Ngrams

Kenneth Ward Church
AT&T Bell Laboratories
Murray Hill, N.J. USA
kwc@research.att.com

- Text is available like never before
  - Dictionaries, corpora, etc.
  - Data Collection Efforts:
    ACL/DCI, BNC, ECI, EDR, ICAME, LDC
  - Information Super-Highway Roadkill:
    email, WWW, bboards, faxes
  - Billions and billions of words/pixels

- What can we do with it all?

- It is better to do something simple,
  than nothing at all.

Simple Stuff

- You can do the simple things yourself

- DIY is more satisfying than begging for ‘‘help’’ from a computer officer.

- Exercises requiring only a few lines of Unix(TM) code
  1. Count words, bigrams, trigrams
  2. Ngram Stats: mutual info, t
  3. Concordances

- Hamming used to say it is much better to do the right problem naively than the wrong problem expertly.

- Infrastructure:
  - We *finally* have data
  - Most people believe it must be good for something
  - Missing: text analysis tools

Exercise 1: Count words in a text

- Input: text file (genesis)
- Output: list of words with freqs

```
        ...
    49 face
     4 faces
     1 fail
     2 failed
     1 faileth
        ...
```

- Algorithm
    1. Tokenize (tr)
    2. Sort (sort)
    3. Count duplicates (uniq –c)
- Solution

```
tr –sc ’A–Za–z’ ’\012’ < genesis |
sort |
uniq –c
```

Exercise 2: Count bigrams

Algorithm

1. tokenize by word
2. print $word_i$ and $word_{i+1}$ on the same line
3. count

```
tr -sc 'A-Za-z' '\012' < genesis > genesis.words
tail +2 genesis.words > genesis.nextwords

paste genesis.words genesis.nextwords

 ...

And    God
God    said
said   Let
Let    there

 ...
```

```
paste genesis.words genesis.nextwords |
sort | uniq –c > genesis.bigrams

sort –nr < genesis.bigrams | sed 5q

372   of     the
287   in     the
192   And    he
185   And    the
178   said   unto
```

Exercise: count trigrams

Mutual Info

$$I(x;y) \;=\; \log_2 \frac{Pr(x,y)}{Pr(x)\,Pr(y)}$$

$$I(x;y) \;\approx\; \log_2 \frac{Nf(x,y)}{f(x)\,f(y)}$$

```
wc –l genesis.words |
cat – genesis.hist genesis.bigrams |
gawk ’
NR==1 {N=$1; next}
NF==2 {f[$2]=$1}
NF==3 {print(log(N*$1/(f[$2]*f[$3]))/log(2),
             f[$2], f[$3], $0)}’ |
gawk ’$4 > 5’ |
sort –nr
```

MI	f(x)	f(y)	f(x,y)	x	y
11.7737	6	11	6	savoury	meat
11.5513	7	11	6	ill	favoured
11.4962	8	10	6	burnt	offering
11.0405	16	8	7	living	creature
10.9112	20	7	7	little	ones

t-scores

$$t \;=\; \frac{Pr(x,y) - Pr(x)\,Pr(y)}{\sqrt{\sigma^2 \left(Pr(x,y) + Pr(x)\,Pr(y)\right)}}$$

$$t \;\approx\; \frac{f(x,y) - \frac{1}{N} f(x)\,f(y)}{\sqrt{f(x,y)}}$$

```
wc –l genesis.words |
cat – genesis.hist genesis.bigrams |
gawk ’
NR==1 {N=$1; next}
NF==2 {f[$2]=$1}
NF==3 {print(($1 – (f[$2]*f[$3])/N)/sqrt($1),
             f[$2], f[$3], $0)}’ |
sort –nr
```

t	f(x)	f(y)	f(x,y)	x	y
14.884	1359	2407	372	of	the
14.772	588	2407	287	in	the
12.794	477	590	178	said	unto
12.3422	1251	646	192	And	he
11.5989	2407	161	154	the	LORD
11.5973	2407	187	157	the	land

Concordances

```
gawk ’
{i=0;
 while(match(substr($0, i+1), pat)) {
   i+=RSTART;
   printf("%15s%s\n",
        substr($0, i–15, i<=15?i–1:15),
        substr($0, i, 15))}}’ pat="$1" |
sort +0.15
```

```
God called the light Day, and
 to divide the light from the
od divided the light from the
cause he had delight in Jacob’
and the lesser light to rule t
s; the greater light to rule t
heaven to give light upon the
```

- Short program (only 8 lines of code)
- Easy to modify
  - sort to the left (no additional lines)
  - don’t count fragments such as ‘‘delight’’

Recap

- Exercises requiring only a few lines of Unix(TM) code

    1. Count words, bigrams, trigrams
    2. Ngram Stats: mutual info, t
    3. Concordances

- Why do these things?

    1. Self-organizing (IBM)
       Statistics can do it all.

    2. Exploratory Data Analysis (AT&T)
       Sensible combination of statistics and intuition.

    3. Stone Soup (Wilks)
       Statistics don't do nothing.
       Statistics + Intuition ≤ Intuition

Using Statistics in Lexical Analysis

Kenneth Church
William Gale
Patrick Hanks
Donald Hindle

Discuss Statistical Methods
for Comparing and Contrasting
the Distribution of Words
in Large Text Corpora

- Mutual Information (highlight associations)

- *t*-test (highlight differences)

  ‘‘You shall know a word by the company it keeps’’
  (Firth, 1957)

Three Types of Evidence in Lexicography

- Intuition: unstable
- Citation Indexes: butterflies
- Concordances: too much and too little

Use statistics to refine the ore

Similar problems in

- Information Retrieval (IR)
- Grammar Development

Proper Role for Human Judgment

- Self-organizing?

- Combination of Statistics and
  Human Interpretation

  1. Choose an appropriate statistic
     (e.g., mutual info, t-score),

  2. preprocess the corpus to highlight properties of interest (with a part of speech tagger or a parser), and

  3. select an appropriate unit of text
     (e.g., bigram, SVO triple, discourse).

Step 1: Select Appropriate Statistic

Mutual Information: A Measure of Similarity

$$I(x;y) \equiv \log_2 \frac{P(x,y)}{P(x)\ P(y)}$$

I(x;y)	fxy	fx	fy	x	y
10.47	7	7809	28	strong	northerly
9.76	23	7809	151	strong	showings
9.30	7	7809	63	strong	believer
9.22	14	7809	133	strong	second-place
9.17	6	7809	59	strong	runup
9.04	10	7809	108	strong	currents
8.85	62	7809	762	strong	supporter
8.84	8	7809	99	strong	proponent
8.68	15	7809	208	strong	thunderstorm
8.45	7	7809	114	strong	odor
8.66	7	1984	388	powerful	legacy
8.58	7	1984	410	powerful	tool
8.35	8	1984	548	powerful	storms
8.32	31	1984	2169	powerful	minority
8.14	9	1984	714	powerful	neighbor
7.98	9	1984	794	powerful	Tamil
7.93	8	1984	734	powerful	symbol
7.74	32	1984	3336	powerful	figure
7.54	10	1984	1204	powerful	weapon
7.47	24	1984	3029	powerful	post

t-test: A Measure of Dissimilarity

> *strong*:
> **4(a)**(capable of) having a great effect on the senses; intense or powerful; *a strong light, colour; a strong feeling of nausea; Her breath is rather strong*, ie has an unpleasant smell. (Oxford Advanced Learner's Dictionary of Current English, Fourth Edition

- *strong* and *powerful* are nearly synonymous,

  but what's the difference?

  strong tea vs. *powerful tea* [Smadja] [Halliday]

- *Her breath is rather strong*

  is different from

  Her breath is rather powerful

- School children often often misuse dictionaries in just this way (George Miller) strong vs. powerful

t	strong w	powerful w	w
12.42	161	0	showing
11.94	175	2	support
10.08	550	68	,
9.97	106	0	defense
9.76	102	0	economy
9.50	97	0	demand
9.40	95	0	gains
9.18	91	0	growth
8.84	137	5	winds
8.02	83	1	opposition
7.78	67	0	sales
–7.44	1	56	than
–5.60	1	32	figure
–5.37	3	31	minority
–5.23	1	28	of
–4.91	0	24	post
–4.63	5	25	new
–4.35	27	36	military
–3.89	0	15	figures
–3.59	6	17	presidency
–3.57	27	29	political
–3.33	0	11	computers

Hanks' Hypothesis

- *strong* denotes an **intrinsic** quality,

  whereas *powerful* denotes an **extrinsic** one

- Any worthwhile politician can expect *strong supporters*, who are enthusiastic, convinced, vociferous, etc.

  But far more valuable are *powerful supporters*, who will bring others with them.

Combination of Stats and Human Interpretation

- Step 1: Choose Appropriate Statistic
  - Mutual Info (highlights associations)
  - t-score (highlights differences)

- Not self-organizing

- Exploratory Data Analysis (EDA)
  (1) Collect data, (2) Analyze it,
  (3) Form hypotheses, and (4) Test.

- Theoretical and Empirical Approaches complement each other.

- Can’t have one without the other
  - Need hypos to collect representative evidence
  - Need evidence to develop & test hypos

Scale Statistics

- Step 1: Choose Appropriate Statistic
  - Mutual Info (highlights associations)
  - t-score (highlights differences)
  - Scale Statistics

Mean and Variance of the Separation Between X and Y				
Relation	Word x	Word y	Separation	
			mean	variance
fixed	*bread*	*butter*	2.00	0.00
	drink	*drive*	2.00	0.00
compound	*computer*	*scientist*	1.12	0.10
	United	*States*	0.98	0.14
semantic	*man*	*woman*	1.46	8.07
	man	*women*	–0.12	13.08
lexical	*refraining*	*from*	1.11	0.20
	coming	*from*	0.83	2.89
	keeping	*from*	2.14	5.53

Step 2: Preprocessing the Corpus

Preprocessing with a Part of Speech Tagger

- tag each word with a part-of-speech
- parse each clause into an SVO triple
- (No preprocessing is, of course, another option.)

Infinitival use of to				Prepositional use of to			
t	w to/to	w to/in	w	t	w to/to	w to/in	w
16.01	266	2	had/hvd	–12.44	10	176	back/rb
15.58	268	6	have/hv	–9.92	0	99	according/in
13.60	245	16	is/bez	–9.50	9	109	went/vbd
13.58	190	1	able/jj	–8.90	7	94	go/vb
12.59	160	0	want/vb	–8.54	29	125	up/rp
12.08	188	11	was/bedz	–8.38	3	77	as/in
11.77	140	0	began/vbd	–8.08	1	68	respect/nn
11.37	135	1	trying/vbg	–7.64	1	61	addition/nn
10.25	122	4	order/nn	–7.63	14	85	down/rp
10.07	107	1	wanted/vbd	–7.57	1	60	close/rb
9.86	202	34	going/vbg	–7.17	0	52	up/in
9.77	97	0	like/vb	–7.17	0	52	related/vbn
9.67	103	2	enough/qlp	–7.10	0	51	due/jj
9.46	156	20	not/*	–6.96	0	49	attention/nn
9.40	90	0	likely/jj	–6.60	31	95	came/vbd
9.14	93	2	tried/vbd	–6.28	0	40	regard/nn
8.95	107	7	seem/vb	–6.28	0	40	approach/nn
8.80	83	1	expected/vbn	–6.20	0	39	relation/nn
8.51	74	0	try/vb	–6.03	0	37	next/in
8.09	67	0	ready/jj	–5.78	0	34	return/vb
8.08	85	5	as/cs	–5.77	1	36	lead/vb
8.05	74	2	difficult/jj	–5.69	0	33	prior/rb
8.03	66	0	how/wrb	–5.69	3	39	said/vbd

SVO Triples in *Three Drowned...*

SVO Triple	Text
Guard search today	NEW BEDFORD, Mass (AP) -- The Coast Guard searched today for a 5-year-old boy
boat capsize ?	missing from an overloaded boat that capsized in New Bedford Harbor,
boat drown sister	drowning his baby sister, their mother and another woman,
official say ?	officials said.
PASSIVE throw people	Fifteen people were thrown into the water Monday night
boat return_from Fourth	as the 22-foot boat was returning from a Fourth of July fireworks display,
Monday say Foley	said Coast Guard Petty Officer David Foley.
survivor say boat	Survivors said
? capsize ?	the boat capsized
it hit wake	when it apparently hit another boat's wake
Foley say ?	while making its way through heavy fog, Foley said.

What does a boat do?

I(x;y)	x	y	I(x;y)	x	y
11.01	boat/S	capsize/V	3.09	boat/S	fail/V
9.30	boat/S	sink/V	2.72	boat/S	stop/V
8.17	boat/S	cruise/V	2.59	boat/S	accord/V
7.40	boat/S	sail/V	2.54	boat/S	reach/V
7.27	boat/S	tow/V	2.14	boat/S	lose/V
7.18	boat/S	turn_in/V	2.09	boat/S	leave/V
6.83	boat/S	collide/V	2.04	boat/S	keep/V
6.61	boat/S	drown/V	2.04	boat/S	kill/V
6.34	boat/S	drag/V	1.69	boat/S	be_in/V
6.28	boat/S	escort/V	1.61	boat/S	put/V
6.04	boat/S	overturn/V	1.38	boat/S	take/V
5.90	boat/S	rescue/V	1.36	boat/S	hold/V
5.43	boat/S	approach/V	1.28	boat/S	use/V
4.64	boat/S	carry/V	1.26	boat/S	become/V
4.43	boat/S	hit/V	0.94	boat/S	have/V
4.18	boat/S	travel/V	0.67	boat/S	begin/V
3.86	boat/S	pass/V	0.57	boat/S	get/V
3.71	boat/S	attack/V	0.17	boat/S	do/V
3.48	boat/S	injure/V	–0.35	boat/S	be/V
3.38	boat/S	fire/V	–0.35	boat/S	make/V
3.30	boat/S	operate/V	–3.38	boat/S	say/V

- Mutual Info ranks the verbs as we intuitively expect: it shows that *boat* is an interesting subject for the verb *sail* but not for the verb *be*

- A lexicographer should now scan the verbs at the top of the list and check for verbs such as *drown* that seem intuitively implausible

- Since there are only a few cases like *drown* where the mutual info value is misleading, it shouldn't be too much trouble for the lexicographer to go back to the original text and see what happened (e.g., parsing error, a piece of loose prose, or an unusual use of a familiar word).

- Applications:

    - Improve parsers: a parser should know that *boat* is probably not the subject of *drowning*

    - Speed up concordance analysis

Food		Water	
I(x;y)	Verb	I(x;y)	Verb
9.62	hoard/V	9.05	conserve/V
8.83	go_without/V	8.98	boil/V
7.68	eat/V	8.64	ration/V
6.93	consume/V	8.45	pollute/V
6.42	run_of/V	8.40	contaminate/V
6.29	donate/V	8.37	pump/V
6.08	distribute/V	7.86	walk_on/V
5.14	buy/V	7.81	drink/V
4.80	provide/V	7.39	spray/V
4.65	deliver/V	7.39	poison/V

t	Verb	t	Verb
7.47	eat/V	–6.93	be_under/V
6.26	buy/V	–5.62	pump/V
4.61	include/V	–5.37	drink/V
4.47	provide/V	–5.20	enter/V
4.18	bring/V	–4.87	divert/V
3.98	receive/V	–4.80	pour/V
3.69	donate/V	–4.25	draw/V
3.55	prepare/V	–4.01	boil/V
3.31	offer/V	–3.89	fall_into/V
3.08	deliver/V	–3.75	contaminate/V

Step 3: Select Appropriate Unit of Text (Discourse Context)

More like food		More like water	
t	w	t	w
50.74	food	–51.11	water
16.90	consumer	–11.31	crew
16.30	products	–11.47	inches
15.90	prices	–11.58	environmental
14.67	goods	–11.73	river
14.28	Food	–12.12	pollution
13.91	stock	–12.16	Water
13.85	market	–12.23	near
13.41	inflation	–12.40	rain
12.99	clothing	–14.35	miles
12.94	price	–15.34	River
12.88	takeover	–16.56	feet
12.44	sales	–11.28	Lake
12.39	rose	–10.99	air
12.29	economic	–10.94	Coast
12.09	consumers	–10.93	Navy
12.06	earnings	–10.90	gallons
11.90	trading	–10.76	vessel
11.89	share	–10.69	boat
11.74	increase	–10.66	waters
11.72	economy	–10.53	accident

Applications

- Information Retrieval (IR)
- Lexicography
  (but might want to pick some other words)
- Cobuild Dictionary
  - *boat*: a small vessel for travelling on water, especially one which only carries a few people.
  - *ship*: a large boat which carries passengers or cargo on sea journeys.

t	Ship	t	Boat
30.2	ship	–29.5	boat
12.5	USS	–11.0	Vietnamese
10.7	Navy	–10.4	refugees
10.6	sailors	–9.8	boats
9.7	Pentagon	–9.7	fishing
9.4	carrier	–9.2	Kong
9.3	WASHINGTON	–9.0	Hong
9.3	turret	–8.7	persecution
9.1	battleship	–8.2	repatriation
8.9	tanker	–8.1	refugee
8.7	ships	–7.7	HONG
8.5	Iowa	–7.7	KONG
8.4	explosion	–7.7	Vietnam
8.3	gallons	–7.5	people
7.9	aground	–7.3	camps
7.8	aircraft	–7.0	colony
7.5	crude	–6.9	drowned
7.4	Adm.	–6.8	Refugees
7.4	spill	–6.7	homeland
7.3	guns	–6.7	fishermen
7.2	Fleet	–6.6	river
7.2	cargo	–6.5	fled
7.0	Iranian	–6.4	woman

Polysemy & Sense Disambiguation

River Sense of Bank		Money Sense of Bank	
t	bank & river	t	bank & money
6.63	river	–15.95	money
4.90	River	–10.70	Bank
4.01	water	–10.60	funds
3.57	feet	–10.46	billion
3.46	miles	–10.13	WASHINGTON
3.44	near	–10.13	Federal
3.27	boat	–9.43	cash
3.06	south	–9.03	interest
2.83	fisherman	–8.79	financial
2.83	along	–8.79	Corp
2.76	border	–8.38	loans
2.74	area	–8.17	loan
2.72	village	–7.57	amount
2.71	drinking	–7.44	fund
2.70	across	–7.38	William
2.66	east	–7.36	company
2.58	century	–7.31	account
2.53	missing	–7.25	deposits
2.52	Perez	–7.25	assets
2.52	barges	–7.12	raised

‘‘On the one hand, *bank* co-occurs with words and expressions such as *money*, *notes*, *loan*, *account*, *investment*, *clerk*, *official*, *manager*, *robbery*, *vaults*, *working in a*, *its actions*, *First National*, *of England*, and so forth. On the other hand, we find *bank* co-occurring with *river*, *swim*, *boat*, *east* (and of course *West* and *South*, which have acquired special meanings of their own), *on top of the*, and *of the Rhine*.’’ (Hanks 1987, p. 127)

Recap

1. Choose an appropriate statistic
    - Mutual Info (highlights associations)
    - t-score (highlights differences)

2. preprocess the corpus to highlight properties of interest (with a part of speech tagger or a parser), and

3. select an appropriate unit of text (e.g., bigram, SVO triple, discourse).

1. Self-organizing???
2. Exploratory Data Analysis!!!
3. Stone Soup???

Why Statistics?

- For many applications (e.g., learners' dictionaries and natural language processing), it is desirable to focus on the ''central and typical'' facts of the language that every speaker is expected to know, and to stay clear of the gray area where the facts seem to be less clear cut.

- Historical precedents
  - ''You shall know a word by the company it keeps'' (Firth, 1957).
  - Harris' ''distributional hypothesis''

- Interest in statistical approaches faded rather suddenly when Chomsky argued quite successfully that statistics should not play a role in his competence model.

- Chomsky (all & only) vs. Shannon (central & typical)

- Both positions are reasonable; different applications lead to different criteria of success

	Rationalism	Empiricism
Advocates	Chomsky, Minsky	Shannon, Skinner, Firth, Harris
Model	Competence Model	Noisy Channel Model
Contexts	Phrase-Structure	N-grams
Goals	All & Only Explanatory Theoretical Perfection	Central & Typical Descriptive Applied Minimal Error (H)
Linguistic Generalizations	Agreement & Wh-movement	Collocations & Word Associations
Parsing Strategies	Principle-Based CKY (Chart), ATNs, Unification	Preference-Based Forward-Backward, Inside-Outside
Applications:	Understanding Who did what to whom	Recognition Noisy Channel Applications

The Stone Soup Debate

- IBM-style MT is obnoxious.
  → agreed
- It has all been done before.
  → agreed
- Stone soup: they’ve been adding intuition to their stats
  → agreed, but sounds more like EDA than stone soup.
- It doesn’t work (Systran is better).
  → Systran is also better than Pangloss
- It isn’t about empiricism, evaluation, etc.
  → Martin Kay’s advice about debating
- Natural Ceiling
  → Chomsky used this arg against Shannon
  In the part of speech case,
  the ceiling was broken with stats.
- The Future: Hybrid Approaches
  → sounds like EDA

Self-organizing: Shannon’s Noisy Channel Model

$$W_i \rightarrow Noisy\ Channel \rightarrow W_o$$

$$\underset{W_i}{ARGMAX}\ \ Pr(W_i)\ \ Pr(W_o \mid W_i)$$

- $Pr(W_i)$ is called the *language* model
- $Pr(W_o \mid W_i)$ is called the *channel* model

- Developed at AT&T Bell Laboratories

- Motivated by Communication Theory

- Applications
    1. Recognition: Speech, OCR, Spelling Correction
    2. Transduction: Part of Speech, MT
    3. Compression
    4. Error Correction

Examples of Channel Confusions
in Different Applications

Application	Input	Output
Speech Recognition	writer	rider
OCR	all	a1l (*A*-*one*-*L*)
	of	o{
	form	farm
Spelling Correction	government	goverment
	occurred	occured
	commercial	commerical
	similar	similiar

sub[X, Y] = Sub of X (incorrect) for Y (correct)

X	**Y (correct)**					
	a	b	c	d	e	f
a	0	0	7	2	342	1
b	1	0	9	9	3	3
c	7	6	0	16	1	9
d	2	10	13	0	12	1
e	388	0	4	11	0	3
f	0	15	1	4	2	0

Context (Language Model)

$$\textit{federal} \quad \begin{bmatrix} \textit{farm} \\ \textit{form} \end{bmatrix} \quad \textit{credit}$$

$$\textit{some} \quad \begin{bmatrix} \textit{farm} \\ \textit{form} \end{bmatrix} \quad \textit{of}$$

- Syntactic constraints will not help in this case.

- If I tell you that the next word is a noun,
  I haven't told you very much.

- There have been quite a number of attempts to use syntactic methods in speech recognition,
  but without much success....

- Syntactic constraints are dominated by
  - word frequencies,
  - collocations (*strong tea* / *powerful drugs*), and
  - word association norms (*doctor* / *nurse*),

  as any psycholinguist knows.

Ngram Language Models

$$W = w_1, w_2 \cdots w_n$$

- Unigrams: $Pr(W) \approx \prod_k Pr(w_k)$
- Bigrams: $Pr(W) \approx \prod_k Pr(w_k | w_{k-1})$
- Trigrams: $Pr(W) \approx \prod_k Pr(w_k | w_{k-2} w_{k-1})$

Parameter Estimation

Maximum Likelihood Estimate (MLE)

$$Pr(w_k) \approx \frac{freq(w_k)}{N}$$

$$Pr(w_k | w_{k-1}) \approx \frac{freq(w_{k-1} w_k)}{freq(w_{k-1})}$$

$$Pr(w_k | w_{k-2} w_{k-1}) \approx \frac{freq(w_{k-2} w_{k-1} w_k)}{freq(w_{k-2} w_{k-1})}$$

Variable Length Ngrams

- Tree Growing Criteria

- Katz' Back-off (code from Lincoln Labs)
```
p(wd2|wd1)= if(bigram exists) p_2(wd1,wd2)
            else bo_wt_1(wd1)*p_1(wd2)
```

- Linear Interpolation

$$\hat{Pr}(x,y) = \lambda \frac{freq(x,y)}{N} + (1-\lambda) \frac{freq(x)}{N} \frac{freq(y)}{N}$$

1. HMM (Hidden Markov Model)
2. linear regression
3. relate λ to σ^2 (variance/confidence)

Smoothing

WARNING: Poor estimates of context
can be worse than none.

- Problem: too many parameters (ngrams) and not enough training data. (Zeros are a particularly nasty case.)
- Reduce the number of parameters by grouping words into classes (e.g., by part of speech, synonymy, etc.)
- Replace trigram estimates with a combination of unigram, bigram and trigram estimates (‘‘backing off’’)

- Adjust frequencies

$r^* = r$ MLE

$r^* = r + 1$ ADD1

$r^* = (r + 1) \dfrac{N_r + 1}{N_r}$ GT

$r^* = C_r / N_r$ HO

Split Text Randomly

r	HO	GT	t
0	.000027041	.000027026	–.7
1	.4476	.4457	–2.9
2	1.254	1.260	2.5
3	2.244	2.237	–1.5
4	3.228	3.236	1.0
5	4.21	4.23	1.8
6	5.23	5.19	–2.8

Split Text Sequentially

r	HO	GT	t
0	0.00001684	0.0001132	–2730
1	0.4076	0.5259	113.
2	1.0721	1.2378	47.0
3	1.9742	2.2685	37.8
4	2.8632	3.1868	26.4
5	3.7982	4.2180	25.8
6	4.7822	5.2221	15.4

Example of the Trigram Model

$W =$ We need to resolve all of the important issues $\cdots$

The Trigram Approximation in Action (Jelinek, 1985)

Word	Rank	More likely alternatives
We	9	The This One Two A Three Please In
need	7	are will the would also do
to	1	
resolve	85	have know do ...
all	9	the this these problems ...
of	2	the
the	1	
important	657	document question first ...
issues	14	thing point to ...

Entropy (H)

- How do we decide if one language model is better than another?
- Entropy (developed by Shannon in 1940s): a measure of the information content of a probabilistic source: $H(Pr(x))$
- H is expressed in *bit*s (*bi*nary digi*t*s)
- H has been used to quantify
    1. noise
    2. redundancy
    3. the capacity (bandwidth) of
       a communication channel (e.g., a telephone),
    4. the efficiency of a code

    and much more...
- For recognition purposes, H characterizes the size of the search space, the number of binary questions that the recognizer will have to answer on average in order to decode a message.

Cross Entropy

- *Cross entropy* is a useful yardstick for measuring the ability of a language model to predict a source of data.

- If the language model is very good at predicting the future output of the source, then the cross entropy will be small.

- Cross entropy $\geq H$
  (H uses the best possible language model,
  the source itself)

Cross Entropy of Various Language Models

Model	Bits / Character
Ascii	8
Huffman code each char	5
Lempel-Ziv (Unix™ *compress*)	3.0
gzip	2.5
Unigram (Huffman code each word)	2.1 (Brown, p.c.)
Trigram	1.76 (Brown *et al.*, 1992)
Human Performance	1.25 (Shannon, 1951)

Variable Length Codes: Huffman and Arithmetic Codes

- The standard ASCII code requires 8 bits per character.

- It would be a perfect code if the source produced each of the $2^8 = 256$ symbols equally often and independently of context.

- However, English is not like this.

- For an English source, it is possible to reduce the average length of the code by assigning shorter codes to more frequent symbols (e.g., e, n, s) and longer codes to less frequent symbols (e.g., j, q, z)

- $codelength(letter) \approx \lceil -\log_2 Pr(letter) \rceil$

Example of Variable Length Coding

Input Text: *ejeqqeje*

Fixed Width Coding (Uniform Pr)

letter	Pr	Code	Length (–log Pr)
e	0.33	00	2
j	0.33	01	2
q	0.33	10	2

$$H = -\sum_{letter} Pr(letter)\log_2 Pr(letter)$$

$$= \frac{1}{3}2 + \frac{1}{3}2 + \frac{1}{3}2 = 2 \ \ bits/letter$$

Variable Length Coding (Non-uniform Pr)

letter	Pr	Code	Length (–log Pr)
e	0.5	0	1
j	0.25	10	2
q	0.25	11	2

$$H = -\sum_{letter} Pr(letter)\log_2 Pr(letter)$$

$$= \frac{1}{2}1 + \frac{1}{4}2 + \frac{1}{4}2 = \frac{3}{2} \ \ bits/letter$$

The cross entropy, *H*, of a code and a source is given by:

$$H(source, code) = -\sum_{s}\sum_{h} Pr(s, h \mid source) \log_2 Pr(s \mid h, code)$$

where $Pr(s,\ h \mid source)$ is the joint probability of a symbol *s* following a history *h* given the source.

$Pr(s \mid h,\ code)$ is the conditional probability of *s* given the history (context) *h* and the code.

In the special case of ASCII, where $Pr(s \mid h, ASCII) = 1/256$, we can carry out the indicated sum, and find, not surprisingly, that ASCII requires 8 bits/char:

$$H(source, ASCII) = -\sum_{s=1}^{256} \frac{1}{256} \log_2 \frac{1}{256} = 8$$

In more difficult cases, cross entropy is estimated by a sampling procedure.

Collect two independent samples of the source: S_1 and S_2.

S_1 is used to fit the values of the parameters of the code, and S_2 is used to test the fit.

It is important in this procedure to use two different samples (don’t test on the training material).

Using Statistics to Fit Probabilistic Models to Data

- Statistical Observations vs. Probabilitistic Models
  - Statistical Observation: $freq(Kennedy) = 140$
  - Probabilistic Model: $Pr_B(m) = \binom{n}{m} p^m (1-p)^{n-m}$
- Example: estimate the probability distribution for ‘‘Kennedy’’ in the Brown Corpus.
- Distribution vs. Expected Value
- $Pr(0) = ?, Pr(1) = ?, Pr(2) = ?, \cdots$
- Assume a *binomial* model, and use the frequency of ‘‘Kennedy’’ in the Brown Corpus (140) to fit the model to the data.

The Binomial Model

- Classic example: coin tossing
- Suppose the prob of head is p, and the probability of tails is $1-p=q$.
- Then the prob of exactly m heads in n tosses is $Pr_B(m)=\binom{n}{m}\ p^m\ (1-p)^{n-m}$.
- $\binom{n}{m}=\frac{n!}{m!\ (n-m)!}$ is the *binomial coefficient*.
- $n!=1\times 2\times \cdots \times n$ is the *factorial* function.
- For example, tossing a fair coin three times (n = 3, p = 1/2) will result in 0, 1, 2, and 3 heads with probability 1/8, 3/8, 3/8, and 1/8, respectively.
- This set of probabilities forms a distribution.
- Its expected value is np and its variance is $\sigma^2 = np(1-p)$.
- Thus, tossing a fair coin three times will produce an average of 3/2 heads with a variance of 3/4.

Fitting the Binomial Model with Statistical Data

- How can the binomial be used to model the distribution of ‘‘Kennedy’’?
- Let p be the probability that the next word is ‘‘Kennedy.’’
- Words are analogous to coin tosses.
- Unfortunately, p is an unknowable theoretical quantity, but it can be estimated from data.

- Method of Moments:
  1. Let k be the number of free parameters in the model.
     (In this case, there is just one free parameter, p; recall that n = sample size = 1,000,000 words.)
  2. Estimate the first k moments both theoretically and empirically, and assume that the two estimates are the same.
  3. Solve the system of k equations for the k unknowns.

- For example, to estimate the distribution of ‘‘Kennedy’’ in the Brown Corpus:
    1. Theoretical estimate of first moment = *np*
    2. Empirical estimate = freq(Kennedy) = 140
    3. Equate two estimates and solve for *p*
       $p = 140/n = 140/10^6$

- Caveots:
    1. Special care such as Good-Turing smoothing should be taken for small frequencies (e.g., less than 10).

    2. It is often convenient to use statistical estimates as if they are the same as the true probs, but this practice can lead to trouble, especially when the data don’t fit the model very well (as we will see).

    3. Method of moments is easy to implement, but other methods such as the maximum likelihood estimate (MLE) are often more accurate.
       (For the binomial, MLE = Method of Moments)

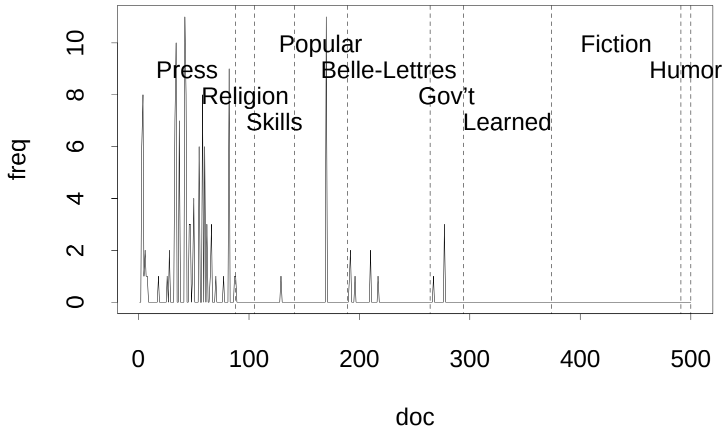
- Strengths of the trigram model:
  - $H(Pr(W)) \approx \log_2 90$
  - Takes advantage of word frequencies and other psycholinguistic factors.
  - Parsers don't do as well because they ignore these factors.

- Weaknesses of the trigram model
  - No syntax (long distance dependencies)
  - Sparse data: $V^3 >> N$
    (and, ironically, it gets worse as $N \rightarrow \infty$)

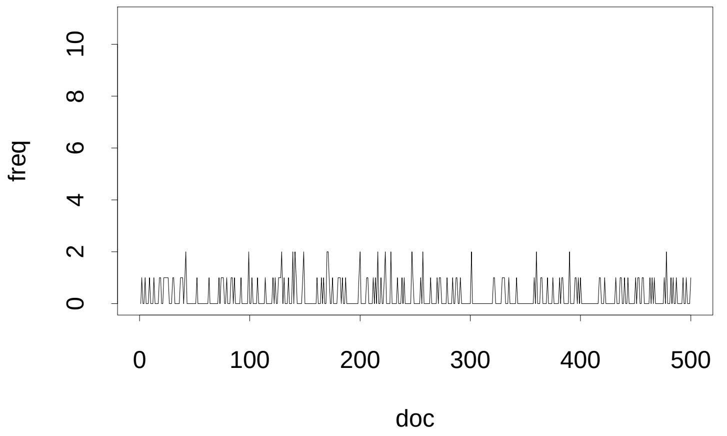
Corpus	N	V
Brown	1M	50k
1988 AP	44M	450k

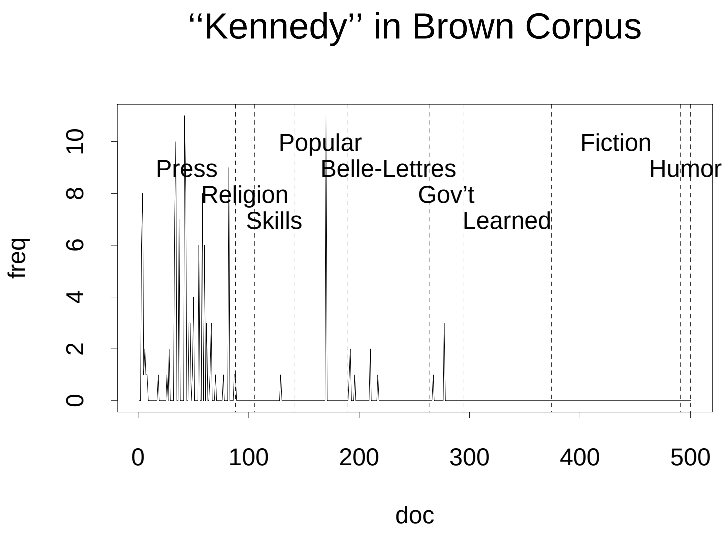
  - Words are ''contagious'' (not binomial)
    If there is one ''Kennedy'' in a doc,
    there'll probably be another.

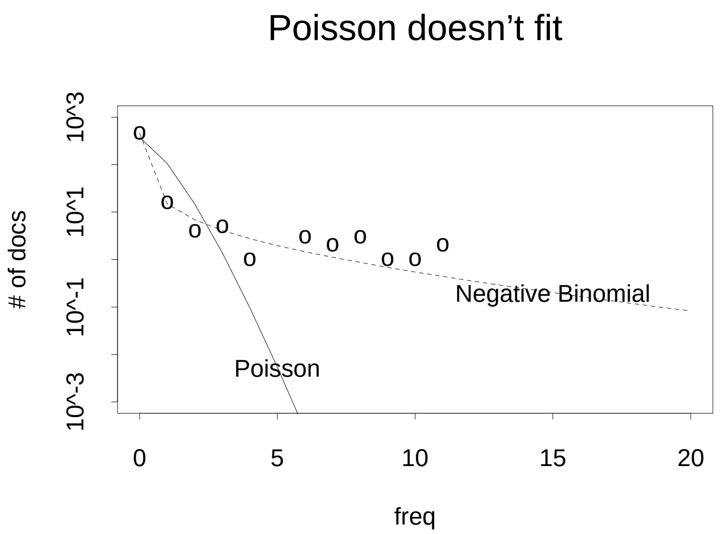
‘‘Kennedy’’ in Brown Corpus

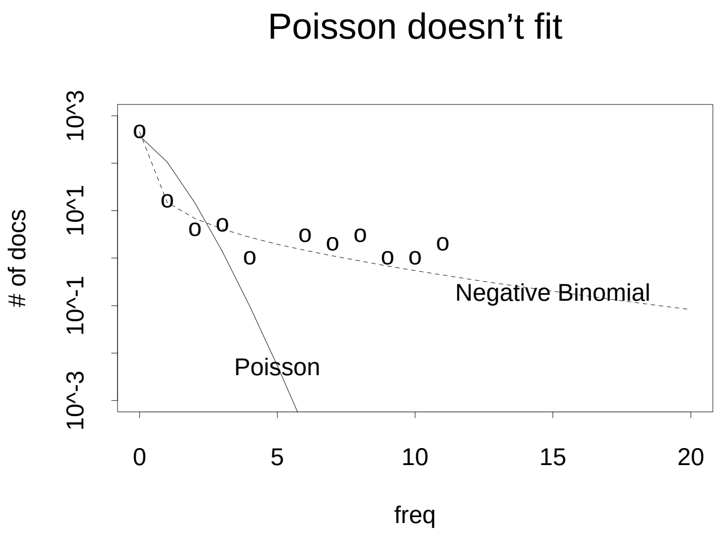


500 Randomly Generated Docs




‘‘Kennedy’’ in Brown Corpus
freq
doc
0
2
4
6
8
10
0
100
200
300
400
500
Press
Religion
Skills
Popular
Belle-Lettres
Gov’t
Learned
Fiction
Humor


Poisson doesn’t fit
# of docs
freq
10^-3
10^-1
10^1
10^3
0
5
10
15
20
Negative Binomial
Poisson



Hidden Variables

- Ngram models assume a single parameter (θ) per ngram.
- No dependencies on hidden variables
  (e.g., genre, author, topic, etc.)

Interpretation of Negative Binomial

Mixture of Poissons

1. Within doc, words are generated by a Poisson (θ)
2. But θ varies from doc to doc (φ)

$$Pr_{NB}(x) = \int_0^\infty \phi(\theta)\ \pi(\theta, x)\, d\theta \quad \text{for } x = 0, 1, \cdots$$

The Negative Binomial

$$(P+Q)^N = \sum_{k=0}^{N} \begin{bmatrix} N \\ k \end{bmatrix} P^k \ Q^{N-k} \qquad \text{Binomial}$$

$$Pr_B(k) = \begin{bmatrix} N \\ k \end{bmatrix} P^k \ Q^{N-k} \quad \text{for } k = 0, 1, \cdots, N$$

$$P+Q=1$$

$$(Q-P)^{-N} = \sum_{k=0}^{\infty} \begin{bmatrix} N+k-1 \\ k \end{bmatrix} P^k \ Q^{-N-k} \qquad \text{NB}$$

$$Pr_{NB}(k) = \begin{bmatrix} N+k-1 \\ k \end{bmatrix} P^k \ Q^{-N-k} \quad \text{for } k = 0, 1, \cdots$$

$$Q-P=1$$

Interpretation of Binomial

- P is probability of success
  (next word is ‘‘Kennedy’’)

- Q is probability of failure
  (next word is not ‘‘Kennedy’’)

- N is number of trials
  (number of words in doc)

Interpretation of Negative Binomial

$Pr_{NB}(x) = \begin{bmatrix} N+x-1 \\ x \end{bmatrix} P^x \ Q^{-N-x}$ can be expressed as an mixture of Poissons: $\pi(\theta,k) = \dfrac{e^{-\theta} \ \theta^k}{k!}$

$$Pr_{NB}(x) = \int_0^{\infty} \phi(\theta) \ \pi(\theta,x)\, d\theta \quad \text{for } x = 0,1, \cdots$$

where ϕ determines how much θ varies
from one doc to the next

$\phi(\theta)$ has a broad distribution for ‘‘Kennedy’’;
and a narrow distribution for ‘‘the’’

Negative Binomial is better than Binomial

- Both binomial and negative binomial have the same symbolic expression for mean (NP) and variance (NPQ).

- But $Q < 1$ in the binomial,
and $Q > 1$ negative binomial.

- Consequently, var < mean in binomial,
and var > mean in negative binomial.

- Empirically, var > mean, and therefore,
negative binomial is more appropriate than binomial.

More Variability → More Content

mean	var	IDF	H	P21	
0.29	3.51	3.45	0.66	0.50	Government
0.27	2.63	3.53	0.60	0.44	Island
0.25	2.23	3.68	0.59	0.54	Church
0.27	1.99	3.65	0.60	0.55	Federal
0.29	1.80	3.53	0.64	0.60	Christian
0.28	1.75	3.72	0.59	0.58	Kennedy
0.26	1.67	3.84	0.55	0.80	Soviet
0.28	1.52	3.02	0.78	0.42	East
0.29	1.31	2.71	0.86	0.36	William
0.29	1.24	2.90	0.82	0.39	North
0.28	1.17	2.81	0.83	0.37	French
0.26	0.96	2.89	0.79	0.32	George
0.27	0.72	2.70	0.86	0.35	City
0.26	0.66	2.64	0.87	0.35	During
0.28	0.58	2.61	0.92	0.43	Well
0.25	0.54	2.71	0.85	0.39	I’ve
0.27	0.43	2.37	0.94	0.28	Yet
0.29	0.38	2.19	0.98	0.25	Here
0.25	0.25	2.18	0.89	0.12	*Poisson*
0.29	0.29	2.00	0.97	0.14	*Poisson*

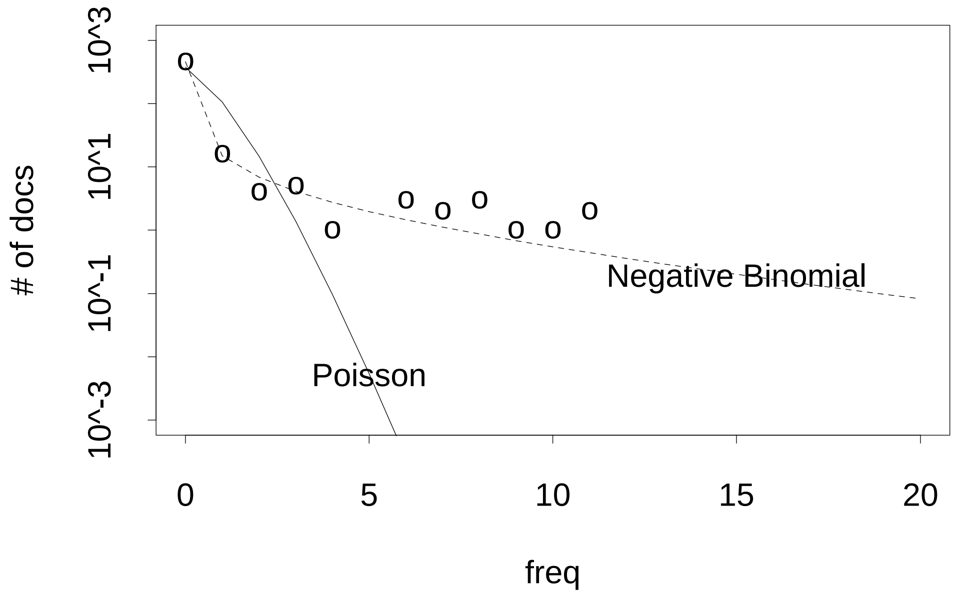
Katz' Observation: Terms Vary more than Cliches

mean	var	IDF	H	P21	
0.04	1.37	4.86	0.09	0.60	menu item
0.04	0.44	4.89	0.08	0.55	access tandem
0.03	0.17	5.12	0.07	0.78	fault density
0.03	0.40	4.86	0.09	0.53	optimal solution
0.04	0.46	4.73	0.09	0.53	signal strength
0.03	0.32	4.96	0.08	0.59	office code
0.03	0.37	4.76	0.09	0.64	function key
0.03	0.28	4.89	0.08	0.62	growth temperature
0.04	0.25	4.50	0.12	0.58	transfer function
0.04	0.38	4.50	0.11	0.37	test session
0.03	0.48	4.86	0.08	0.53	data design
0.04	0.72	4.29	0.13	0.47	overload control
0.04	0.35	4.18	0.15	0.49	execution time
0.04	0.10	3.85	0.18	0.34	high resolution
0.04	0.12	3.76	0.20	0.36	error detection
0.03	0.07	4.18	0.14	0.42	communication link
0.04	0.09	3.78	0.19	0.38	solid line
0.03	0.06	4.09	0.15	0.34	new technology
0.04	0.08	3.57	0.22	0.27	short term
0.03	0.05	4.03	0.15	0.25	above table
0.03	0.04	3.73	0.18	0.15	latter case
0.03	0.03	3.77	0.17	0.09	small amount

Symptoms of Fat Tails

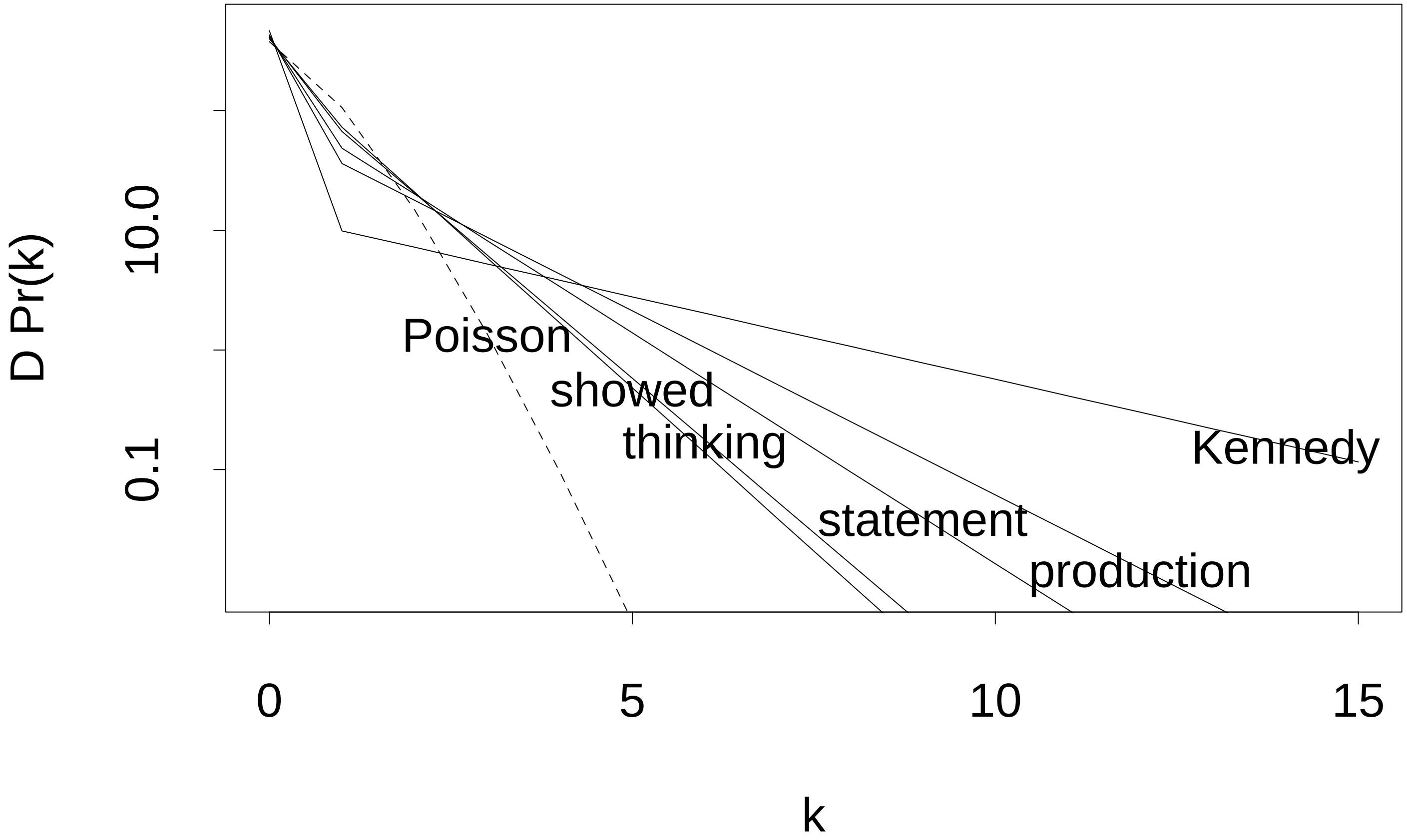
1. large variance (σ^2)
2. small doc freq (*df*)
3. small entropy (*H*)
4. lots of content

Poisson doesn’t fit



More content $\rightarrow$ Less df

freq	df	Word
140	38	Kennedy
141	62	East
140	68	letter
140	71	production
140	75	son
140	82	Well
141	83	statement
141	90	increased
141	90	results
140	97	thinking
140	99	start
141	99	addition
141	101	showed
141	107	decided
	122	*Binomial* or *Poisson*


D Pr(k)
10.0
0.1
Poisson
showed
thinking
Kennedy
statement
production
0
5
10
15
k

Bursty words tend to have more content

freq	df = 1 (bursty)	df = freq (diffuse)
15	Blackman, Dandy, Drug’s, Eugenia, Fromm’s, Hardy’s, Juanita, Selden, Ulyate, collage, tappet	Naturally, Norman, Otherwise, Somehow, Thank, cease, claiming, clue, confident, indispensable, landed, originated, plunged, restricted, sweep, termed
16	Gilborn, Handley, Hanford, Nicolas, Styka, Willis, clover, leveling, secants, thyroglobulin	Already, Back, None, Right, absurd, appearing, collect, delighted, deserves, devised, discussing, faster, inherited, legitimate, lined, link, men’s, persuade, piled, praise, refuse, severely, shops, sole, spreading, thereafter, unnecessary, waved
17	Angie, BOD, Giffen, Krim, Lalaurie, Lizzie, Moreland, Nadine, TSH, Trevelyan, accelerometer	35, Go, K., artificial, capture, consistently, designated, expecting, formally, grasp, lit, obscure, pushing, respective, spontaneous, surprisingly, vitality
18	Andrei, Barco, Helion, Keys, Kitti, Langford, Madden, Saxon, Stevie, Upton, effluent, nonspecific	Beyond, avoided, birthday, emphasized, escaped, gather, instantly, packed, proceed, repeatedly, sixty, submit, surrounded
19	Haney, Killpath, Letch, tetrachloride, tsunami	Which, alike, amazing, bold, happily, notable, overwhelming, remainder, rid, rush, savage, whereby

Recap

- Standard criticism of ngram models:
  Unbounded Dependencies
  (Is $n \approx 3$ enough context???)

- True, but there are more serious problems:

  1. Sparseness (too many ngrams and never enough training data)

  2. Contagiousness (if there is one ‘‘Kennedy,’’ there’ll probably be another)

Upcoming Topics

- Suffix Arrays: Efficient Calculation of Long Ngrams
- Dotplots: Visualizing Ngrams
- Applications:
  - Part of Speech Tagging
  - Spelling Correction

Suffix Arrays

- Suffix Array Data-Structure:
  Encodes the frequency and location of long ngrams

- Finding Ngrams in a Suffix Array

- Longest Common Prefix (LCP)

- Distribution of LCPs:
  Long Ngrams are not unusual.
  Trigrams???

Computing Suffix Arrays

Simple but slow algorithm:

1. Input a text of length N
2. Construct an array, *suf*, consisting of the integers from 1 to N.
3. Let each integer, i, denote the suffix starting at position i in the input text.
4. Sort *suf* by lexicographic order

Complexity: $O(N^2 \log N)$ time and $O(N)$ space;
there is an $O(N \log N)$ solution

```
#include <fcntl.h>
#include <malloc.h>
#include <stdio.h>
#include <sys/mman.h>
#include <sys/stat.h>
#include <sys/types.h>
/* usage: sufsort1 text > text.suf */
char *text;

suffix_compare(int *a, int *b)
{
    return strcmp(text + *a, text + *b);
}

main(int ac, char **av)
{
    struct stat stat_buf;
    int N, i, *suf;
    FILE *fd = fopen(av[1], "r");
    fstat(fileno(fd), &stat_buf);
    N = stat_buf.st_size;
    text = (char *)malloc(N+1);
    fread(text, sizeof(char), N, fd);
    text[N] = 0;      /* pad with null */

    suf = (int *)malloc(N * sizeof(int));
    for(i=0;i<N;i++) suf[i]=i;

    qsort(suf, N, sizeof(int), suffix_compare);
    fwrite(suf, N, sizeof(int), stdout);
}
```

Finding Ngrams in a Suffix Array

- Use a binary search to find the freq and location of long ngrams, *key*.

- This algorithm takes $O(n \log N)$ time; there is an $O(n + \log N)$ solution.

```
int *lookup(char *key, char *text, int *suf,
            int N, int roundup)
{
  int *left = suf;
  int *right = suf + N;
  int n = strlen(key);
  for(;;) {
    int *mid = left + (right - left)/2;
    int c = strncmp(key, text + *mid, n);
    if(mid == left) {
      if(roundup && c >= 0) return right;
      if(c >= 0) return right;
      return left;
    }
    if(c < 0) right = mid;
    else if(c == 0 && !roundup) right=mid;
    else left = mid;
  }
}
```

- $roundup = 0 \rightarrow$ find the first instance of *key* in *suf*
  $roundup = 1 \rightarrow$ find the last instance of *key* in *suf*

- Take the difference to compute freq of *key*

- De-reference pointers to obtain locations

Longest Common Prefix (LCP)

LCP: an array of N integers, indicating the length of the common prefix between the i^{th} and $i+1^{st}$ suffix.

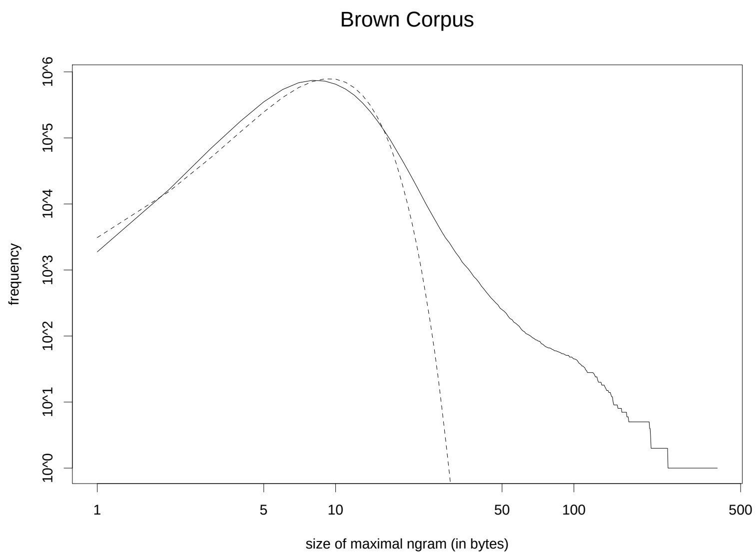
Common_Prefix("abcd", "abce") $\rightarrow$ 3

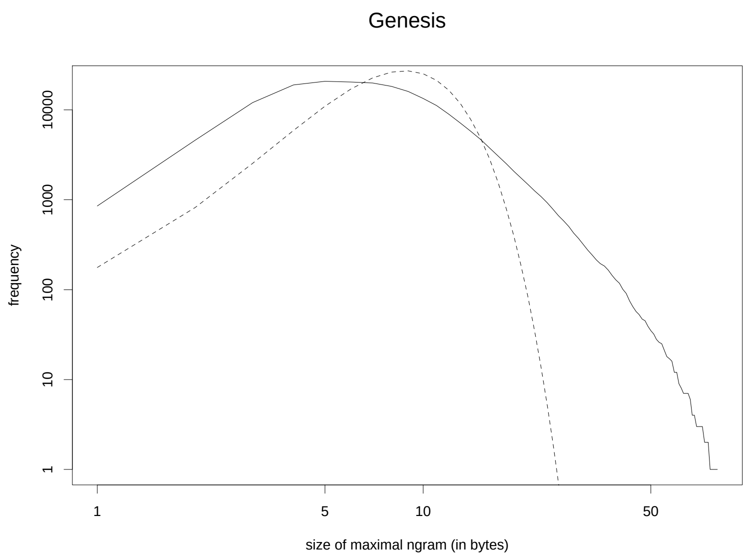
```
int common_prefix(char *a, char *b)
{
    int result = 0;
    while(*a && *a++ == *b++)
          result++;
    return result;
}
```

```
main(int ac, char **av)
{
    ...
     suf = (int *)malloc(N * sizeof(int));
     lcp = (int *)malloc(N * sizeof(int));
     for(i=0;i<N;i++) suf[i]=i;
     qsort(suf, N, sizeof(int), suffix_compare);
     for(i=0;i<N;i++)
       lcp[i] = common_prefix(text + suf[i], text + suf[i+1])
     fwrite(suf, N, sizeof(int), stdout);
    ...
}
```

This algorithm takes $O(N^2)$ time;
there is an $O(N)$ solution.

```
# create the suffix array
$ sufsort1 data/genesis > data/genesis.suf

# make a concordance of a phrase
$ echo ’she conceived again’ | \
     suflookup data/genesis | \
     pcontext –l15 –r40 data/genesis
evi._29:35 And she conceived again, and bare a son: and
 me._29:33 And she conceived again, and bare a son; and
eon._29:34 And she conceived again, and bare a son; and
e Er._38:4 And she conceived again, and bare a son; and

# compute the shared lengths (LCPs)
$ sufshared data/genesis > data/genesis.shared

# find suffixes with long LCPs (LCP > 50 chars)
$ itoa < data/genesis.shared | \
     awk ’$1 > 50 {print $1 "\t" NR–1}’ > /tmp/long

# print the first few suffixes with long LCPs
$ awk ’{print $2}’ /tmp/long | atoi | \
     aref –s4 –a data/genesis.suf | \
     pcontext –l0 –r50 data/genesis | head
 And she conceived again, and bare a son; and said
 And, behold, there came up out of the river seven
 Asenath the daughter of Potipherah priest of On b
 Be fruitful, and multiply, and replenish the eart
 Behold now, I have taken upon me to speak unto th
 I lifted up my voice and cried, that he left his
 Thou shalt not take a wife of the daughters of Ca
 Ye shall not see my face, except your brother [be
 after his kind: and God saw that [it was] good._1
 after his kind: and God saw that [it was] good._1
```


Brown Corpus
frequency
10^0
10^1
10^2
10^3
10^4
10^5
10^6
1
5
10
50
100
500
size of maximal ngram (in bytes)


Genesis
frequency
1
10
100
1000
10000
1
5
10
50
size of maximal ngram (in bytes)

Trigrams???

- Trigrams are the method of choice in many applications such as speech recognition

- Possible explanations:
    1. Trigrams are good enough in most applications

    2. Few have looked at longer ngrams,
       because they were believed to be
       computationally prohibitive.
       (looking under the lamppost)

- Open question:
  Now that it is possible to go beyond trigrams,
  should trigrams remain the method of choice?

- Fears: long ngrams demand better smoothing methods, and highlight weaknesses in binomial assumptions.

Dotplot

Applications

- Part of Speech Tagging
- Spelling Correction

Part of Speech Tagging

Examples

- He/PPS will/MD table/VB the/AT motion/NN ./.
- The/AT table/NN is/BEZ ready/JJ ./.

[A/AT former/AP top/NN aide/NN] to/IN [Attorney/NP/NP General/NP/NP Edwin/NP/NP Meese/NP/NP] interceded/VBD to/TO extend/VB [an/AT aircraft/NN company/NN ’s/$ government/NN contract/NN] ,/, then/RB went/VBD into/IN [business/NN] with/IN [a/AT lobbyist/NN] [who/WPS] worked/VBD for/IN [the/AT defense/NN contractor/NN] ,/, according/IN to/IN [a/AT published/VBN report/NN] ./.

Motivation

Part of Speech Tagging is an important practical problem with numerous applications.

- Speech Synthesis (TTS)
- Speech Recognition
- Optical Character Recognition (OCR)
- Information Retrieval (IR)
- Spelling Correction
- Proof-Reading (WWB)
- Query Answering (Q&A)
- Machine Translation (MT)
- Tagging Corpora for future research (COBUILD)

Speech Synthesis Applications

- The *WIND* is strong.
- Don’t forget to *wind* your watch.

- Did you see *THAT*?
- It is a shame *that* he’s leaving.

- oily *FLUID*
- *TRANSMISSION* fluid

OCR Application

Input	Output			
Byzantine	1.00	Byzantine		
icons	1.00	icons		
could	1.00	could		
murder	1.00	murder	0.33	warder
the	1.00	the		
divine	1.00	divine		
identity	1.00	identity	0.57	identify

Byzantine/JJ
icons/NNS
could/MD
murder_warder/VB
the/AT
divine/JJ
identity_identify/NN

Syntactic Constraints May Not Help Recognition Very Much

which_Which/WDT
is/BEZ
only_Only/RB
the/AT
pure_pare_pave/JJ
form_farm/NN
of_Of/IN
triumphal/JJ
ethnology/NN
,/,

- identity vs. identify
- pure form vs. pure farm

The State-of-the-Art

- As a result of corpus collection efforts such as

    1. the Tagged Brown Corpus,
    2. the Penn Treebank, and
    3. similar efforts within the ICAME community,

  there are now quite a number of extremely successful part of speech tagging programs which make use of probabilities derived from corpus data.

- These programs work on unrestricted texts,
- with reasonable accuracy and efficiency.

- 95-99% of the words are ‘‘correctly’’ tagged,

- which is generally regarded as a major advance over previously available alternatives such as ATNs.

Is 95-99% performance good enough?

- On the one hand, it is better than we have been doing before n-gram part of speech taggers came into fashion,

- but on the other hand, it still means that a large fraction of sentences will contain at least one fatal error.

- If subsequent processing (e.g., parsing, semantic analysis) require perfect part of speech analysis, then 95% performance is clearly not nearly good enough, and probably 99% isn't either.

- Perhaps we need to modify these subsequent steps so they can tolerate an error rate of 1-5%. Alternatively, we may need to aim for somewhat higher levels of tagging performance than we can currently achieve.

How Hard is the Problem?

- 95% might sound good,

- but really dumb methods do almost as well.

- If we simply ignore the context, and just select the most likely part of speech given the word, we will achieve nearly 90% correct.

- (Some methods manage to fall below this baseline by focusing on the grammar rather than the lexicon.)

- 95% may not sound so good when we realize that the lexicon gives you the first 90%, and context contributes only about half of the remaining 10%.

Intuition

- Many people who have not worked in computational linguistics have a strong intuition that lexical ambiguity is usually not much of a problem.

- It is commonly believed that most words have just one part of speech, and that the few exceptions such as ‘‘table’’ are easily disambiguated by context in most cases.

- This intuition is largely supported by the numbers just cited.

- That is, most cases can be resolved without context (e.g., 90%), and that simple n-gram models of context are sufficient for more than half of the remainder.

Key Lesson: Focus on the Lexicon

- Focusing on lexicon has produced results:

  - Much of this progress has been achieved because lexical probabilities, $Pr(pos_i | word_i)$, are now being estimated directly from corpus data,

  - and can therefore be estimated much more accurately than before.

  - Until recently, it had been common practice for most researchers in computational linguistics to concentrate their energies on modeling contextual constraints (e.g., grammar),

  - which appears to be much less important than lexical probabilities (e.g., the dictionary), at least for the part of speech tagging application.

Suggestions for the Future

- We should continue to focus on the lexicon...
  - Lexical probabilities are actually much more difficult to estimate than one might have thought.
  - The lexical probabilities tend to have more parameters than the contextual probabilities ($40\ V >> 40^3$).
  - Moreover, the relationships among lexical items turn out to be extremely subtle.
  - One would hope that it would be possible to predict the lexical probabilities by making use of what we know about morphologically related forms.
  - But even this apparently benign step is fraught with peril, we as will see.

The Statistical Approach

- These days, most part of speech programs take a statistical approach,
    - Leech *et al.* (1983), Jelinek (1985), Deroualt and Merialdo (1986), Church (1988), DeRose (1988), Kupiec (1989), Ayuso *et al.* (1990), de Marcken (1990), Boggess *et al.* (1991), Merialdo (1991)
- though there are a few recent exceptions
    - Heidorn *et al.* (1982), Martin *et al.* (1987), Hindle (1989), Karlsson (1990)
- Most of these statistical programs use a linear time dynamic programming algorithm to find an assignment of parts of speech to words that optimizes the product of
    1. lexical probabilities, $Pr(pos_i | word_i)$, and
    2. contextual probabilities, $Pr(pos_i | pos_{i-1}\ pos_{i-2})$.

The Noisy Channel Model

pos $\rightarrow$ Noisy Channel $\rightarrow$ words

$$\underset{pos}{ARGMAX}\ Pr(pos)\ Pr(words \mid pos)$$

Under certain indep assumptions, this can be approximated as:

$$\underset{pos}{ARGMAX} \prod_i Pr(pos_i \mid pos_{i-1}\, pos_{i-2})\ Pr(words_i \mid pos_i)$$

There is a dynamic programming algorithm which finds the best assignment of n pos to n input words in $O(n)$ time.

Notational Convenience

- I prefer to rewrite

  $$Pr(words_i | pos_i)$$

  as

  $$Pr(pos_i | words_i) \ \frac{Pr(words_i)}{Pr(pos_i)}$$

  since I find it convenient to think of $Pr(pos_i | words_i)$ as a dictionary,

  and to think of $Pr(pos_i | pos_{i-1} \, pos_{i-2})$ as a grammar.

- The crux of the problem, in my view, is to estimate $Pr(pos_i | words_i)$.

- Everything else is relatively easy and relatively unimportant.

Lexical Probabilities of Some Common Words

Word	Pos	P(Pos | Word)	Pos	P(Pos | Word)
costs	VBZ	0.08	NNS	0.92
human	NN	0.12	JJ	0.88
humans	NNS	1.00		
amount	VB	0.18	NN	0.82
amounts	VBZ	0.46	NNS	0.54
meeting	VBG	0.20	NN	0.80
attack	VB	0.25	NN	0.75
mark	VB	0.30	NN	0.70
support	VB	0.31	NN	0.69
change	VB	0.32	NN	0.68
sacrifice	VB	0.37	NN	0.63
use	VB	0.38	NN	0.62
gathering	VBG	0.45	NN	0.55
strike	VB	0.46	NN	0.54
sink	VB	0.48	NN	0.52
travel	VB	0.49	NN	0.51
landing	VBG	0.52	NN	0.48
count	VB	0.58	NN	0.42
finish	VB	0.58	NN	0.42
ride	VB	0.68	NN	0.32
dancing	VBG	0.69	NN	0.31
draw	VB	0.80	NN	0.20
remains	VBZ	0.87	NNS	0.13

Typical Contextual Probabilities

log P	(Pos |	__ Pos,	Pos)	log P	(Pos |	__ Pos,	Pos)
–0.81	IN	AT	NNS	–4.31	VBN	AT	NN
–0.83	IN	AT	NN	–4.47	VBN	AT	NNS
–2.28	.	AT	NNS	–4.49	NN	AT	NN
–2.51	.	AT	NN	–4.67	NN	AT	NNS
–2.54	VB	AT	NNS	–5.27	NNS	AT	NN
–2.72	VB	AT	NN	–5.60	JJ	AT	NN
–2.97	,	AT	NNS	–5.64	NNS	AT	NNS
–2.98	,	AT	NN	–5.95	NP	AT	NN
–3.12	VBD	AT	NN	–5.96	NP	AT	NNS
–3.28	VBD	AT	NNS	–6.26	JJ	AT	NNS
–3.41	VBG	AT	NNS	–8.21	PPSS	AT	NNS
–3.62	VBG	AT	NN	–8.90	AT	AT	NNS
–3.88	RB	AT	NN	–9.42	PPSS	AT	NN
–4.17	RB	AT	NNS				

The Proposed Method

- Conceptually, enumerate all assignments
- Score each path (product of lexical and contextual probabilities)
- Select best

I	see	a	bird
PPSS	VB	AT	NN
PPSS	VB	IN	NN
PPSS	UH	AT	NN
PPSS	UH	IN	NN
NP	VB	AT	NN
NP	VB	IN	NN
NP	UH	AT	NN
NP	UH	IN	NN

- Conceptually, there could be k^n part of speech sequences, where n is the length of the input sentence, and k is the (worst case) lexical ambiguity.
- Fortunately, there is a linear time dynamic programming solution.
- If two paths are the same within the ngram window of 3 words, then keep the just better one.
- This way, there will be at most nk^3 paths to consider.

	.	.	I	see	a	bird	.	.	
A1	.	.	PPSS	VB	AT	NN	.	.	
context	0.99	0.20	0.07	0.07	0.23	0.25	1.00	1.00	e-4
lex	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	
A2	.	.	PPSS	VB	IN	NN	.	.	
context	0.99	0.20	0.08	0.03	0.13	0.25	1.00	1.00	e-9
lex	1.00	1.00	1.00	1.00	e-4	1.00	1.00	1.00	
A3	.	.	PPSS	UH	AT	NN	.	.	
context	0.99	1.00	0.00	0.00	0.23	0.25	1.00	1.00	0
lex	1.00	1.00	1.00	e-3	1.00	1.00	1.00	1.00	
A4	.	.	PPSS	UH	IN	NN	.	.	
context	0.99	1.00	0.00	0.00	0.13	0.25	1.00	1.00	0
lex	1.00	1.00	1.00	e-3	e-4	1.00	1.00	1.00	
A5	.	.	NP	VB	AT	NN	.	.	
context	0.97	0.03	0.01	0.07	0.23	0.25	1.00	1.00	e-10
lex	1.00	1.00	e-4	1.00	1.00	1.00	1.00	1.00	
A6	.	.	NP	VB	IN	NN	.	.	
context	0.97	0.03	0.01	0.03	0.13	0.25	1.00	1.00	e-15
lex	1.00	1.00	e-4	1.00	e-4	1.00	1.00	1.00	
A7	.	.	NP	UH	AT	NN	.	.	
context	0.97	0.00	0.00	0.00	0.23	0.25	1.00	1.00	0
lex	1.00	1.00	e-4	e-3	1.00	1.00	1.00	1.00	
A8	.	.	NP	UH	IN	NN	.	.	
context	0.97	0.00	0.00	0.00	0.13	0.25	1.00	1.00	0
lex	1.00	1.00	e-4	e-3	e-4	1.00	1.00	1.00	

On the Inadequacy of Ngram Models

It is surprising that a local ‘‘bottom-up’’ approach can perform so well. One might have thought that ngram models weren’t adequate for the task. Recall that statistical ngram models were quite common in the 1950s when Information Theory was hot, but lost popularity among computational linguistics when it was demonstrated by Chomsky that ngram models lacked the generative capacity to capture certain syntactic generalizations, especially subject-verb agreement.

> ‘‘We find that no finite-state Markov process that produces symbols with transition from state to state can serve as an English grammar. Furthermore, the particular subclass of such processes that produce n-order statistical approximations to English do not come closer, with increasing *n*, to matching the output of an English grammar.’’ [Chomsky, p. 113]

- Ngram models are inadequate for many applications,
- but may be acceptable for tagging since long distance dependencies do not seem to be very important (most of the time).
- The ngram approximation is not as bad as some others that are often made...

Three Approaches

- Non-deterministic (ATN): try all possibilities and hope the bad ones are ungrammatical (punt and return all possibilities)

- Deterministic (Marcus): make a single pass over the input, delay bindings as long as possible, and guess only when people seem to (ambitious)

- Statistical: make a single pass over the input, delay bindings for 3 words, and then optimize (guess).

Deterministic and statistical approaches are similar; both try to find a single ‘‘best’’ interpretation with limited resources (linear time)

Statistical approach is more likely to work in short term.

The Non-deterministic Non-Solution

- Although most naive people think that most words are unambiguous, so-called ‘‘experts’’ know better.
- It is said that practically any word is noun-verb-adj ambiguous.
- The literature is full of examples where no amount of context will help:
  - Time flies like an arrow.
  - Flying planes can be dangerous.
- These example sentences are generally taken to indicate that the parser must allow for multiple possibilities, and that subsequent levels of processing (e.g., semantics, pragmatics) will be required in order to resolve the ambiguity.
- Unfortunately, this strategy has not worked out well in practice because it tends to ignore the lexical probabilities, which are, in fact, the single most important set of constraints.
- Precision vs. Recall

What's wrong with the ATN approach?

- Consider the trivial sentence: *I see a bird.*

- As we have seen, this is easy for the statistical method because every word is (almost) unambiguous:

- Prob(I is a pronoun) = 5837/5838

  Prob(see is a verb) = 771/772

  Prob(a is an article) = 23013/23019

  Prob(bird is a noun) = 26/26

- But, according to Websters, every word is ambiguous...

Word	**Parts of Speech**	
I	pronoun	noun
see	verb	noun
a	article	noun
bird	noun	verb

One might hope that these spurious assignments could be ruled out by the parser as syntactically ill-formed...

But unfortunately, this is unlikely to work. If the parser is going to accept noun phrases of the form:

- [NP [N city] [N school] [N committee] [N meeting]]

then it can’t rule out

- [NP [N I] [N see] [N a] [N bird]]

Similarly, the parser probably also has to accept ‘‘bird’’ as an intransitive verb, since there is nothing syntactically wrong with:

- [S [NP [N I] [N see] [N a]] [VP [V bird]]]

These part of speech assignments aren’t wrong; they are just extremely improbable.

Supervised vs. Unsupervised Training

- I have always prefered to train on hand-tagged text,
- though many others have advocated the use of self-organizing re-estimation techniques that do not require the availability of hand-tagged training material.

- Are hand-tagging efforts worthwhile?

    1. the Tagged Brown Corpus,
    2. the Penn Treebank, and
    3. similar efforts within the ICAME community

- Can they be replaced with self-organizing methods such as Baum-Welch re-estimation?

Yes and No

- Yes, data collection is a good thing, and
  No, there is no free lunch.

- Merialdo (1991) found that tagged text is preferable to re-estimation when tagged text is available in sufficient quantity.

- Someone ought to find a way to combine ‘‘small’’ amounts of hand-tagged text (e.g., 0.01 Gwords) and large amounts of untagged text (e.g., 1-5 Gwords).

Modern Hand-Tagging (Semi-Automatic Re-Estimation)

- Two Modes:

  1. Tagging
  2. Correcting

- Marcus & Santorini found that Correcting was faster and more accurate than Tagging.

- Note that human ‘‘error’’ rates are fairly high (%5); most people would think that the task is easier than it is.

- Two kinds of ‘‘errors’’:

  1. blunders
  2. differences of opinion

Smoothing Issues

- Must do something with Zeros
- Zipf’s Law: there will always be a large tail of low frequency words
- 40,000 words in the Brown Corpus have freq < 5
- If ‘‘yawn’’ appears once as a noun and once as a verb, what is the probability that it could be an adjective?
- Dictionaries (and morphology)
- The fact that none of our dictionaries list ‘‘yawn’’ as an adjective should suggest that the probability is fairly small.
- Unfortunately, it turns out to be very difficult to combine evidence from different sources in a principled way.

Smoothing Across Morphological Variants

- One would hope that it would be possible to predict the lexical probabilities by making use of what we know about morphologically related forms.

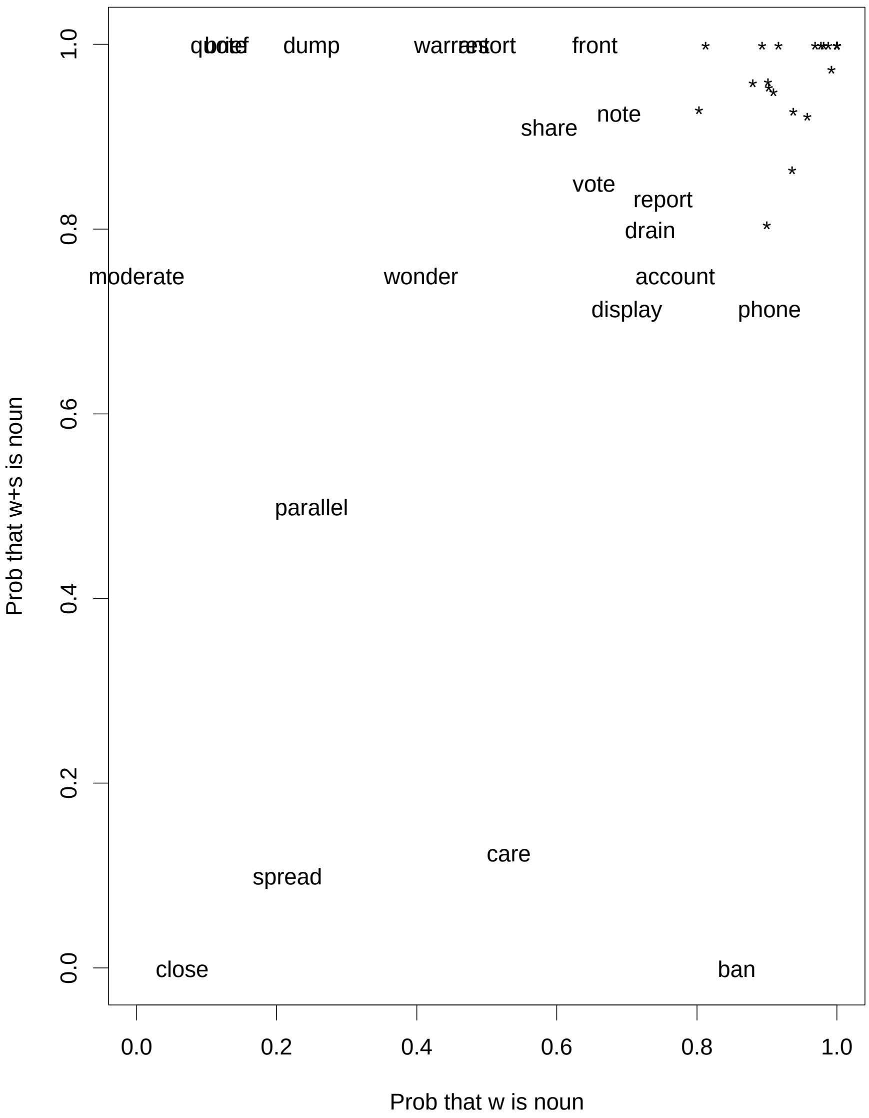
- Even this apparently benign step is fraught with peril.

- One might think that it would be fairly safe to assume that adding an ’s’ to the end of a word would not change its lexical probabilities very much.

- Unfortunately, even this seemingly innocuous move can lead to serious difficulties.

Word	**Base Form**			**Base Form + s**		
	noun	non–noun	%	noun	non–noun	%
abuse	13	3	81	7	0	100
account	90	27	77	27	9	75
ban	6	1	86	0	1	0
bar	68	3	96	36	3	92
brief	9	61	13	1	0	100
care	85	75	53	1	7	12
center	175	12	94	45	7	87

Figure 1


quote
brief
dump
warrant
resort
front
share
note
vote
report
drain
moderate
wonder
account
display
phone
parallel
care
spread
close
ban
0.0
0.2
0.4
0.6
0.8
1.0
Prob that w is noun
Prob that w+s is noun

Smoothing Contextual Probabilities

- The lexical probabilities are not the only probabilities that require smoothing.
- The contextual probabilities also raise some interesting estimation questions.
- They too tend to have a very skewed distribution, and consequently, even after looking at a very large training corpus, there will still be many n-grams that have not been observed.
- It is clear that the contextual frequencies require smoothing. Zeros should be avoided.
- Nevertheless, I have relatively little to say about estimating the contextual probabilities for three reasons:
    1. it relatively well-studied,
    2. it is relatively easy since there are relatively few parameters to estimate ($40^3 << 40\ V$),
    3. it is relatively unimportant since the contextual probabilities don't matter very much compared with the lexical probabilities.

Part-of-Speech Conclusion

- Statistical part of speech programs currently dominate the practice,
- because they are are *real*,
- (unlike previously available alternatives such as ATNs).
- This approach is extremely empirical.
- Empiricism fell out of favor in the 1950s,
when Chomsky and others pointed out some of its limitations.
- But it has recently enjoyed a strong come-back in NLP, because data is becoming so much easier to collect.
- Back in 1960, it was a big deal to collect the Brown Corpus (1 Mword), but these days I talk about Gwords.

Lesson from Speech Recognition Research

- Empirical methods are often helpful when:
  - Data rates are high,
  - There is plenty of training material, and
  - Nothing else seems to work very well (because we don’t know what we’re doing).
- Probability vs Possibility
- Psycholinguistics is hard; if you want to find a syntactic effect, you have to learn to control for everything that matters: word frequencies, word association norms, etc. Maybe these factors are important for something ...
- Breadth vs Depth

Problems

- Flying Planes and friends
  [Time/NN] flies/VBZ like/CS [an/AT arrow/NN] ./.
  [Fruit/NN] flies/VBZ like/CS [a/AT banana/NN] ./.

  [Flying/VBG planes/NNS] can/MD be/BE dangerous/JJ ./.
  [They/PPSS] are/BER flying/VBG [planes/NNS] ./.

- Inadequate window size
  [The/AT horse/NN] have/HV raced/VBN past/IN [the/AT barn/NN] ./.
  [The/AT horse/NN] has/HVZ slipped/VBN ./.
  [The/AT horse/NN] has/HVZ raced/VBN past/IN [the/AT barn/NN] and/CC slipped/VBD ./.

- Unknown words
  Do/DO [you/PPSS] know/VB [what/WDT] [a/AT xxx/NN] is/BEZ ?/.
  [I/PPSS] know/VB [care/NN] if/CS [you/PPSS] xxx/VB !/.
  [I/PPSS] need/MD xxx/VB ./.

- Lack of word association norms,
  semantics, pragramatics
  [I/PPSS] like/VB to/TO work/VB ./.
  [I/PPSS] went/VBD to/TO work/VB ./.
  [I/PPSS] went/VBD to/IN [school/NN] ./.

- Garden Paths
  [The/AT horse/NN] raced/VBD past/IN [the/AT barn/NN] fell/VBD ./.
  [The/AT horse/NN] taken/VBN past/IN [the/AT barn/NN] fell/VBD ./.

  [The/AT ship/NN] floated/VBD sank/VBD ./.
  [The/AT ship/NN] ,/, [which/WDT] was/BEDZ floated/VBN ,/, sank/V

Spelling Correction

```
echo absorbant adusted ambitios afte |
spell |
correct
```

absorbant	absorbent	
adusted	adjusted	100%
	dusted	0%
afte	after	100%
	fate	0%
	aft	0%
	ate	0%
	ante	0%
ambitios	ambitious	77%
	ambitions	23%
	ambition	0%

- lots of typos to train on
- 2000 / month (6% of lowercase word types)

sub[X, Y] = Sub of X (incorrect) for Y (correct)

X	Y (correct)					
	a	b	c	d	e	f
a	0	0	7	2	342	1
b	1	0	9	9	3	3
c	7	6	0	16	1	9
d	2	10	13	0	12	1
e	388	0	4	11	0	3
f	0	15	1	4	2	0

$$\underset{c}{ARGMAX}\ Pr(c)\ Pr(t|c)$$

$P(c)$ is a unigram model (no context for now)

$$Pr(t|c) \approx \begin{cases} \dfrac{del[c_{p-1},\ c_p]}{chars[c_{p-1},\ c_p]} \\ \dfrac{add[c_{p-1},\ t_p]}{chars[c_{p-1}]} \\ \dfrac{sub[t_p,\ c_p]}{chars[c_p]} \\ \dfrac{rev[c_p,\ c_{p+1}]}{chars[c_p,\ c_{p+1}]} \end{cases}$$

Some typos are frequent

AP Freq (44M words)	WSJ Freq (22M words)	Typo	Correction
106	15	goverment	government
71	21	occured	occurred
61	6	responsiblity	responsibility
47	2	negotations	negotiations
45	8	benefitted	benefited
45	13	commerical	commercial
41	0	assocations	associations
39	26	televison	television
38	1	millenium	millennium
38	9	possiblity	possibility
34	3	accomodate	accommodate
32	16	similiar	similar

‘‘goverment’’ is more frequent than many words

AP Freq	Word	AP Freq	Word
99	extinct	93	standby
99	pellets	92	attends
98	remorse	92	condors
97	lighted	91	coaches
97	marital	88	averted

Evaluation

- absurb, absorb, absurd
  ...financial community. ‘‘It is **absurb** and probably obscene for any person so engaged to...

	Judge 1	Judge 2	Judge 3
choice 0 (spell error)	99	124	93
choice 1	188	176	167
choice 2	175	159	151
other	28	26	30
?	74	79	123
total	564	564	564

The Judges found the task harder than anticipated.

Performance

Method	Discrimination	%
correct	286/ 329	87 ± 1.9
Judge 1	271/ 273	99 ± 0.5
Judge 2	271/ 275	99 ± 0.7
Judge 3	271/ 281	96 ± 1.1
channel-only	263/ 329	80 ± 2.2
prior-only	247/ 329	75 ± 2.4
chance	172/ 329	52 ± 2.8

The Task is Hard without Context

Typo	Choice 1	Choice 2
actuall	actual	actually
constuming	consuming	costuming
conviced	convicted	convinced
confusin	confusing	confusion
workern	worker	workers

Easier With Context

- actuall, actual, actually
  ...in determining whether the defendant **actuall** will die. In the 1985 decision, the...

- constuming, consuming, costuming
  ...on Friday night, a show as lavish in **constuming** and lighting as those the late Liberace used to...

- conviced, convicted, convinced
  ...of the area. ‘‘When we’re **conviced** and the Peruvians are convinced (the base camp)...

- confusin, confusing, confusion
  ...The political situation grew more **confusin** today, with an official media report indicating...

Syntax generally doesn’t help.

Bigram context helps some,
but people still do better.

	Model	%
1	channel	80
1	prior	76
1	left	78
1	right	77
2	channel + prior	87
2	channel + left	87
2	channel + right	88
2	prior + left	83
2	prior + right	80
2	left + right	86
3	channel + prior + left	90
3	channel + prior + right	88
3	channel + left + right	90
3	prior + left + right	86
4	channel + prior + left + right	90
	Judge 1	99
	Judge 2	99
	Judge 3	96

Conclusions

- Text is available like never before

- Exercises requiring only a few lines of Unix(TM) code
  1. Count words, bigrams, trigrams
  2. Ngram Stats: mutual info, t
  3. Concordances

- Hamming used to say it is much better to do the right problem naively than the wrong problem expertly.

- Infrastructure:
  - We *finally* have data
  - Most people believe it must be good for something
  - Missing: text analysis tools

- ‘‘You shall know a word by the company it keeps’’ (Firth, 1957)
    - Mutual Info and t-scores
    - Self-organizing vs. EDA vs. Stone Soup
- Hot Research Topics
    - Long Ngrams (Suffix Arrays)
    - Contagiousness (Negative Binomials)
    - Visualization (Dotplots)
- Applications
    - Part of Speech Tagging
    - Recognition