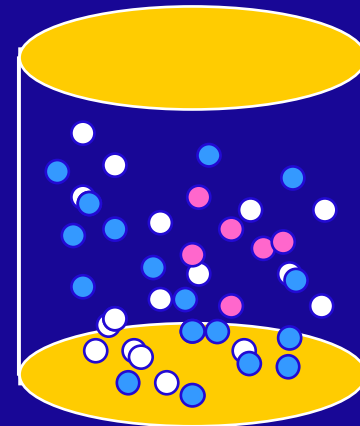
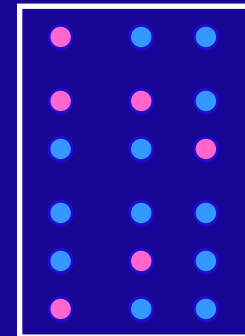
retrieved

■ Recall

relevant retrieved

relevant in collection

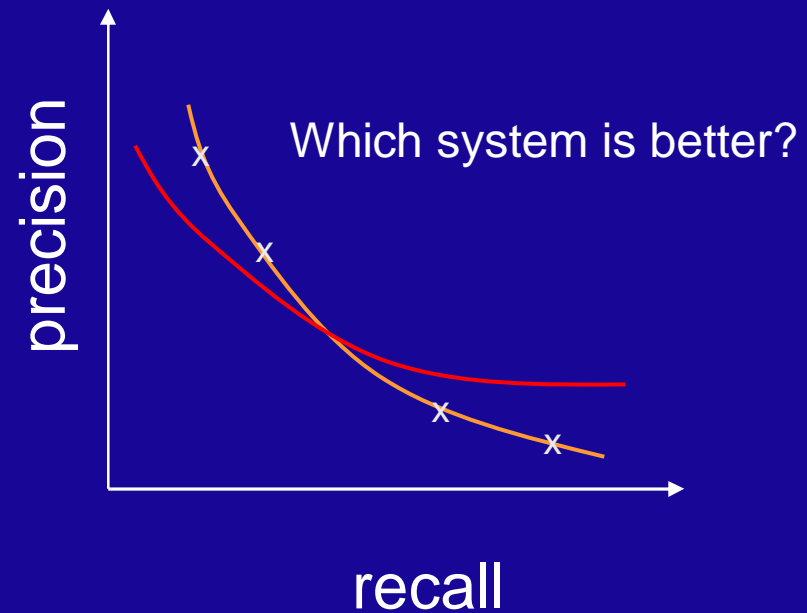
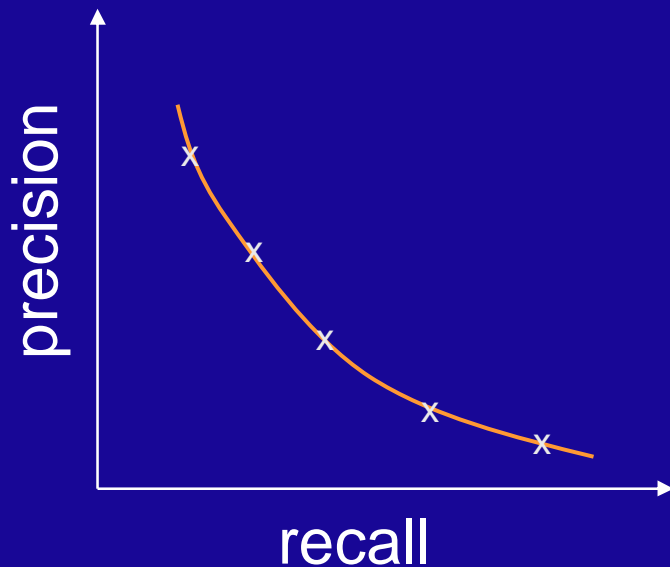
Retrieved Documents



Collection

Precision/Recall Curves

- There is a tradeoff between Precision and Recall
- Measure average precision over many queries at different levels of Recall



Automatic Content Representation

- Using natural language understanding?
 - Computationally too expensive in real-world settings
 - Coverage
 - Language dependence
 - The resulting representations may be too explicit to deal with the vagueness of a user's information need
- Alternative: a document is simply an unstructured set of words appearing in it: **bag of words**

Bag-of-Words Approach

- A document is an unordered list of words
 - Grammatical information is lost
- | Tokenization: What is a word?
 - Is 'White House' one or two words?
- Case folding
 - 'President Bush' becomes 'president' , 'bush'
- Stemming or lemmatization
 - Morphological information is thrown away:
'agreements' becomes 'agreement'
(lemmatization) or even 'agree' (stemming)

Example Bag of Words

Scientists have found compelling new evidence of possible ancient microscopic life on Mars, derived from magnetic crystals in a meteorite that fell to Earth from the red planet, NASA announced on Monday.

a, ancient, announced, compelling, crystals, derived, earth, evidence, fell, found, from (2), have, in, life, magnetic, mars, meteorite, microscopic, monday, nasa, new, of, on (2), planet, possible, red, scientists, that, the, to

What is this about?

added, al, an, and, ballots, been, completed, count, county (2), even, former, gore, ground, had, hand, have (2), he, if, in (2), independent, lost, many, miami-dade, might, new, not, of, president, presidential, requested, shows, study, that, the, vice, votes, would

An independent study shows former Vice President Al Gore would not have added many new votes in Miami-Dade County and might even have lost ground in that county, if the hand count of presidential ballots he requested had been completed.

Boolean Retrieval

- Boolean operators are: AND (NEAR), OR, NOT
- The semantics of the Boolean operators:
 - $t1 \text{ AND } t2 = \{d \mid t1 \in r(d)\} \cap \{d \mid t2 \in r(d)\}$
Documents whose representation contains $t1$ and $t2$
 - $t1 \text{ OR } t2 = \{d \mid t1 \in r(d)\} \cup \{d \mid t2 \in r(d)\}$
Documents whose representation contains $t1$ or $t2$
 - $\text{NOT } t1 = \{d \mid t1 \notin r(d)\}$
Documents whose representation doesn't contain $t1$

Searching the Collection

- Finding a word by linear search can be inefficient
- Solution: Construct a matrix which indexes documents and words
 - Matrix can be extremely large but will be sparse (Zipf's law)
 - Only a few words occur many times, and a lot of words occur only once
- Better: Construct an inverted index
 - A word points to the documents in which it occurs
 - This is an implementational (not a modeling) issue!

Pros and Cons of Boolean Retrieval

- Pros of Boolean Retrieval:
 - Clean and simple formalism
 - Firm grip on query formulation
- Cons of Boolean Retrieval:
 - Most non-experts cannot handle Boolean expressions, and query formulation may be time consuming
 - No ranking of retrieved documents
 - Exact matching may lead to too few or too many retrieved documents
- Alternatives:
 - Vector space retrieval
 - Probabilistic retrieval

Vector Space Retrieval

- By far the most common modern retrieval system
- Features:
 - Users can enter free text
 - Documents are ranked
 - Relaxation of the matching criterion
- Key idea: Everything (documents, queries, terms) is a vector in a high-dimensional space
- Example system: SMART, developed by Salton and students at Cornell in the 1960s; still in use.

Vector Space Representation

- Documents are vectors of terms
- Terms are vectors of documents
- Similarly, a query is a vector terms

	t_1	t_2	t_3	t_4	...
d_1	1	0	0	1	...
d_2	0	1	0	1	...
d_3	0	0	1	1	...
d_4	1	1	1	0	...
i	\vdots	\vdots	\vdots	\vdots	\vdots

Similarity in the Vector Space

- Given a query vector q and a document vector d , both of length n similarity between q and d is defined as:

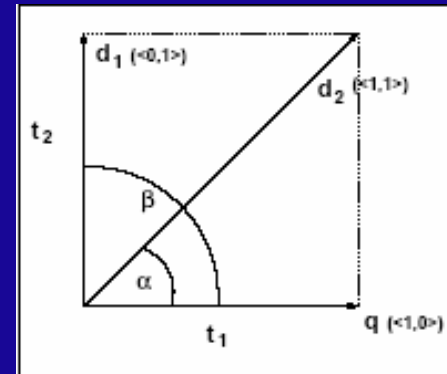
- The inner product of q and d : $q \cdot d$, where

$$q \cdot d = \sum_{i=1}^n (q_i \cdot d_i)$$

- and $q_i(d_i)$ is value of the i -th position of q (d)
- With binary values, this amounts to counting the matching terms between q and d

Similarity from a Different Angle

- The similarity between a query and a document can be measured as the angle between their representations in vector space



- Ranging between:
 - 90 degrees: No similarity (orthogonal)
 - 0 degrees: Maximum similarity (identical)
- d_2 is more similar to q than d_1 since the angle α is smaller than β

Cosine Similarity

- Let a query vector q and a document vector d , both of length n , be given.
- The the cosine similarity is defined as

$$\begin{aligned}\text{sim}(q, d) &= \frac{\sum_{i=1}^n (q_i \cdot d_i)}{\sqrt{\sum_{i=1}^n q_i^2} \cdot \sqrt{\sum_{i=1}^n d_i^2}} \\ &= (\|q\|^{-1} \cdot q) \bullet (\|d\|^{-1} \cdot d)\end{aligned}$$

Term Weights

- Up to now, we've only considered binary term weights
 - 1: a term occurs in the document
 - 0: it doesn't occur in the document
- There are two shortcomings:
 - It doesn't reflect the frequency of terms
 - All terms are equally important (cf. president vs. the)
- Term Weights
 - Store the frequency in the vector (e.g. 4) instead of 1
 - **tf-score**
 - `Meaningful terms' occur only in a few documents, they are good discriminators that distinguish those documents from the rest)
 - **idf-score**

TF.IDF-Score

- tf-score: $tf_{i;j}$ = frequency of term i in document j
- idf-score: $idf_i = \log(N/n_i)$, where
 - N is the size of the collection (no. documents)
 - n_i is the number of documents in which term i occurs
 - the logarithm is used for dampening
- The term weight of term i in document j is then computed as:
 - $tf_{i;j} \cdot idf_i$