CS474 Natural Language Processing

- Smoothing
  - Add-one
  - Discounting
- Combining estimators
  - Linear interpolation
  - Backoff
- Training issues

Language models: n-grams

- *I'd like to make a collect* ____
- *to make a collect* ***call***
- *make a collect* ***call***
- *a collect* ***call***

- Markov assumption: only the prior local context --- the last few words --- matters

$$P(w_n \mid w_1^{n-1}) \approx P(w_n \mid w_{n-N+1}^{n-1})$$

Training N-gram models

- N-gram models can be trained by counting and normalizing
  - $\mathrm{P}(call \mid a\ collect) = \dfrac{\mathrm{Count}(a\ collect\ call)}{\mathrm{Count}(a\ collect)}$
  - MLE estimates from relative frequencies
  - Bigram model

$$P(w_n \mid w_1^{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})}$$

Sparse data

- Problem with the maximum likelihood estimate: *sparse data*

ATIS corpus (~500 sentences, ~400 words):

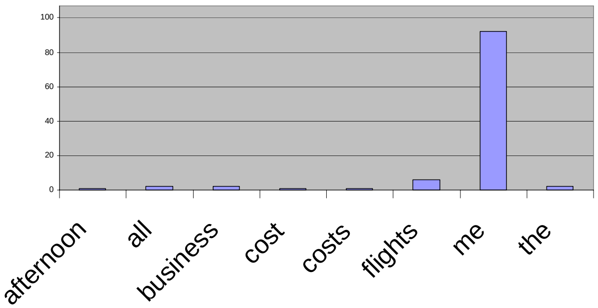
Show me flights from Boston to Chicago

I need to return on Tuesday

I would like to travel to Westchester

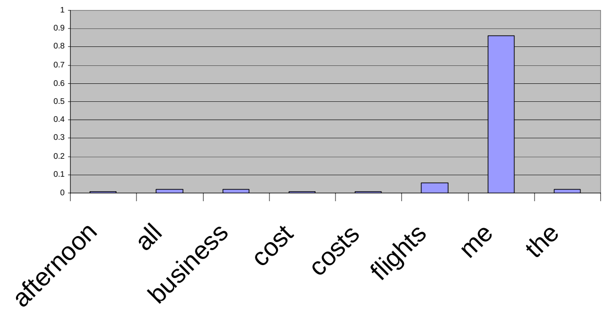
Sparse data (counts)

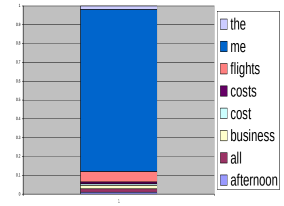
	Show	me	flights	from	Boston	to	Chicago
Show	**0**	**92**	**6**	**0**	**0**	**0**	**0**
me	0	0	14	0	0	0	0
flights	0	0	0	96	0	0	0
from	0	0	0	0	4	0	4
Boston	0	0	0	0	0	4	0
to	0	0	0	0	3	2	3
Chicago	0	0	0	0	0	4	0



Sparse data (probabilities)

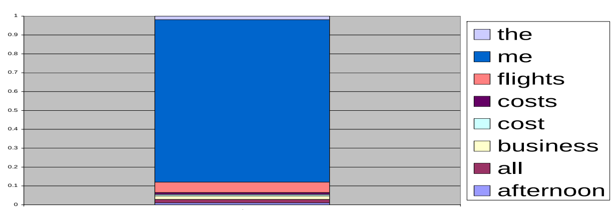
	Show	me	flights	from	Boston	to	Chicago
Show	**0**	**0.8598**	**0.0561**	**0**	**0**	**0**	**0**
me	0	0	0.1414	0	0	0	0
flights	0	0	0	0.4615	0	0	0
from	0	0	0	0	0.0148	0	0.0148
Boston	0	0	0	0	0	0.5714	0
to	0	0	0	0	0.0099	0.0066	0.0099
Chicago	0	0	0	0	0	0.4	0





Smoothing

- Need better estimators for rare events
- Approach
  - Somewhat decrease the probability of previously seen events, so that there is a little bit of probability mass left over for previously unseen events



Add-one smoothing

- Add one to all of the counts before normalizing into probabilities
- Normal unigram probabilities

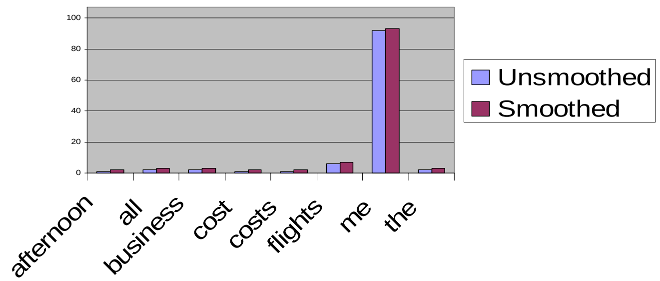
$$P(w_x) = \frac{C(w_x)}{N}$$

- Smoothed unigram probabilities

$$P(w_x) = \frac{C(w_x)+1}{N+V}$$

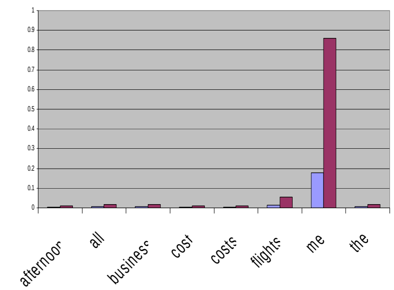
+1 Smoothed ATIS (counts)

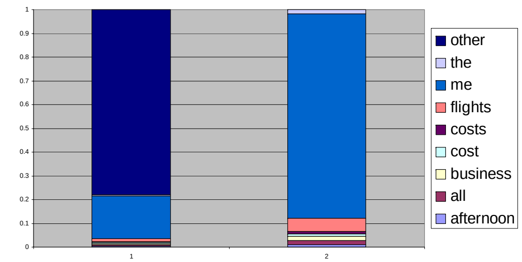
	Show	me	flights	from	Boston	to	Chicago
Show	**1**	**93**	**7**	**1**	**1**	**1**	**1**
me	1	1	15	1	1	1	1
flights	1	1	1	97	1	1	1
from	1	1	1	1	5	1	5
Boston	1	1	1	1	1	5	1
to	1	1	1	1	4	3	4
Chicago	1	1	1	1	1	5	1



+1 Smoothed ATIS (probabilities)

	Show	me	flights	from	Boston	to	Chicago
Show	**0.0019**	**0.1782**	**0.0134**	**0.0019**	**0.0019**	**0.0019**	**0.0019**
me	0.0019	0.0019	0.0292	0.0019	0.0019	0.0019	0.0019
flights	0.0016	0.0016	0.0016	0.1557	0.0016	0.0016	0.0016
from	0.0015	0.0015	0.0015	0.0015	0.0073	0.0015	0.0073
Boston	0.0024	0.0024	0.0024	0.0024	0.0024	0.0118	0.0024
to	0.0014	0.0014	0.0014	0.0014	0.0056	0.0042	0.0056
Chicago	0.0024	0.0024	0.0024	0.0024	0.0024	0.0118	0.0024





Too much probability mass is moved

- Estimated bigram frequencies
- AP data, 44million words
- Church and Gale (1991)
- In general, add-one smoothing is a poor method of smoothing
- Much worse than other methods in predicting the actual probability for unseen bigrams
- Variances of the counts are worse than those from the unsmoothed MLE method

$r = f_{MLE}$	f_{emp}	$f_{add\text{-}1}$
0	0.000027	0.000137
1	0.448	0.000274
2	1.25	0.000411
3	2.24	0.000548
4	3.23	0.000685
5	4.21	0.000822
6	5.23	0.000959
7	6.21	0.00109
8	7.21	0.00123
9	8.26	0.00137

Aside: Methodology

- Cardinal sin: Testing on the training corpus
- Divide data into training set and test set
  - Train the statistical parameters on the training set; use them to compute probabilities on the test set
  - Test set: 5-10% of the total data, but large enough for reliable results
- Divide training into training and validation/held out set
  - » Obtain counts from training
  - » Tune smoothing parameters on the validation set
- Divide test set into development and final test set
  - Do all algorithm development by testing on the dev set, save the final test set for the very end…

Back to smoothing: solutions

- Discounting
  - Better estimates for how much probability to siphon away for unseen words
  - Use higher frequency words to estimate mass of lower frequency words
  - See book: Witten-Bell, Good-Turing (& many more)
- Combining estimators...

Combining estimators

- Discounting methods
  - Provide the same estimate for all unseen (or rare) n-grams
  - Make use only of the raw frequency of an n-gram
- But there is an additional source of knowledge we can draw on --- the n-gram "hierarchy"
  - If I haven't seen *a collect call*, maybe I've seen *collect call*
  - ... *collect call* *call*
- For n-gram models, suitably combining various models of different orders is the secret to success.

Simple linear interpolation

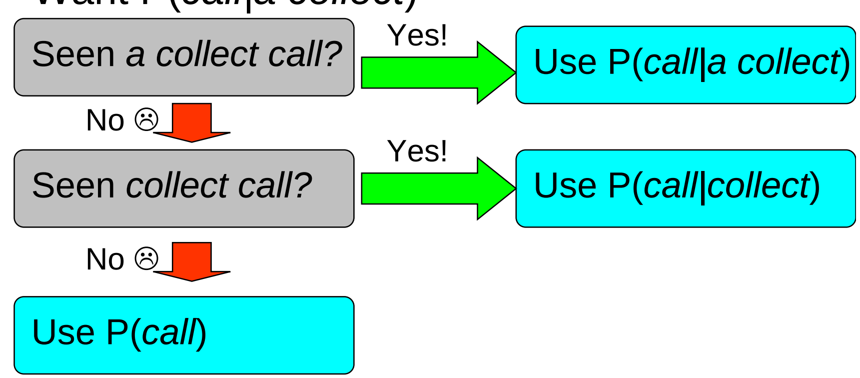
- Construct a linear combination of the multiple probability estimates.
  - Weight each contribution so that the result is another probability function.

$$P(w_n \mid w_{n-1}, w_{n-2}) = \lambda_3 P(w_n \mid w_{n-1} w_{n-2}) + \lambda_2 P(w_n \mid w_{n-1}) + \lambda_1 P(w_n)$$

  - λs sum to 1.
  - λs trained on validation set

Backoff (Katz 1987)

- *(this is a lie)*
- Want P(*call*|*a collect*)



Backoff: details

- Ps need to sum to 1!

- Discount each MLE prob (W-B, G-T, ...)
- Apportion the saved mass to lower-orders

$$\hat{P}(w_n \mid w_{n-N+1}^{n-1}) = \tilde{P}(w_n \mid w_{n-N+1}^{n-1}) +$$

$$\theta\,(P(w_n \mid w_{n-N+1}^{n-1}))\,\alpha(w_{n-N+1}^{n-1})\,\hat{P}(w_n \mid w_{n-N+2}^{n-1})$$