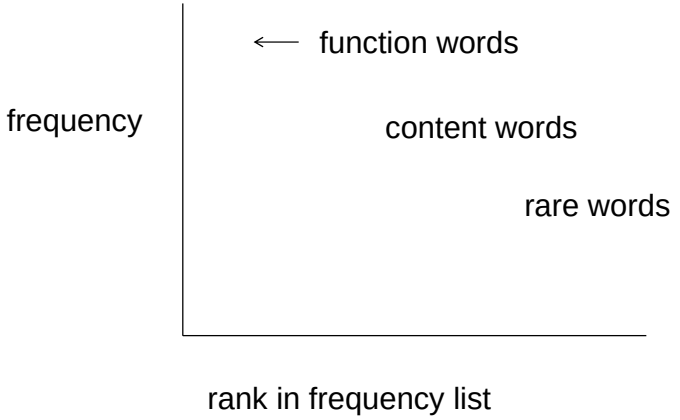
CS674 Natural Language Processing

- Last class
  - Introduction to generative models of language
    - » What are they?
    - » Why they're important
    - » Issues for counting words
    - » Statistics of natural language
- Today
  - Introduction to generative models of language
    - » Statistics of natural language
    - » Unsmoothed N-grams

How are they distributed?



Statistical Properties of Text

- The most frequent words in one corpus may be rare words in another corpus
  - Example: "computer" in CACM vs. National Geographic
- Each corpus has a different, fairly small "working vocabulary"

These properties hold in a wide range of languages

Zipf's Law

- Zipf's Law relates a term's frequency to its rank
  - frequency ∝ 1/rank
  - There is a constant k such that $freq * rank = k$
  - Rank the terms in a vocabulary by frequency, in descending order
    - f_r : frequency of term at rank r
    - N : total number of word tokens
    - $p_r = f_r / N$ and $\sum_{r=1}^{V} p_r = A$
  - Empirical observation: $p_r \approx A/r$, $A \approx 0.1$
  - Hence:

$$p_r = \frac{f_r}{N} = \frac{A}{r} \rightarrow rf_r = AN$$

  - $k \approx N/10$ for English

Zipf's Law

Word	Frequency	$r \times p_r$	Word	Frequency	$r \times p_r$
the	1,130,021	0.059	by	118.863	0.081
of	547,311	0.058	as	109,135	0.080
to	516,635	0.082	at	101,779	0.080
a	464,736	0.098	mr	101,679	0.086
in	390,819	0.103	with	101,210	0.091
and	387,703	0.122	from	96,900	0.092
that	204,351	0.075	he	94,585	0.095
for	199,340	0.084	million	93,515	0.098
is	152,483	0.072	year	90,104	0.100
said	148,302	0.078	its	86,774	0.100
it	134,323	0.078	be	85,588	0.104
on	121,173	0.077	was	83,398	0.105

WSJ87 collection (46,449 articles, 19 million term occurrences, 132 MB)

Zipf's Law (*Tom Sawyer*)

Word	Freq. (f)	Rank (r)	$f \cdot r$	Word	Freq. (f)	Rank (r)	$f \cdot r$
the	3332	1	3332	turned	51	200	10200
and	2972	2	5944	you'll	30	300	9000
a	1775	3	5235	name	21	400	8400
he	877	10	8770	comes	16	500	8000
but	410	20	8400	group	13	600	7800
be	294	30	8820	lead	11	700	7700
there	222	40	8880	friends	10	800	8000
one	172	50	8600	begin	9	900	8100
about	158	60	9480	family	8	1000	8000
more	138	70	9660	brushed	4	2000	8000
never	124	80	9920	sins	2	3000	6000
Oh	116	90	10440	Could	2	4000	8000
two	104	100	10400	Applausive	1	8000	8000

Manning and Schutze SNLP

Zipf's Law

- Useful as a rough description of the frequency distribution of words in human languages
- Behavior occurs in a surprising variety of situations
  - English verb polysemy
  - References to scientific papers
  - Web page in-degrees, out-degrees
  - Royalties to pop-music composers
- Zipf postulated a general law regarding human behavior: "principle of least effort"
  - Speaker: small vocabulary is best
  - Hearer: large vocabulary of unambiguous words best
  - Maximally economical compromise solution (Mandelbrot 1954): reciprocal relationship between frequency and rank

Topics for today

- Statistics of natural language
- Unsmoothed N-grams

Models of word sequences

- Simplest model
  - Let any word follow any other word
    - » P (word1 follows word2) = 1/# words in English
- Probability distribution at least obeys actual relative word frequencies
    - » P (word1 follows word2) = # occurrences of word1 / # words in English
- Pay attention to the preceding words
  - "Let's go outside and take a [ ]"
    - » walk very reasonable
    - » break quite reasonable
    - » lion less reasonable
  - Compute conditional probability P (walk| let's go…)

Probability of a word sequence

- P (w_1, w_2,…, w_{n-1}, w_n)
- Problem?
- Solution: approximate the probability of a word given all the previous words…

N-gram approximations

- Bigram model
- Trigram model
- N-gram approximation
- Markov assumption: probability of some future event (next word) depends only on a limited history of preceding events (previous words)

Bigram grammar fragment

- Berkeley Restaurant Project

eat on	.16	eat Thai	.03
eat some	.06	eat breakfast	.03
eat lunch	.06	eat in	.02
eat dinner	.05	eat Chinese	.02
eat at	.04	eat Mexican	.02
eat a	.04	eat tomorrow	.01
eat Indian	.04	eat dessert	.007
eat today	.03	eat British	.001

- Can compute the probability of a complete string
  - P (I want to eat British food) = P(I|<s>) P(want|I) P(to|want) P(eat|to) P(British|eat) P(food|British)

Training N-gram models

- N-gram models can be trained by counting and normalizing
  - Bigrams
  - General case…
  - An example of Maximum Likelihood Estimation (MLE)
    - » Resulting parameter set is one in which the likelihood of the training set T given the model M (i.e. P(T|M)) is maximized.

Bigram counts

	I	want	to	eat	Chinese	food	lunch
I	8	1087	0	13	0	0	0
want	3	0	786	0	6	8	6
to	3	0	10	860	3	0	12
eat	0	0	2	0	19	2	52
Chinese	2	0	0	0	0	120	1
food	19	0	17	0	0	0	0
lunch	4	0	0	0	0	1	0

- Note the number of 0's…
- Will look soon at improvements to MLE for dealing with this sparse data problem.

Accuracy of N-gram models

- Accuracy increases as N increases
  - Train various N-gram models and then use each to generate random sentences.
  - Corpus: Complete works of Shakespeare
    - » **Unigram:** *Will rash been and by I the me loves gentle me not slavish page, the and hour; ill let*
    - » **Bigram:** *What means, sir. I confess she? Then all sorts, he is trim, captain.*
    - » **Trigram:** *Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, 'tis done.*
    - » **Quadrigram:** *They say all lovers swear more performance than they are wont to keep obliged faith unforfeited!*

Strong dependency on training data

- Trigram model from WSJ corpus
  - *They also point to ninety nine point six billion dollars from two hundred four oh six three percent of the rates of interest stores as Mexico and Brazil on market conditions*