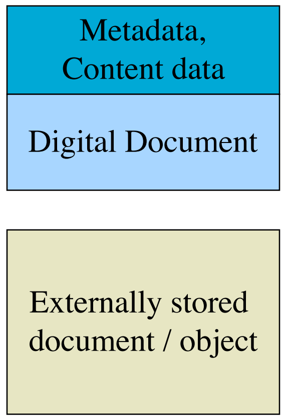
INFSCI 2140
Information Storage and Retrieval
Lecture 5: Text Analysis

Peter Brusilovsky
http://www2.sis.pitt.edu/~peterb/2140-051/

Overview

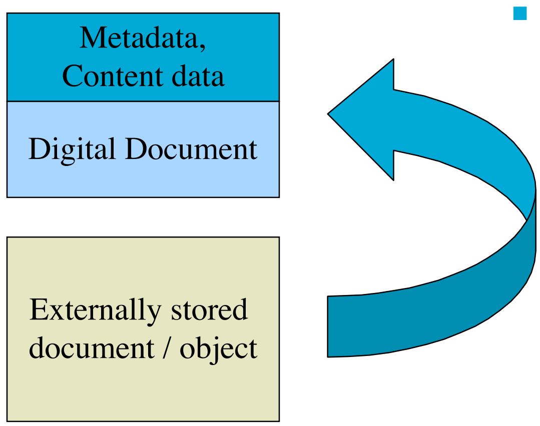
- Large picture: document processing, storage, search
- Indexing
- Term significance and term weighting
  - Zipf's law, TF*IDF, Signal to Noise Ratio
- Document similarity
- Processing: stop lists and stemming
- Other problems of text analysis

Documents and Surrogates



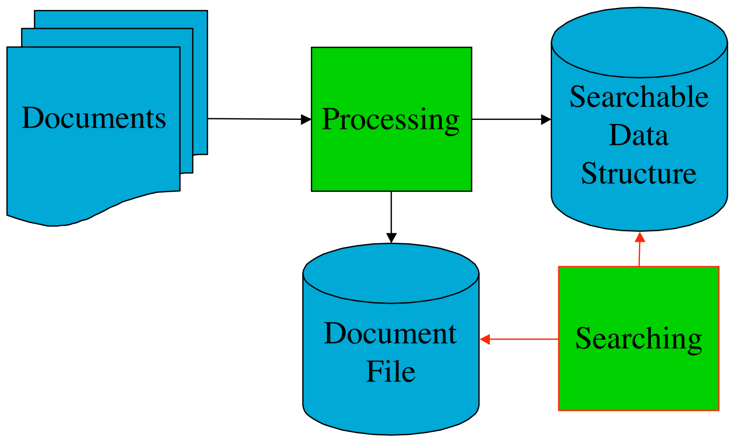
- Digitally stored, used for search, presentation, and selection
- Digitally stored, used for presentation and selection, not used for search
- Externally stored, not used for search

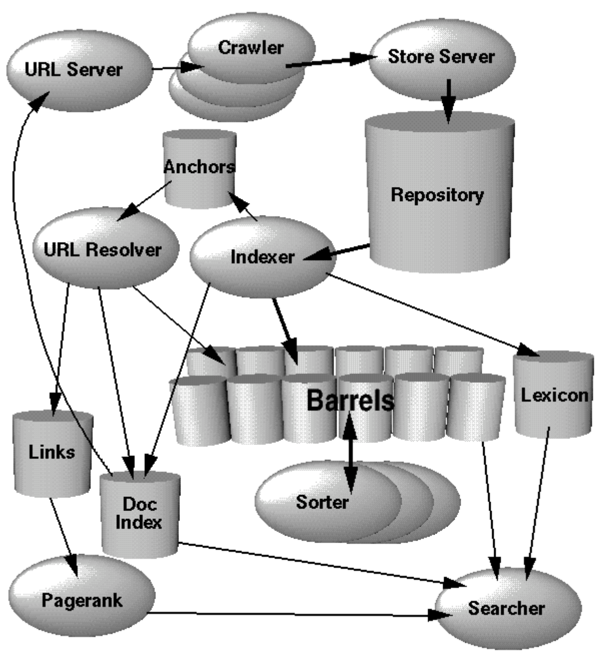
Document Processing



- The focus of document processing is
  – Extracting useful information from a document
  – Creating searchable document surrogates

Document processing and search

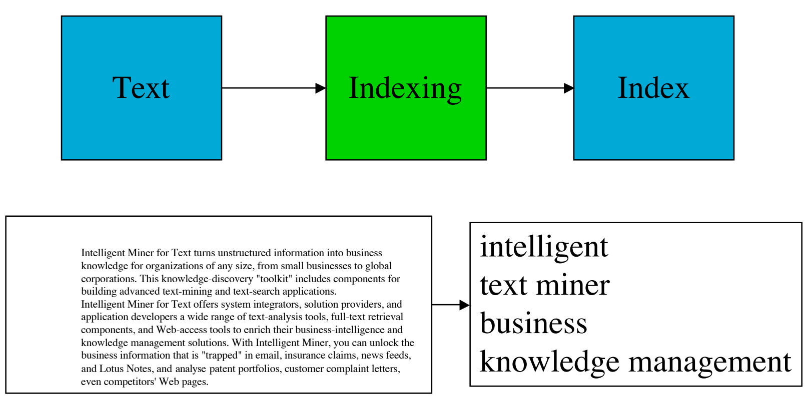




Indexing

- Act of assigning index terms to a document
- Identify important information and represent it in a useful way
- Indexing in traditional books
  – Book index (term index, topic index)
  – Figure index, citations, formula index

Indexing: From text to index



Why indexing?

- Need some representation of content
- Can not use the full document for search
- Using plain surrogates in inefficient
  - We want to avoid a “brute force” approach to searching (string searching, pattern matching)
- Used in:
  - Find documents by topic
  - Define topic areas, relate documents to each other
  - Predict relevance between documents and information needs

Indexing language (vocabulary)

- A set of index terms
  - words, phrases
- Controlled vocabulary
  - Indexing language is restricted to a set of terms predefined by experts
- Uncontrolled vocabulary
  - Any term satisfying some broad criteria is legible for indexing

Characteristics of an Indexing Language

- *Exhaustivity* refers to the breadth coverage
  - The extent to which all topics are covered
- *Specificity* refers to the depth of coverage
  - The ability to express specific details
- Domain dependent - snow example

Indexing: Choices and problems

- Who does the indexing
  - Humans (manual)
  - Computers (automatic)
- Problems and trade-offs
  - Presence of digital documents
  - Cost
  - Consistency
  - Precision

Manual indexing

- High precision (human understanding)
- Supports advance forms of indexing
  - Role-based indexing, phrase indexing
- Problems
  - Expensive
  - Inherently inconsistent
  - Indexer-user mismatch
- Addressing problems
  - Indexing rules
  - Precoordinated indexing
    - (vodka, gin, rum) -> liquor

Thesauri

- Roget Thesaurus vs. IR thesaurus
- IR thesaurus provides a controlled vocabulary and connections between words. It specifies:
  - Standard words that has to be used for indexing (vodka, *see* liquor)
  - Relationships between words (broader, narrower, related, opposite terms)

Features of thesauri

- Coordination level
  - Precoordination, postcoordination
- Represented term relationships
- Number of entries for each term
- Specificity of vocabulary
- Control on term frequency
- Normalization of vocabulary

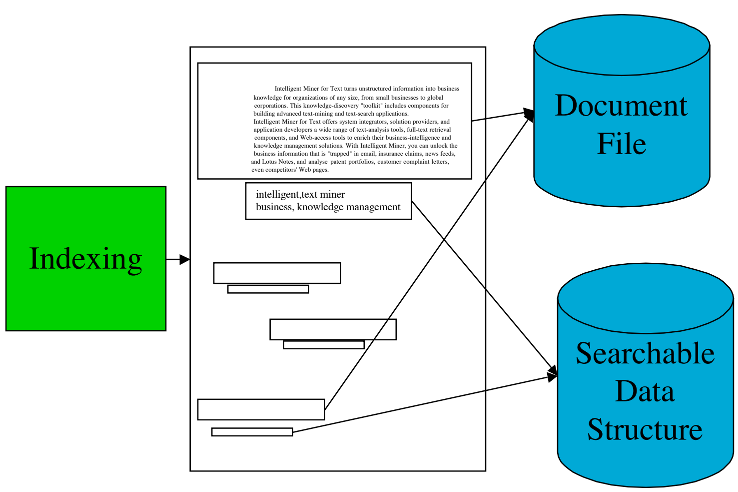
Working with thesauri

- Construction
  - User, automated, or automatic
- Usage
  - Using a thesaurus for indexing
  - Using a thesaurus for search
- Some years ago a thesaurus was a handbook for an IR system

Automatic indexing

- Inexpensive
  - The only practical solution for large volume of data
- Consistent
- Requires digital documents
- Problems
  - Less precise (computer does not *understand* text!)
  - Typically supports simple forms of indexing

Document processing for search



From Indexing to Search

- The results of indexing are used to create a searchable data structure:
  - an inverted file
  - a term document matrix

Inverted File

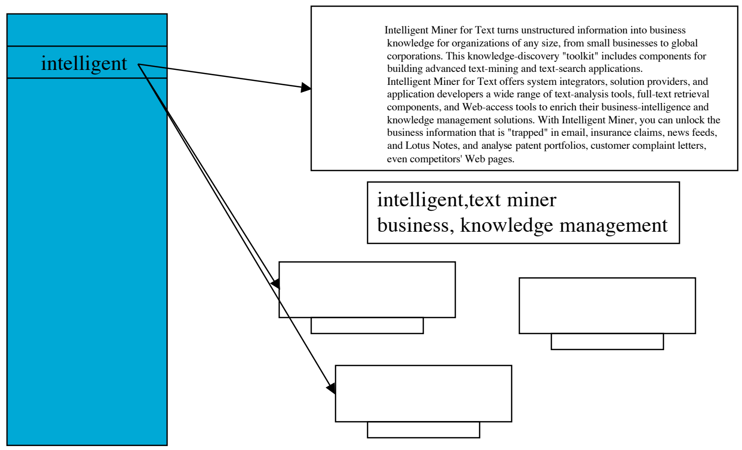
- Also known as a *Posting file* or *concordance*

  Contains, for each term of the lexicon, an inverted list that stores a list of pointers to all the occurrences of that term in the document collection

  Lexicon (or vocabulary) is a list of all terms that appear in the document collection

Inverted File

- Document file and inverted file



Inverted file

Doc1: `the cat is on the mat`

Doc2: `the mat is on the floor`

Inverted file

```
cat:doc1,1
floor:doc2,5
mat:doc1,5;doc2,1
```

Granularity

- The granularity of an index is the accuracy to which it identifies the location of a term
- The granularity depends on the document collection.
- The usual granularity is to individual documents

Matrix representation

- Many-to-many relationship
- Term-document matrix
  – indexing
- Term-term matrix
  – co-occurrence
- Document-document matrix
  – Similarity

Term-Document matrix

- Rows represent document terms
- Columns represent documents

Doc1: `the cat is on the mat`

Doc2: `the mat is on the floor`

	Doc1	Doc2
cat	1	0
floor	0	1
mat	1	1

The word *floor* is present in the document 2

Term-Document matrix

- The cells can also represent word counts or other frequency indicator
- Storage problems
  – n. of cells=n. of terms X n. of documents
- Matrix is sparse (i.e. many terms are 0)
- Practically use topologically equivalent representations

Term-term matrix

- Square matrix whose rows and columns represent the vocabulary terms
- a nonzero value in a cell t_{ij} means that the two terms occur together in some document or have some relationship

Document-document matrix

- Square matrix whose rows and columns represent the documents
- a nonzero value in a cell d_{ij} means that the two documents have some terms in common or have some relationship (e.g. an author in common)

Principles of automatic indexing

- Grammatical and content-bearing words
- Specific vs. generic
- Frequent vs. non frequent
  - The more often the word is found in the document - the better term is it
  - The less often the word is found in other documents - the better term is it
- Words of phrases?

Zipf's Law

- If the words that occurs in a document collection are ranked in order of decreasing frequency, they follow the *Zipf's law*

rank x frequency ≅ constant

If this law hold strictly the second most common world would occur only half as often as the the most frequent one

Optimal Term Selection

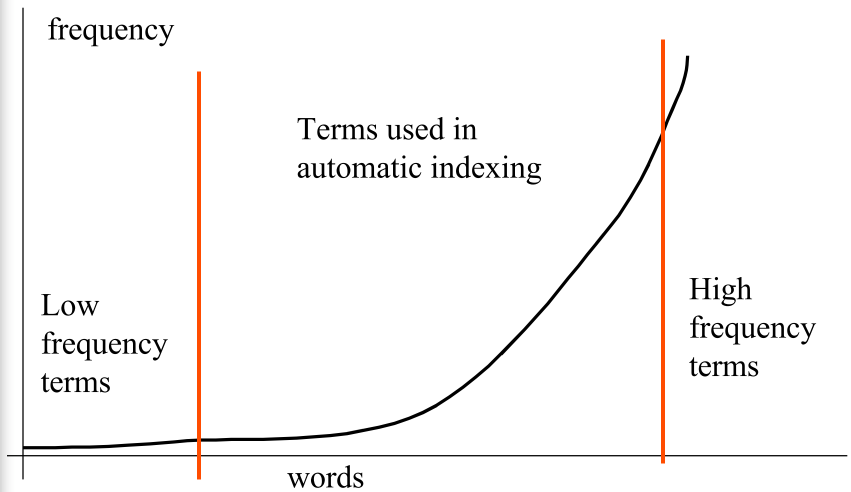
- The most frequently occurring words are those included by grammatical necessity (i.e. stopwords)

  the, of, and, a
- The words at the other end of the scale are poor index terms: very few documents will be retrieved when indexed by these terms

Thresholds

- Two thresholds can be defined when an automatic indexing algorithm is used:
  - high-frequency terms are not desirable because are often not significant
  - very low frequency terms are not desirable because their inability to retrieve many documents

Term Selection with Thresholds



What is a term?

- "bag of words"
  – In simple indexing we are neglecting the relationships among different words just considering the frequency
- Term Association
  – If two or more words occur often together then the pair should be included in the vocabulary (e.g. "information retrieval")
  – It can be useful to consider the word proximity (e.g. "retrieval of information" and "information retrieval")

Term Weighting

- With the term weighting we try to understand the importance of an index term for a document.
- A simple mechanism can be the use of the frequency of the term (tf) in the document, but it also necessary to consider the length of the documents and the kind of the documents.

Advanced Term Weighting

- Taking document into account
  - The frequency of a term in a documents should be compared with the length of the document
  - Relative frequency (frequency / length)
- Taking collection into account
  - Depending on the kind of document collection the same term can be more or less important.
  - The term *computer* can be very important in a collection of medical papers, but very common in a collection of document about programming

TF*IDF Term Weighting

- A relatively successful approach to automatic indexing uses TF*IDF term weighting
- Calculate the frequency of each word in the text, assign a weight to each term in each document which is
  - proportional to the frequency of the word in the document **(TF)**
  - inversely proportional to the frequency of the word in the document collection **(IDF)**

TF*IDF Term Weighting

k_i is an index term

d_j is a document

$w_{ij} \geq 0$ is a weight associated with (k_i,d_j)

- Assumption of mutual independence *("bag of words" representation)*

Calculating TF*IDF

$$w_{ik} = f_{ik} \times \left(\log_2 \frac{N}{D_k} + 1 \right)$$

Where:

N number of document in the collection

D_k number of documents containing term k (at least once)

f_{ik} frequency of term k in document i

TF*IDF matrix

	$term_1$	$term_2$			$term_n$
doc_1	w_{11}	w_{12}	w_{13}	...	w_{1n}
doc_2	w_{21}	w_{22}	w_{23}	...	w_{2n}
	...				
doc_m	w_{m1}	w_{m2}	w_{m3}	...	w_{mn}

Term Weighting with Signal to Noise Ratio

- Based on Shannon's information theory
- In information theory information has nothing to do with *meaning* but refers to the unexpectedness of a word
  - If a word is easy to forecast the information carried is very little. There is no information in something that can be precisely predicted
- Common words do not carry much information (e.g. stopwords).
- Less common words are much more informative

Information as messages

- Suppose that we have a set of n possible messages (words) i=1,2,3,…,n with probabilities of occurring p_i
- Since some message will occur,

$$\sum_{i=1}^{n} p_i = 1$$

Information Content

- We would like to define the *information content* H of the sequence of messages
- The entropy function satisfies some necessary assumptions

$$H = \sum_{i=1}^{n} p_i \log_2\left(\frac{1}{p_i}\right)$$

Information Content

- The *information content* of the single word i is calculated as:

$$\log_2\left(\frac{1}{p_i}\right)$$

- The more probable is the word less information it carries
- H is an average information content

Noise of an Index Term

- The noise associated to an index term K for a collection of N documents is calculated as

p_i

$$n_k = \sum_{i=1}^{N} \frac{f_{ik}}{t_k} \log_2\left(\frac{t_k}{f_{ik}}\right)$$

Where $t_k = \sum_{i=1}^{N} f_{ik}$ is the total frequency of the word k in the document collection

Noise of an Index Term

- Note that if f_{ik}=0 for a particular document then

$$\frac{f_{ik}}{t_k} \log_2\left(\frac{t_k}{f_{ik}}\right) = 0$$

Noise of an Index Term

- If a term appears just in *one* document K (repeated *a* times) then the noise is minimal: $t_k = a$

$$n_k = \frac{a}{a} * \log_2 \frac{a}{a} = \log_2 1 = 0$$

- On the contrary the noise is max if the term do not carry any information (appears in many documents)

Signal to Noise Ratio

- The signal of term k is

$$s_k = \log_2 t_k - n_k$$

- the weight w_{ik} of the term k in the document i is

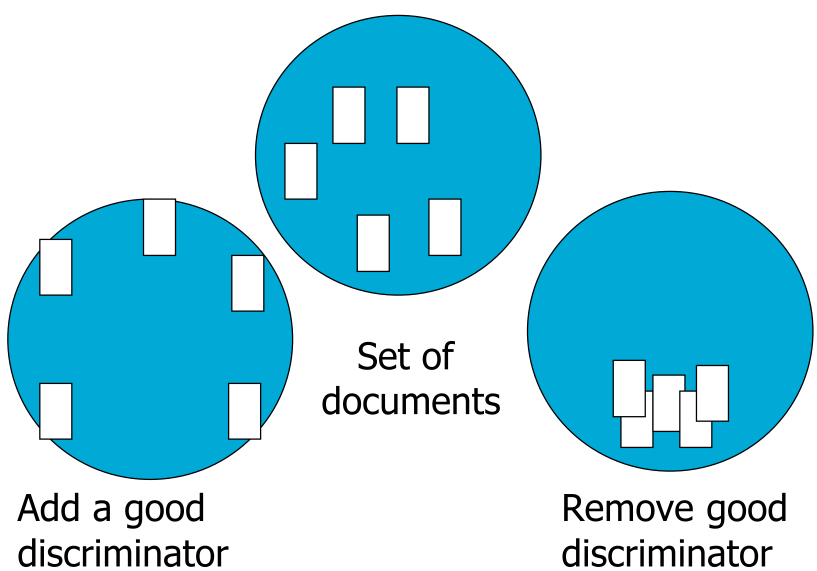
$$w_{ik} = f_{ik} \cdot s_k = f_{ik} \cdot [\log_2 t_k - n_k]$$

Term Discrimination Value TDV

- Measures the degree to which the use of a term will help to distinguish the document from one to another
- A measure of how much a given term k contributes to separating a set of documents into distinct subsets
- AVSIM= average similarity for the documents in the collection

$$TDV = AVSIM_N - AVSIM_{N(no\ k)}$$

Term Discrimination Value TDV



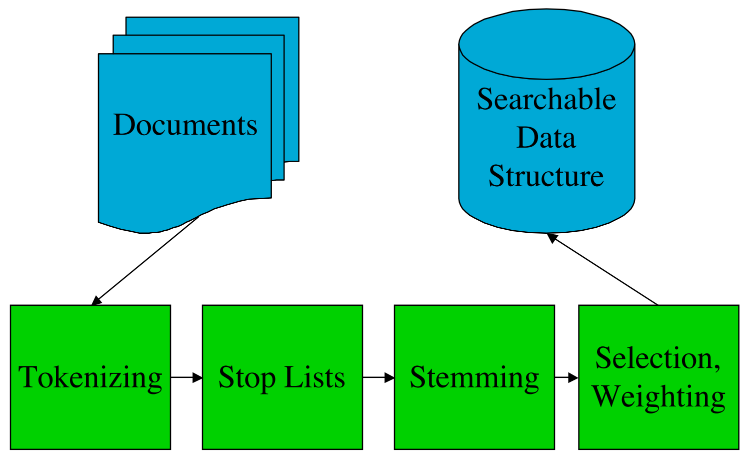
Term Discrimination Value TDV

- If TDV >>0 term is a good discriminator
- If TDV << 0 term is a poor discriminator
- If TDV ≅ 0 term is a mediocre discriminator
- TDV can be used as a term weight (together with term frequency) or used to select terms for indexing (as a threshold)

Simple Automatic Indexing

- Every character string not a *stopword* can be considered an index term
- Positional index: include information on filed and location
- Use some normalized form of the word
- Use of a threshold: eliminate high and low frequency terms as index terms
- Assign a term weight using statistics or some other mechanism

Automatic indexing



Stop lists

- Language-based stop list: words that bear little meaning (stopwords) and dropped from further processing
  - 20-500 English words *(an, and, by, for, of, the, ...)*
  - Subject-dependent stop lists
- Improve storage efficiency
- May cause problems
  - “to be or not to be”, AT&T, programming
- Removing stop words
  - From document
  - From query

Stoplist examples

CACM text collection:

a, about, above, accordingly, across, after, afterwards, again, against, all, almost, alone, along, already, also, although, always, am, among, amongst, an, and, another, any, anybody, anyhow, anyone, anything, anywhere, apart, are, around, as, aside, at, away, awfully, b, be, became, because, become, becomes, becoming, been, before, beforehand, behind, being, below, beside, besides, best, better, between, beyond, both, brief, but, by, c, can, cannot, cant, certain, co, consequently, could, d, did, do, does,

…..

x, y, yet, you, your, yours, yourself, yourselves, z, zero, /*, manual, unix, programmer's, file, files, used, name, specified, value, given, return, use, following, current, using, normally, returns, returned, causes, described, contains, example, possible, useful, available, associated, would, cause, provides, taken, unless, sent, followed, indicates, currently, necessary, specify, contain, indicate, appear, different, indicated, containing, gives, placed, uses, appropriate, automatically, ignored, changes, way, usually, allows, corresponding, specifying.

see also
http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/stop_words

Stemming

- Are there different index terms?
  - retrieve, retrieving, retrieval, retrieved, retrieves…
- Stemming algorithm:
  - (retrieve, retrieving, retrieval, retrieved, retrieves) ⇨ retriev
  - Strips prefixes of suffixes (-s, -ed, -ly, -ness)
  - Morphological stemming

Porter’s stemming algorithm

- Based on a measure of vowel-consonant sequences
  - measure **m** for a stem is **$[C](VC)^m[V]$** where **C** is a sequence of consonants and **V** is a sequence of vowels (including “y”) (**[ ]** indicates optional)
  - **m=0** (tree, by), **m=1** (trouble, oats, trees, ivy), **m=2** (troubles, private)
- Some Notation:
  - *<X> --> stem ends with letter X
  - *v* --> stem contains a vowel
  - *d --> stem ends in double consonant
  - *o --> stem ends with a **cvc** sequence where the final consonant is not w, x, y
- Algorithm is based on a set of condition action rules
  - old suffix --> new suffix
  - rules are divided into steps and are examined in sequence
- Good average recall and precision

Porter, M.F., "An Algorithm For Suffix Stripping," Program 14 (3), July 1980, pp. 130-137.

Porter’s stemming algorithm

- A selection of rules from Porter’s algorithm:

STEP	CONDITION	SUFFIX	REPLACEMENT	EXAMPLE
1a	NULL	sses	ss	stresses -> stress
	NULL	ies	I	ponies -> poni
	NULL	ss	ss	caress -> caress
	NULL	s	NULL	cats -> cat
1b	*v*	ing	NULL	making -> make
	. . .	. . .	. . .	. . .
1b1	NULL	at	ate	inflat(ed) -> inflaste
	. . .	. . .	. . .	. . .
1c	*v*	y	I	happy -> happi
2	m > 0	aliti	al	formaliti > formal
	m > 0	izer	ize	digitizer -> digitize
	. . .	. . .	. . .	. . .
3	m > 0	icate	ic	duplicate -> duplic
	. . .	. . .	. . .	. . .
4	m > 1	able	NULL	adjustable -> adjust
	m > 1	icate	NULL	microscopic -> microscop
	. . .	. . .	. . .	. . .
5a	m > 1	e	NULL	inflate -> inflat
	. . .	. . .	. . .	. . .
5b	M > 1, *d, *<L>	NULL	single letter	controll -> control, roll -> roll

Connections between document preparation and search

- If case conversion was used - can't distinguish lower and upper cases in a query
- If stop list was used - can't search by stop words
- If stemming is used can't distinguish different forms of the same word

Document similarity

- Similarity measure is a key IR problem
- How to calculate document similarity?
- Lexical measures
  – Count term occurrences
  – Count term frequencies
- Document as a vector of terms
  – 0-1 vector
  – Weighted vector

Document Similarity: 0-1 Vector

- Any document can be represented by a vector or a list of terms that occur in it

$$D=<t_1, t_2, t_3, \dots t_N>$$

where the component t_i corresponds to the i^{th} term in the vocabulary

- t_i=0 if the term does not occur
- t_i=1 or w_i if the term occurs

Document Similarity

Let D_1 and D_2 two document vectors with components t_{1i} t_{2i} for i=1,2,…N

we define:

- w=number of terms for which $t_{1i}=t_{2i}=1$ (present in both)
- x=number of terms for which $t_{1i}=1$ and $t_{2i}=0$ (present in 1st)
- y=number of terms for which $t_{1i}=0$ and $t_{2i}=1$ (present in 2nd)
- z=number of terms for which $t_{1i}=t_{2i}=0$ (absent in both)
- $n_1=w+x$
- $n_2=w+y$

Matching document terms

w	x
y	z

$n_1 = w + x$

$n_2 = w + y$

$N = w + x + y + z$

- w - terms present in both
- z - terms absent in both
- x and y - terms present in one of the documents

Measures

- Basic measure:
  $\delta = w - n_1 n_2 / N$
- Measures of similarity:
  $C(D_1, D_2) = \delta(D_1, D_2) / \alpha$
  Where α is:
  $\alpha(S) = N/2$ - separation
  $\alpha(R) = \max(n_1, n_2)$ - rectangular distance

Document Similarity

- Define the *basic comparison unit*

$$\delta\left(D_1, D_2\right) = \delta\left(D_2, D_1\right) = w - \frac{n_1 n_2}{N}$$

- The basic comparison unit can be used as a measure of similarity defining a *coefficient of association*

$$C_\alpha(D_1, D_2) = \frac{\delta(D_1, D_2)}{\alpha}$$

Document Similarity

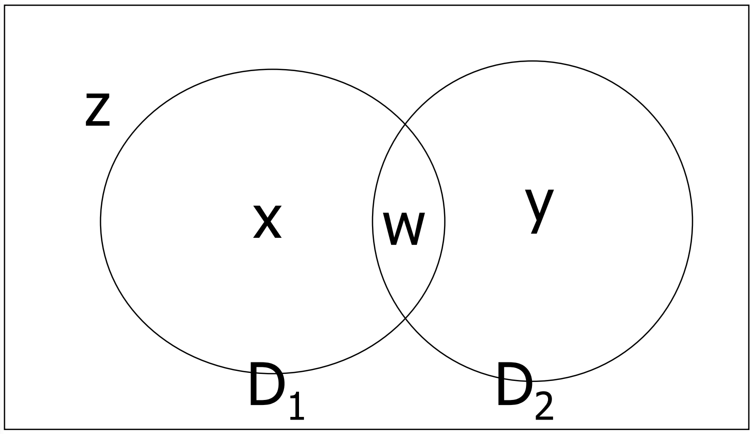
- There are many different definition of α and so many “similarity” definitions
- Some typical examples:

  α is:

  $\alpha(S) = N/2$ - separation coefficient

  $\alpha(R) = \max(n_1, n_2)$ - rectangular distance

Document Similarity: Separation



Document Similarity: Weighted Vector

- Similarity measures that depends on the frequency with which terms occur in a document can be based on a metric (distance measure)
- The greater the distance between documents, the less similar they are

Properties of a Metric

- A metrics has three defining properties
  - its values are nonnegative, the distance between two points is 0 *iff* the points are identical d(A,B)=0 $\rightarrow$ A≡B
  - it is symmetric d(A,B)=d(B,A)
  - it satisfies the triangle inequality d(A,B)+d(B,C) ≥ d(A,C) for any points A,B and C

L_p Metrics

- Let D_1 and D_2 two document vectors with components t_{1i} t_{2i} for i=1,2,…N

$$D_1=<t_{11}, t_{12}, t_{13}, \dots t_{1N}>$$
$$D_2=<t_{21}, t_{22}, t_{23}, \dots t_{2N}>$$

- The L_p metrics can be defined

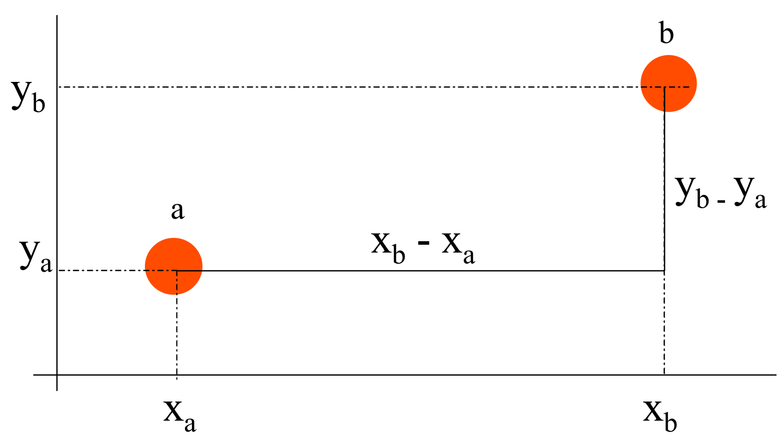
$$L_p(D_1, D_2) = \left[\sum_i |t_{1i} - t_{2i}|^p \right]^{1/p}$$

Three Popular L_p Metrics

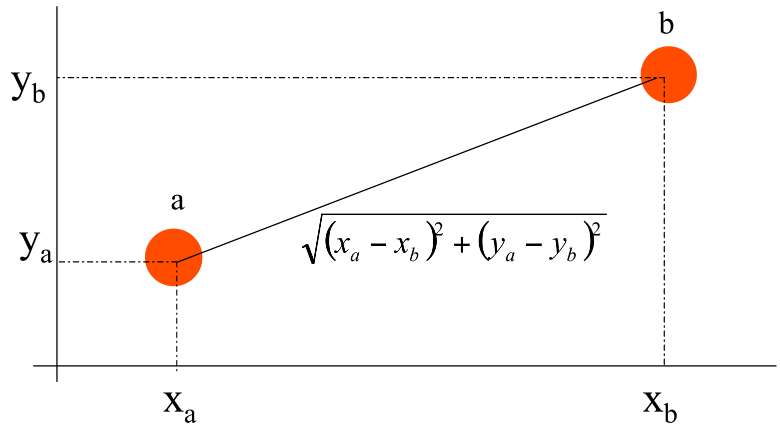
- City block distance if p=1
- Euclidean distance if p=2
- Maximal direction if p=∞

$$L_\infty(D_1, D_2) = \max_i \left(\left| t_{1i} - t_{2i} \right| \right)$$

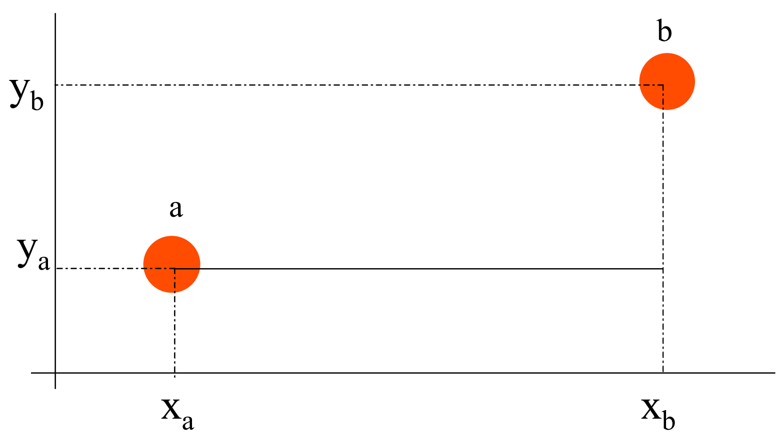
City Block Distance



Euclidean Distance



Maximal Direction



Analysis beyond counting words?

- Natural Language Processing
- Pragmatics processing
  - Weighting sources, authors
- User-depending factors
  - User adaptation

Multi-language retrieval

- Most progress with English, but now there are IR systems for every language
- English is simple!
  - Separated verbs in German
  - Suffixes in Russian and Turkish
  - Vowels in Hebrew and Arabic
- Translation and multi-language systems

Homework

Exercise 1

Given the document representations

D_1=<4,2,0,4>

D_2=<0,3,1,0>

D_3=<1,2,0,5>

D_4=<2,0,4,3>

- calculate the distances between all the documents pairs for the three L metrics

Homework

Exercise 2

For a set of N=20 documents, calculate the noise associated to a term that appears twice in documents 1,2,3,…, 19 and once in document 20.

Compare it with the noise associated to a term that appears 2 times in ALL documents.

Explain the results