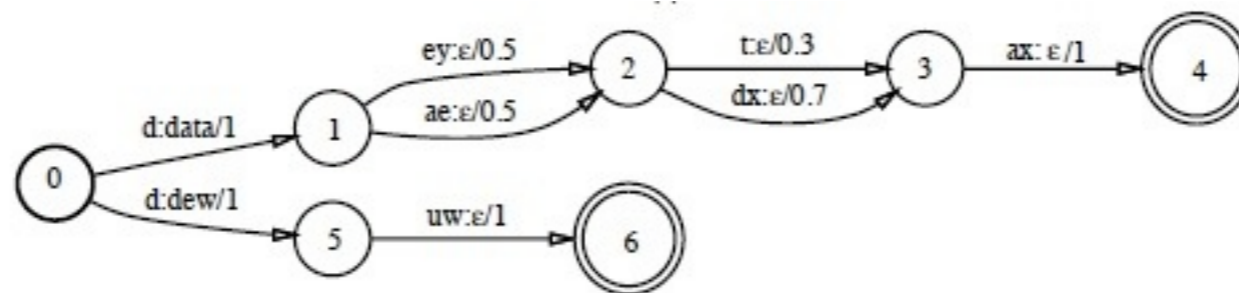


CS 562: Empirical Methods in Natural Language Processing

Unit 1: Sequence Models

Language Models



Fall 2011

Liang Huang (lihuang@isi.edu)

Python Review: Styles

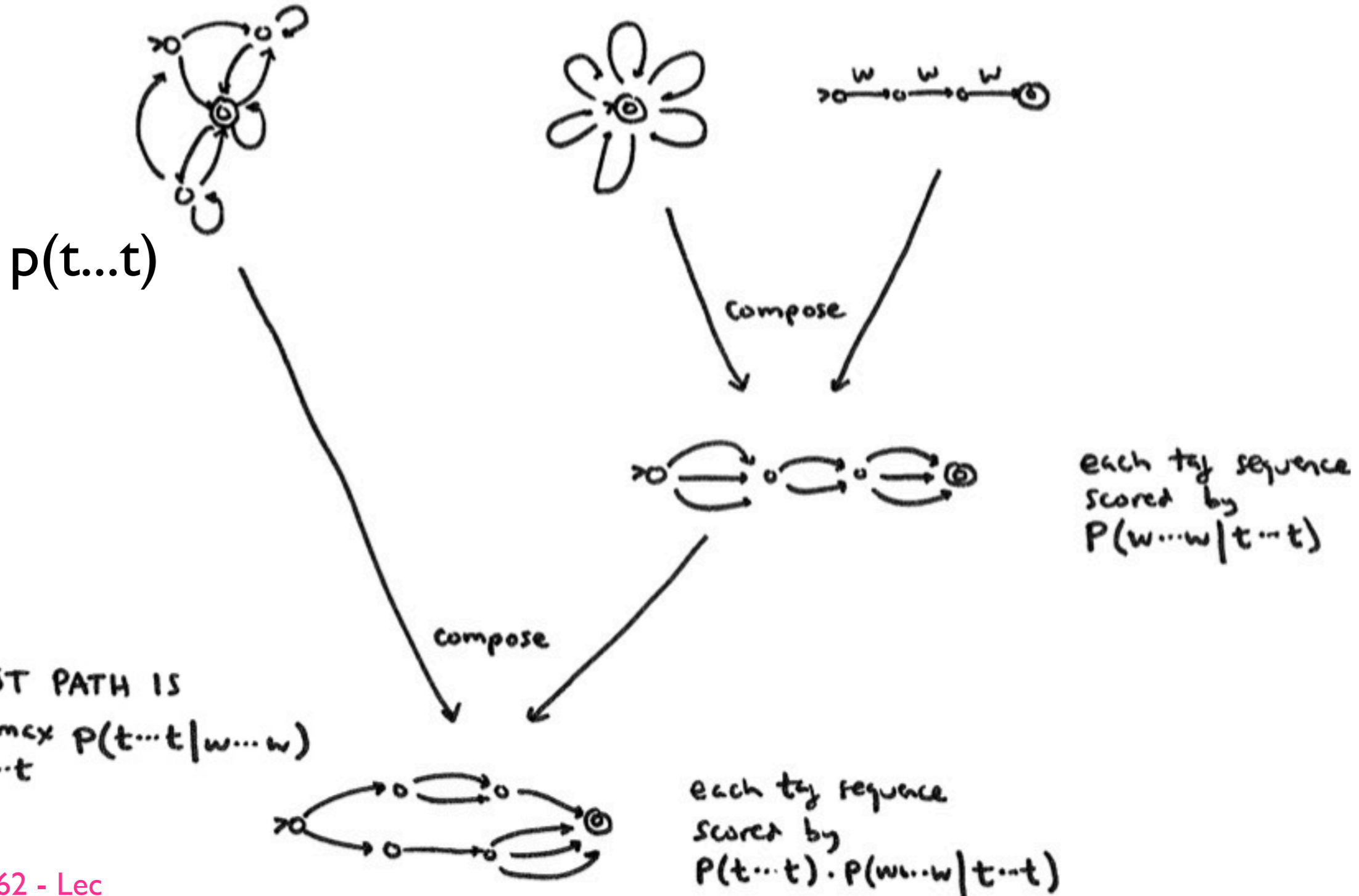
- do not write ... when you can write ...

<pre>for key in d.keys():</pre>	<pre>for key in d:</pre>
<pre>if d.has_key(key):</pre>	<pre>if key in d:</pre>
<pre>i = 0 for x in a: ... i += 1</pre>	<pre>for i, x in enumerate(a):</pre>
<pre>a[0:len(a) - i]</pre>	<pre>a[:-i]</pre>
<pre>for line in \ sys.stdin.readlines():</pre>	<pre>for line in sys.stdin:</pre>
<pre>for x in a: print x, print</pre>	<pre>print " ".join(map(str, a))</pre>
<pre>s = "" for i in range(lev): s += " " print s</pre>	<pre>print " " * lev</pre>

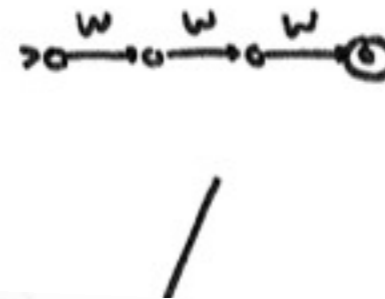
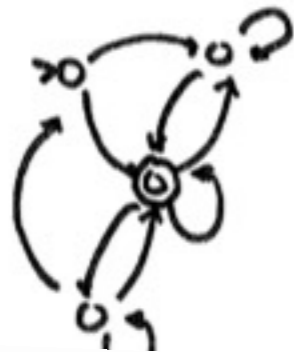
Noisy-Channel Model



Noisy-Channel Model



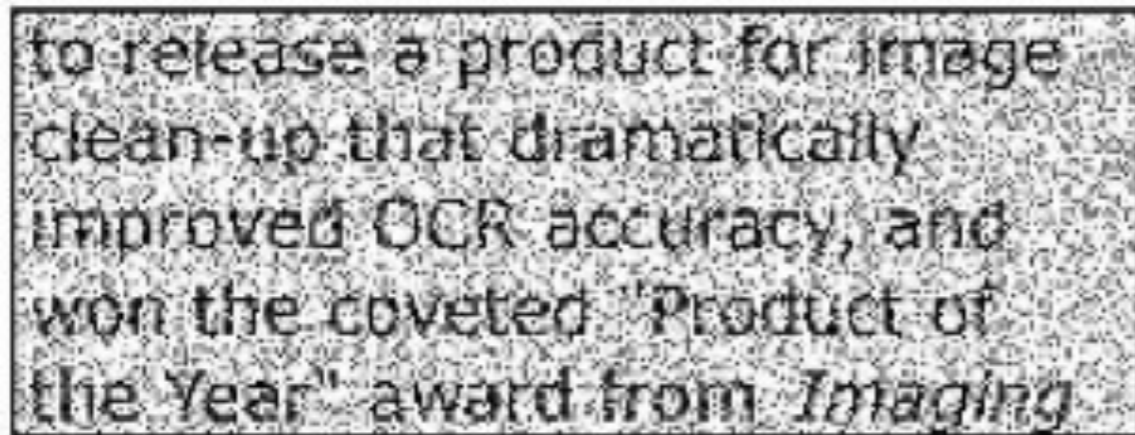
Applications of Noisy-Channel



Application	Input	Output	$p(i)$	$p(o i)$
Machine Translation	L_1 word sequences	L_2 word sequences	$p(L_1)$ in a language model	translation model
Optical Character Recognition (OCR)	actual text	text with mistakes	prob of language text	model of OCR errors
Part Of Speech (POS) tagging	POS tag sequences	English words	prob of POS sequences	$p(w t)$
Speech recognition	word sequences	speech signal	prob of word sequences	acoustic model

spelling correction correct text text with mistakes prob. of language text noisy spelling

Noisy Channel Examples



Th qck brwn fx jmps vr th lzy dg.
Ths sntnc hs ll twnty sx ltrrs n th lphbt.

I cnduo't bvliee taht I culod aulacly
uesdtannrd waht I was rdnaieg. Unisg the
icndeblire pweor of the hmuan mnid, aocdcnig
to rsecrah at Cmabrigde Uinervtisy, it dseno't
mttaer in waht oderr the lterets in a wrod are,
the olny irpoamtnt tihng is taht the frsit and lsat
ltteer be in the rhgit pclae.

Therestcanbeatotalmessandyocanstillreaditwi
thoutaproblem.Thisisbecausethehumanminddo
esnotreadeveryletterbyitself,butthewordasawh
ole.

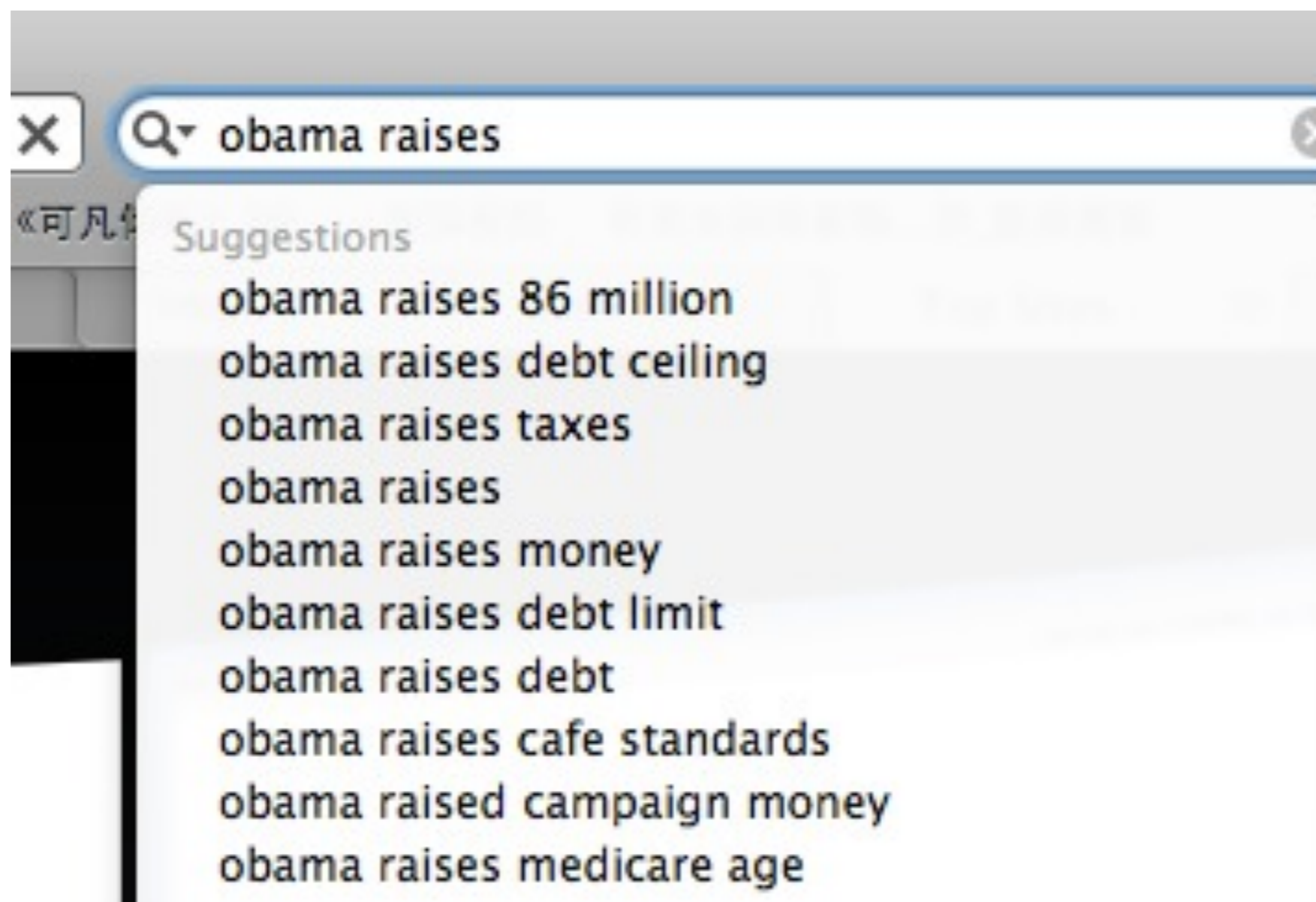


Noisy Channel Examples



Language Model for Generation

- search suggestions



Language Models

- problem: what is $P(\mathbf{w}) = P(w_1 w_2 \dots w_n)$?
- ranking: $P(\text{an apple}) > P(\text{a apple})=0$, $P(\text{he often swim})=0$
- prediction: what's the next word? use $P(w_n | w_1 \dots w_{n-1})$
- Obama gave a speech about _____.

sequence prob, not just joint prob.

- $P(w_1 w_2 \dots w_n) = P(w_1) P(w_2 | w_1) \dots P(w_n | w_1 \dots w_{n-1})$
- $\approx P(w_1) P(w_2 | w_1) P(w_3 | w_1 w_2) \dots P(w_n | w_{n-2} w_{n-1})$ trigram
- $\approx P(w_1) P(w_2 | w_1) P(w_3 | w_2) \dots P(w_n | w_{n-1})$ bigram
- $\approx P(w_1) P(w_2) P(w_3) \dots P(w_n)$ unigram
- $\approx P(w) P(w) P(w) \dots P(w)$ 0-gram

Estimating n -gram Models

"In person she was inferior superior to both sisters"

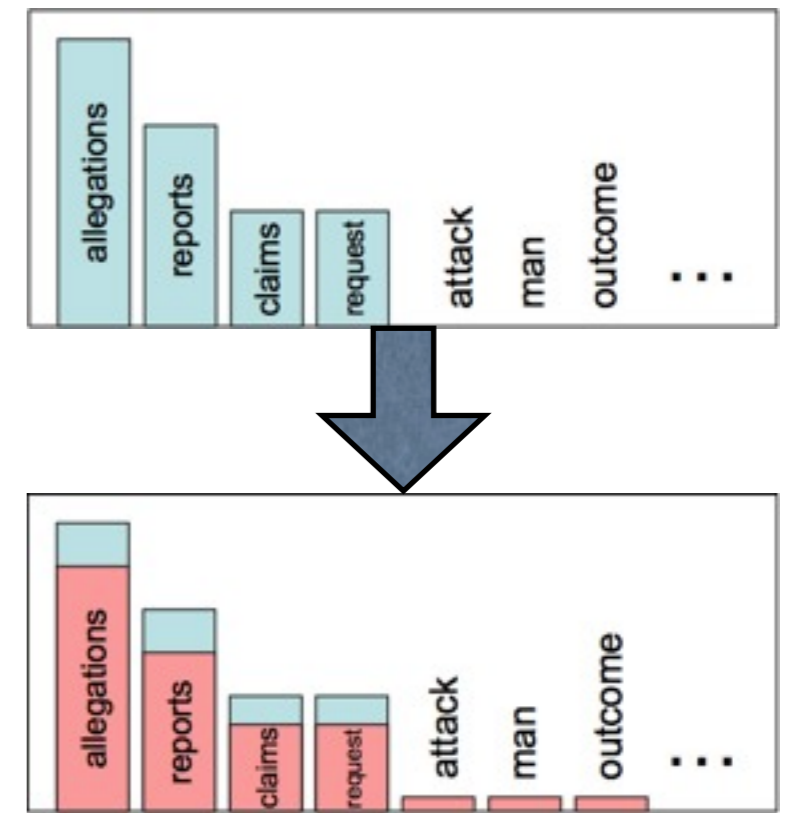
	In	person	she	was	inferior superior	to	both	sisters	
0-gram	10^{-6}	10^{-6}	10^{-6}	10^{-6}	10^{-6}	10^{-6}	10^{-6}	10^{-6}	$\approx 10^{-36}$
unigram	.011	.015	.015	.00005	.032	.0005	.0003		$= 4 \times 10^{-17}$
bigram	.009	.122	.122	0	.212	.0004	.006		$=$
trigram	?	.5	.5	0	?	0	0		$=$
4-gram	?	?	?	0	?	?	?		$=$

(textbook, table 6.3)

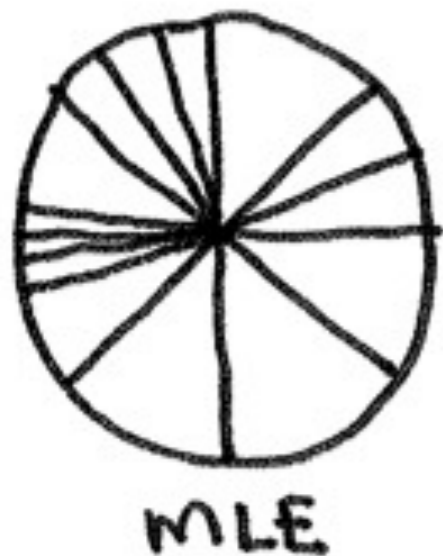
- maximum likelihood: $p_{ML}(x) = c(x)/N$; $p_{ML}(xy) = c(xy)/c(x)$
- problem: unknown words/sequences (unobserved events)
- sparse data problem
- solution: smoothing

Smoothing

- have to give some probability mass to unseen events
 - (by discounting from seen events)
- Q1: how to divide this wedge up?
- Q2: how to squeeze it into the pie?



(D. Klein)



new wedge (one tiny slice for each character sequence of length ≤ 20 that was never observed in training data)

ML, MAP, and smoothing

- simple question: what's $P(H)$ if you see H H H H?
- always maximize **posterior**: what's the best m given d ?
- with uniform **prior**, same as **likelihood (explains data)**
 - $\operatorname{argmax}_m p(m|d) = \operatorname{argmax}_m p(m) p(d|m)$ bayes, and $p(d)=1$
 - $= \operatorname{argmax}_m p(d|m)$ when $p(m)$ uniform

Suppose $d = H H T H$

m_1 coin is unbiased

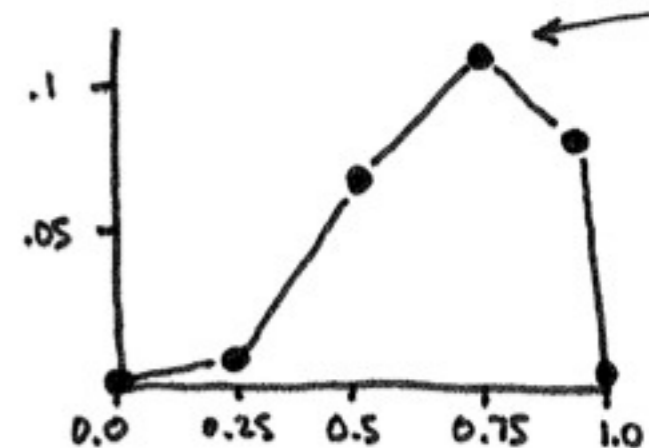
$$P(d|m) = \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} = 0.0625$$

m_2 coin is biased
so that $P(H) = 3/4$

$$P(d|m) = \frac{3}{4} \cdot \frac{3}{4} \cdot \frac{1}{4} \cdot \frac{3}{4} = 0.105$$

m_3 coin is biased
so that $P(H) = 9/10$

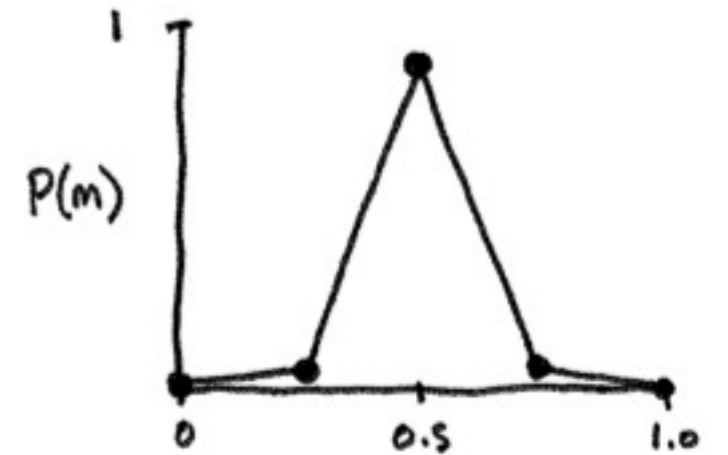
$$P(d|m) = \frac{9}{10} \cdot \frac{9}{10} \cdot \frac{1}{10} \cdot \frac{9}{10} = 0.073$$



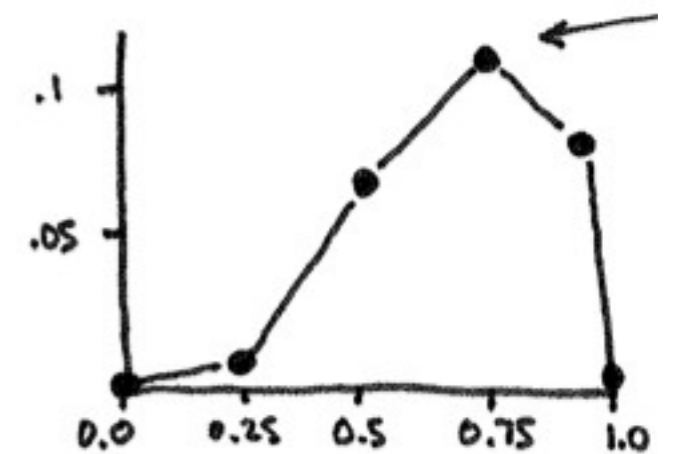
ML, MAP, and smoothing

m_1	coin is unbiased	$P(m) = 0.90$	} just "made up"!
m_2	coin is biased $3/4$	$P(m) = 0.01$	
m_3	coin is biased $1/4$	$P(m) = 0.01$	

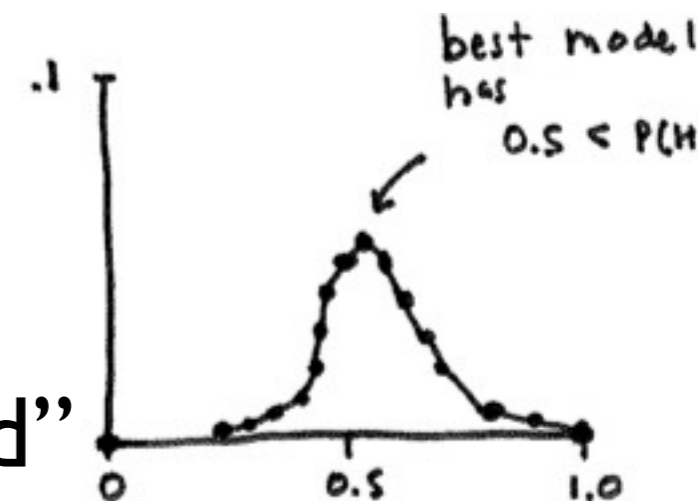
- what if we have arbitrary prior
 - like $p(\theta) = \theta (1-\theta)$
- maximum a posteriori estimation (MAP)
- MAP approaches MLE with infinite
- MAP = MLE + smoothing
 - this prior is just "extra two tosses, unbiased"
 - you can inject other priors, like "extra 4 tosses, 3 Hs"



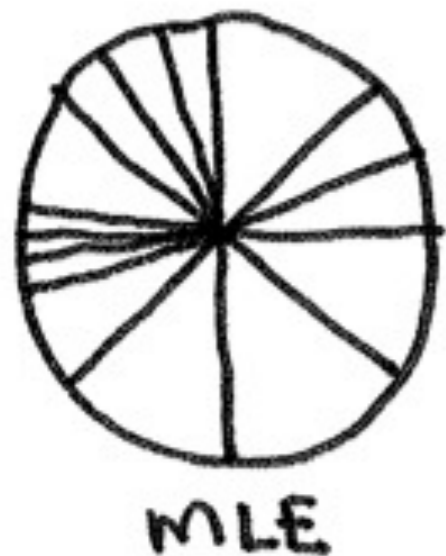
$P(d|m)$ prob



$P(m) \cdot P(d|m)$



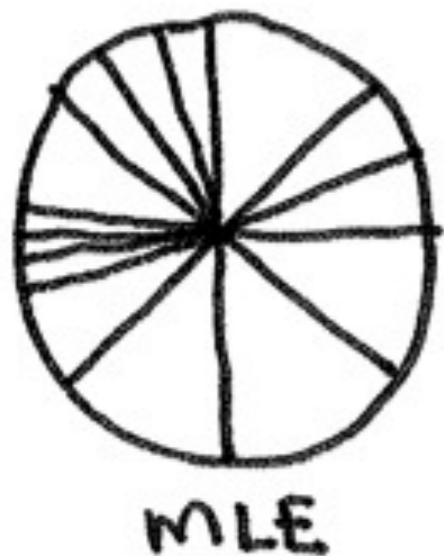
Smoothing: Add One (Laplace)



new wedge (one tiny slice for each character sequence of length < 20 that was never observed in training data)

- MAP: add a “pseudocount” of 1 to every word in Vocab
- $P_{\text{lap}}(x) = (c(x) + 1) / (N + V)$ V is Vocabulary size
- $P_{\text{lap}}(\text{unk}) = 1 / (N+V)$ same probability for all unks
- how much prob. mass for unks in the above diagram?
 - e.g., $N=10^6$ words, $V=26^{20}$, $V_{\text{obs}} = 10^5$, $V_{\text{unk}} = 26^{20} - 10^5$

Smoothing: Add Less than One



new wedge (one tiny slice for each character sequence of length ≤ 20 that was never observed in training data)

- add one gives too much weight on unseen words!
- solution: add less than one (Lidstone) to each word in V
- $P_{\text{lid}}(x) = (c(x) + \lambda) / (N + \lambda V)$ $0 < \lambda < 1$ is a parameter
- $P_{\text{lid}}(\text{unk}) = \lambda / (N + \lambda V)$ still same for unks, but smaller
- Q: how to tune this λ ? on held-out data!

Smoothing: Witten-Bell

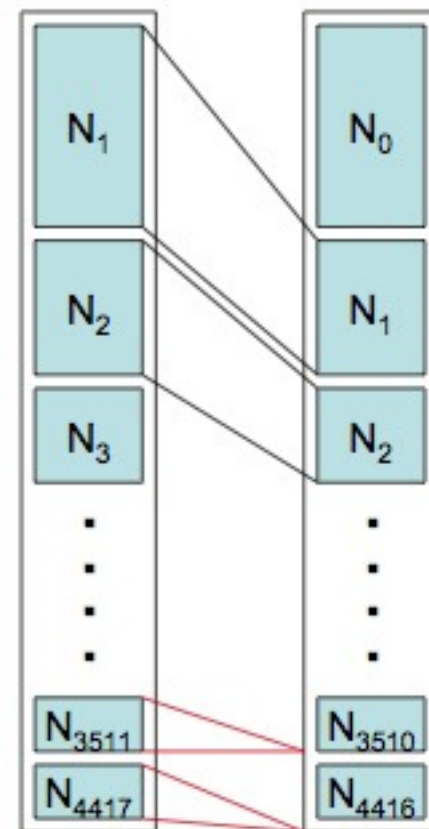
- key idea: use one-count things to guess for zero-counts
 - recurring idea for unknown events, also for Good-Turing
- prob. mass for unseen: $T / (N + T)$ T : # of seen types
 - 2 kinds of events: one for each token, one for each type
 - = MLE of seeing a new type (T among $N+T$ are new
 - divide this mass evenly among $V-T$ unknown words
- $p_{wb}(x) = T / (V-T)(N+T)$ unknown word
 $= c(x) / (N+T)$ known word
- bigram case more involved; see J&M Chapter for details

Smoothing: Good-Turing

- again, one-count words in training ~ unseen in test
- let $N_c = \#$ of words with frequency r in training
- $P_{GT}(x) = c'(x) / N$ where $c'(x) = (c(x)+1) N_{c(x)+1} / N_{c(x)}$
- total adjusted mass is $\sum_c c' N_c = \sum_c (c+1) N_{c+1} / N$
- remaining mass: N_1 / N : split evenly among unks

EXAMPLE:

r	N_r	N_{r+1}	r^*	r^*/N
0	1000	100	-	$1-z$
1	100	40	0.8	Sums to z
2	40	20	1.5	
3	20	10	2.0	
4	10	6	3.0	
5	6	3	3.0	
⋮	⋮	⋮	⋮	⋮



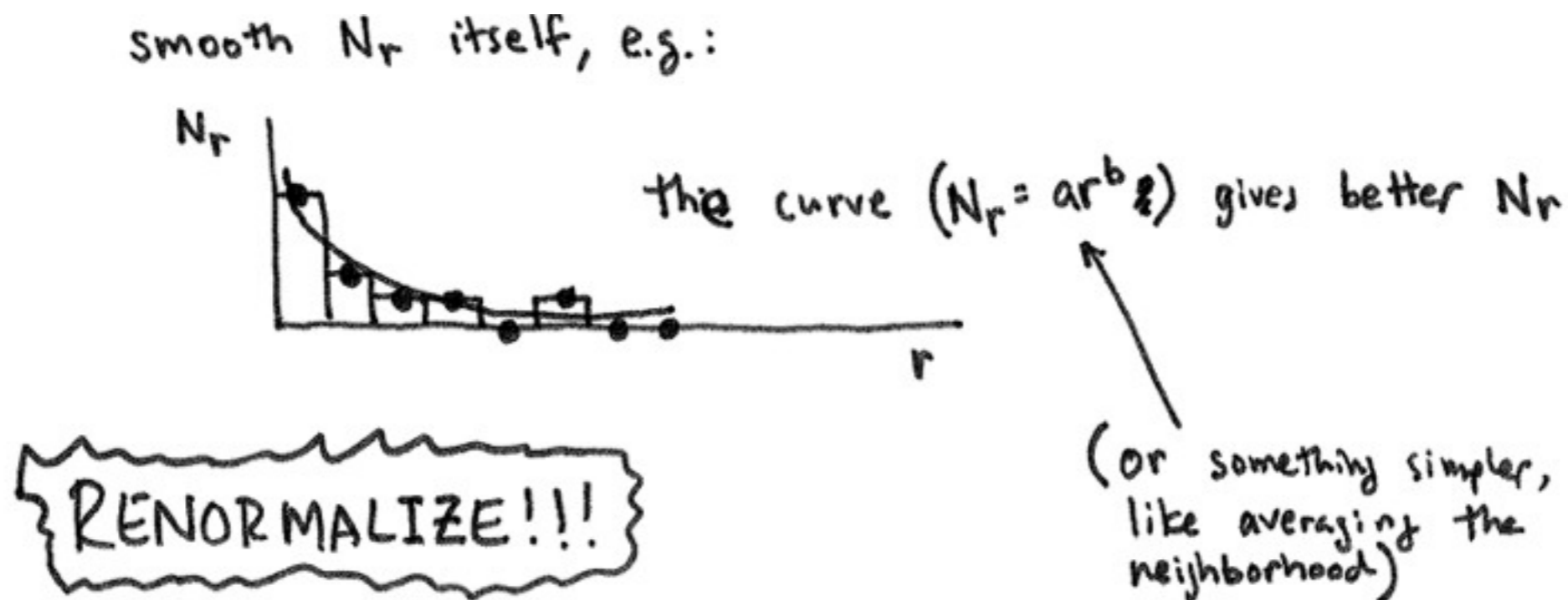
Smoothing: Good-Turing

- from Church and Gale (1991).
bigram LMs. unigram vocab size = 4×10^5 .
 T_r is the frequencies in the held-out data (see $f_{\text{empirical}}$).

$r = f_{\text{MLE}}$	$f_{\text{empirical}}$	f_{Lap}	f_{del}	f_{GT}	N_r	T_r
0	0.000027	0.000137	0.000037	0.000027	74 671 100 000	2 019 187
1	0.448	0.000274	0.396	0.446	2 018 046	903 206
2	1.25	0.000411	1.24	1.26	449 721	564 153
3	2.24	0.000548	2.23	2.24	188 933	424 015
4	3.23	0.000685	3.22	3.24	105 668	341 099
5	4.21	0.000822	4.22	4.22	68 379	287 776
6	5.23	0.000959	5.20	5.19	48 190	251 951
7	6.21	0.00109	6.21	6.21	35 709	221 693
8	7.21	0.00123	7.18	7.24	27 710	199 779
9	8.26	0.00137	8.18	8.25	22 280	183 971

Smoothing: Good-Turing

- Good-Turing is much better than add (less than) one
- problem 1: $N_{c_{\max}+1} = 0$, so $c'_{\max} = 0$
 - solution: only adjust counts for those less than k (e.g., 5)
- problem 2: what if $N_c = 0$ for some middle c ?
 - solution: smooth N_c itself



Smoothing: Backoff

$$\hat{p}(w_i | w_{i-2} w_{i-1}) = \begin{cases} \tilde{p}(w_i | w_{i-2} w_{i-1}), & \text{if } C(w_{i-2} w_{i-1} w_i) > 0 \\ \alpha_1 p(w_i | w_{i-1}), & \text{if } C(w_{i-2} w_{i-1} w_i) = 0 \\ & \text{and } C(w_{i-1} w_i) > 0 \\ \alpha_2 p(w_i), & \text{otherwise.} \end{cases}$$

Smoothing: Interpolation

$$\begin{aligned}\hat{p}(w_i|w_{i-2}w_{i-1}) &= \lambda_1 p(w_i|w_{i-2}w_{i-1}) \\ &\quad + \lambda_2 p(w_i|w_{i-1}) \\ &\quad + \lambda_3 p(w_i)\end{aligned}$$

subject to the constraint that $\sum_j \lambda_j = 1$

Entropy and Perplexity

- classical entropy: $H(X) = - \sum p(x) \log p(x)$
- $H(L) = \lim 1/n H(w_1 \dots w_n)$
- $= \lim 1/n \sum_{\{w \text{ in } L\}} p(w_1 \dots w_n) \log p(w_1 \dots w_n)$
- Shannon-McMillan-Breiman theorem:
- $H(L) = \lim -1/n \log p(w_1 \dots w_n)$
- perplexity is $2^{\{H(L)\}}$