Smoothing

BM1: Advanced Natural Language Processing

University of Potsdam

Tatjana Scheffler

tatjana.scheffler@uni-potsdam.de

November 1, 2016

Last Week

- Language model: $P(X_t = w_t \mid X_1 = w_1, \ldots, X_{t-1} = w_{t-1})$
- Probability of string $w_1 \ldots w_n$ with bigram model:
  $P(w_1 \ldots w_n) = P(w_1)P(w_2 | w_1) \ldots P(w_n | w_{n-1})$
- Maximum likelihood estimation using relative frequencies:

$$P(w_t \mid w_1, \ldots, w_{t-1}) = \frac{C(w_1 \ldots w_{t-1} w_t)}{C(w_1 \ldots w_{t-1})}$$

low n ⟷ high n

modeling errors ⟷ estimation errors

Today

- More about dealing with sparse data
- Smoothing
- Good-Turing estimation
- Linear interpolation
- Backoff models

An example

JOHN READ MOBY DICK
MARY READ A DIFFERENT BOOK
SHE READ A BOOK BY CHER

$$
\begin{aligned}
& p(\text{JOHN READ A BOOK}) \\
= \; & p(\text{JOHN}|\bullet) \; p(\text{READ}|\text{JOHN}) \; p(\text{A}|\text{READ}) \; p(\text{BOOK}|\text{A}) \; p(\bullet|\text{BOOK}) \\
= \; & \frac{c(\bullet\ \text{JOHN})}{\sum_w c(\bullet\ w)} \; \frac{c(\text{JOHN READ})}{\sum_w c(\text{JOHN}\ w)} \; \frac{c(\text{READ A})}{\sum_w c(\text{READ}\ w)} \; \frac{c(\text{A BOOK})}{\sum_w c(\text{A}\ w)} \; \frac{c(\text{BOOK}\ \bullet)}{\sum_w c(\text{BOOK}\ w)} \\
= \; & \frac{1}{3} \; \frac{1}{1} \; \frac{2}{3} \; \frac{1}{2} \; \frac{1}{2} \\
\approx \; & 0.06
\end{aligned}
$$

(Chen/Goodman, 1998)

An example

JOHN READ MOBY DICK
MARY READ A DIFFERENT BOOK
SHE READ A BOOK BY CHER

$$
\begin{aligned}
& p(\text{CHER READ A BOOK}) \\
&= p(\text{CHER}|\bullet) \quad p(\text{READ}|\text{CHER}) \quad p(\text{A}|\text{READ}) \quad p(\text{BOOK}|\text{A}) \quad p(\bullet|\text{BOOK}) \\
&= \frac{c(\bullet\ \text{CHER})}{\sum_w c(\bullet\ w)} \quad \frac{c(\text{CHER READ})}{\sum_w c(\text{CHER}\ w)} \quad \frac{c(\text{READ A})}{\sum_w c(\text{READ}\ w)} \quad \frac{c(\text{A BOOK})}{\sum_w c(\text{A}\ w)} \quad \frac{c(\text{BOOK}\ \bullet)}{\sum_w c(\text{BOOK}\ w)} \\
&= \frac{0}{3} \quad \frac{0}{1} \quad \frac{2}{3} \quad \frac{1}{2} \quad \frac{1}{2} \\
&= 0
\end{aligned}
$$

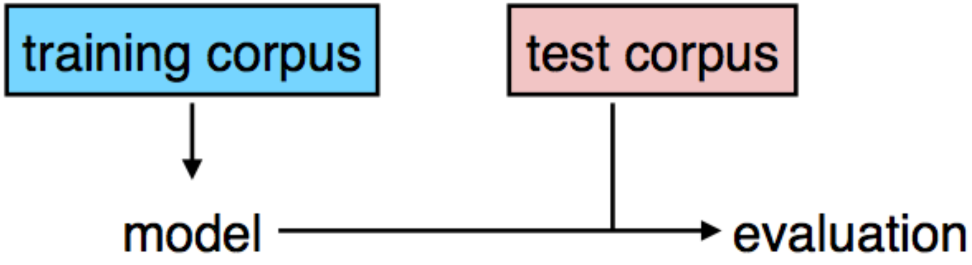
(Chen/Goodman, 1998)

Unseen data

- ML estimate is “optimal” only for the corpus from which we computed it.
- Usually does not generalize directly to new data.
- Ok for unigrams, but there are *so many* bigrams.
- Extreme case: $P(\text{unseen} \mid w_{k-1}) = 0$ for all w_{k-1}
- This is a disaster because product with 0 is always 0.

Honest evaluation

- To get an honest picture of a model's performance, need to try it on a separate *test corpus*.



- Maximum likelihood for training corpus is not necessarily good for the test corpus.
  - In Cher corpus, likelihood L(test) = 0.

Measures of quality

- (Cross) Entropy: Average number of bits per word in corpus T in an optimal compression scheme:

$$H_p(T) = -\frac{1}{N}\log_2 p(T)$$

- Good language model should minimize entropy of observations.
- Equivalently, represent in terms of perplexity:

$$\mathrm{PP}_p(T) = 2^{H_p(T)}$$

Smoothing techniques

- Replace ML estimate

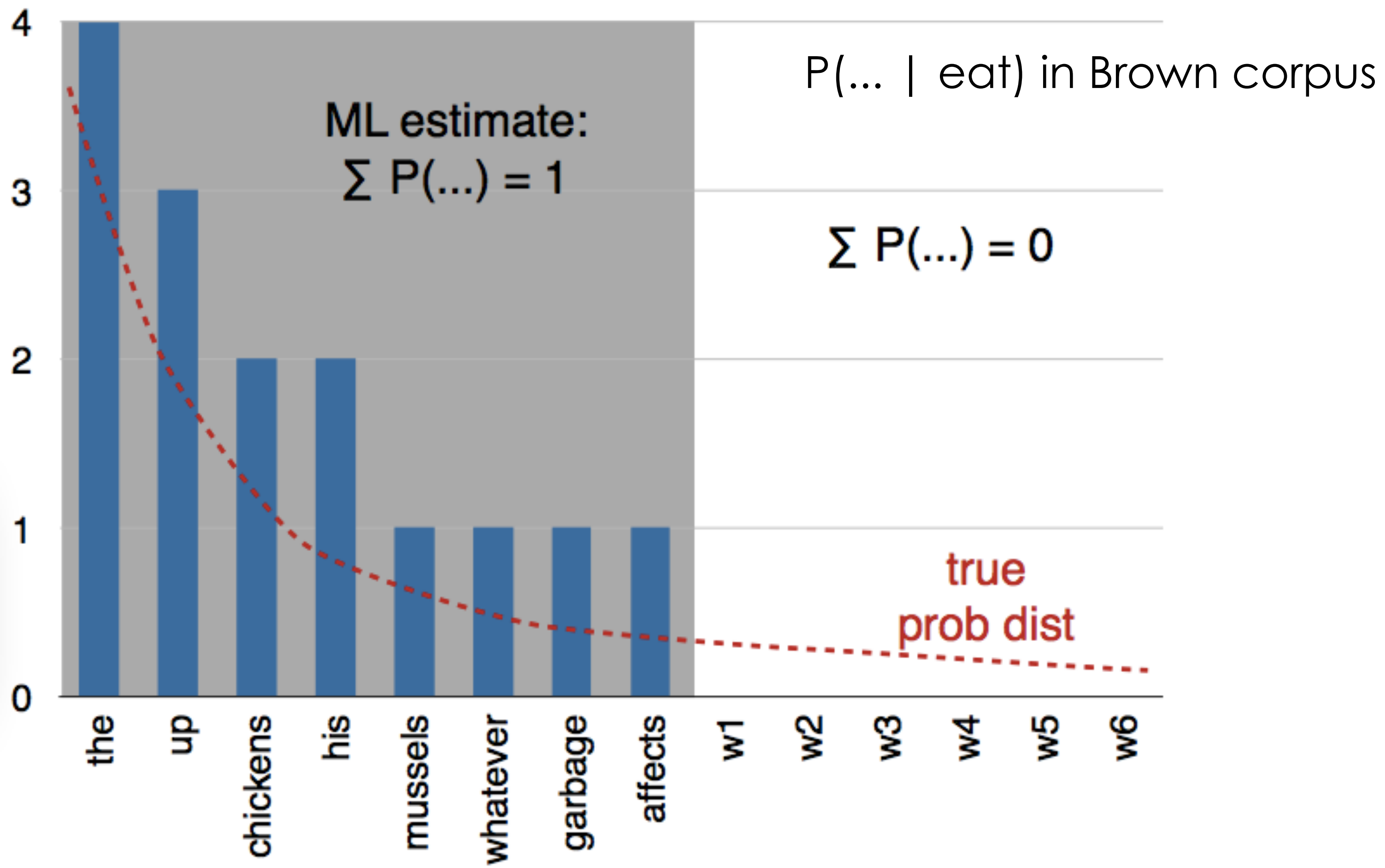
$$P_{\mathrm{ML}}(w_i \mid w_{i-1}) = \frac{C(w_{i-1}w_i)}{C(w_{i-1})}$$

- by an adjusted bigram count

$$P^*(w_i \mid w_{i-1}) = \frac{C^*(w_{i-1}w_i)}{C(w_{i-1})}$$

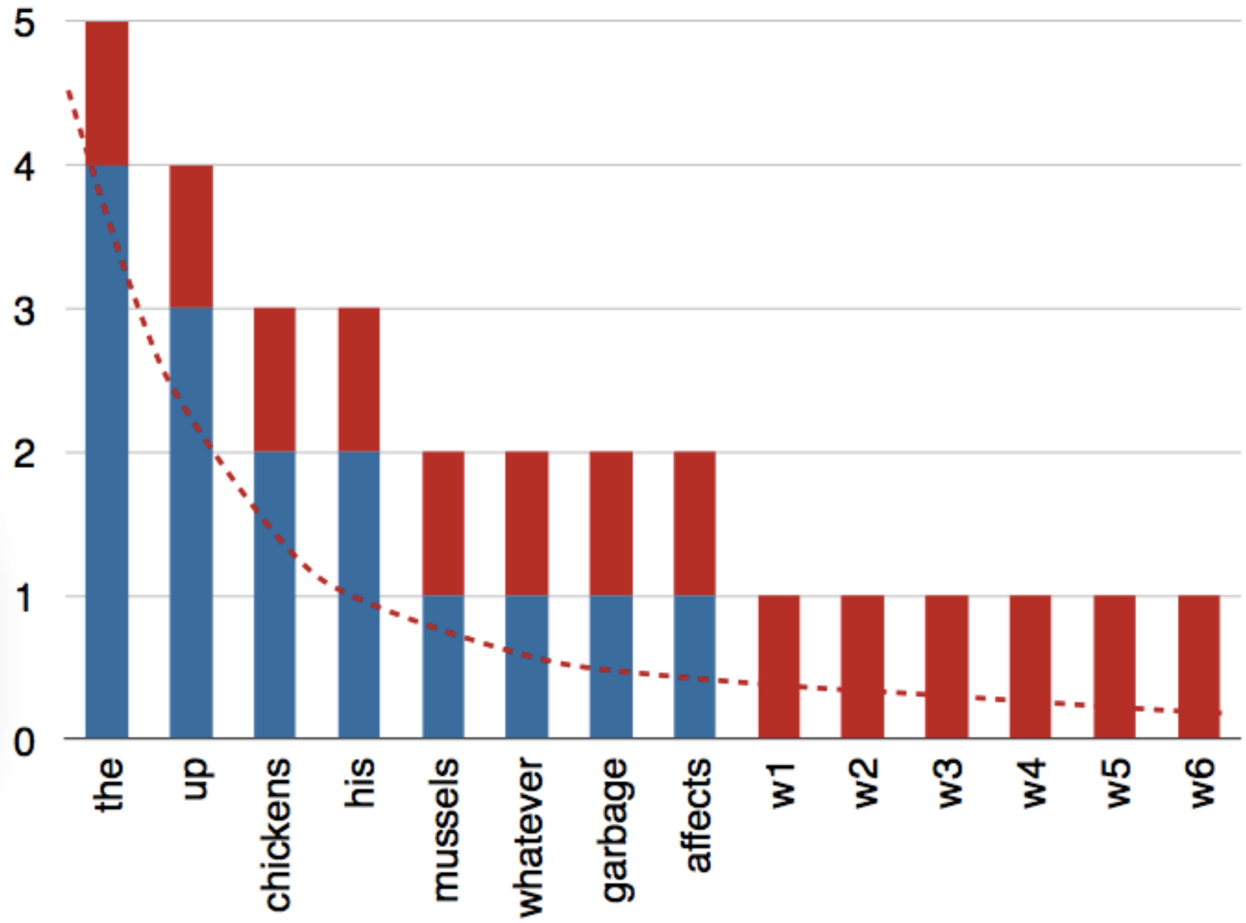
- Redistribute counts from seen to unseen bigrams.
- Generalizes easily to n-gram models with n > 2.

Smoothing

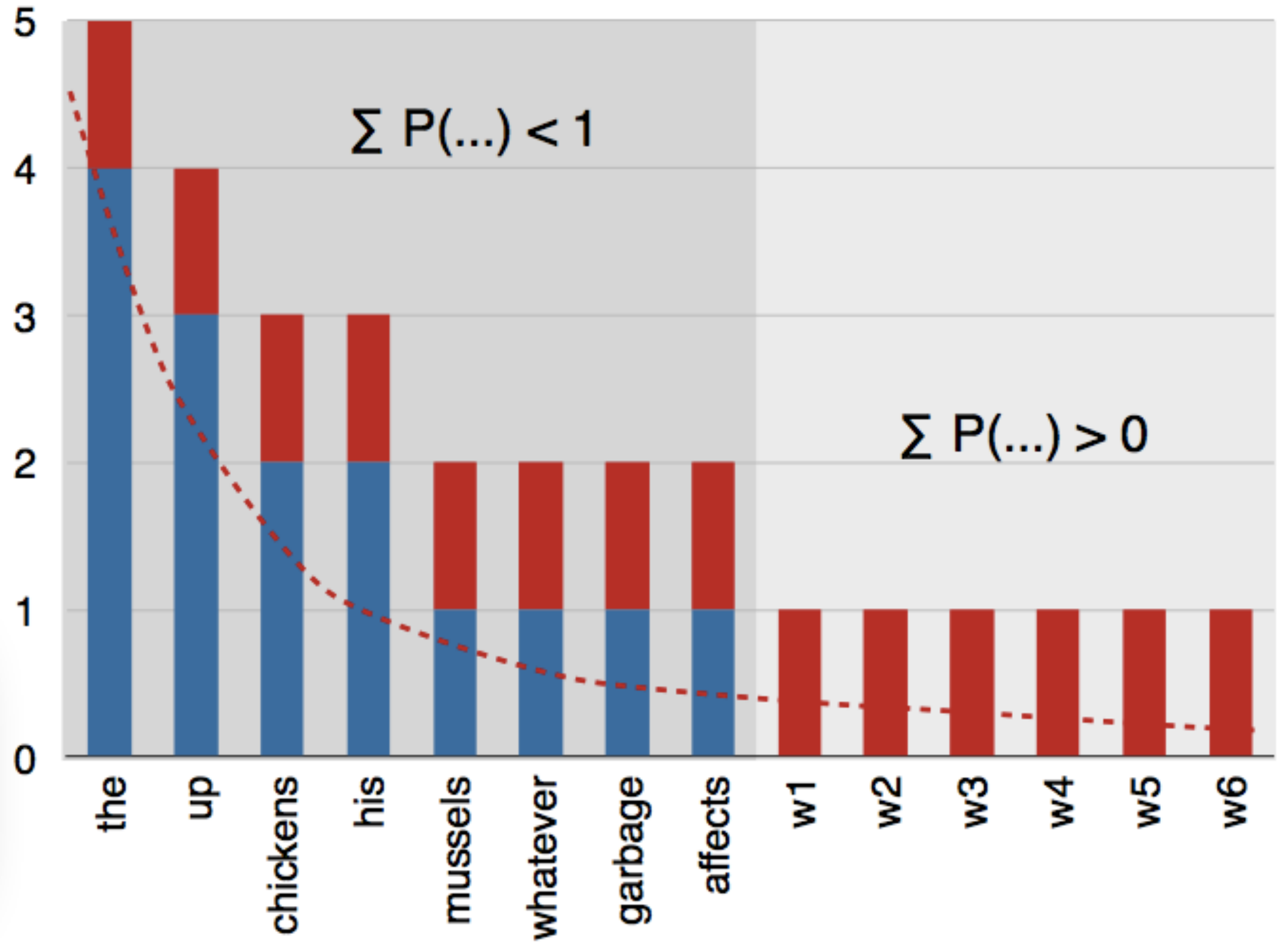


4
3
2
1
0
ML estimate:
∑ P(...) = 1
P(... | eat) in Brown corpus
∑ P(...) = 0
true
prob dist
the
up
chickens
his
mussels
whatever
garbage
affects
w1
w2
w3
w4
w5
w6

Laplace Smoothing



Laplace Smoothing



Laplace Smoothing

- Count every bigram (seen or unseen) one more time than in corpus and normalize:

$$P_{\text{lap}}(w_i \mid w_{i-1}) = \frac{C(w_{i-1}w_i) + 1}{\sum_w (C(w_{i-1}w) + 1)} = \frac{C(w_{i-1}w_i) + 1}{C(w_{i-1}) + |V|}$$

- Easy to implement, but dramatically overestimates probability of unseen events.
- Quick fix: Additive smoothing with some $0 < \delta \leq 1$.

$$P_{\text{add}}(w_i \mid w_{i-1}) = \frac{C(w_{i-1}w_i) + \delta}{C(w_{i-1}) + \delta|V|}$$

Cher example

- $|V| = 11$,
  $|\text{seen bigram types}| = 11$
  $\Rightarrow$ 110 unseen bigrams
- $P_{lap}(\text{unseen} \mid w_{i-1}) \geq 1/14$; thus "count"($w_{i-1}$ unseen) $\approx 110 * 1/14 = 7.8$.
- Compare against 12 bigram tokens in training corpus.

JOHN READ MOBY DICK
MARY READ A DIFFERENT BOOK
SHE READ A BOOK BY CHER

$$p(\text{JOHN READ A BOOK})$$

$$= \frac{1+1}{11+3} \frac{1+1}{11+1} \frac{1+2}{11+3} \frac{1+1}{11+2} \frac{1+1}{11+2}$$

$$\approx 0.0001$$

$$p(\text{CHER READ A BOOK})$$

$$= \frac{1+0}{11+3} \frac{1+0}{11+1} \frac{1+2}{11+3} \frac{1+1}{11+2} \frac{1+1}{11+2}$$

$$\approx 0.00003$$

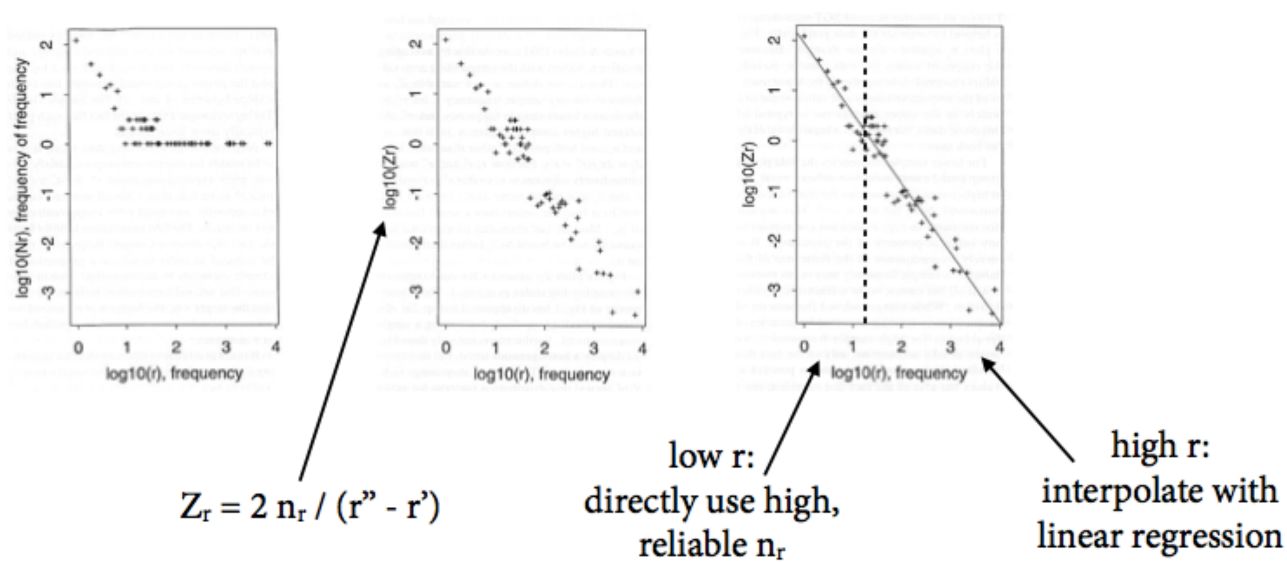
Good-Turing Estimation

- For each bigram count r in corpus, look how many bigrams had the same count:
  - "count count" n_r
- Now re-estimate bigram counts as $r^* = (r+1)\frac{n_{r+1}}{n_r}$
- One intuition:
  - 0* is now greater than zero.
  - Total sum of counts stays the same:

$$\sum_{r=0}^{\infty} n_r r^* = \sum_{r=0}^{\infty} n_r (r+1)\frac{n_{r+1}}{n_r} = \sum_{r=1}^{\infty} n_r r = N$$

Good-Turing Estimation

- Problem: n_r becomes zero for large r.
- Solution: need to smooth out n_r in some way, e.g. Simple G-T (Gale/Sampson 1995):



Good-Turing > Laplace

$r = f_{MLE}$	$f_{empirical}$	f_{Lap}	f_{del}	f_{GT}
0	0.000027	0.000137	0.000037	0.000027
1	0.448	0.000274	0.396	0.446
2	1.25	0.000411	1.24	1.26
3	2.24	0.000548	2.23	2.24
4	3.23	0.000685	3.22	3.24
5	4.21	0.000822	4.22	4.22
6	5.23	0.000959	5.20	5.19
7	6.21	0.00109	6.21	6.21
8	7.21	0.00123	7.18	7.24
9	8.26	0.00137	8.18	8.25

(Manning/Schütze after Church/Gale 1991)

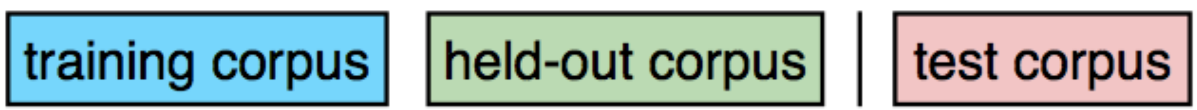
Linear Interpolation

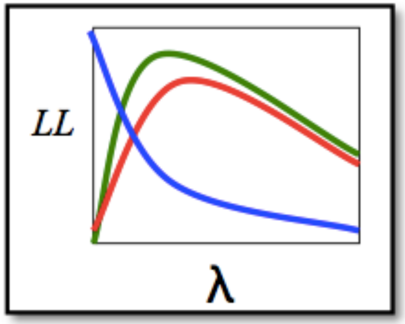
- One problem with Good-Turing:
  All unseen events are assigned the same probability.
- Idea: $P^*(w_i \mid w_{i-1})$ for unseen bigram w_{i-1} w_i should be higher if w_i is a frequent word.
- Linear interpolation: combine multiple models with a weighting factor λ.

$$P^*(w_i \mid w_{i-1}) = \lambda_{w_{i-1}w_i} \cdot P_2(w_i \mid w_{i-1}) + (1 - \lambda_{w_{i-1}w_i}) \cdot P_1(w_i)$$

Linear interpolation

- Simplest variant: λ_{wi-1wi} the same λ for all bigrams.
- Estimate from *held-out* data:





- Can also *bucket* bigrams in various ways and have one λ for each bucket, for better performance.
- Linear interpolation generalizes to higher n-grams.

(graph from Dan Klein)

Backoff models

- Katz: try fine-grained model first; if not enough data available, *back off* to lower-order model.
  - By contrast, interpolation always *mixes* different models.
- General formula (e.g., k=5):

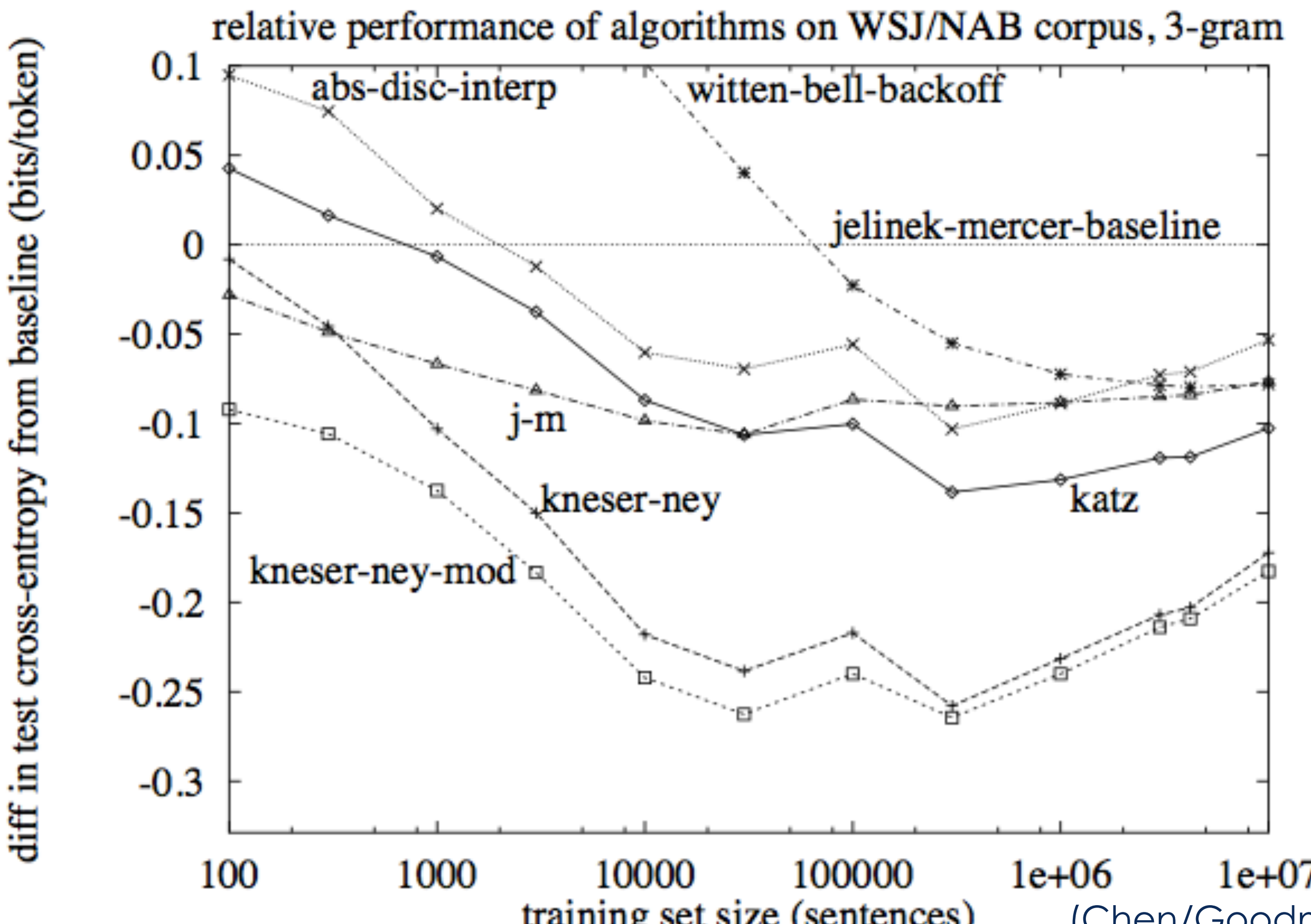
$$C_{\mathrm{katz}}(w_{i-1}w_i) = \begin{cases} d_r \cdot r & \text{if } r = C(w_{i-1}w_i) > k \\ \alpha(w_{i-1}) \cdot C(w_i) & \text{if } r \leq k \end{cases}$$

- Choose α and d appropriately to redistribute probability mass in a principled way.

Kneser-Ney smoothing

- Interpolation and backoff models that rely on unigram models can make mistakes if there was a reason why a bigram was rare:
  - "I can't see without my reading ______"
  - C_1(Francisco) > C_1(glasses), but appears only in very specific contexts (example from Jurafsky & Martin).
- Kneser-Ney smoothing: P(w) models how likely w is to occur after words that we haven't seen w with.
  - captures "specificity" of "Francisco" vs. "glasses"
  - originally formulated as backoff model, nowadays interpolation

Smoothing performance



(Chen/Goodman 1998)

Summary

- In practice (speech recognition, SMT, etc.):
  - unigram, bigram models not accurate enough
  - trigram models work much better
  - higher models only if we have lots of training data
- Smoothing is important and surprisingly effective.
  - permits use of "deeper" model with same amount of data
  - "If data sparsity is not a problem for you, your model is too simple."

Friday

- Part of Speech Tagging