

# indexing

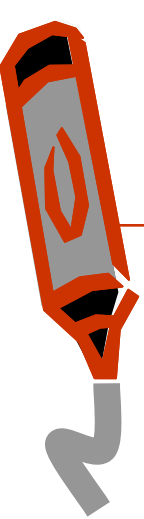
---



# vocabulary

---

- File organizations or *indexes* are used to increase performance of system
  - Will talk about how to store indexes later
- Text *indexing* is the process of deciding what will be used to represent a given document
- These *index terms* are then used to build indexes for the documents
- The *retrieval model* described how the indexed terms are incorporated into a model
  - Relationship between retrieval model and indexing model



# manual vs automatic

---

## Manual vs. Automatic Indexing

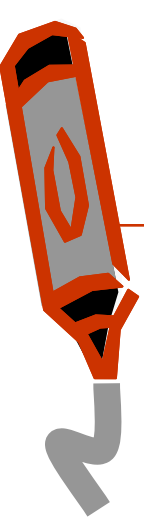
- Manual or human indexing:
  - Indexers decide which keywords to assign to document based on *controlled vocabulary*
  - e.g. MEDLINE, MeSH, LC subject headings, Yahoo
  - Significant cost
- Automatic indexing:
  - Indexing program decides which words, phrases or other features to use *from text of document*
  - Indexing speeds range widely
- Indri (CIIR research system) indexes approximately 10GB/hour



# terminology

---

- *Index language*
  - Language used to describe documents and queries
- *Exhaustivity*
  - Number of different topics indexed, completeness
- *Specificity*
  - Level of accuracy of indexing
- *Pre-coordinate indexing*
  - Combinations of index terms (e.g. phrases) used as indexing label
  - E.g., author lists key phrases of a paper
- *Post-coordinate indexing*
  - Combinations generated at search time
  - Most common and the focus of this course

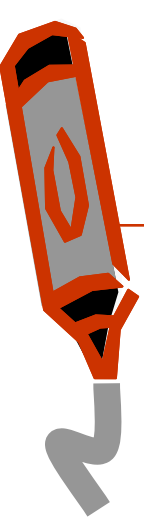


# library of Congress headings

---

- A -- GENERAL WORKS
- B -- PHILOSOPHY. PSYCHOLOGY. RELIGION
- C -- AUXILIARY SCIENCES OF HISTORY
- D -- HISTORY: GENERAL AND OLD WORLD
- E -- HISTORY: AMERICA
- F -- HISTORY: AMERICA
- G -- GEOGRAPHY. ANTHROPOLOGY. RECREATION
- H -- SOCIAL SCIENCES
- J -- POLITICAL SCIENCE
- K -- LAW
- L -- EDUCATION
- M -- MUSIC AND BOOKS ON MUSIC
- N -- FINE ARTS
- P -- LANGUAGE AND LITERATURE
- Q -- SCIENCE
- R -- MEDICINE**
- S -- AGRICULTURE
- T -- TECHNOLOGY
- U -- MILITARY SCIENCE
- V -- NAVAL SCIENCE
- Z -- BIBLIOGRAPHY. LIBRARY SCIENCE. INFORMATION RESOURCES  
(GENERAL)

# where is computer science ?



Subclass Q	Subclass Q	
Subclass QA	Q1-390	Science (General)
Subclass QB	Q1-295 Q300-390	General Cybernetics
Subclass QC	Q350-390	Information theory
Subclass QD	Subclass QA	
Subclass QE	QA1-939	Mathematics
Subclass QH	QA1-43	General
Subclass QK	QA47-59	Tables
Subclass QL	QA71-90	Instruments and machines
Subclass QM	QA75-76.95	Calculating machines
Subclass QP	QA75.5-76.95	Electronic computers. Computer science
Subclass QR	QA76.75-76.765	Computer software
Subclass QS	QA101-(145)	Elementary mathematics. Arithmetic
Subclass QT	QA150-272.5	Algebra
Subclass QU	QA273-280	Probabilities. Mathematical statistics
Subclass QV	QA299.6-433	Analysis
Subclass QW	QA440-699	Geometry. Trigonometry. Topology
Subclass QX	QA801-939	Analytic mechanics
Subclass QY	Microbiology	



# manual vs automatic indexing

---

	<b>Manual</b>	<b>Automatic</b>
<b>Controlled Vocabulary</b>	Current indexing practice	Text categorization “Intelligent” IR
<b>Free Text</b>	Current indexing practice	Text search engines “Statistical” IR

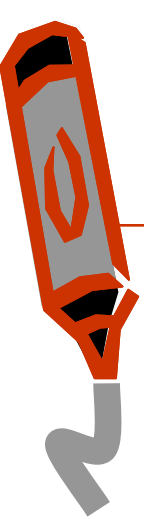


# manual vs automatic indexing

---

- Experimental evidence is that retrieval effectiveness using automatic indexing can be at least as effective as manual indexing with controlled vocabularies
  - original results were from the Cranfield experiments in the 60s
  - considered counter-intuitive
  - other results since then have supported this conclusion
  - broadly accepted at this point
  
- Experiments have also shown that using *both* manual and automatic indexing improves performance
  - “combination of evidence”

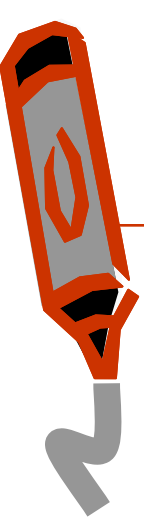




# basic automatic indexing

---

- Parse documents to recognize structure
  - e.g. title, date, other fields
  - clear advantage to XML
- Scan for word tokens
  - numbers, special characters, hyphenation, capitalization, etc.
  - languages like Chinese need *segmentation*
  - record positional information for *proximity* operators
- Stopword removal
  - based on short list of common words such as “the”, “and”, “or”
  - saves storage overhead of very long indexes
  - can be dangerous (e.g., “The Who”, “and-or gates”, “vitamin a”)



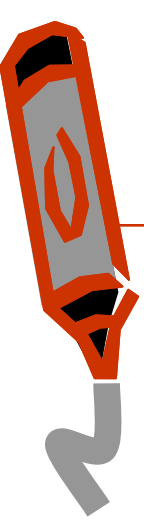
# basic automatic indexing

---


- Stem words
  - morphological processing to group word variants such as plurals
  - better than string matching (e.g. comput\*)
  - can make mistakes but generally preferred
  - not done by most Web search engines (why?)
- Weight words
  - want more “important” words to have higher weight
  - using frequency in documents and database
  - frequency data independent of retrieval model
- Optional
  - phrase indexing
  - thesaurus classes (probably will not discuss)
  - others...

# basic indexing

---



- Parse and tokenize
- Remove stop words
- Stemming
- Weight terms



# words vs terms vs concepts

---

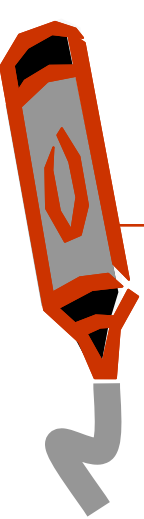
- Simple indexing is based on words or word stems
  - More complex indexing could include phrases or thesaurus classes
  - *Index term* is general name for word, phrase, or feature used for indexing
- *Concept-based retrieval* often used to imply something beyond word indexing
- In virtually all systems, a *concept* is a name given to a set of recognition criteria or rules
  - similar to a thesaurus class
- Words, phrases, synonyms, linguistic relations can all be evidence used to infer presence of the concept
- e.g. the concept “information retrieval” can be inferred based on the presence of the words “information”, “retrieval”, the phrase “information retrieval” and maybe the phrase “text retrieval”



# phrases

---

- Both statistical and syntactic methods have been used to identify “good” phrases
- Proven techniques include finding all word pairs that occur more than  $n$  times in the corpus or using a part-of-speech tagger to identify simple noun phrases
  - 1,100,000 phrases extracted from all TREC data (more than 1,000,000 WSJ, AP, SJMS, FT, Ziff, CNN documents)
  - 3,700,000 phrases extracted from PTO 1996 data
- Phrases can have an impact on both effectiveness and efficiency
  - phrase indexing will speed up phrase queries
  - finding documents containing “Black Sea” better than finding documents containing both words
  - effectiveness not straightforward and depends on retrieval model
- e.g. for “information retrieval”, how much do individual words count?

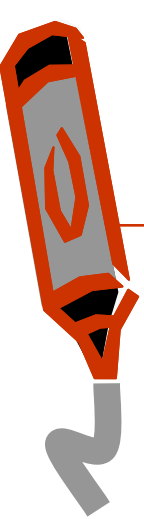


# top phrases on TREC 2-8

---

65824 United States  
61327 Article Type  
33864 Los Angeles  
18062 Hong Kong  
17788 North Korea  
17308 New York  
15513 San Diego  
15009 Orange County  
12869 prime minister  
12799 first time  
12067 Soviet Union  
10811 Russian Federation  
9912 United Nations  
8127 Southern California  
7640 South Korea  
7620 end recording  
7524 European Union  
7436 South Africa  
7362 San Francisco  
7086 news conference  
6792 City Council  
6348 Middle East  
6157 peace process  
5955 human rights  
5837 White House

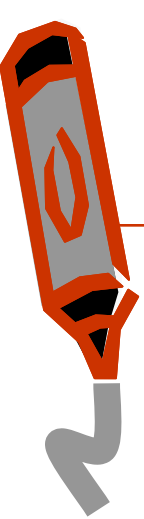
5778 long time  
5776 Armed Forces  
5636 Santa Ana  
5619 Foreign Ministry  
5527 Bosnia-Herzegovina  
5458 words indistinct  
5452 international community  
5443 vice president  
5247 Security Council  
5098 North Korean  
5023 Long Beach  
4981 Central Committee  
4872 economic development  
4808 President Bush  
4652 press conference  
4602 first half  
4565 second half  
4495 nuclear weapons  
4448 UN Security Council  
4426 South Korean  
4219 first quarter  
4166 Los Angeles County  
4107 State Duma  
4085 State Council  
3969 market economy  
3941 World War II



# phrases from 50 T R E C queries

---

14	international criminal activity	5	theft of trade secret
9	international criminal	1324	trade secret
1436	criminal activity	573	sources of information
84	hubble telescope	530	trade journal
188	passenger vehicle	334	business meet
9086	civil war	506	patent office
255	hydroelectric project	1870	trade show
5261	detailed description	26	competitor's product
183	rap music	63	growing plant
1449	negative effect	41	magnetic levitate
8081	young people	38	commercial harvest
297	radio wave	58	highway accident
26	radio tower		
404	car phone		
135	brain cancer		

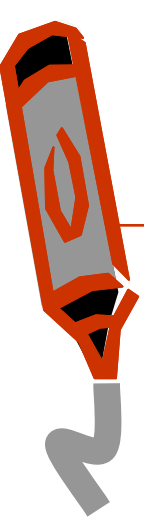


# information extraction

---

- Special recognizers for specific concepts
  - people, organizations, places, dates, monetary amounts, products, ...
- “Meta” terms such as #COMPANY, #PERSON can be added to indexing
- e.g., a query could include a restriction like “...the document must specify the location of the companies involved...”
- Could potentially customize indexing by adding more recognizers
  - difficult to build
  - problems with accuracy
  - adds considerable overhead
- Key component of question answering systems
  - To find concepts of the right type (e.g., people for “who” questions)





# indexing example

---

- **Original text:**

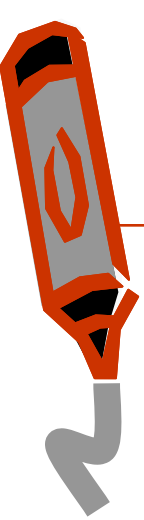
John Davenport, ~~52 years old, was~~ appointed chief executive officer ~~of this~~ international telecommunications concern's ~~U.S.~~ subsidiary, Cable & Wireless North America Inc. ~~Mr. Davenport, who~~ succeeds John Zrno, ~~is~~ currently general manager ~~of the~~ group's operations ~~in~~ Bermuda.

- **One indexing result:**

john davenport appoint chief executive officer international telecommunication concern subsidiary cable wireless north america davenport succeed john zrno current general manager group operation bermuda

- **Another possibility:**

John\_Davenport #person 52 years\_old #age appoint chief\_executive\_officer international telecommunication concern #USA subsidiary Cable\_&\_Wireless\_North\_America #company Davenport #person succeed John\_Zrno #person general\_manager group operation Bermuda #foreigncountry



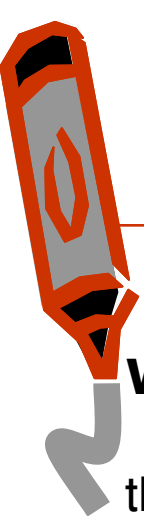
# stopwords

---

- Remove non-content-bearing words
  - Function words that do not convey much meaning
- Can be as few as one word
  - What might that be?
- Can be several hundreds
  - Surprising(?) examples from Inquiry at UMass (of 418)
  - Halves, exclude, exception, everywhere, sang, saw, see, smote, slew, year, cos, ff, double, down
- Need to be careful of words in phrases
  - Library of Congress, Smoky the Bear
- Primarily an efficiency device, though sometimes helps with spurious matches

# stopwords

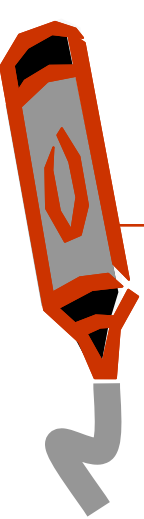
---



<b>Word</b>	<b>Occurrences</b>	<b>Percentage</b>
the	8,543,794	6.8
of	3,893,790	3.1
to	3,364,653	2.7
and	3,320,687	2.6
in	2,311,785	1.8
is	1,559,147	1.2
for	1,313,561	1.0
that	1,066,503	0.8
said	1,027,713	0.8

Frequencies from 336,310 documents in the 1GB TREC Volume 3 Corpus  
125,720,891 total word occurrences; 508,209 unique words

# stopwords



a about above according across after afterwards again against albeit all almost alone along already also although always am among amongst an and another any anybody anyhow anyone anything anyway anywhere apart are around as at av be became because become becomes becoming been before beforehand behind being below beside besides between beyond both but by can cannot canst certain cf choose contrariwise cos could cu day do does doesn't doing dost doth double down dual during each either else elsewhere enough et etc even ever every everybody everyone everything everywhere except excepted excepting exception exclude excluding exclusive far farther farthest few ff first for formerly forth forward from front further furthermore furthest get go had halves hardly has hast hath have he hence henceforth her here hereabouts hereafter hereby herein hereto hereupon hers herself him himself hindmost his hither hitherto how however howsoever i ie if in inasmuch inc include included including indeed indoors inside insomuch instead into inward inwards is it its itself just kind kg km last latter latterly less lest let like little ltd many may maybe me meantime meanwhile might moreover most mostly more mr mrs ms much must my myself namely need neither never nevertheless next no nobody none nonetheless noone nope nor not nothing notwithstanding now nowadays nowhere of off often ok on once one only onto or other others otherwise ought our ours ourselves out outside over own per perhaps plenty provide quite rather really round said sake same sang save saw see seeing seem seemed seeming seems seen seldom selves sent several shalt she should shown sideways since slept slew slung slunk smote so some somebody somehow someone something sometime sometimes somewhat somewhere spake spat spoke spoken sprang sprung stave staves still such supposing than that the thee their them themselves then thence thenceforth there thereabout therabouts thereafter thereby therefore therein thereof thereon thereto thereupon these they this those thou though thrice through throughout thru thus thy thyself till to together too toward towards ugh unable under underneath unless unlike until up upon upward upwards us use used using very via vs want was we week well were what whatever whatsoever when whence whenever whensoever where whereabouts whereafter whereas whereat whereby wherefore wherefrom wherein whereinto whereof whereon wheresoever whereto whereunto whereupon wherever wherewith whether whew which whichever whichsoever while whilst whither who whoa whoever whole whom whomever whomsoever whose whosoever why will wilt with within without worse worst would wow ye yet year yippee you your yours yourself yourselves



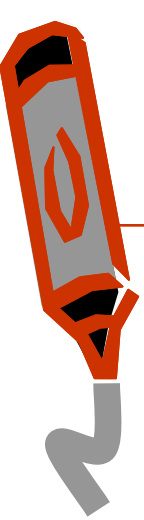
# stemming

---

- Stemming is commonly used in IR to conflate morphological variants
- Typical stemmer consists of collection of rules and/or dictionaries
  - simplest stemmer is “suffix s”
  - Porter stemmer is a collection of rules
  - KSTEM [Krovetz] uses lists of words plus rules for inflectional and derivational morphology
  - similar approach can be used in many languages
  - some languages are difficult, e.g. Arabic
- Small improvements in effectiveness and significant usability benefits
  - With huge document set such as the Web, less valuable

# stemming

---



servomanipulator | servomanipulators servomanipulator

logic | logical logic logically logics logicals logical logicial logically

login | login logins

microwire | microwires microwire

overpressurize | overpressurization overpressurized overpressurizations  
overpressurizing overpressurize

vidrio | vidrio

sakhuja | sakhuja

rockel | rockel

pantopon | pantopon

knead | kneaded kneads knead kneader kneading kneaders

linxi | linxi

rocket | rockets rocket rocketed rocketing rocketings rocketeer

hydroxytoluene | hydroxytoluene

ripup | ripup



# Porter stemmer

---

- 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 (inc.  $y$ ),  $[\ ]$  = optional
  - $m=0$  (tree, by),  $m=1$  (trouble, oats, trees, ivy),  $m=2$  (troubles, private)
- Algorithm is based on a set of condition action rules
  - old suffix  $\rightarrow$  new suffix
  - rules are divided into steps and are examined in sequence
- Longest match in a step is the one used
  - e.g. Step 1a:
    - sses  $\rightarrow$  ss (*caresses*  $\rightarrow$  *caress*)
    - ies i (*ponies*  $\rightarrow$  *poni*)
    - s NULL (*cats*  $\rightarrow$  *cat*)
  - e.g. Step 1b:
    - if  $m > 0$  eed  $\rightarrow$  ee (*agreed*  $\rightarrow$  *agree*)
    - if  $*v^*ed$   $\rightarrow$  NULL (*plastered*  $\rightarrow$  *plaster* but *bled*  $\rightarrow$  *bled*)
    - then at  $\rightarrow$  ate (*conflat(ed)*  $\rightarrow$  *conflate*)
- Many implementations available
  - <http://www.tartarus.org/~martin/PorterStemmer/>
- Good average recall and precision



# stemming example

---

- **Original text:**

Document will describe marketing strategies carried out by U.S. companies for their agricultural chemicals, report predictions for market share of such chemicals, or report market statistics for agrochemicals, pesticide, herbicide, fungicide, insecticide, fertilizer, predicted sales, market share, stimulate demand, price cut, volume of sales

- **Porter Stemmer:**

market strateg carr compan agricultur chemic report predict market share chemic report market statist agrochem pesticid herbicid fungicid insecticid fertil predict sale stimul demand price cut volum sale

- **KSTEM:**

marketing strategy carry company agriculture chemical report prediction market share chemical report market statistic agrochemic pesticide herbicide fungicide insecticide fertilizer predict sale stimulate demand price cut volume sale

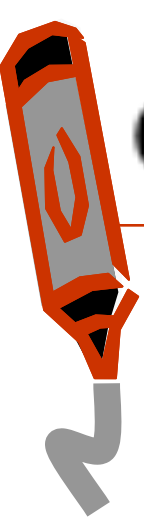




# stemming issues

---

- Lack of domain-specificity and context can lead to occasional serious retrieval failures
- Stemmers are often difficult to understand and modify
- Sometimes too aggressive in conflation
  - e.g. “policy”/“police”, “execute”/“executive”, “university”/“universe”, “organization”/“organ” are conflated by Porter
- Miss good confluations
  - e.g. “European”/“Europe”, “matrices”/“matrix”, “machine”/“machinery” are not conflated by Porter
- Produce stems that are not words and are often difficult for a user to interpret
  - e.g. with Porter, “iteration” produces “iter” and “general” produces “gener”
- Corpus analysis can be used to improve a stemmer or replace it



# corpus-based stemming

---

- Hypothesis: Word variants that should be conflated will co-occur in documents (text windows) in the corpus
- Modify equivalence classes generated by a stemmer or other “aggressive” techniques such as initial n-grams
  - more aggressive classes mean less conflations missed
- New equivalence classes are clusters formed using (modified) EMIM scores between pairs of word variants
- Can be used for other languages



# equivalence classes

---

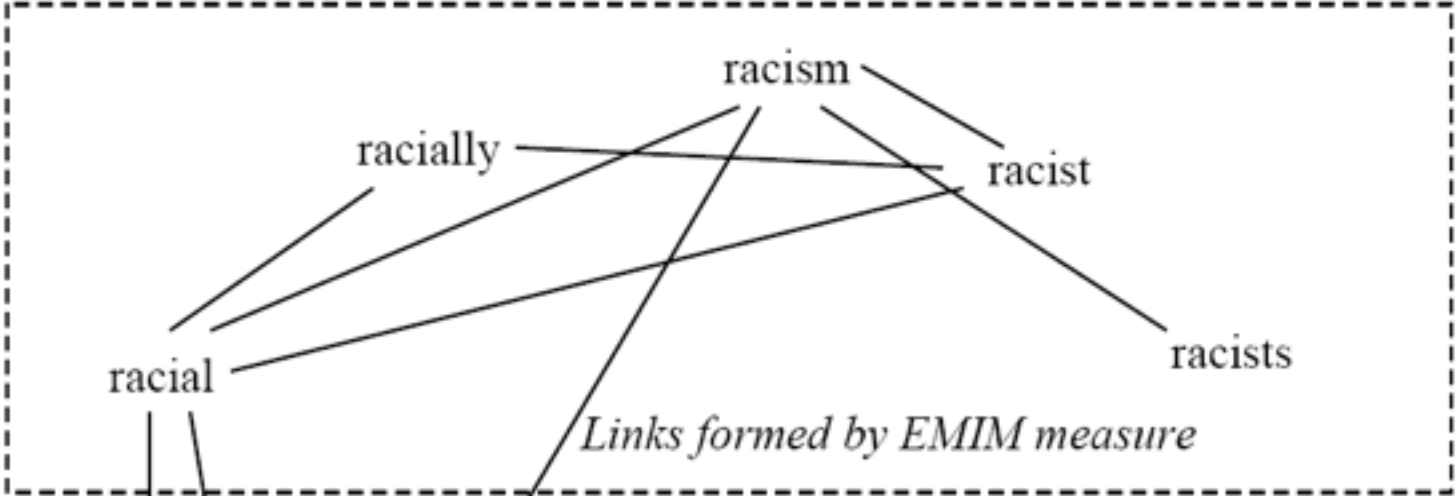
## **Some Porter Classes for a WSJ Database**

abandon abandoned abandoning abandonment abandonments abandons  
abate abated abatement abatements abates abating  
abrasion abrasions abrasive abrasively abrasiveness abrasives  
absorb absorbable absorbables absorbed absorbencies absorbency absorbent  
absorbents absorber absorbers absorbing absorbs  
abusable abuse abused abuser abusers abuses abusing abusive abusively  
access accessed accessibility accessible accessing accession

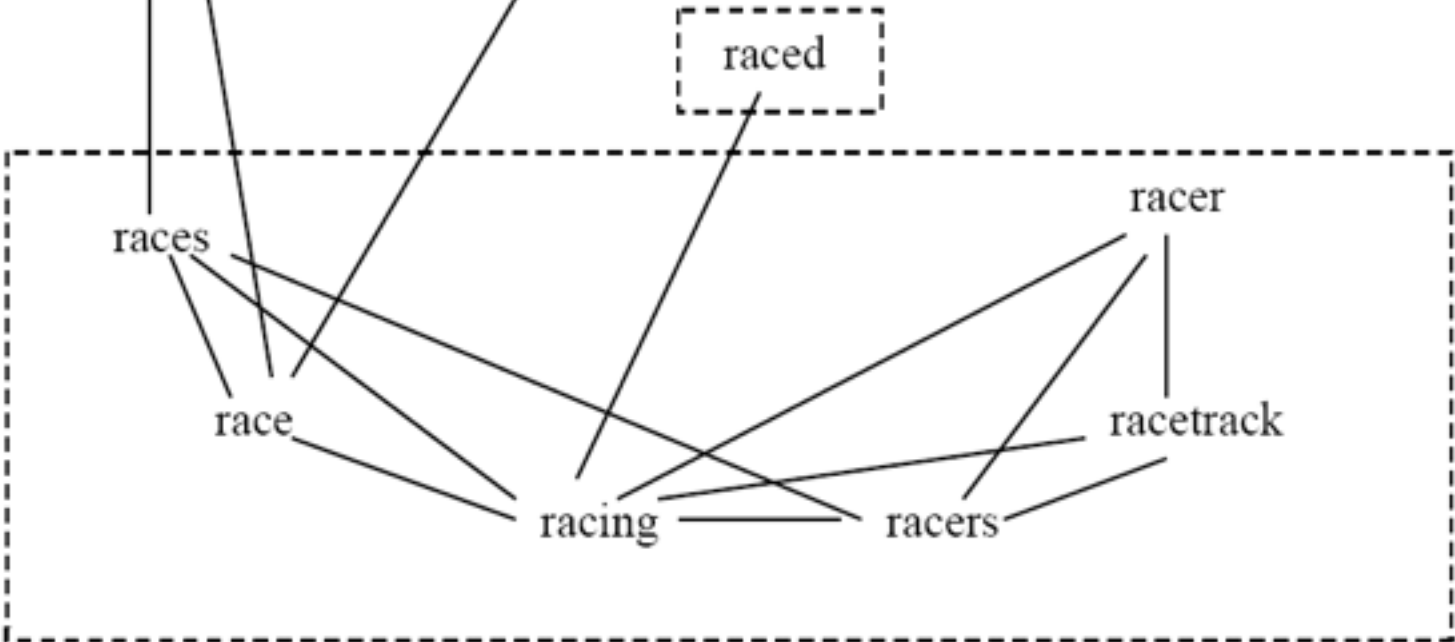
## **Classes refined through corpus analysis (singleton classes omitted)**

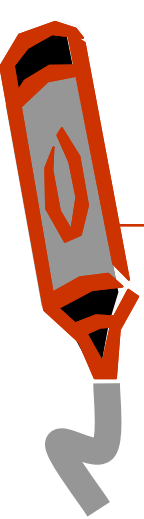
abandonment abandonments  
abated abatements abatement  
abrasive abrasives  
absorbable absorbables  
absorbencies absorbency absorbent  
absorber absorbers  
abuse abusing abuses abusive abusers abuser abused  
accessibility accessible

# partitions



*“optimal”  
partition*

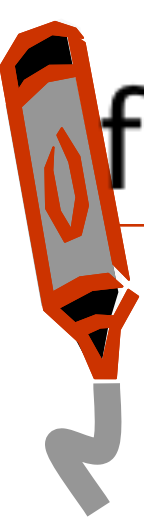




# corpus-based stemming

---

- Clustering technique used has an impact
- Both Porter and KSTEM stemmers are improved slightly by this technique (max. of 4% avg. precision on WSJ)
- N-gram stemmer gives same performance as improved “linguistic” stemmers
- N-gram stemmer gives same performance as baseline Spanish linguistic stemmer
- Suggests advantage to this technique for
  - building new stemmers
  - building stemmers for new languages



# feature selection/weighting

---

- Basic Issue: Which terms should be used to index (describe) a document?
- Different focus than retrieval model, but related
- Sometimes seen as *term weighting*
- Some approaches
  - TF·IDF
  - Term Discrimination model
  - 2-Poisson model
  - Clumping model
  - Language models

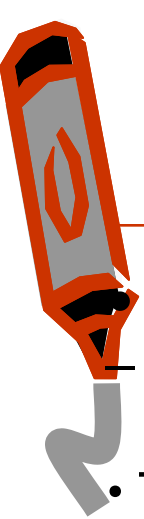


# index models

---

- What makes a term good for indexing?
  - Trying to represent “key” concepts in a document
  
- What makes an index term good for a query?

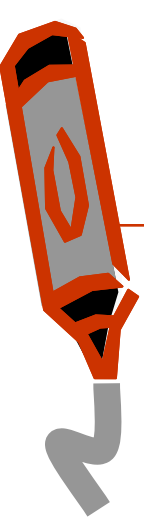
# tf weights



- Standard weighting approach for many IR systems
  - many different variations of exactly how it is calculated
- TF component - the more often a term occurs in a document, the more important it is in describing that document
  - normalized term frequency
  - normalization can be based on maximum term frequency or could include a document length component
  - often includes some correction for estimation using small samples
  - some bias towards numbers between 0.4-1.0 to represent fact that a single occurrence of a term is important
  - logarithms used to smooth numbers for large collections
  - e.g. where  $c$  is a constant such as 0.4,  $tf$  is the term frequency in the document, and  $max\_tf$  is the maximum term frequency in any document

$$c + (1 - c) \frac{\log(tf + 0.5)}{\log(max\_tf + 1.0)}$$





# tf = term frequency

---

<sup>2</sup> raw tf (called tf) = count of 'term' in document

<sup>2</sup> robinson tf (okpitf): 
$$\text{okpitf} = \frac{tf}{tf + 1.5 + 1.5 \frac{\text{doclen}}{\text{avgdoclen}}}$$

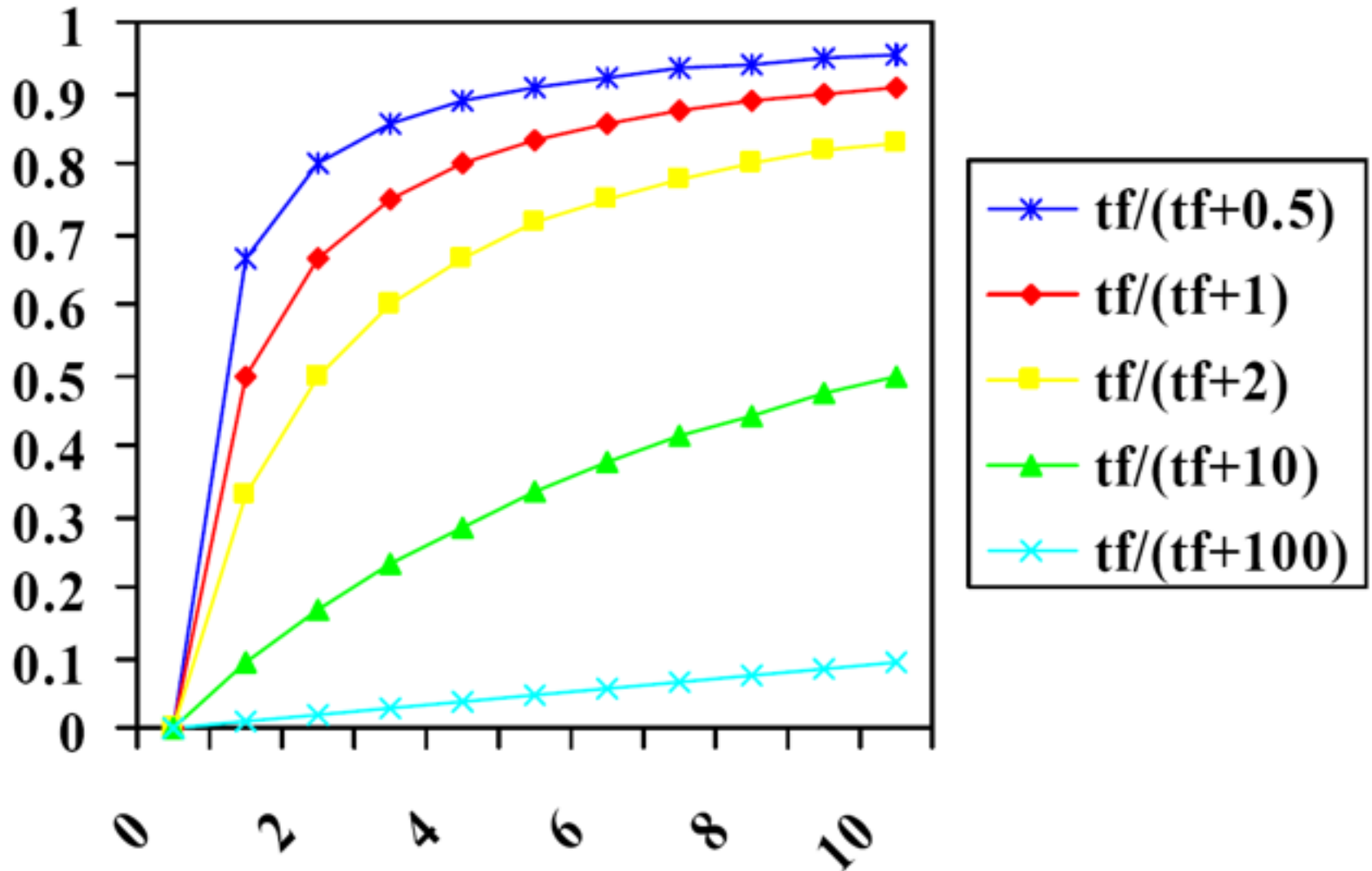
- Based on a set of simple criteria loosely connected to the 2-Poisson model

- Basic formula is  $tf / (k + tf)$  where  $k$  is a constant (approx. 1-2)

- Document length introduced as a verbosity factor

<sup>2</sup> many variants

# Robertson tf





# IDF weights

---

² Invers Document Frequency

² used to weight terms based on frequency in the corpus (or language)

² Fixed, it can be precomputed for every term

²  $IDF(t) = \log\left(\frac{N}{N_t}\right)$  where

$N = \#$  of docs

$N_t = \#$  of docs containing term  $t$

# tf-idf

2 in fact  $tf \cdot idf$

2 the weight on every term is  $tf(t,d) \cdot idf(t)$

Often :  $IDF = \log(N/d) + 1$  where  $N$  is the number of documents in the collection,  $d$  is the number of documents the term occurs in

$IDF = -\log p$ , where  $p$  is the term probability

sometimes normalized when in  $TF \cdot IDF$  combination

e.g. for INQUERY:  $\frac{\log(\frac{N+0.5}{d})}{\log(N+10)}$

2  $TF$  and  $IDF$  combined using multiplication

2 No satisfactory model behind these combinations



# term discrimination model

---

- Proposed by Salton in 1975
- Based on vector space model
  - documents and queries are vectors in an n-dimensional space for n terms
- Compute *discrimination value* of a term
  - degree to which use of the term will help to distinguish documents
- Compare average similarity of documents both with and without an index term



# term discrimination model

---

- Compute average similarity or “density” of document space

$$AVGSIM = K \sum_{i=1}^n \sum_{\substack{j=1 \\ i \neq j}}^n similar(D_i, D_j)$$

- *AVGSIM* is the density
- where  $K$  is a normalizing constant (e.g.,  $1/n(n-1)$ )
- *similar()* is a similarity function such as cosine correlation
- Can be computed more efficiently using an average document or *centroid*
  - frequencies in the centroid vector are average of frequencies in document vectors

$$AVGSIM = K \sum_{i=1}^n similar(\bar{D}, D_i)$$



# term discrimination model

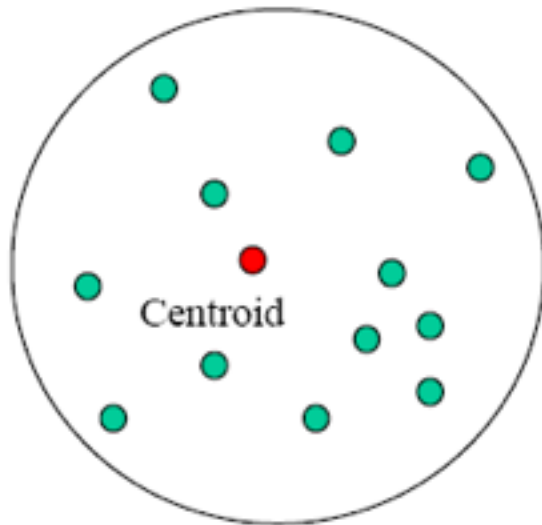
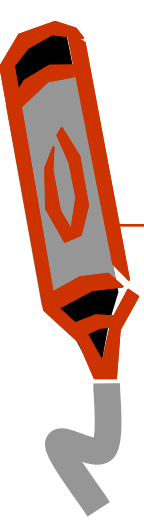
---

- Let  $(AVGSIM)_k$  be density with term  $k$  removed from documents
- Discrimination value for term  $k$  is

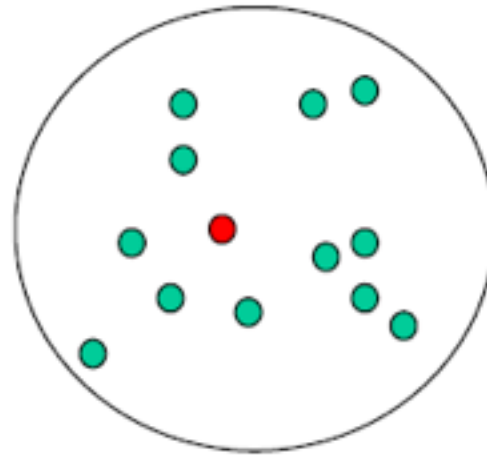
$$DISCVALUE_k = (AVGSIM)_k - AVGSIM$$

- Good discriminators have positive  $DISCVALUE_k$ 
  - introduction of term decreases the density (moves some docs away)
  - tend to be medium frequency
- Indifferent discriminators have  $DISCVALUE$  near zero
  - introduction of term has no effect
  - tend to be low frequency
- Poor discriminators have negative  $DISCVALUE$ 
  - introduction of term increases the density (moves all docs closer)
  - tend to be high frequency
- Obvious criticism is that discrimination of *relevant* and *nonrelevant* documents is the important factor

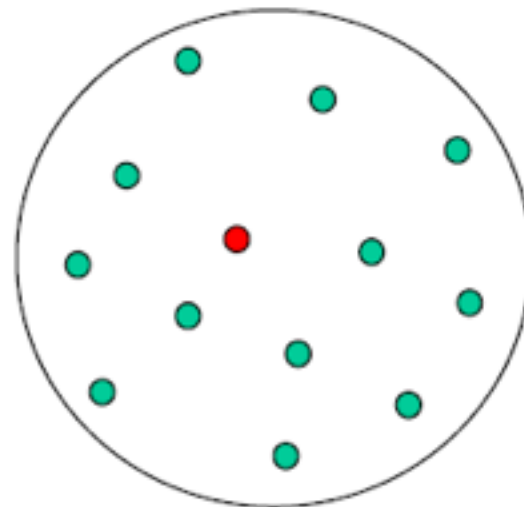
# term discrimination model



Document space with all terms



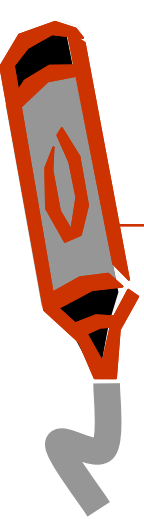
After removal of a good discriminator



After removal of a poor discriminator



# term discrimination model



Cranfield 424	MED 450	Time 425
<b>Best Discriminators</b>		
panel flutter jet cone separate shell yaw nozzle transit degree	marrow Amyloidosis Lymphostasis Hepatitis Hela antigan chromosome irradiate tumor virus	Buddhist Diem Lao Arab Viet Kurd Wilson Baath Park Nenni
<b>Worst Discriminators</b>		
equate theo bound effect solution method press result number flow	clinic children act high develop treat increase result cell patient	work lead Red minister nation party commune U.S. govern new



# summary

---

- Index model identifies how to represent documents
  - Manual
  - Automatic
- Typically consider content-based indexing
  - Using features that occur within the document
- Identifying features used to represent documents
  - Words, phrases, concepts, ...
- Normalizing them if needed
  - Stopping, stemming, ...
- Assigning a weight (significance) to them
  - TF·IDF, discrimination value
- Some decisions determined by retrieval model
  - E.g., language modeling incorporates “weighting” directly