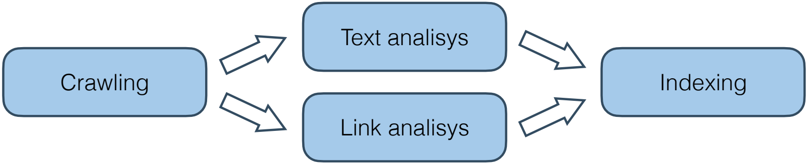
Information Retrieval 1

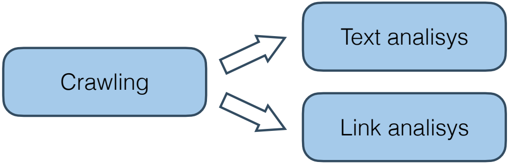
Text Analysis

Ilya Markov
i.markov@uva.nl

University of Amsterdam

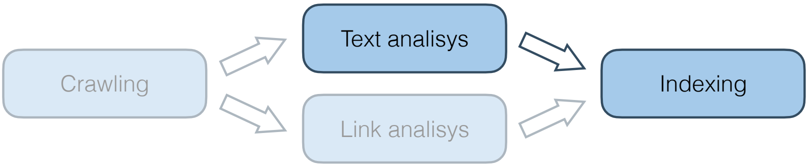
Recap IR0


Crawling
Text analisys
Link analisys
Indexing

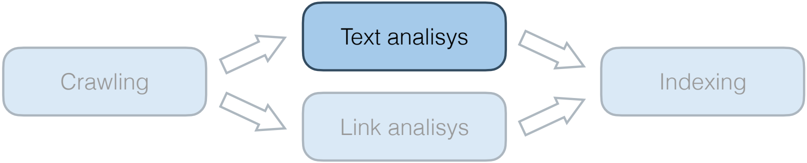

Crawling
Text analisys
Link analisys


Text analisys

Recap IR0



Text analysis



Outline

Outline

Outline

1. Statistical properties of written text
   - Zipf's law
   - Heaps' law

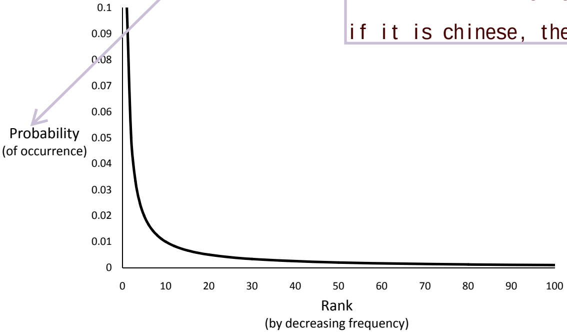
Zipf's law

对于所有language，都符合zipf's law
即：如果我们对这个语言中的所有单词根据它们的出现频率从高到低来排序，则
rank*frequency=constant
prob = func(freq)
rank*probability =another constant

if the chosen language is english, then constant=0.1

if it is chinese, then constant= maybe 0.5

我们定义：大概率出现的单词=高频单词=rank靠前的=rank接近于1



$$rank \cdot freq = const$$

$$rank \cdot P_r = const'$$

For English, $const' \approx 0.1$

Croft et al., "Search Engines, Information Retrieval in Practice"

High-frequency words

r=rank=根据frequency 把word从高频到低频进行排序

最初的这些单词r*P deviate from 0.1

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

稳定在≈0.1

Croft et al., "Search Engines, Information Retrieval in Practice"

Low-frequency words

大部分单词，constant约等于0.1

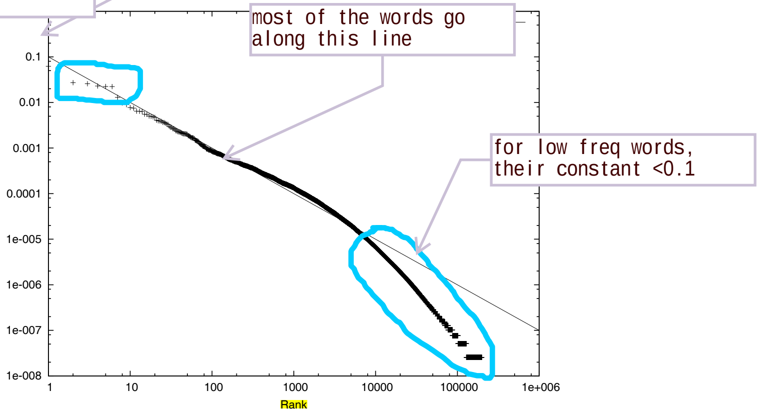
Word	*Freq.*	r	$P_r(\%)$	$r.P_r$
assistant	5,095	1,021	.013	0.13
sewers	100	17,110	.000256	0.04
toothbrush	10	51,555	.000025	0.01
hazmat	1	166,945	.000002	0.04

for low freq words, their r*P is likely to be different from 0.1

Croft et al., "Search Engines, Information Retrieval in Practice"

Zipf's law vs. real data

y axis: probability



Croft et al., "Search Engines, Information Retrieval in Practice"

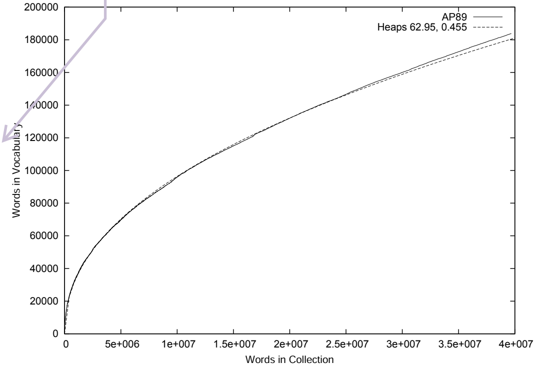
Outline

1. Statistical properties of written text
   - Zipf's law
   - Heaps' law

Heaps' law

we choose any language, eg. english.
# of unique words is Not linear to # of words in a collection
当写的是一本书，全书的字数统计与所用的 total # of unique words 构成正相关，但是不是线性正相关。因为字数越多，重复使用的词也越多

# of unique words
1个words出现10次，被记作1



左图符合如下函数
即非线性的函数

$$vocab = const \cdot words^{\beta}$$

$$10 \leq k \leq 100, \beta \approx 0.5$$

# of total words
1个words出现10次，被记作10

Croft et al., … l in Practice"

Outline

Text analysis pipeline

1. Remove white-spaces and punctuation
2. Convert terms to lower-case
3. Remove stop-words
4. Convert terms to their stems
5. Deal with phrases
6. Apply language-specific processing rules

Example

1. To prepare a text for indexing, one needs to split it into tokens, remove stop-words and perform stemming.
2. to prepare a text for indexing one needs to split it into tokens remove stop words and perform stemming
3. prepare indexing needs split tokens remove stop perform stemming
4. prepar index need split token remov stop perform stem

Stop-word removal

- Frequency-based
  - Set a frequency threshold f
  - Remove words with the frequency higher than f
- Dictionary-based
  - Create a dictionary of stop-words
  - Remove words that occur in this dictionary

Outline

Stemming

1. Algorithmic
2. Dictionary-based
3. Hybrid

Algorithmic stemming (Porter stemmer)

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).
- Whew!

Croft et al., "Search Engines, Information Retrieval in Practice"

http://snowball.tartarus.org/algorithms/porter/stemmer.html

Algorithmic stemming (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

Croft et al., "Search Engines, Information Retrieval in Practice"

Dictionary-based stemming

- Large dictionary of related words
- Semi-automatic: run $\rightarrow$ running, runs, runned, runly
- New-words problem

Hybrid stemming (Krovetz stemmer)

- Approach
  1. Check the word in a dictionary
  2. If found, leave it as is
  3. If not found, apply algorithmic stemming (remove suffixes)
  4. Check the dictionary again
  5. If not found, apply rules to modify the ending
- Produces words not stems
- Comparable effectiveness with the Porter stemmer

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:
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

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

Croft et al., "Search Engines, Information Retrieval in Practice"

Outline

1. Statistical properties of written text
2. Text analysis pipeline
3. Stemming
4. Phrases

Example

To be or not to be...

Dealing with phrases

1. Detect **noun phrases** using a part-of-speech tagger
   - sequences of nouns
   - adjectives followed by nouns
2. Detect phrases at the query processing time
   - Use an index with word positions
3. Use frequent **n-grams**, e.g., bigrams and trigrams

Summary

- Statistical properties of written text
  - Zipf's law
  - Heaps' law
- Text processing pipeline
  - Lexical analysis
  - Stop-word removal
  - Stemming
  - Phrases

Materials

- Croft et al., Chapters 4.1–4.3, 6.2.1–6.2.2
- Manning et al., Chapters 2.1–2.2, 3.3–3.4, 5.1
- Serrano et al.
  Modeling Statistical Properties of Written Text
  PLoS ONE, 2009