Information Retrieval
Part 1

Günter Neumann

LT lab, DFKI

(Using slides from Raymond Mooney's IR course
http://www.cs.utexas.edu/users/mooney/ir-course/)

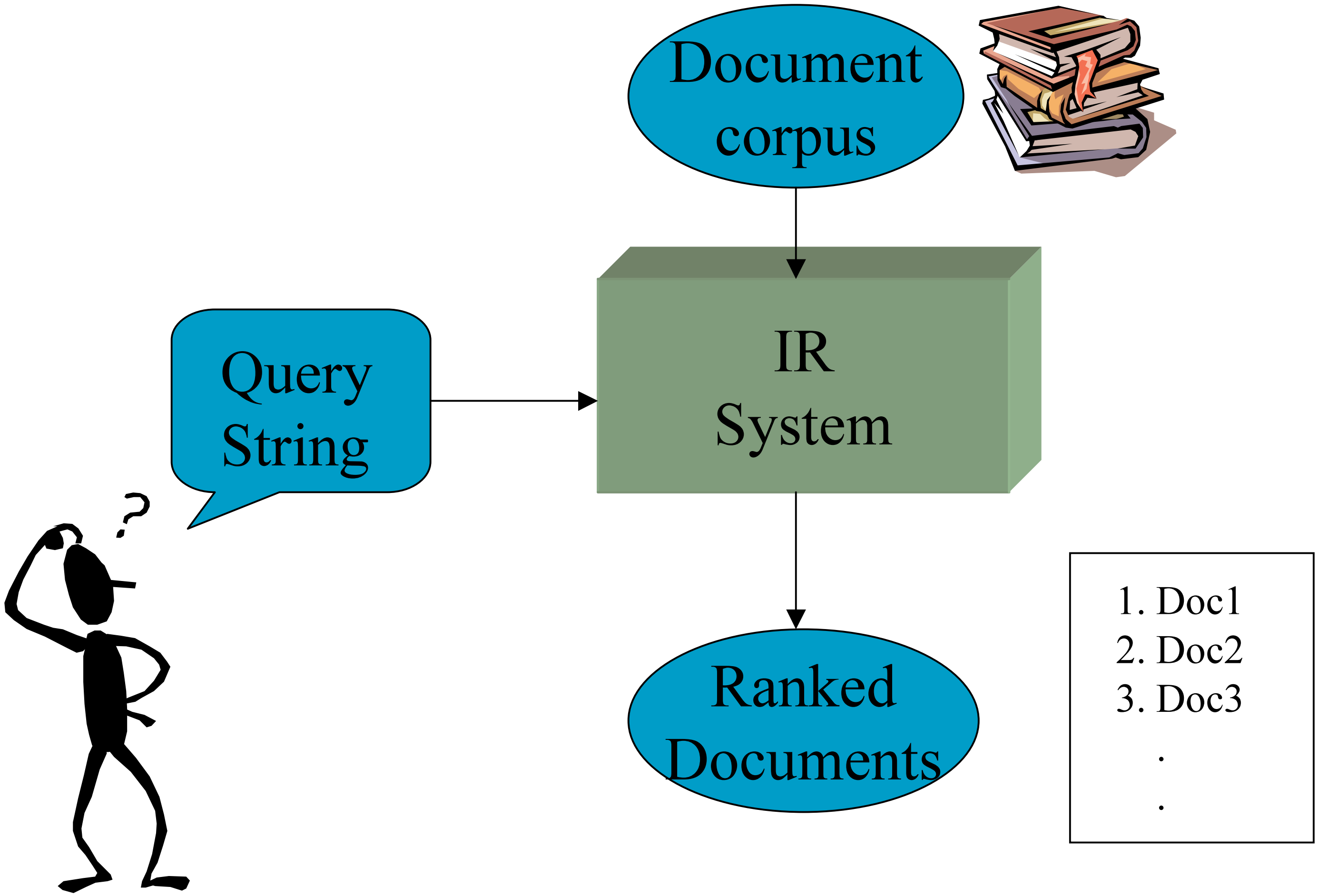
Information Retrieval (IR)

- The indexing and retrieval of textual documents.
- Searching for pages on the World Wide Web is the most recent “killer app.”
- Concerned firstly with retrieving *relevant* documents to a query.
- Concerned secondly with retrieving from *large* sets of documents *efficiently*.

Typical IR Task

- 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.

IR System


Document corpus
Query String
IR System
Ranked Documents
1. Doc1
2. Doc2
3. Doc3
.
.

Relevance

- Relevance is a subjective judgment and may include:
  - Being on the proper subject.
  - Being timely (recent information).
  - Being authoritative (from a trusted source).
  - Satisfying the goals of the user and his/her intended use of the information (*information need*).

Keyword Search

- Simplest notion of relevance is that the query string appears verbatim in the document.
- Slightly less strict notion is that the words in the query appear frequently in the document, in any order (*bag of words*).
  - Documents have to be *about* the query terms

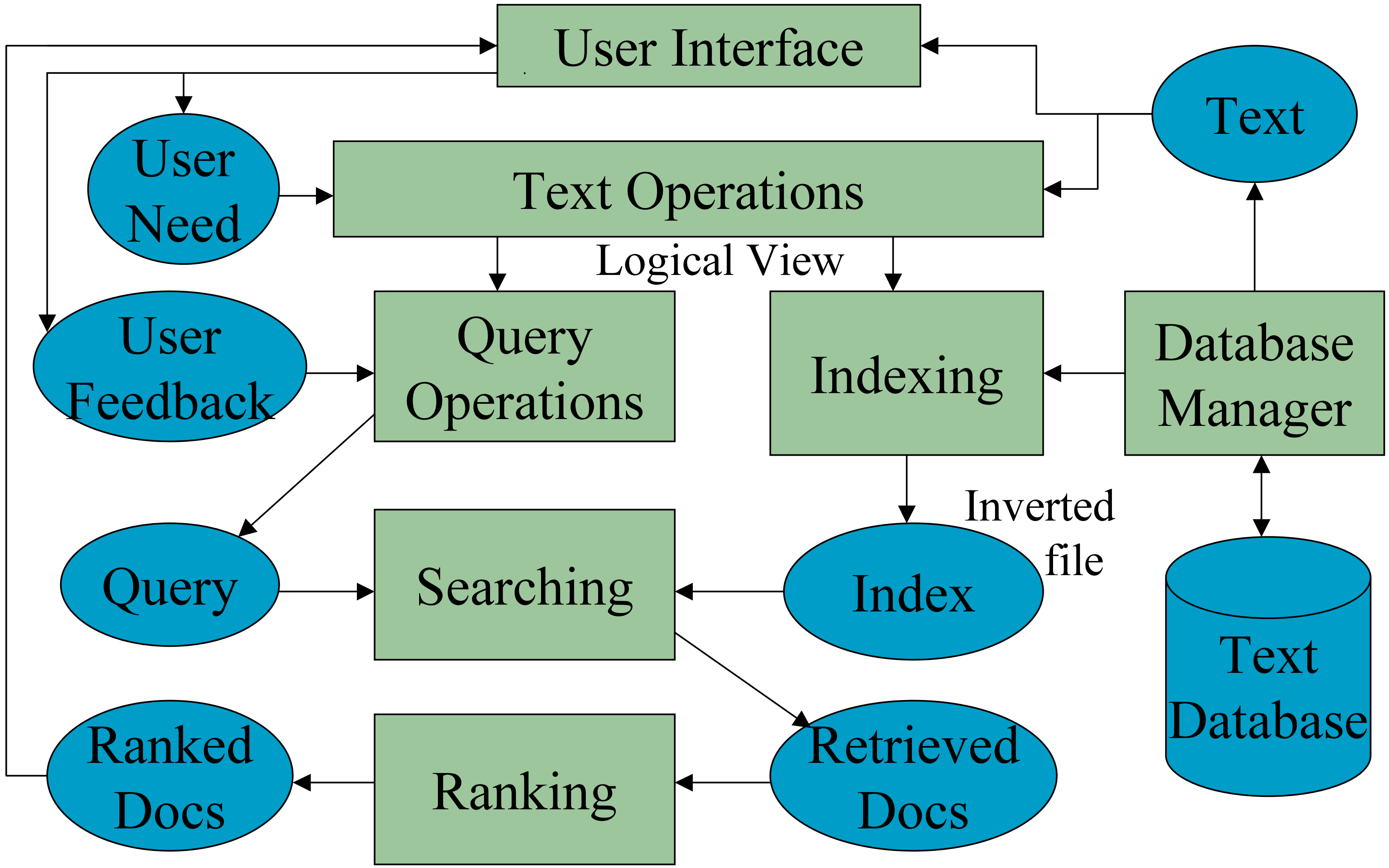
Problems with Keywords

- May not retrieve relevant documents that include synonymous terms.
  - "restaurant" vs. "café"
  - "PRC" vs. "China"
- May retrieve irrelevant documents that include ambiguous terms.
  - "bat" (baseball vs. mammal)
  - "Apple" (company vs. fruit)
  - "bit" (unit of data vs. act of eating)

Intelligent IR

- Taking into account the *meaning* of the words used.
- Taking into account the *order* of words in the query.
- *Adapting* to the user based on direct or indirect feedback.
- Taking into account the *authority* of the source.

IR System Architecture



IR System Components

- Text Operations forms index words (tokens/terms).
  - Stopword removal
  - Stemming
- Indexing constructs an *inverted index* of word to document pointers.
- Searching retrieves documents that contain a given query token from the inverted index.
- Ranking scores all retrieved documents according to a relevance metric.

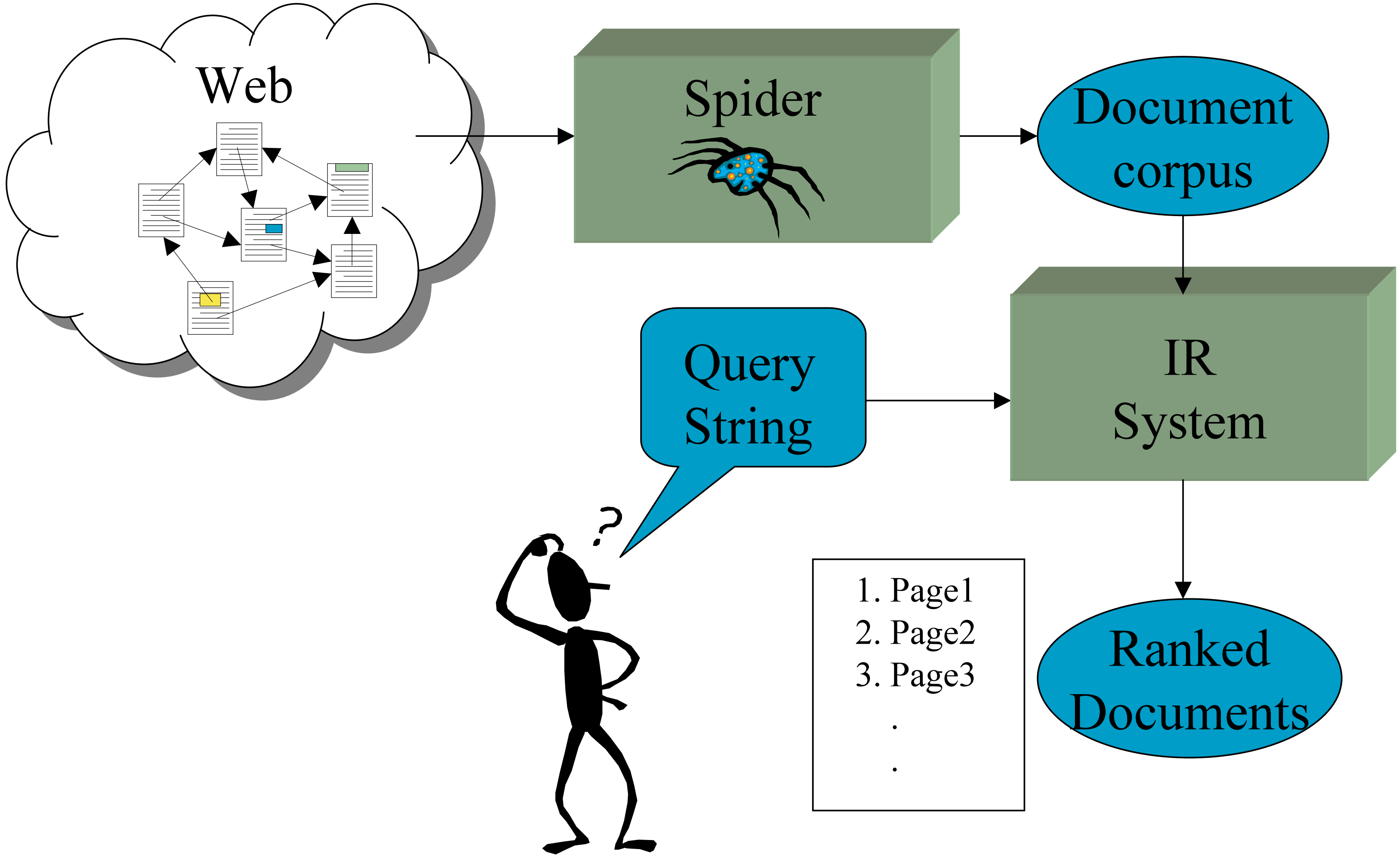
IR System Components (continued)

- User Interface manages interaction with the user:
  - Query input and document output.
  - Relevance feedback.
  - Visualization of results.
- Query Operations transform the query to improve retrieval:
  - Query expansion using a thesaurus.
    - Query transformation using relevance feedback.

Web Search

- Application of IR to HTML documents on the World Wide Web.
- Differences:
  - Must assemble document corpus by spidering the web.
  - Can exploit the structural layout information in HTML (XML).
  - Documents change uncontrollably.
  - Can exploit the link structure of the web.

Web Search System



Recent IR History

- 2000's
  - Link analysis for Web Search
    - Google
  - Automated Information Extraction
    - Whizbang
    - Fetch
    - Burning Glass
  - Question Answering
    - TREC Q/A track
    - Clef multilingual QA track

Recent IR History

- 2000’s continued:
  - Multimedia IR
    - Image
    - Video
    - Audio and music
  - Cross-Language IR
    - DARPA Tides
  - Document Summarization
    - DUC

Related Areas

- Database Management
- Library and Information Science
- Artificial Intelligence
- Natural Language Processing
- Machine Learning

Database Management

- Focused on *structured* data stored in relational tables rather than free-form text.
- Focused on efficient processing of well-defined queries in a formal language (SQL).
- Clearer semantics for both data and queries.
- Recent move towards *semi-structured* data (XML) brings it closer to IR.

Library and Information Science

- Focused on the human user aspects of information retrieval (human-computer interaction, user interface, visualization).
- Concerned with effective categorization of human knowledge.
- Concerned with citation analysis and *bibliometrics* (structure of information).
- Recent work on *digital libraries* brings it closer to CS & IR.
  - http://citeseer.ist.psu.edu/ CiteSeer.IST Scientific Literature Digital Library

Artificial Intelligence

- Focused on the representation of knowledge, reasoning, and intelligent action.
- Formalisms for representing knowledge and queries:
  - First-order Predicate Logic
  - Bayesian Networks
- Recent work on web ontologies and intelligent information agents brings it closer to IR.
  - Semantic Web

Natural Language Processing

- Focused on the syntactic, semantic, and pragmatic analysis of natural language text and discourse.
- Ability to analyze syntax (phrase structure) and semantics could allow retrieval based on *meaning* rather than keywords.

Natural Language Processing: IR Directions

- Methods for determining the sense of an ambiguous word based on context (*word sense disambiguation*).
- Methods for identifying specific pieces of information in a document (*information extraction*).
- Methods for answering specific NL questions from document corpora (*open domain QA*)

Machine Learning

- Focused on the development of computational systems that improve their performance with experience.
- Automated classification of examples based on learning concepts from labeled training examples (*supervised learning*).
- Automated methods for clustering unlabeled examples into meaningful groups (*unsupervised learning*).

Machine Learning: IR Directions

- Text Categorization
  - Automatic hierarchical classification (Yahoo).
  - Adaptive filtering/routing/recommending.
  - Automated spam filtering.
- Text Clustering
  - Clustering of IR query results.
  - Automatic formation of hierarchies (Yahoo).
- Learning for Information Extraction
- Text Mining

Topics to be covered in the next slides

- Vector space model
- Text processing aspects
- Evaluation (part 2)
- Concept-based IR (part 2)

This can only give an overview

Issues for Vector Space Model

- How to determine important words in a document?
  - Word sense?
  - Word n-grams (and phrases, idioms,…) → terms
- How to determine the degree of importance of a term within a document and within the entire collection?
- How to determine the degree of similarity between a document and the query?
- In the case of the web, what is a collection and what are the effects of links, formatting information, etc.?

The Vector-Space Model

- Assume *t* distinct terms remain after preprocessing; call them index terms or the vocabulary.
- These "orthogonal" terms form a vector space.

  Dimension = *t* = |vocabulary|

- Each term, *i*, in a document or query, *j*, is given a real-valued weight, w_{ij}.
- Both documents and queries are expressed as t-dimensional vectors:

  $d_j = (w_{1j}, w_{2j}, \ldots, w_{tj})$

Document Collection

- A collection of *n* documents can be represented in the vector space model by a term-document matrix.
- An entry in the matrix corresponds to the "weight" of a term in the document; zero means the term has no significance in the document or it simply doesn't exist in the document.

$$
\begin{pmatrix}
 & T_1 & T_2 & \cdots & T_t \\
D_1 & w_{11} & w_{21} & \cdots & w_{t1} \\
D_2 & w_{12} & w_{22} & \cdots & w_{t2} \\
\vdots & \vdots & \vdots & & \vdots \\
\vdots & \vdots & \vdots & & \vdots \\
D_n & w_{1n} & w_{2n} & \cdots & w_{tn}
\end{pmatrix}
$$

Term Weights: Term Frequency

- More frequent terms in a document are more important, i.e. more indicative of the topic.

  f_{ij} = frequency of term *i* in document *j*

- May want to normalize *term frequency* (*tf*) across the entire corpus:

  $$tf_{ij} = f_{ij} \; / \; max\{f_{ij}\}$$

Term Weights: Inverse Document Frequency

- Terms that appear in many *different* documents are *less* indicative of overall topic.

  df_i = document frequency of term i

  = number of documents containing term i

  idf_i = inverse document frequency of term i,

  $= \log_2(N/\, df_i)$

  (N: total number of documents)

- An indication of a term's *discrimination* power.
- Log used to dampen the effect relative to *tf*.

TF-IDF Weighting

- A typical combined term importance indicator is *tf-idf weighting*:

$$w_{ij} = tf_{ij}\, idf_i = tf_{ij} \log_2 (N/ df_i)$$

- A term occurring frequently in the document but rarely in the rest of the collection is given high weight.
- Many other ways of determining term weights have been proposed.
- Experimentally, *tf-idf* has been found to work well.

Computing TF-IDF - An Example

Given a document containing terms with given frequencies:

A(3), B(2), C(1)

Assume collection contains 10,000 documents and

document frequencies of these terms are:

A(50), B(1300), C(250)

Then:

A: tf = 3/3; idf = log(10000/50) = 5.3; tf-idf = 5.3

B: tf = 2/3; idf = log(10000/1300) = 2.0; tf-idf = 1.3

C: tf = 1/3; idf = log(10000/250) = 3.7; tf-idf = 1.2

Similarity Measure

- A similarity measure is a function that computes the *degree of similarity* between two vectors.

- Using a similarity measure between the query and each document:
  - It is possible to rank the retrieved documents in the order of presumed relevance.
  - It is possible to enforce a certain threshold so that the size of the retrieved set can be controlled.

Similarity Measure - Inner Product

- Similarity between vectors for the document $\boldsymbol{d}_i$ and query $\boldsymbol{q}$ can be computed as the vector inner product:

$$\text{sim}(\boldsymbol{d}_j, \boldsymbol{q}) = \sum_{i=1}^{t} \boldsymbol{d}_j \bullet \boldsymbol{q} = w_{ij} \cdot w_{iq}$$

where w_{ij} is the weight of term i in document j and w_{iq} is the weight of term i in the query

- For binary vectors, the inner product is the number of matched query terms in the document (size of intersection).
- For weighted term vectors, it is the sum of the products of the weights of the matched terms.

Properties of Inner Product

- Favors long documents with a large number of unique terms.
- Measures how many terms matched but not how many terms are *not* matched.

Inner Product -- Examples

Binary:

	retrieval	database	architecture	computer	text	management	information
– D =	1,	1,	1,	0,	1,	1,	0
– Q =	1,	0 ,	1,	0,	0,	1,	1

Size of vector = size of vocabulary = 7
0 means corresponding term not found in document or query

sim(D, Q) = 3

Weighted:

$D_1 = 2T_1 + 3T_2 + 5T_3$ $\quad D_2 = 3T_1 + 7T_2 + 1T_3$

$Q = 0T_1 + 0T_2 + 2T_3$

sim(D_1 , Q) = 2*0 + 3*0 + 5*2 = 10

sim(D_2 , Q) = 3*0 + 7*0 + 1*2 = 2

Comments on Vector Space Models

- Simple, mathematically based approach.
- Considers both local (*tf*) and global (*idf*) word occurrence frequencies.
- Provides partial matching and ranked results.
- Tends to work quite well in practice despite obvious weaknesses.
- Allows efficient implementation for large document collections.

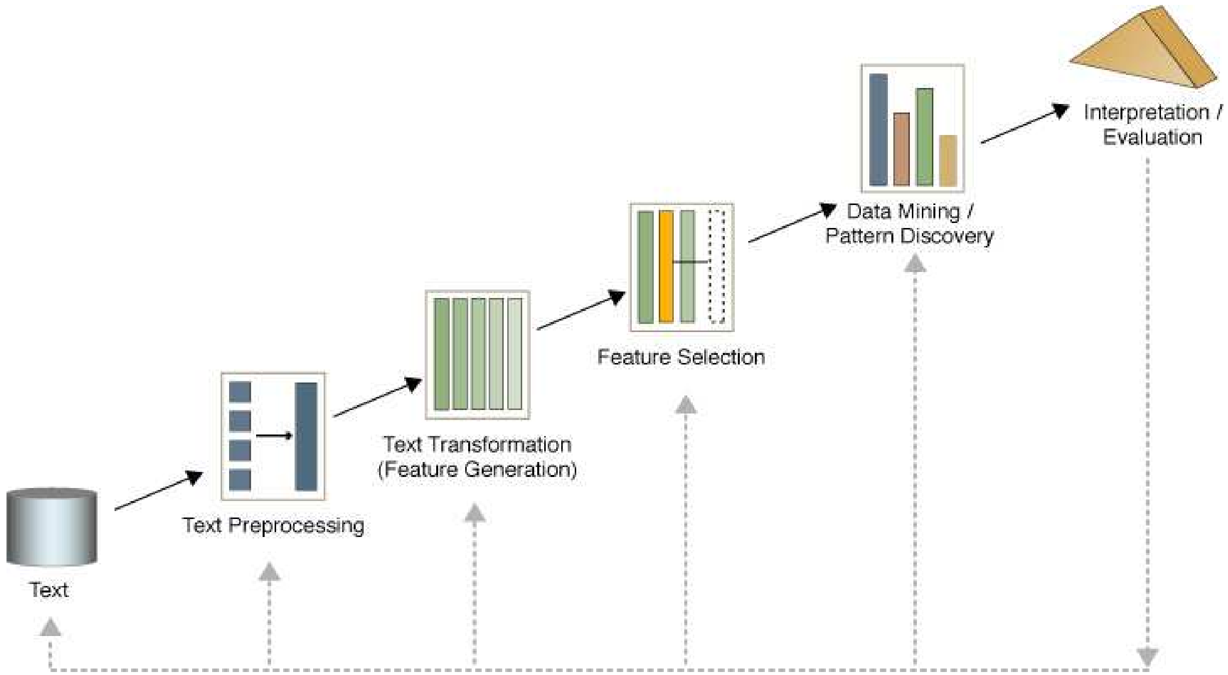
Problems with Vector Space Model

- Missing semantic information (e.g. word sense).
- Missing syntactic information (e.g. phrase structure, word order, proximity information).
- Assumption of term independence (e.g. ignores synonomy).
- Lacks the control of a Boolean model (e.g., *requiring* a term to appear in a document).
  - Given a two-term query "A B", may prefer a document containing A frequently but not B, over a document that contains both A and B, but both less frequently.

Text Processing Aspects

Text Processing Aspect

- Text Preprocessing
  - Syntactic/Semantic Text Analysis
- Features Generation
  - Bag of Words
- Feature Selection
  - Simple Counting
  - Statistics
- Text/Data Mining
  - Classification- Supervised Learning
  - Clustering- Unsupervised Learning
- Analyzing Results



Text Characteristics (1)

- Large textual database
  - Web is growing
  - Publications are electronic (e.g., PubMed with > 11 Million articles)
- High dimensionality
  - Consider each word/phrase as a dimension
- Dependency
  - Relevant information is a complex conjunction of words/phrases
    - e.g., Document categorization and Pronoun disambiguation
- Ambiguity
  - Word ambiguity
    - Pronouns (he, she …)
    - Synonyms (buy, purchase)
    - Words with multiple meanings (bat – it is related to baseball or mammal)
  - Semantic ambiguity
    - The king saw the rabbit with his glasses. (multiple meanings)

Text Characteristics (2)

- Noisy data
  - Spelling mistakes
  - Abbreviations
  - Acronyms
- Not well structured text
  - Email/Chat rooms
    - “r u available ?”
    - “Hey whazzzzzz up”
  - Speech

Text Characteristics (3)

- Order of words in the query
  - hot dog stand in the amusement park
  - hot amusement stand in the dog park
- User dependency for the data
  - direct feedback
  - indirect feedback
- Authority of the source
  - IBM is more likely to be an authorized source then my second far cousin

German Language

- Free word order
  - No fix order of phrases in a sentence
- Rich morphology
  - Inflection
  - derivation
- Compounds
  - Very productive
- Complex verb groups
  - Den Bericht rechtzeitig zu schreiben geglaubt zu haben.
  - Der Termin findet ….. Statt.
- Challenge for finite-state approaches of shallow parsing

Text PreProcessing: Syntactic / Semantic Text Analysis

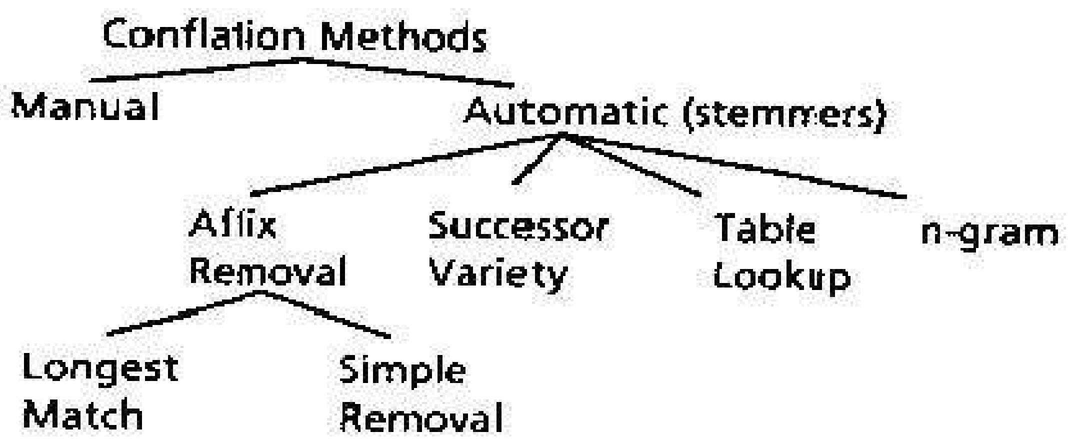
- Part Of Speech (PoS) Tagging
  - Find the corresponding PoS for each word
  - e.g., John (noun) gave (verb) the (det) ball (noun)
  - ~98% accurate
- Word Sense Disambiguation
  - Context based or proximity based
  - Very accurate
- Parsing
  - Generates a parse tree (graph) for each sentence
  - Each sentence is a stand alone graph

Feature Generation: Bag of Words

- Text document is represented by the words it contains (and their occurrences)
  - e.g., “Lord of the rings” $\rightarrow$ {“the”, “Lord”, “rings”, “of”}
  - Highly efficient
  - Makes learning far simpler and easier
  - Order of words is not that important for certain applications
- Stemming
  - Reduce dimensionality
  - Identifies a word by its root
  - e.g., flying, flew $\rightarrow$ fly
- Stop words
  - Identifies the most common words that are unlikely to help with text mining
  - e.g., “the”, “a”, “an”, “you”

Stemming

- Stemming is one technique to provide ways of finding morphological variants of search terms.
- Used to improve retrieval effectiveness and to reduce the size of indexing files.
- Taxonomy for stemming algorithms



Stemming (con’t)

- Criteria for judging stemmers
  - Correctness
    - Overstemming: too much of a term is removed.
    - Understemming: too little of a term is removed.
  - Retrieval effectiveness

    measured with recall and precision, and on their speed, size, and so on
  - compression performance

Type of stemming algorithms

- Table lookup approach
- Successor Variety
- n-gram stemmers
- Affix Removal Stemmers

Porter Stemmer

- Simple procedure for removing known affixes in English without using a dictionary.
- Can produce unusual stems that are not English words:
  - "computer", "computational", "computation" all reduced to same token "comput"
- May conflate (reduce to the same token) words that are actually distinct.
- Not recognize all morphological derivations.

Porter Stemmer Errors

- Errors of “comission”:
  - organization, organ $\rightarrow$ organ
  - police, policy $\rightarrow$ polic
  - arm, army $\rightarrow$ arm
- Errors of “omission”:
  - cylinder, cylindrical
  - create, creation
  - Europe, European

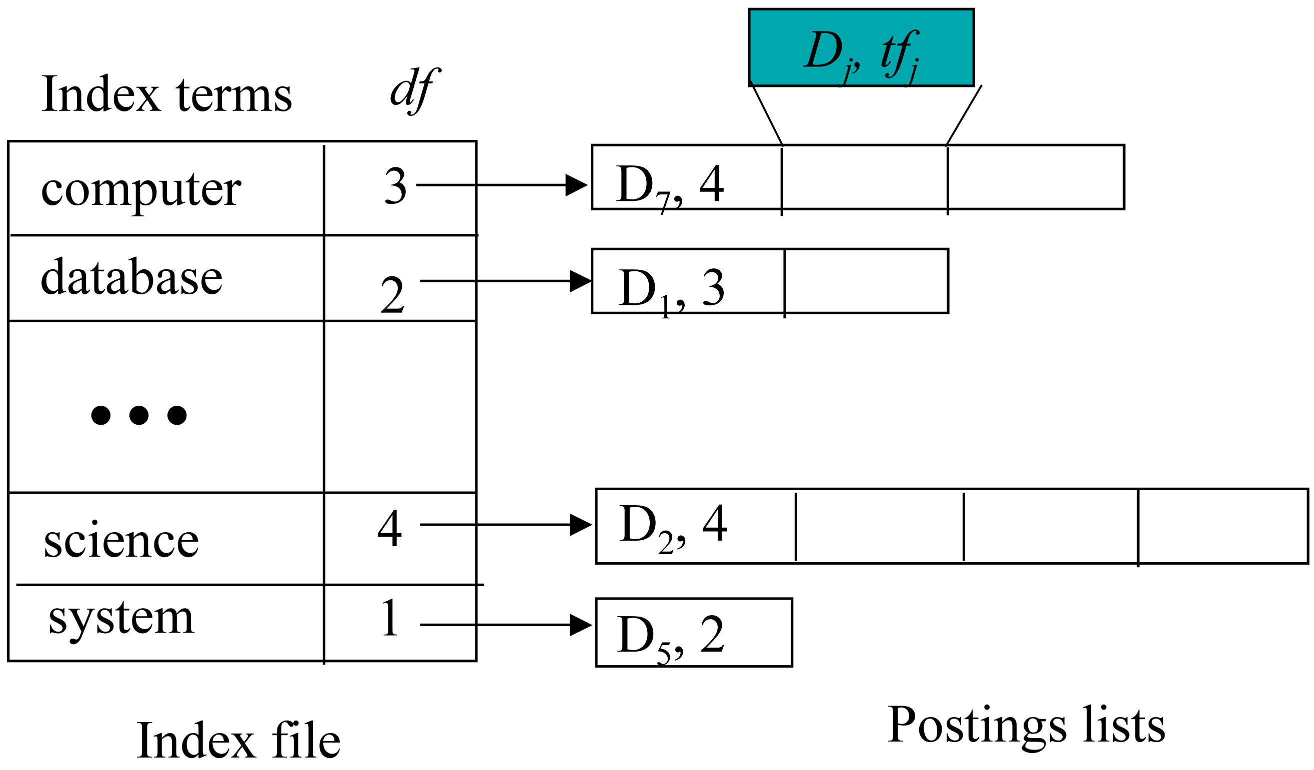
Sparse Vectors

- Vocabulary and therefore dimensionality of vectors can be very large, ~10^4 .
- However, most documents and queries do not contain most words, so vectors are sparse (i.e. most entries are 0).
- Need efficient methods for storing and computing with sparse vectors.

Implementation Based on Inverted Files

- In practice, document vectors are not stored directly; an inverted organization provides much better efficiency.
- The keyword-to-document index can be implemented as a hash table, a sorted array, or a tree-based data structure (trie, B-tree).
- Critical issue is logarithmic or constant-time access to token information.

Inverted Index

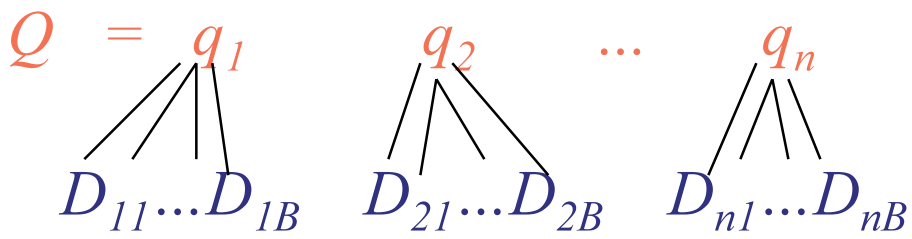


Retrieval with an Inverted Index

- Tokens that are not in both the query and the document do not effect cosine similarity.
  - Product of token weights is zero and does not contribute to the dot product.
- Usually the query is fairly short, and therefore its vector is *extremely* sparse.
- Use inverted index to find the limited set of documents that contain at least one of the query words.

Inverted Query Retrieval Efficiency

- Assume that, on average, a query word appears in *B* documents:



- Then retrieval time is O(|*Q*| *B*), which is typically, **much** better than naïve retrieval that examines all *N* documents, O(|*V*| *N*), because |*Q*| << |*V*| and *B* << *N*.