IT Systems Engineering | Universität Potsdam

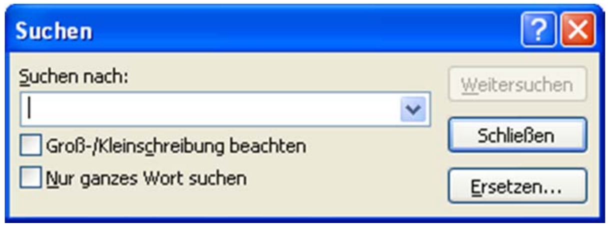
Search Engines
Chapter 4 – Processing Text

3.5.2011

Felix Naumann

Processing Text

- Converting documents to *index terms*
  - *"Text processing"* or *"Text transformation"*
- Easy: Do nothing



- Why?
  - Matching the exact string of characters typed by the user is too restrictive.
    - Poor effectiveness
  - Not all words are of equal value in a search.
  - Sometimes not clear where words begin and end
    - Not even clear what a word is in some languages
      - e.g., Chinese, Korean

Processing Text

- NLP (natural language processing)
  - Syntactic analysis
  - Semantic analysis
- Text statistics
  - Counting words
  - Counting co-occurrences
- Many simple techniques
  - Lower case
  - Punctuation
  - Tokenization
  - Stopping
  - Stemming
  - Structure and format
  - Links
- But profound impact

Overview

- Text statistics
- Document parsing
- Link Analysis
- Information Extraction

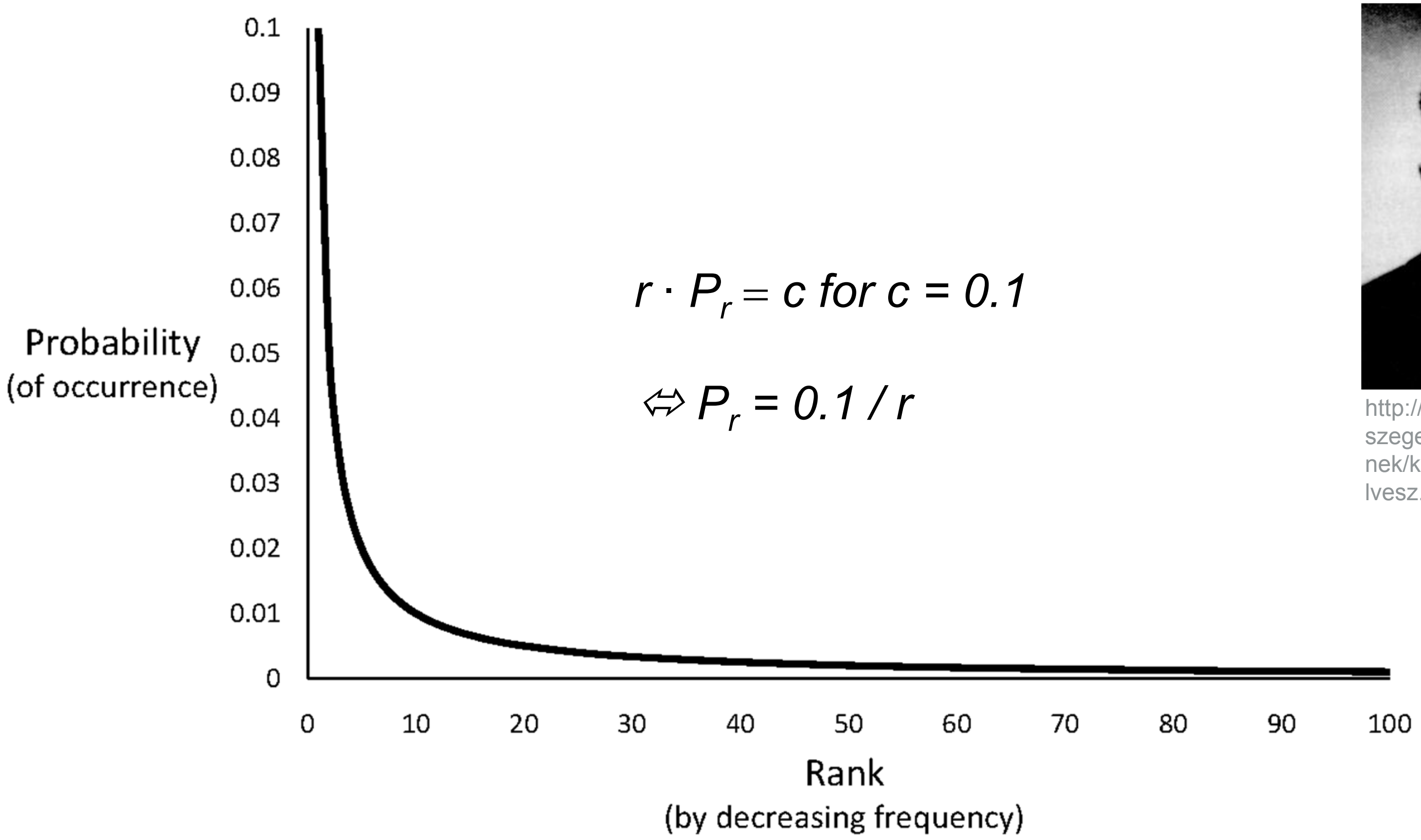
Text Statistics

- Huge variety of words used in text but…
- Many statistical characteristics of word occurrences are predictable
  - e.g., distribution of word counts
- Retrieval models and ranking algorithms depend heavily on statistical properties of words.
  - e.g., important words occur often in a document but are not of high frequency in entire collection
  - *tf-idf* (term-frequency – inverse-document-frequency)

Zipf's Law

- Distribution of word frequencies is very *skewed*.
  - A few words occur very often, many words hardly ever occur
  - Two most common words ("*the*", "*of*") make up about 10% of all word occurrences in text documents
  - Top 6 words account for 20% of text.
  - Top 50 words account for 40% of text.
  - And: 50% of all words in a large sample occur only once.
- Zipf's "law":
  - Observation that rank *r* of a word times its frequency *f* is approximately a constant *k*.
    - Assuming words are ranked in order of decreasing frequency
  - $r \cdot f \approx \mathrm{k}$ or $r \cdot P_r \approx c$
    - where P_r is probability of word occurrence and $c \approx 0.1$ for English

Zipf's Law



$r \cdot P_r = c$ *for* $c = 0.1$

$\Leftrightarrow P_r = 0.1 / r$

George Kingsley Zipf
(1902–1950)

http://www.lib.jgypk.u-szeged.hu/alknyelv/idegenek/klasszikusok/Zipf/nyelvesz.htm

News Collection (AP89) Statistics

- Associated Press from 1989

Total documents	84,678
Total word occurrences	39,749,179
Vocabulary size	198,763
Words occurring > 1000 times	4,169
Words occurring once	70,064

Top 50 Words from AP89

Word	Freq.	r	$P_r(\%)$	$r.P_r$	Word	Freq	r	$P_r(\%)$	$r.P_r$
the	2,420,778	1	6.49	0.065	has	136,007	26	0.37	0.095
of	1,045,733	2	2.80	0.056	are	130,322	27	0.35	0.094
to	968,882	3	2.60	0.078	not	127,493	28	0.34	0.096
a	892,429	4	2.39	0.096	who	116,364	29	0.31	0.090
and	865,644	5	2.32	0.120	they	111,024	30	0.30	0.089
in	847,825	6	2.27	0.140	its	111,021	31	0.30	0.092
said	504,593	7	1.35	0.095	had	103,943	32	0.28	0.089
for	363,865	8	0.98	0.078	will	102,949	33	0.28	0.091
that	347,072	9	0.93	0.084	would	99,503	34	0.27	0.091
was	293,027	10	0.79	0.079	about	92,983	35	0.25	0.087
on	291,947	11	0.78	0.086	i	92,005	36	0.25	0.089
he	250,919	12	0.67	0.081	been	88,786	37	0.24	0.088
is	245,843	13	0.65	0.086	this	87,286	38	0.23	0.089
with	223,846	14	0.60	0.084	their	84,638	39	0.23	0.089
at	210,064	15	0.56	0.085	new	83,449	40	0.22	0.090
by	209,586	16	0.56	0.090	or	81,796	41	0.22	0.090
it	195,621	17	0.52	0.089	which	80,385	42	0.22	0.091
from	189,451	18	0.51	0.091	we	80,245	43	0.22	0.093
as	181,714	19	0.49	0.093	more	76,388	44	0.21	0.090
be	157,300	20	0.42	0.084	after	75,165	45	0.20	0.091
were	153,913	21	0.41	0.087	us	72,045	46	0.19	0.089
an	152,576	22	0.41	0.090	percent	71,956	47	0.19	0.091
have	149,749	23	0.40	0.092	up	71,082	48	0.19	0.092
his	142,285	24	0.38	0.092	one	70,266	49	0.19	0.092
but	140,880	25	0.38	0.094	people	68,988	50	0.19	0.093

r x P_r value always close to 0.1

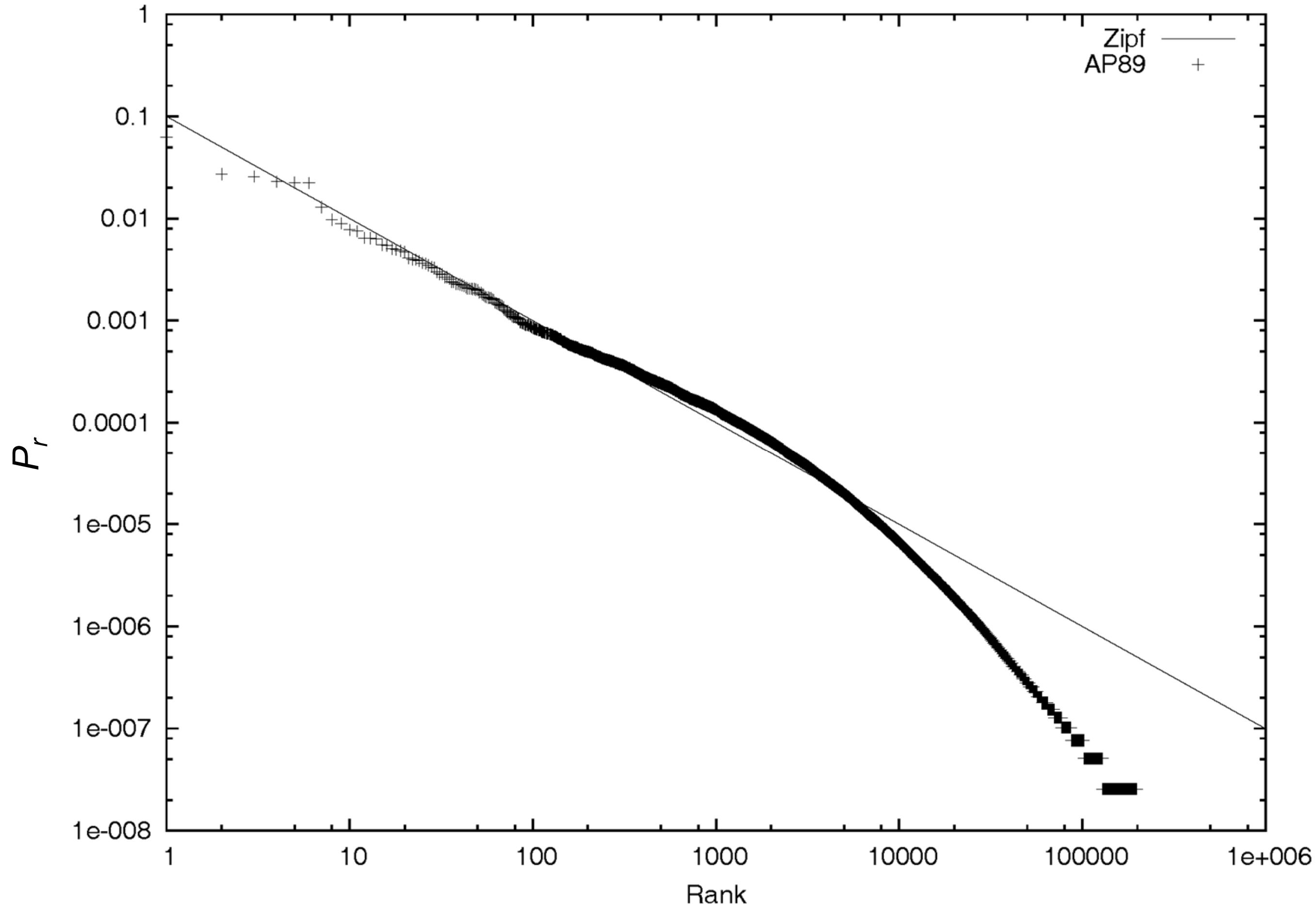
Low frequency words from AP89

- Zipf is most inaccurate for very frequent and very infrequent words.

Word	Freq.	r	$P_r(\%)$	$r.P_r$
the	2,420,778	1	6.49	0.065
of	1,045,733	2	2.80	0.056
to	968,882	3	2.60	0.078
a	892,429	4	2.39	0.096
and	865,644	5	2.32	0.120
in	847,825	6	2.27	0.140
said	504,593	7	1.35	0.095
for	363,865	8	0.98	0.078
that	347,072	9	0.93	0.084
was	293,027	10	0.79	0.079
on	291,947	11	0.78	0.086

Word	Freq.	r	Pr(%)	r.Pr
Assistant	5,095	1,021	.013	0.13
Sewers	100	17,110	2.56×10^{-4}	0.04
Toothbrush	10	51,555	2.56×10^{-5}	0.01
Hazmat	1	166,945	2.56×10^{-6}	0.04

Zipf's Law for AP89



Note problems at high and low frequencies

Zipf's Law – example calculations

- Reminder: $r \cdot f \approx$ k
- What is the proportion of words with a given frequency?
  - Word that occurs n times has rank $r_n = k/n$
  - Multiple words can have same frequency
    - r_n is associated with last word in group
  - Number of words with same frequency n is
    - $r_n - r_{n+1} = k/n - k/(n+1) = k/n(n+1)$
  - Proportion found by dividing by total number of words
    - = rank of last word with freq. 1 = highest rank = $k/1 = k$
  - So, proportion with frequency n is $1/n(n+1)$
    - => half of all words appear once
      - (n=1 => proportion = ½)

Zipf's Law – example calculation

- Example word frequency ranking

Rank	*Word*	*Frequency*
1000	concern	5,100
1001	spoke	5,100
1002	summit	5,100
1003	bring	5,099
1004	star	5,099
1005	immediate	5,099
1006	chemical	5,099
1007	african	5,098

- To compute number of words with frequency 5,099
  - rank of "chemical" minus the rank of "summit"
  - 1006 − 1002 = 4
- Proportion: $1/n(n+1) = 1/5{,}099(5{,}100) = 1/\ 26{,}004{,}900$

Example

Number of Occurrences (n)	*Predicted Proportion (1/n(n+1))*	*Actual Proportion*	*Actual Number of Words*
1	.500	.402	204,357
2	.167	.132	67,082
3	.083	.069	35,083
4	.050	.046	23,271
5	.033	.032	16,332
6	.024	.024	12,421
7	.018	.019	9,766
8	.014	.016	8,200
9	.011	.014	6,907
10	.009	.012	5,893

- Proportions of words occurring n times in 336,310 TREC documents
- Vocabulary size is 508,209

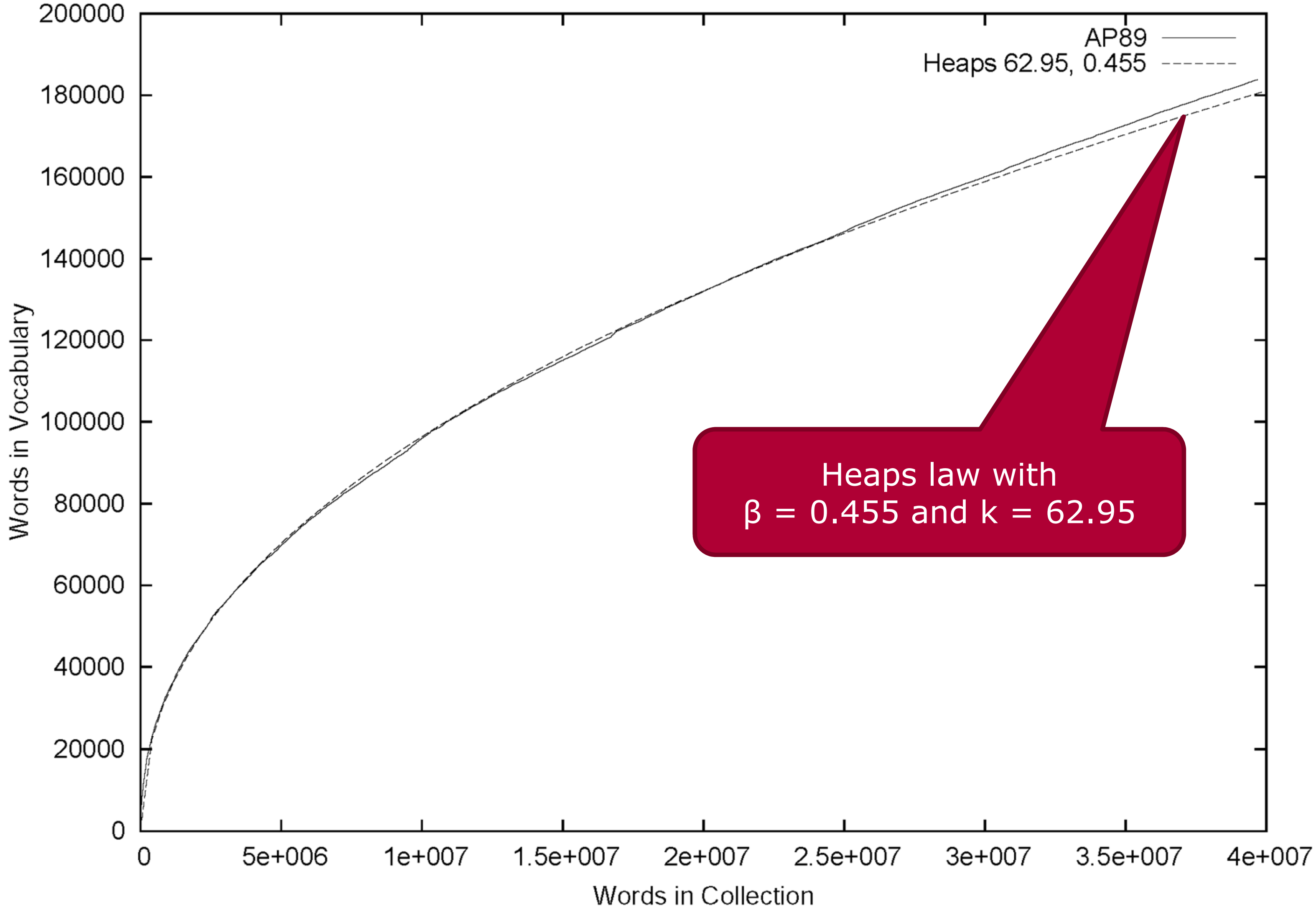
Vocabulary Growth

- As corpus grows, so does vocabulary size
  - But: Fewer new words when corpus is already large
- Observed relationship (*Heaps' Law, found empirically*):

$$v = k \cdot n^{\beta}$$

  - where v is vocabulary size (number of unique words)
  - n is the number of words in corpus (non-unique)
  - k, β are parameters that vary for each corpus
    - typical values given are $10 \leq k \leq 100$ and $\beta \approx 0.5$
- Example
  - $n = 1{,}000{,}000$ $k = 50$ $\beta = 0.5$
  - $v = 50 \cdot 1{,}000{,}000^{0.5} = 50{,}000$

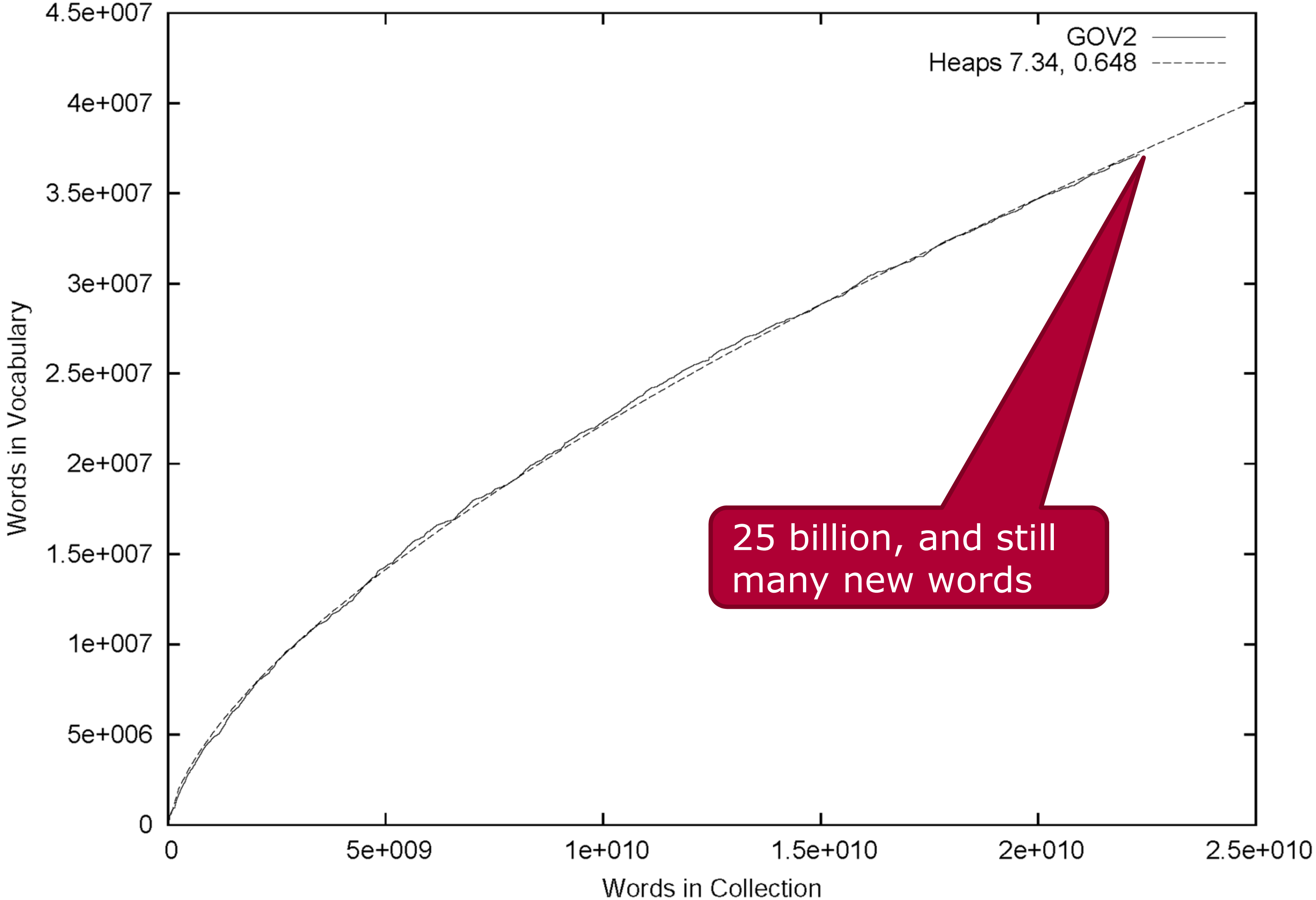
TREC AP89 Example



Heaps' Law Predictions

- Predictions for TREC collections are accurate for large numbers of words.
  - E.g., first 10,879,522 words of the AP89 collection scanned
  - Prediction is 100,151 unique words
  - Actual number is 100,024
- Predictions for small numbers of words (i.e. < 1,000) are much worse.

GOV2 (Web) Example



Web Example

- Heaps' Law works with very large corpora
  - New words occurring even after seeing 30 million!
  - Parameter values on Web different than typical TREC values
- New words come from a variety of sources
  - Spelling errors, invented words (e.g., product, company names), code, other languages, email addresses, etc.
- Search engines must deal with these large and growing vocabularies

Estimating Result Set Size

Hasso Plattner Institut

tropical fish aquarium | Search

Web results Page 1 of 3,880,000 results

- How many pages contain *all* of the query terms?
  - Not always conjunctive semantics
- For the query "*a b c*":

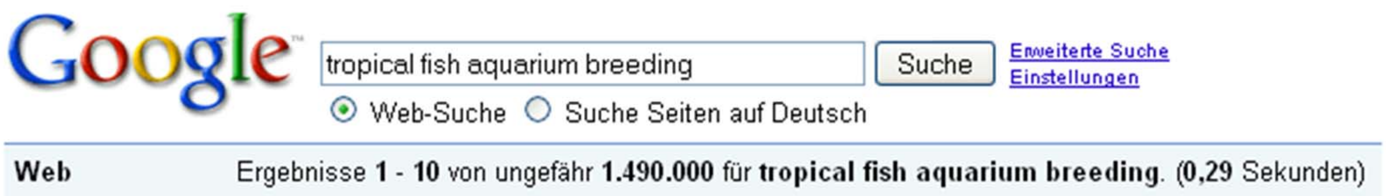
$$f_{abc} = N \cdot f_a/N \cdot f_b/N \cdot f_c/N = (f_a \cdot f_b \cdot f_c)/N^2$$

  - Assuming that terms occur independently
  - f_{abc} is the estimated size of the result set
  - f_a, f_b, f_c are the number of documents that terms *a*, *b*, and *c* occur in
    - Available through index
    - Document frequency (not word occurrences)
  - N is the number of documents in the collection

TREC GOV2 Example

Word(s)	*Document Frequency*	*Estimated Frequency*
tropical	120,990	
fish	1,131,855	
aquarium	26,480	
breeding	81,885	
tropical fish	18,472	5,433
tropical aquarium	1,921	127
tropical breeding	5,510	393
fish aquarium	9,722	1,189
fish breeding	36,427	3,677
aquarium breeding	1,848	86
tropical fish aquarium	1,529	6
tropical fish breeding	3,629	18

Collection size (*N*) is 25,205,179



Result Set Size Estimation

- Poor estimates because words are not independent
  - e.g., fish and aquarium
- Better estimates possible if pair-wise co-occurrence information is available:
  - $P(a \cap b \cap c) = P(a \cap b) \cdot P(c|(a \cap b))$
  - Approximate $P(c|(a \cap b))$ with $\max[P(c|a), P(c|b)]$.
    - Reminder: $P(c|a) = P(c \cap a)/P(a)$
  - $f_{tropical \cap fish \cap aquarium} = f_{tropical \cap aquarium} \cdot f_{fish \cap aquarium} / f_{aquarium}$
    $= 1921 \cdot 9722/26480 = 705$
  - $f_{tropical \cap fish \cap breeding} = f_{tropical \cap breeding} \cdot f_{fish \cap breeeding} / f_{breeding}$
    $= 5510 \cdot 36427/81885 = 2451$
- Still too low, because still some independence assumptions.
  - But: Storing deeper co-occurrence (triples, quadruples, …) is too expensive.

Result Set Estimation – New Idea

- Even better estimates using *initial result set* during processing
  - Estimate is simply C/s, where
    - s is the proportion of the total documents that have been ranked.
    - C is the number of documents found that contain all of the query words.
- E.g., "tropical fish aquarium" in GOV2
  - After processing 3,000 out of the 26,480 documents that contain "aquarium", $C = 258$
    $f_{tropical \cap fish \cap aquarium} = 258/(3000 \div 26480) = 2{,}277$
    $(= 26480 \cdot 258/3000)$
  - After processing 20% of the documents
    $f_{tropical \cap fish \cap aquarium} = 1{,}778$ (1,529 is real value)
- Total number of documents in collection irrelevant here

Estimating Collection Size

- Important issue for Web search engines
  - Academia: How big is the web?
  - Business: Which search engine has best coverage?
- Simple technique: Use independence model
  - Given two words *a* and *b* that are (probably) independent

    $f_{ab}/N = f_a/N \cdot f_b/N$

    $N = (f_a \cdot f_b)/f_{ab}$

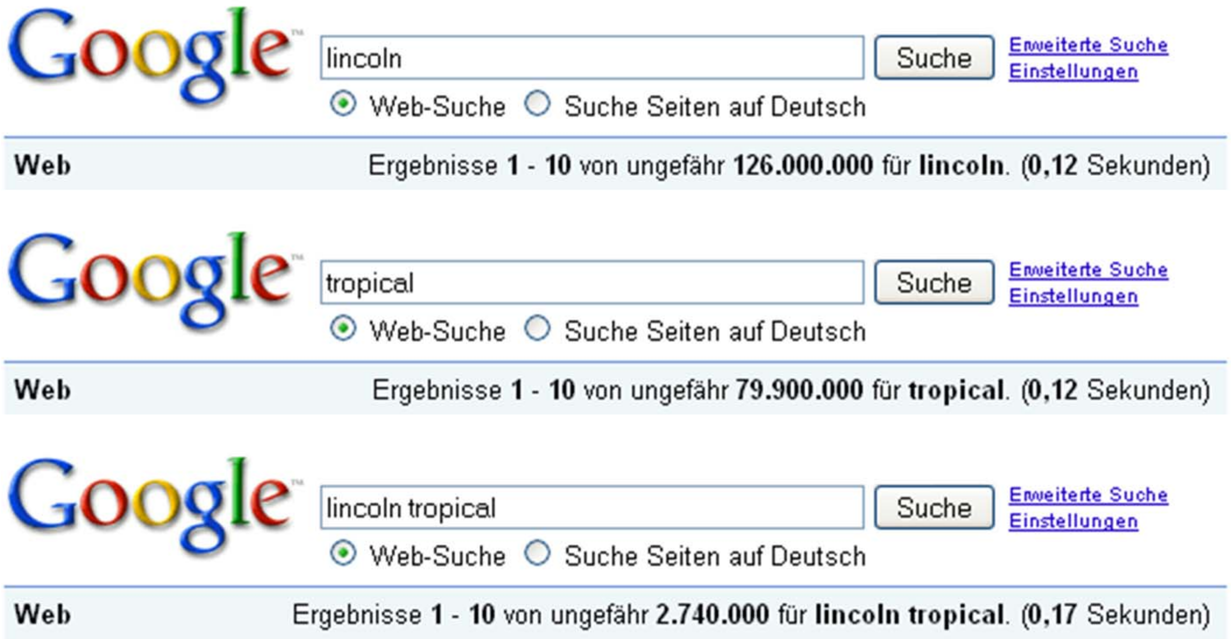
  - e.g., for GOV2

    $f_{lincoln}$ = 771,326 $f_{tropical}$ = 120,990 $f_{lincoln \cap tropical}$ = 3,018

    N = (120990 · 771326)/3018 = 30,922,045

    (actual number is 25,205,179)

Estimating Google's Size (*GS*) 2009



GS = (126,000,000 · *79,900,000*) / 2,740,000 = 3,674,233,577

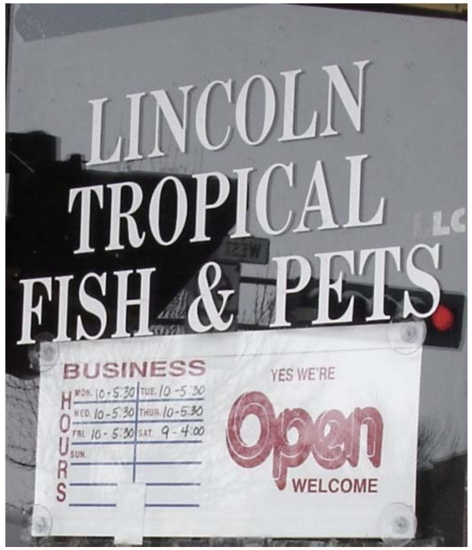
Actual size: 1,000,000,000,000

Estimating Google's Size (*GS*) 2011

Google: lincoln
About 225,000,000 results (0.18 seconds)

Google: tropical
About 145,000,000 results (0.10 seconds)

Google: lincoln tropical
About 15,500,000 results (0.19 seconds)



Google: 225,000,000 x 145,000,000 / 15500000 — Search
Advanced search

Everything
More

$(225\,000\,000 \times 145\,000\,000) / 15\,500\,000 = 2.10483871 \times 10^9$
More about calculator.

= 2 billion (2,104,838,710)

Google: a
About 25,270,000,000 results (0.14 seconds)

Overview

- Text statistics
- Document parsing
- Link Analysis
- Information Extraction

Motivation

- Document parsing =
  Recognition of *content* and *structure* of document
- Tokenizing / lexical analysis =
  Recognition of words in sequence of characters
- Syntactic analysis =
  Recognition of structure for content
  - Uses markup
- Parsing very tolerant – represent every document in index!

- Input: Result of crawling – textual representation of web page
  - With some markup
- Output: Data structure used for index

Tokenizing

- Forming words from sequence of characters
- Surprisingly complex in English, can be harder in other languages
- Early IR systems:
  - Any sequence of alphanumeric characters of length > 3
  - Terminated by a space or other special character
  - Any upper-case changed to lower-case (aka. *case-folding* or *downcasing*)
- Example:
  - "*Bigcorp's 2007 bi-annual report showed profits rose 10%.*"
  - becomes "*bigcorp 2007 annual report showed profits rose*"
- Too simple for search applications or even large-scale experiments
- Why? Too much information lost
  - Small decisions in tokenizing can have major impact on effectiveness of some queries.

Tokenizing Problems

- Small words can be important in some queries, usually in combinations
  - *xp, ma, pm, ben e king, el paso, system r*
  - *master p, gm, j lo, world war II*
- Both hyphenated and non-hyphenated forms of many words are common
  - Sometimes hyphen is *not needed*
    - *e-bay, wal-mart, active-x, cd-rom, t-shirts*
  - Sometimes hyphens *should be considered* either as part of the word or a word separator
    - *winston-salem, mazda rx-7, e-cards, pre-diabetes, t-mobile, spanish-speaking*

Tokenizing Problems

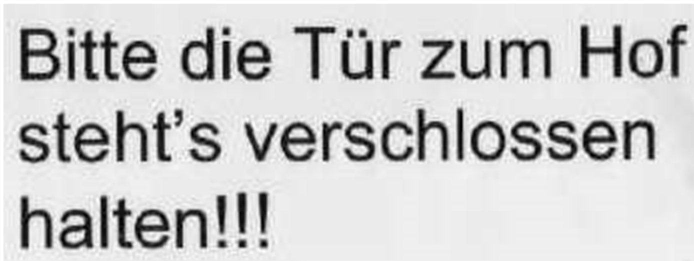
- Special characters are an important part of
  tags, URLs, code in documents, …

- Capitalized words can have different meaning from lower case words
  - *Bush, Apple*
  - *bush, apple*

- Apostrophes can be a part of a word, a part of a possessive, or just a mistake
  - *rosie o'donnell, can't, don't, 80's, 1890's, men's straw hats, master's degree, england's ten largest cities, shriner's*

Die Kapostroph-Gruselgalerie – Kategore „Völlig willenlos“

http://www.apostroph.de/



Tokenizing Problems

- Numbers can be important, including decimals
  - *nokia 3250, top 10 courses, united 93, quicktime 6.5 pro, 92.3 the beat, 288358*
- Periods can occur in numbers, abbreviations, URLs, ends of sentences, and other situations
  - *I.B.M., Ph.D., cs.umass.edu, F.E.A.R.*

- Note: Tokenizing steps for queries (later) must be identical to steps for documents

Tokenizing Process

- Step 1: Parse for markup
  - Allow for syntax errors
  - Identify appropriate parts of document to tokenize
- Step 2: Parse for content
  - Defer complex decisions to other components
    - Stemming, dates, NER
  - Word is any sequence of alphanumeric characters, terminated by a space or special character, with everything converted to lower-case
    - Let query transformation component deal with ambiguities
  - Example: 92.3 → 92 3 but search finds documents with 92 and 3 adjacent
  - Incorporate additional rules to handle some special characters (so query and document will match).

Tokenizing Process

Hasso Plattner Institut

- Not that different than simple tokenizing process used in past
- Examples of rules used with TREC
  - Apostrophes in words ignored
    - o’connor → oconnor bob’s → bobs
  - Periods in abbreviations ignored
    - I.B.M. → ibm
    - But Ph.D. → ph d because ph is word not character.

Stopping

- Function words (determiners, prepositions) have little meaning on their own
  - Determiners: The, a, an, that, those, …
  - Prepositions: Over, under, above, below, …
- High occurrence frequencies
- Little relevance (except for phrases)
- Treated as *stopwords* (i.e., removed)
  - Reduce index space
  - Improve response time
  - Improve effectiveness
- Can be important in combinations
  - e.g., “to be or not to be”

Word	*Freq.*	*r*	$P_r(\%)$	$r.P_r$	*Word*	*Freq*	*r*	$P_r(\%)$	$r.P_r$
the	2,420,778	1	6.49	0.065	has	136,007	26	0.37	0.095
of	1,045,733	2	2.80	0.056	are	130,322	27	0.35	0.094
to	968,882	3	2.60	0.078	not	127,493	28	0.34	0.096
a	892,429	4	2.39	0.096	who	116,364	29	0.31	0.090
and	865,644	5	2.32	0.120	they	111,024	30	0.30	0.089
in	847,825	6	2.27	0.140	its	111,021	31	0.30	0.092
said	504,593	7	1.35	0.095	had	103,943	32	0.28	0.089
for	363,865	8	0.98	0.078	will	102,949	33	0.28	0.091
that	347,072	9	0.93	0.084	would	99,503	34	0.27	0.091
was	293,027	10	0.79	0.079	about	92,983	35	0.25	0.087
on	291,947	11	0.78	0.086	i	92,005	36	0.25	0.089
he	250,919	12	0.67	0.081	been	88,786	37	0.24	0.088
is	245,843	13	0.65	0.086	this	87,286	38	0.23	0.089
with	223,846	14	0.60	0.084	their	84,638	39	0.23	0.089
at	210,064	15	0.56	0.085	new	83,449	40	0.22	0.090
by	209,586	16	0.56	0.090	or	81,796	41	0.22	0.090
it	195,621	17	0.52	0.089	which	80,385	42	0.22	0.091
from	189,451	18	0.51	0.091	we	80,245	43	0.22	0.093
as	181,714	19	0.49	0.093	more	76,388	44	0.21	0.090
be	157,300	20	0.42	0.084	after	75,165	45	0.20	0.091
were	153,913	21	0.41	0.087	us	72,045	46	0.19	0.089
an	152,576	22	0.41	0.090	percent	71,956	47	0.19	0.091
have	149,749	23	0.40	0.092	up	71,082	48	0.19	0.092
his	142,285	24	0.38	0.092	one	70,266	49	0.19	0.092
but	140,880	25	0.38	0.094	people	68,988	50	0.19	0.093

Stopping

- Stopword list can be created from high-frequency words or based on a standard list
  - With caution
- Lists are customized for applications, domains, and even parts of documents.
  - E.g., "click" is a good stopword for anchor text
- Best policy is to index all words in documents, and then make decisions about which words to use at query time.
  - Stopwords are removed from query, except with "+"-sign
  - But: Space consuming

Stemming

- Also: "Conflation"
- Many morphological variations of words
  - *inflectional* (plurals, tenses)
    - Flexion, Beugung: Kasus, Numerus, Genus, Tempus
  - *derivational* (making verbs nouns etc.)
    - Ableitung und Zusammensetzung (Komposition)
- In most cases, these have the same or very similar meanings
- Stemmers attempt to reduce morphological variations of words to a common stem
  - Usually involves removing suffixes
- Can be done at indexing time or as part of query processing (like stopwords).

Stemming

- Generally a small but significant effectiveness improvement
  - can be crucial for some languages
  - e.g., 5-10% improvement for English, up to 50% in Arabic

kitab	*a book*
kitabi	*my book*
al**kitab**	*the book*
kitabuki	*your book* (f)
kitabuka	*your book (m)*
kitabuhu	*his book*
ka**t**a**b**a	*to write*
ma**kt**a**b**a	*library, bookstore*
ma**ktab**	*office*

Words with the Arabic root **ktb**

Stemming

- Two basic types of stemmers
  - Dictionary-based: uses lists of related words
  - Algorithmic: uses program to determine related words
- Algorithmic stemmers
  - *suffix-s:* remove 's' endings assuming plural
    - e.g., cats → cat, lakes → lake, wiis → wii
    - Many *false negatives*: supplies → supplie
    - Some *false positives*: ups → up
- More complex stemmers include more endings
  - -ing, -ed
  - Fewer false negatives, more false positives

Porter Stemmer

- Algorithmic stemmer used in IR experiments since the 70s
- Consists of a series of rules
  - Find the longest possible suffix at each step
  - Some non-intuitive
- Effective in TREC
- Produces *stems* not *words*
- Makes a number of errors and difficult to modify

Martin Porter: http://tartarus.org/~martin/

Porter Stemmer: Example step (1 of 5)

Hasso Plattner Institut

Step 1a:

- Replace ***sses*** by ***ss*** (e.g., stresses → stress).
- Delete ***s*** if the preceding word part contains a vowel not immediately before the ***s*** (e.g., gaps → gap but gas → gas).
- Replace ***ied*** or ***ies*** by ***i*** if preceded by more than one letter, otherwise by ***ie*** (e.g., ties → tie, cries → cri).
- If suffix is ***us*** or ***ss*** do nothing (e.g., stress → stress).

Step 1b:

- Replace ***eed***, ***eedly*** by ***ee*** if it is in the part of the word after the first non-vowel following a vowel (e.g., agreed → agree, feed → feed).
- Delete ***ed***, ***edly***, ***ing***, ***ingly*** if the preceding word part contains a vowel, and then if the word ends in ***at***, ***bl***, or ***iz*** add ***e*** (e.g., fished → fish, pirating → pirate), or if the word ends with a double letter that is not ***ll***, ***ss*** or ***zz***, remove the last letter (e.g., falling→ fall, dripping → drip), or if the word is short, add ***e*** (e.g., hoping → hope).

Porter Stemmer

- Some errors of Porter stemmer

False positives	*False negatives*
organization/organ	european/europe
generalization/generic	cylinder/cylindrical
numerical/numerous	matrices/matrix
policy/police	urgency/urgent
university/universe	create/creation
addition/additive	analysis/analyses
negligible/negligent	useful/usefully
execute/executive	noise/noisy
past/paste	decompose/decomposition
ignore/ignorant	sparse/sparsity
special/specialized	resolve/resolution
head/heading	triangle/triangular

- Porter2 stemmer addresses some of these issues
- Approach has been used with other languages

Dictionary-based Stemmers

- Word-relationships stored explicitly
- Difficult cases are caught
  - Is, be, was
  - Few false positives
- But: Language evolves
- Observation
  - Old words are irregular
  - Newer words are more regular
- Thus: Hybrid approach
  - Dictionary-based for old/difficult words
  - Algorithmic-based for new words

Krovetz Stemmer

- Hybrid algorithmic-dictionary
  - Word checked in dictionary and exception set
    - If present, either left alone or replaced with “exception”
    - If not present, word is checked for suffixes that could be removed
    - After removal, dictionary is checked again
    - If still not present, different endings are tried
- Produces words not stems
- Comparable effectiveness
- Lower false positive rate, somewhat higher false negative

Stemmer Comparison

- 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
  - *document describ market strategi carri compani agricultur chemic report predict market share chemic report market statist agrochem pesticid herbicid fungicid insecticid fertil predict sale market share stimul demand price cut volum sale*

Stems

- Krovetz stemmer
  - *document describe 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*

Words (mostly)

Phrases

- Many queries are 2-3 word phrases.
- Phrases are
  - more precise than single words
    - e.g., documents containing "*black sea*" vs. two words "*black*" and "*sea*"
  - less ambiguous
    - e.g., "*big apple*" vs. "*rotten apple*" vs. "*apple*"
- Can be difficult for ranking
    - e.g., given query "*fishing supplies*", how do we score documents with
      - exact phrase many times
      - exact phrase just once
      - individual words in same sentence, same paragraph, whole document
      - variations on words?

Phrases

- Ranking: See retrieval model
  - But: Deal with phrases during text processing?
- Text processing issue – how are phrases recognized?
- Three possible approaches:
  - Identify syntactic phrases using a *part-of-speech* (POS) tagger.
  - Use word *n-grams*.
  - Store word positions in indexes and use *proximity operators* in queries.

POS Tagging

- POS taggers use statistical models or rule-based models of text to predict syntactic tags of words
- Trained on large corpora
  - Example tags:
    - **NN** (singular noun), **NNS** (plural noun), **VB** (verb), **VBD** (verb, past tense), **VBN** (verb, past participle), **IN** (preposition), **JJ** (adjective), **CC** (conjunction, e.g., "*and*", "*or*"), **PRP** (pronoun), and **MD** (modal auxiliary, e.g., "*can*", "*will*").
- Phrases can then be defined as simple noun groups (*noun phrase*)
  - Or simpler: Sequence of nouns, or nouns plus adjective
- Disadvantage: Slow

Pos Tagging 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.*
- Brill tagger
  - *Document/NN will/MD describe/VB* ***marketing/NN strategies/NNS*** *carried/VBD out/IN by/IN U.S./NNP companies/NNS for/IN their/PRP* ***agricultural/JJ chemicals/NNS*** *,/, report/NN predictions/NNS for/IN market/NN share/NN of/IN such/JJ chemicals/NNS ,/, or/CC report/NN market/NN statistics/NNS for/IN agrochemicals/NNS ,/, pesticide/NN ,/, herbicide/NN ,/, fungicide/NN ,/, insecticide/NN ,/, fertilizer/NN ,/,* ***predicted/VBN sales/NNS*** *,/, market/NN share/NN ,/, stimulate/VB demand/NN ,/, price/NN cut/NN ,/, volume/NN of/IN sales/NNS ./.*

Noun phrase

Noun phrase

Not recognized as noun phrase

http://research.microsoft.com/en-us/um/people/brill/

Example Noun Phrases

TREC data		Patent data	
Frequency	*Phrase*	*Frequency*	*Phrase*
65824	united states	975362	present invention
61327	article type	191625	u.s. pat
33864	los angeles	147352	preferred embodiment
18062	hong kong	95097	carbon atoms
17788	north korea	87903	group consisting
17308	new york	81809	room temperature
15513	san diego	78458	seq id
15009	orange county	75850	brief description
12869	prime minister	66407	prior art
12799	first time	59828	perspective view
12067	soviet union	58724	first embodiment
10811	russian federation	56715	reaction mixture
9912	united nations	54619	detailed description
8127	southern california	54117	ethyl acetate
7640	south korea	52195	example 1
7620	end recording	52003	block diagram
7524	european union	46299	second embodiment
7436	south africa	41694	accompanying drawings
7362	san francisco	40554	output signal
7086	news conference	37911	first end
6792	city council	35827	second end
6348	middle east	34881	appended claims
6157	peace process	33947	distal end
5955	human rights	32338	cross-sectional view
5837	white house	30193	outer surface

Many proper nouns

Many topical phrases

Fewer content related

Word positions

- POS tagging too slow for large collections
- Instead: Store word position information in index
- Identify phrases only when query is processed
- More flexible in types of phrases
  - Not restricted to adjacent words
  - Identification of phrases using proximity / occurrence within a window

- Indexing positions and retrieval model for positions: Later

Word N-Grams

- Simpler definition – phrase is any sequence of *n* words – known as *n-grams*
  - *bigram*: 2 word sequence, *trigram*: 3 word sequence, *unigram*: single words
  - N-grams also used at character level for applications such as OCR
  - Also useful for indexing Chinese text
- N-grams typically formed from *overlapping* sequences of words
  - i.e., move n-word "window" one word at a time in document
- Indexes grow larger

N-Grams

- Frequent n-grams are more likely to be meaningful phrases
- N-grams also form a Zipf distribution
  - Better fit than words alone
- Could index all n-grams up to specific length
  - Much faster than POS tagging
  - Uses a lot of storage:
    - Document containing 1,000 words would contain 3,990 instances of word n-grams of length $2 \leq n \leq 5$
  - Remove stopword n-grams: "*and the*", "*there is*", …
    - But again: "*to be or not to be*"

Google N-Grams
"All Our N-gram are Belong to You"

- Web search engines index n-grams
- Google sample (http://googleresearch.blogspot.com/2006/08/all-our-n-gram-are-belong-to-you.html):
  - Number of tokens: 1,024,908,267,229
  - Number of sentences: 95,119,665,584
  - Number of unigrams: 13,588,391
  - Number of bigrams: 314,843,401
  - Number of trigrams: 977,069,902
  - Number of fourgrams: 1,313,818,354
  - Number of fivegrams: 1,176,470,663

- Most frequent trigram in English is "*all rights reserved*"
  - In Chinese, "*limited liability corporation*"
  - Not dominated by patterns of speech ("*and will be*")

Document Structure and Markup

- Some parts of documents are more important than others.
  - Similar to databases: Column-names
- Document parser recognizes structure using markup, such as HTML tags
  - Headers, anchor text, bolded text all likely to be important
  - Metadata can also be important
  - Links used for *link analysis*

Tropical fish

From Wikipedia, the free encyclopedia

Tropical fish include fish found in tropical environments around the world, including both freshwater and salt water species. Fishkeepers often use the term *tropical fish* to refer only those requiring fresh water, with saltwater tropical fish referred to as *marine fish*.

Tropical fish are popular aquarium fish , due to their often bright coloration. In freshwater fish, this coloration typically derives from iridescence, while salt water fish are generally pigmented.

Example Web Page

```
<html>
<head>
<meta name="keywords" content="Tropical fish, Airstone, Albinism, Algae eater,
Aquarium, Aquarium fish feeder, Aquarium furniture, Aquascaping, Bath treatment
(fishkeeping),Berlin Method, Biotope" />
…
<title>Tropical fish - Wikipedia, the free encyclopedia</title>
</head>
<body>
…
<h1 class="firstHeading">Tropical fish</h1>
…
<p><b>Tropical fish</b> include <a href="/wiki/Fish" title="Fish">fish</a> found in <a
href="/wiki/Tropics" title="Tropics">tropical</a> environments around the world,
including both <a href="/wiki/Fresh_water" title="Fresh water">freshwater</a> and <a
href="/wiki/Sea_water" title="Sea water">salt water</a> species. <a
href="/wiki/Fishkeeping" title="Fishkeeping">Fishkeepers</a> often use the term
<i>tropical fish</i> to refer only those requiring fresh water, with saltwater tropical fish
referred to as <i><a href="/wiki/List_of_marine_aquarium_fish_species" title="List of
marine aquarium fish species">marine fish</a></i>.</p>
<p>Tropical fish are popular <a href="/wiki/Aquarium" title="Aquarium">aquarium</a>
fish , due to their often bright coloration. In freshwater fish, this coloration typically
derives from <a href="/wiki/Iridescence" title="Iridescence">iridescence</a>, while salt
water fish are generally <a href="/wiki/Pigment" title="Pigment">pigmented</a>.</p>
…
</body></html>
```

Document Structure and Markup

- URL itself is source for words
- http://en.wikipedia.org/wiki/Tropical_fish
- Depth of URL: Where is IBM‘s homepage?
  - www.ibm.com vs.
  - www.pcworld.com/businesscenter/article/698/ibm_buys_apt!

- HTML for layout and presentation
- XML for semantic markup
  - Simple **Dublin Core Metadata Element Set**
    - Title, Creator, Subject, Description, Publisher, Contributor, Date, Type, Format, Identifier, Source, Language, Relation, Coverage, Rights
  - Geotagging
    - <meta name="geo.position" content="50.167958;-97.133185"> <meta name="geo.placename" content="Rockwood Rural Municipality, Manitoba, Canada"> <meta name="geo.region" content="ca-mb">

Overview

- Text statistics
- Document parsing
- Link Analysis
- Information Extraction

Link Analysis

- Links are a key component of the Web.
  - Relationships
- Important for navigation, but also for search
  - e.g., Example website
  - “Example website” is the anchor text.
  - “http://example.com” is the destination link.
  - Both are used by search engines.
- No relevance for desktop search

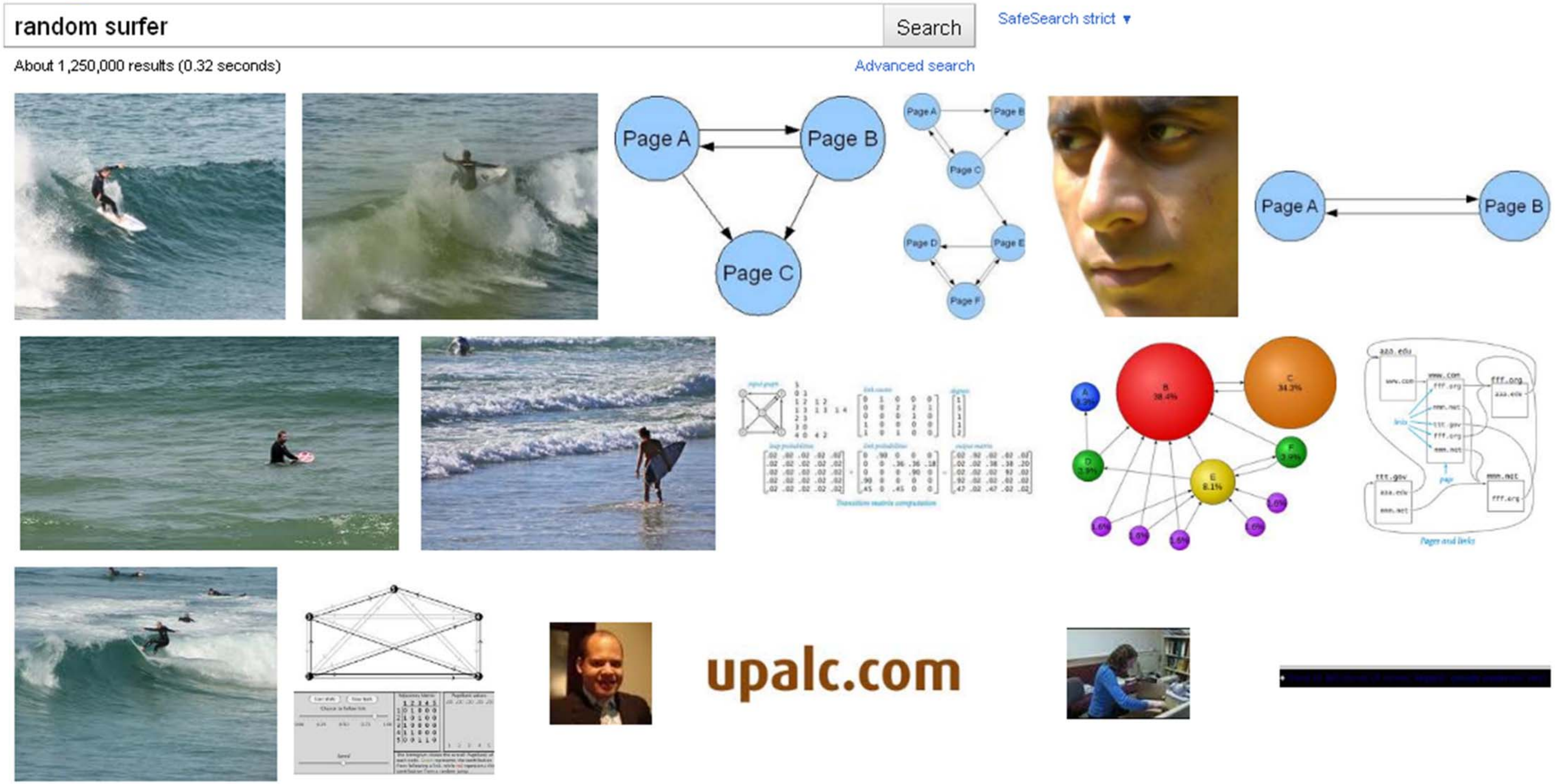
Anchor Text

- Used as a description of the content of the *destination page*
  - Collection of anchor texts in all links pointing to a page used as an additional text field
- Anchor text tends to be short, descriptive, and similar to query text.
  - ebay
  - But: click here
- Written by people who are not author of page
  - Description from a different perspective
  - Description of most important aspect
- Link itself is also a vote for importance
- Retrieval experiments have shown that anchor text has significant impact on effectiveness for *some types of queries.*
  - Especially homepages
  - More effective than PageRank

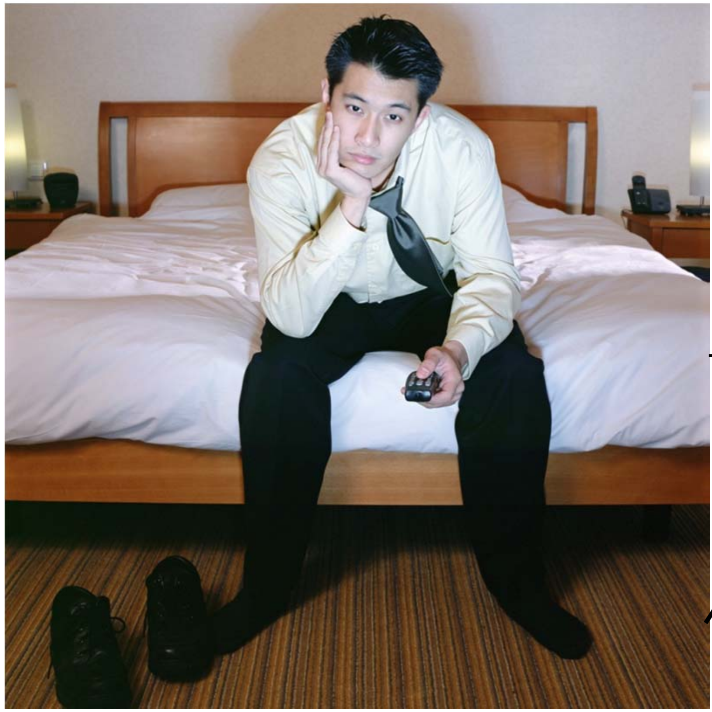
PageRank

- Tens of billions of web pages, some more informative than others
  - Spam vs. personal homepage/photo album vs. news site vs. corporate homepage
  - Ranking difficult
- Links can be viewed as information about the *popularity* (*authority*?) of a web page
  - Can be used by ranking algorithm
- *Inlink* count could be used as simple measure
  - Susceptible to link spam
- Link analysis algorithms like PageRank provide more reliable ratings
  - Less susceptible to link spam

PageRank: Random Surfer



Surfer Bob is bored



Surprise me!

?

Random link from current page

Prof. Dr. Felix Naumann
Information Systems

Home Lehre Projekte Veröffentlichungen M

uche

>> Home

Fachgebiet Informationssyst

Unter der Leitung von Prof. Dr. Felix Naumann untersucht da
Informationssysteme den effizienten und effektiven Umga
heterogenen Informationen in großen, autonomen Systemen
Methoden der Informationsintegration, der Informationsqualit
Informationssuche und des Metadatenmanagements.

Forschungsthemen des Fachgebiets

Ein Überblick über einige der Forschungsthemen bietet ein jü
Engineering Bulletin: "Data Fusion in Three Steps: Resolving
Inconsistencies" (2006). Einen weiteren Überblick bieten uns
Publikationsseiten. Seit kurzem pflegen wir eine Repeatability
Code und Daten zur Verfügung zu stellen.

Im regelmäßigen InfoLunch hören Sie mehr über die aktuelle

Aktuell

Datenqualität / Informationsqualität

Die Qualität von Daten wird in einer Vielzahl von Dimensione
können aggregiert werden, um z.B. die Qualität von Anfrage
einem jüngsten Artikel im Informatik Spektrum wird dieses F
"Aktuelles Schlagwort: Datenqualität".

Duplikaterkennung

Im Kundendatenmanagement (CRM) aber auch in anderen D
eine Person mehrfach mit leicht unterschiedlichen Werten ge
Duplikaterkennung finden effizient solche doppelten Einträge
Probleme und Verfahren der Duplikaterkennung werden hier
beschrieben.

Aktuell

Datenfusion

Für viele Anwendungen müssen erkannte Duplikate, also meh

PageRank: Random Surfer Model

- Browse the Web using the following algorithm:
  - Choose a random number *r* between 0 and 1
  - If $r < \lambda$:
    - Go to a random page
  - If $r \geq \lambda$:
    - Click a link at random on the current page
  - Start again
- "PageRank" of a page is the probability that the "random surfer" will be looking at that page
  - Links from popular pages will increase PageRank of pages they point to, because they are more often visited than non-popular pages
  - Many pages will be reached very often (thousands of time more often than others)
- λ is typically small

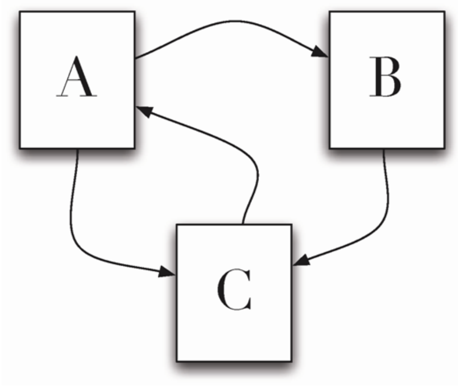
Dangling Links

- Random jump guarantees that every page will be reached at some point in time.
- Random jump prevents getting stuck on pages that
  - do not have links,
  - contain only links that no longer point to other pages, or
  - have links forming a loop.
- Links that point to the first two types of pages are called *dangling links*.
  - May also be links to pages that have not yet been crawled

- Problem: Bob does not have enough time…

PageRank – Random Link

- PageRank (*PR*) of page C:
  $PR(C) = PR(A)/2 + PR(B)/1$



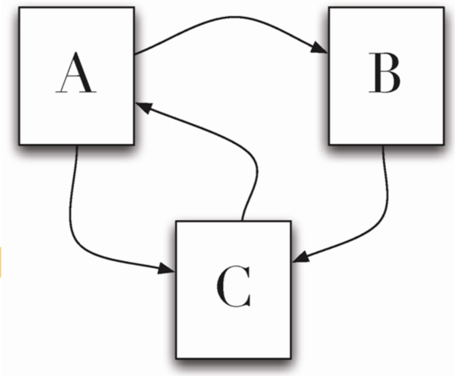
- More generally,

$$PR(u) = \sum_{v \in B_u} \frac{PR(v)}{L_v}$$

  - where B_u is the set of pages that point to u, and L_v is the number of outgoing links from page v (not counting duplicate links)
  - But: What is $PR(v)$?

PageRank – Random Link

$$PR(u) = \sum_{v \in B_u} \frac{PR(v)}{L_v}$$



- Don't know PageRank values at start
- Idea: Assume equal values (1/3 in this case), then iterate:
  - First iteration:
    PR(C) = 0.33/2 + 0.33 = 0.5, *PR*(A) = 0.33, *PR*(B) = 0.17
  - Second iteration:
    PR(C) = 0.33/2 + 0.17 = 0.33, *PR*(A) = 0.5, *PR*(B) = 0.17
  - Third iteration:
    PR(C) = 0.42, *PR*(A) = 0.33, *PR*(B) = 0.25
- Converges to *PR*(C) = 0.4, *PR*(A) = 0.4, and *PR*(B) = 0.2

PageRank – Random Page

- Taking random page jump into account, 1/3 chance of going to any page when $r < \lambda$
- $PR(C) = \lambda \cdot 1/3 + (1 - \lambda) \cdot (PR(A)/2 + PR(B)/1)$
- More generally,

$$PR(u) = \frac{\lambda}{N} + (1 - \lambda). \sum_{v \in B_u} \frac{PR(v)}{L_v}$$

  - where N is the number of pages, λ typically 0.15
- Equivalent to $R = T \cdot R$
  - Where R is vector of PageRank values and T is transition probability matrix:

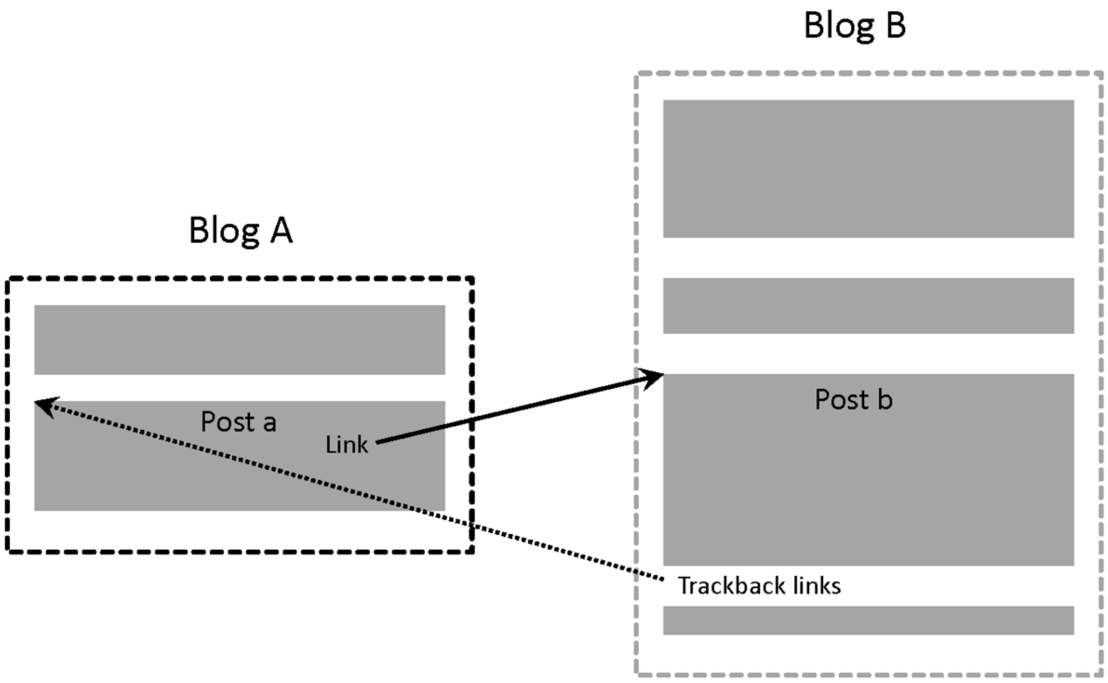
$$T_{ij} = \frac{\lambda}{N} + (1 - \lambda)\frac{1}{L_i}$$

- R is Eigenvector of T

procedure PAGERANK(G)
    ▷ G is the web graph, consisting of vertices (pages) and edges (links).
    $(P, L) \leftarrow G$ ▷ Split graph into pages and links
    $I \leftarrow$ a vector of length $|P|$ ▷ The current PageRank estimate
    $R \leftarrow$ a vector of length $|P|$ ▷ The resulting better PageRank estimate
    for all entries $I_i \in I$ **do**
        $I_i \leftarrow 1/|P|$ ▷ Start with each page being equally likely
    end for
    while R has not converged **do**
        for all entries $R_i \in R$ **do**
            $R_i \leftarrow \lambda/|P|$ ▷ Each page has a $\lambda/|P|$ chance of random selection
        end for
        for all pages $p \in P$ **do**
            $Q \leftarrow$ the set of pages p such that $(p, q) \in L$ and $q \in P$
            if $|Q| > 0$ **then**
                for all pages $q \in Q$ **do**
                    $R_q \leftarrow R_q + (1 - \lambda) I_p / |Q|$ ▷ Probability I_p of being at page p
                end for
            else
                for all pages $q \in P$ **do**
                    $R_p \leftarrow R_q + (1 - \lambda) I_p / |P|$
                end for
            end if
            $I \leftarrow R$ ▷ Update our current PageRank estimate
        end for
    end while
    return R
end procedure

Link Quality

- Link quality is affected by spam and other factors
  - e.g., *link farms* to increase PageRank
  - *Trackback links* in blogs can create loops
  - Trackback links are links of a different nature



Link Quality

- Link quality is affected by spam and other factors
  - Links from comments section of popular blogs boost own web page
    - Blog services modify comment links to contain `rel=nofollow` attribute
      - To help search engines
      - Initiatied by Google in 2005
    - e.g., “Come visit my <a `rel=nofollow` href="http://www.page.com">web page.”

rel="nofollow" Action	Google	Yahoo!	Bing	Ask.com
Uses the link for ranking	No	No	No	?
Follows the link	Yes	Yes	Yes	No
Indexes the "linked to" page	No	Yes	No	No
Shows the existence of the link	Only for a previously indexed page	Yes	Yes	Yes
In results pages for anchor text	Only for a previously indexed page	Yes	Only for a previously indexed page	Yes

http://en.wikipedia.org/wiki/Nofollow

Overview

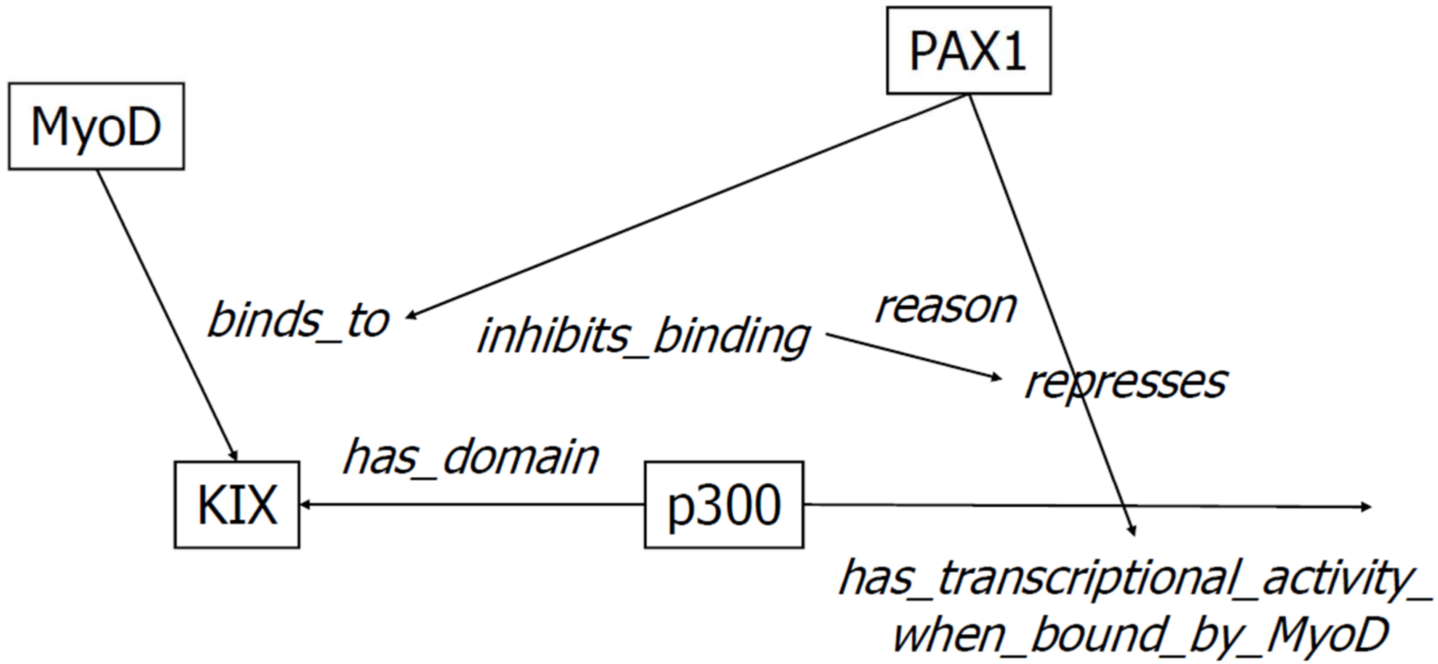
- Text statistics
- Document parsing
- Link Analysis
- Information Extraction

Information Extraction

- Automatically extract structure from text
  - Annotate document using tags to identify extracted structure
  - Near-term goal: Improve ranking
  - Far-term goal: Turn search problem into database problem
- Already some information extraction
  - HTML structure
  - XML annotations
- *Named entity recognition (NER)*
  - Identify word or sequence of words that refer to something of interest in a particular application.
  - e.g., people, companies, locations, dates, product names, prices, drug names, etc.
  - Also: Semantic annotation (domain-specific)

Understanding Text is Difficult (even for us)

„The PAX1 protein represses MyoD-dependent transcription by inhibiting MyoD-binding to the KIX domain of p300."



What we Need to do

Z-100 is an arabinomannan extracted from Mycobacterium tuberculosis that has various immunomodulatory activities, such as the induction of interleukin 12, interferon gamma (IFN-gamma) and beta-chemokines. The effects of Z-100 on human immunodeficiency virus type 1 (HIV-1) replication in human monocyte-derived macrophages (MDMs) are investigated in this paper. In MDMs, Z-100 markedly suppressed the replication of not only macrophage-tropic (M-tropic) HIV-1 strain (HIV-1JR-CSF), but also HIV-1 pseudotypes that possessed amphotropic Moloney murine leukemia virus or vesicular stomatitis virus G envelopes. Z-100 was found to inhibit HIV-1 expression, even when added 24 h after infection. In addition, it substantially inhibited the expression of the pNL43lucDeltaenv vector (in which the env gene is defective and the nef gene is replaced with the firefly luciferase gene) when this vector was transfected directly into MDMs. These findings suggest that Z-100 inhibits virus replication, mainly at HIV-1 transcription. However, Z-100 also downregulated expression of the cell surface receptors CD4 and CCR5 in MDMs, suggesting some inhibitory effect on HIV-1 entry. Further experiments revealed that Z-100 induced IFN-beta production in these cells, resulting in induction of the 16-kDa CCAAT/enhancer binding protein (C/EBP) beta transcription factor that represses HIV-1 long terminal repeat transcription. These effects were alleviated by SB 203580, a specific inhibitor of p38 mitogen-activated protein kinases (MAPK), indicating that the p38 MAPK signalling pathway was involved in Z-100-induced repression of HIV-1 replication in MDMs. These findings suggest that Z-100 might be a useful immunomodulator for control of HIV-1 infection.

Find Entities

Z-100 is an *arabinomannan* extracted from Mycobacterium tuberculosis that has various immunomodulatory activities, such as the induction of **interleukin 12**, **interferon gamma** (**IFN-gamma**) and beta-chemokines. The effects of *Z-100* on human immunodeficiency virus type 1 (HIV-1) replication in human **monocyte-derived macrophages** (**MDMs**) are investigated in this paper. In **MDMs**, *Z-100* markedly suppressed the replication of not only macrophage-tropic (M-tropic) HIV-1 strain (HIV-1JR-CSF), but also HIV-1 pseudotypes that possessed amphotropic Moloney murine leukemia virus or vesicular stomatitis virus G envelopes. *Z-100* was found to inhibit HIV-1 expression, even when added 24 h after infection. In addition, it substantially inhibited the expression of the pNL43lucDeltaenv vector (in which the *env* gene is defective and the *nef* gene is replaced with the *firefly luciferase* gene) when this vector was transfected directly into **MDMs**. These findings suggest that *Z-100* inhibits virus replication, mainly at HIV-1 transcription. However, *Z-100* also downregulated expression of the cell surface receptors **CD4** and **CCR5** in **MDMs**, suggesting some inhibitory effect on HIV-1 entry. Further experiments revealed that *Z-100* induced **IFN-beta** production in these cells, resulting in induction of the 16-kDa **CCAAT/enhancer binding protein** (**C/EBP**) **beta transcription factor** that represses HIV-1 long terminal repeat transcription. These effects were alleviated by SB 203580, a specific inhibitor of **p38 mitogen-activated protein kinases** (**MAPK**), indicating that the **p38 MAPK** signalling pathway was involved in *Z-100*-induced repression of HIV-1 replication in **MDMs**. These findings suggest that *Z-100* might be a useful immunomodulator for control of HIV-1 infection.

Find Relationships

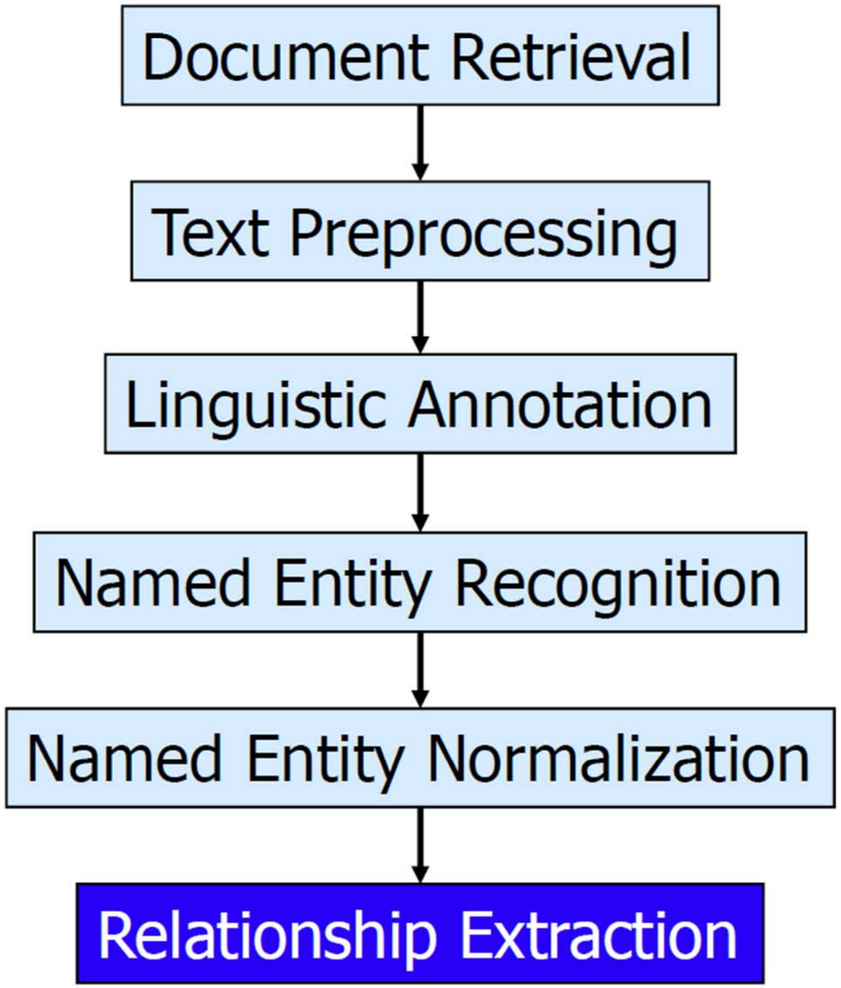
Z-100 is an *arabinomannan* extracted from Mycobacterium tuberculosis that has various immunomodulatory activities, such as the induction of **interleukin 12**, **interferon gamma** (**IFN-gamma**) and beta-chemokines. The effects of *Z-100* on human immunodeficiency virus type 1 (HIV-1) replication in human **monocyte-derived macrophages** (**MDMs**) are investigated in this paper. In **MDMs**, *Z-100* markedly suppressed the replication of not only macrophage-tropic (M-tropic) HIV-1 strain (HIV-1JR-CSF), but also HIV-1 pseudotypes that possessed amphotropic Moloney murine leukemia virus or vesicular stomatitis virus G envelopes. *Z-100* was able to inhibit HIV-1 expression, even when added 24 h after infection. In addition, it substantially inhibited the expression of the pNL43lucDeltaenv vector (in which the ***env*** gene is defective and the ***nef*** gene is replaced with the ***firefly luciferase*** gene) when this vector was transfected directly into **MDMs**. These findings suggest that *Z-100* inhibits virus replication, mainly at HIV-1 transcription. However, *Z-100* also downregulated expression of the cell surface receptors **CD4** and **CCR5** in **MDMs**, suggesting some inhibitory effect on HIV-1 entry. Further experiments revealed that *Z-100* induced **IFN-beta** production in these cells, resulting in induction of the 16-kDa **CCAAT/enhancer binding protein** (**C/EBP**) **beta transcription factor** that represses HIV-1 long terminal repeat transcription. These effects were alleviated by SB 203580, a specific inhibitor of **p38 mitogen-activated protein kinases** (**MAPK**), indicating that the **p38 MAPK** signalling pathway was involved in *Z-100*-induced repression of HIV-1 replication in **MDMs**. These findings suggest that *Z-100* might be a useful immunomodulator for control of HIV-1 infection.

induction

inhibit

induces

Information Extraction Workflow

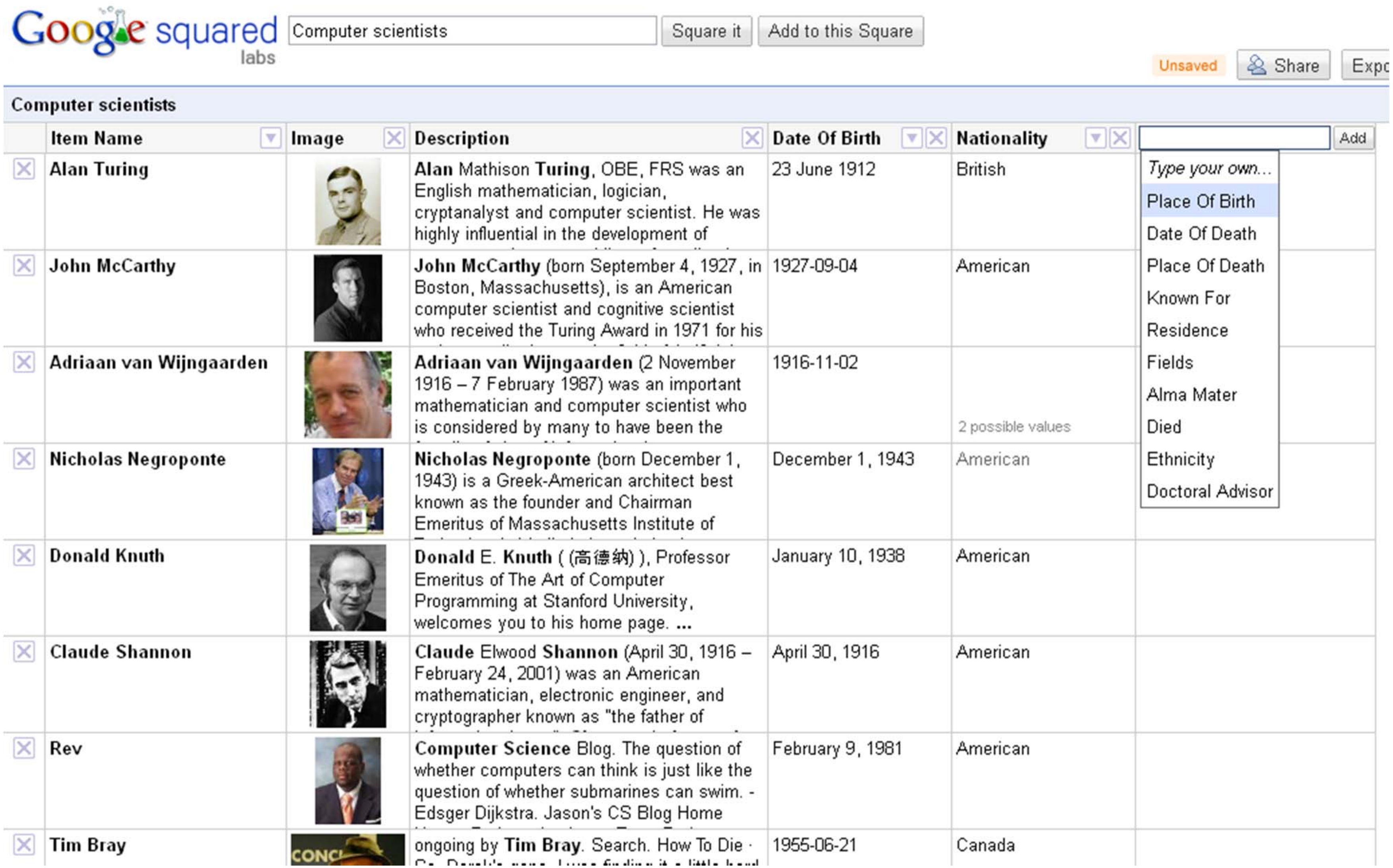


Detecting Gene Names (Leser & Hakenberg, 2005)

The human T cell leukemia lymphotropic virus type 1 Tax protein represses MyoD-dependent transcription by inhibiting MyoD-binding to the KIX domain of p300.

- Also: hedgehog, soul, the, white, …
- State-of-the-art methods reach ~85% in NEN
  - Plus 10% for less stringent boundary definitions
  - Large dicts, CRF, species classification, large background corpus, …
  - That's about as high as inter-annotator agreement
- Different performance for other classes (mutations, diseases, functional terms, cell lines, …)

Google Squared



Named Entity Recognition

- “Fred Smith, who lives at 10 Water Street, Springfield, MA, is a long-time collector of **tropical fish**.”
    - **<p><PersonName><GivenName>Fred</GivenName> <Sn>Smith</Sn></PersonName>, who lives at <address><Street>10 Water Street</Street>, <City>Springfield</City>, <State>MA</State></address>, is a long-time collector of tropical fish.</p>**
- Example shows semantic annotation of text using XML tags
- Information extraction also includes document structure and more complex features such as *relationships* and *events*
- Uses
    - Faceted search
    - Improved browsing (clickable locations, phone-numbers, etc.)

Named Entity Recognition

- *Rule-based*
  - Uses *lexicons* (lists of words and phrases) that categorize names
    - e.g., locations, person names, organizations, etc.
  - Rules (patterns) also used to verify or find new entity names, e.g.,
    - "<number> <word> street" for addresses
    - "<street address>, <city>" or "in <city>" to verify city names
    - "<street address>, <city>, <state>" to find new cities
    - "<title> <name>" to find new names
- Rules either developed manually by trial and error or using machine learning techniques

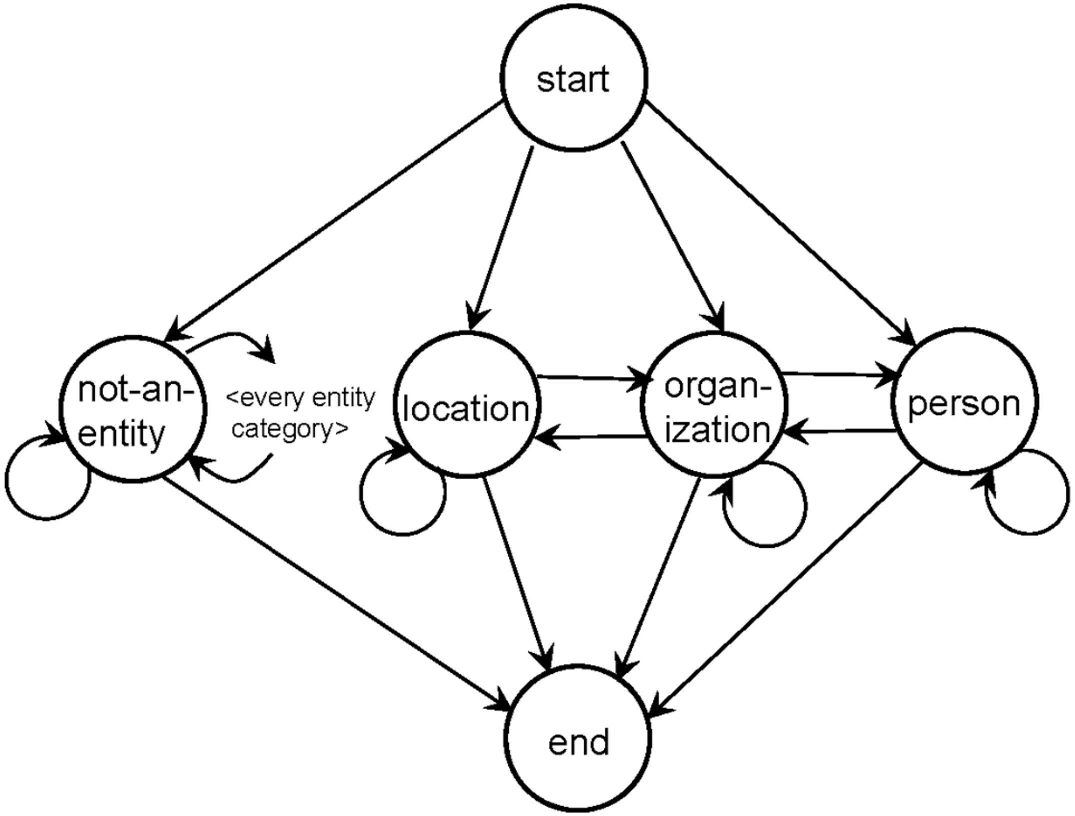
Named Entity Recognition

- *Statistical*
  - Uses a probabilistic model of the words in and around an entity
  - Probabilities estimated using *training data* (manually annotated text)
  - Hidden Markov Model (HMM) is one approach
- HMM for Extraction
  - Resolve ambiguity (homonyms) in a word using *context*
    - Like humans
    - e.g., "marathon" is a location or a sporting event, "boston marathon" is a specific sporting event
  - Model the context using a *generative* model of the sequence of words
    - *Markov property*: the next word in a sequence depends only on a small number of the previous words

HMM for Extraction

- *Markov Model* describes a process as a collection of states with transitions between them.
  - Each transition has a probability associated with it.
  - Next state depends only on current state and transition probabilities
- *Hidden Markov Model*
  - Each state has a set of possible outputs.
  - Outputs have probabilities.
  - “Hidden”, because sequence of states not visible
    - Output is visible, however

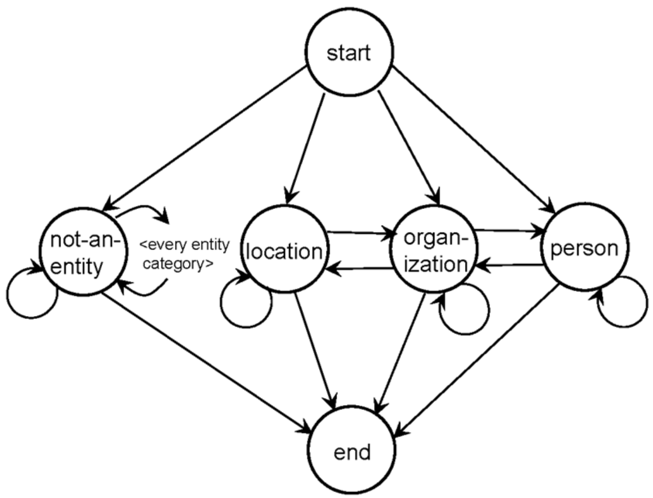
HMM Sentence Model



- Each state is associated with a probability distribution over words (the output)

HMM for Extraction

- Could generate sentences with this model
- To recognize named entities, find sequence of "labels" that give highest probability for the sentence
  - Only the outputs (words) are visible or observed, states are "hidden".
  - "Fred Smith, who lives at 10 Water Street, Springfield, MA, is a long-time collector of **tropical fish.**"
  - <start><name><not-an-entity><location><not-an-entity><end>
- *Viterbi* algorithm used for recognition
  - Dynamic programming

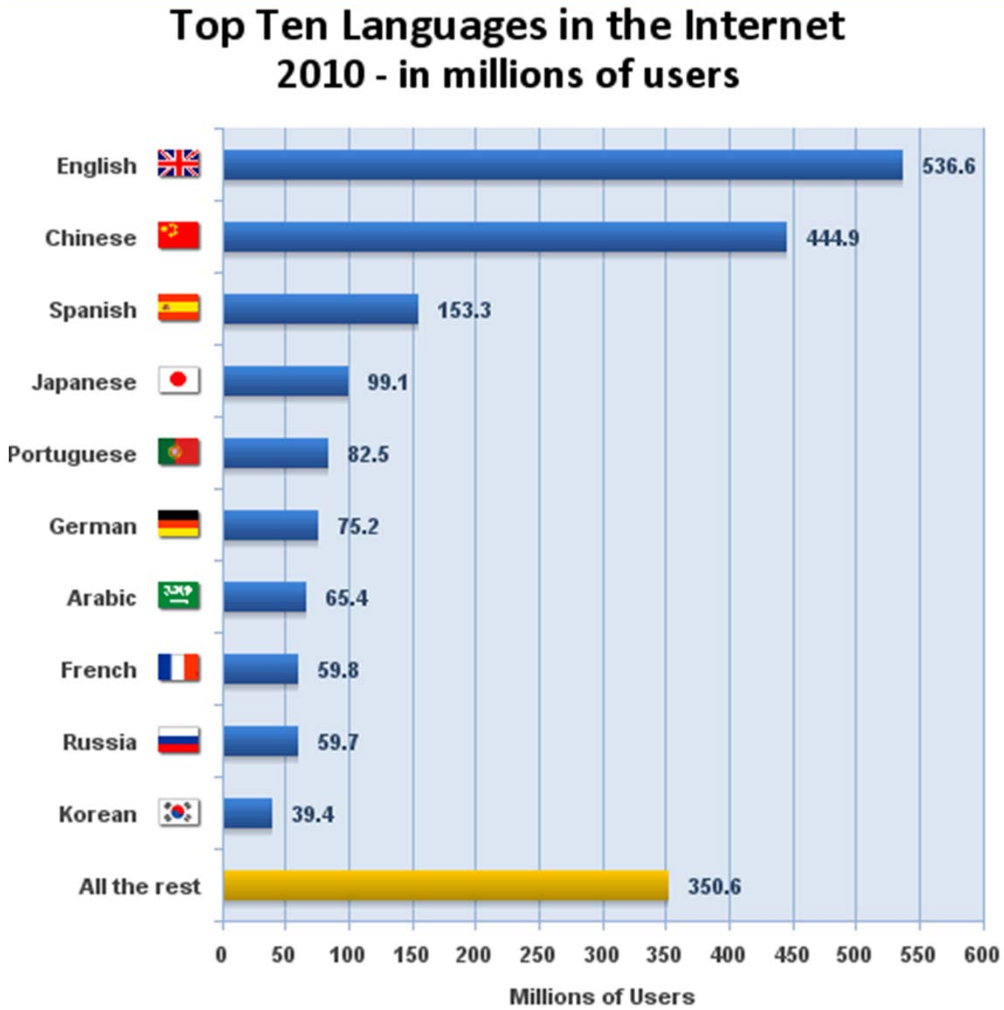


Named Entity Recognition

- Accurate recognition requires about 1 million words of training data (1,500 news stories)
  - May be more expensive than developing rules for some applications
- Both rule-based and statistical approaches can achieve about 90% effectiveness for categories such as names, locations, organizations.
  - Others, such as product name or genes, can be much worse

Internationalization

Hasso Plattner Institut



Source: Internet World Stats - www.internetworldstats.com/stats7.htm
Estimated Internet users are 1,966,514,816 on June 30, 2010

Internationalization

- 2/3 of the Web is in English
  - But decreasing
- At least 50% of Web users do not use English as their primary language
- Many (maybe most) search applications have to deal with multiple languages
  - *monolingual search*: search in one language, but with many possible languages
  - *cross-language search*: search in multiple languages at the same time

Position	Language	Internet users	% of all
1	English	536,564,837	27.5
2	Chinese (Mandarin)	444,948,013	22.6
3	Spanish	153,309,074	7.8
4	Japanese	99,143,700	5.3
5	Portuguese	82,548,200	4.3
6	German	75,158,584	4.0
7	Arabic	65,365,400	3.3
8	French	59,779,525	3.2
9	Russian	59,700,000	2.5
10	Korean	39,440,000	2.1

http://en.wikipedia.org/wiki/Global_Internet_usage

Internationalization

- Many aspects of search engines are language-neutral
- Major differences are in text processing:
  - Text encoding (converting to Unicode)
  - Tokenizing (many languages have no word separators)
  - Stemming
- Cultural differences may also impact interface design and features provided

Chinese "Tokenizing"

- Auch im Deutschen
  - Donaudampfschifffahrts-gesellschaft

1. Original text

旱灾在中国造成的影响

(the impact of droughts in China)

2. Word segmentation

旱灾 在 中国 造成 的 影响

drought at china make impact

3. Bigrams

旱灾 灾在 在中 中国 国造

造成 成的 的影 影响

Donaudampfschiffahrtselektrizitätenhauptbetriebswerkbauunterbeamtengesellschaft

Donaudampfschiffahrtselektrizitätenhauptbetriebswerkbauunterbeamtengesellschaft ist ein Wort, das in verschiedenen Ausgaben des Guinness-Buchs der Rekorde[1] mit einer Länge von 79 Buchstaben als das längste veröffentlichte Wort in der deutschen Sprache angegeben wird. Gemäß der Rechtschreibreform von 1996 ist das Wort mit drei aufeinanderfolgenden 'f' („-schifffahrt-") und somit 80 Buchstaben zu schreiben.

Während Donaudampfschiffahrtsgesellschaft (auch „Erste Donau-Dampfschiffahrts-Gesellschaft", abgekürzt DDSG oder EDDG) als Name einer 1829 gegründeten Gesellschaft, die von 1830 bis 1995 Personen- und Güterschiffahrt auf der Donau betrieb, historisch belegt ist, sind für die besagte „Unterbeamtengesellschaft" keine Belege dafür bekannt, dass jemals eine Gesellschaft dieses Namens existierte und es sich bei diesem Namen nicht bloß um ein Kunstwort handelt, das zur Erzielung einer besonderen Wortlänge erzeugt wurde.

Amtlich belegt ist hingegen das *Rindfleischetikettierungsüberwachungsaufgabenübertragungsgesetz* in Mecklenburg-Vorpommern.