ANLP

05 - n-grams (sequences, part I)

David Schlangen
University of Potsdam, MSc Cognitive Systems
Winter 2019 / 2020

Week	Date	Theme	Focus	Readings	Assignment
1	2019-10-16		Intro	E.1	
	2019-10-17		Probability	https://mml-book.github.io ; [Sharon Goldwater's tutorial]()	
2	2019-10-23	Words, Representations	words, relations	JM-3.6	A1 released
	2019-10-24		words, embeddings	JM-3.6, E.3.3.4, E.14.5-6	
3	2019-10-30	Sequences I	n-grams	JM-3.3	
	2019-10-31		/	/	/
4	2019-11-06	Tools / Framings: Classification	binary classification	E.2.0-5, E.4.2-4.4.1, JM-3.4, JM-3.5.0-6	A1 due; A2 released
	2019-11-07		multiclass classification	E.4.2, JM-3.5.6	
5	2019-11-13		discussion of A1		
	2019-11-14	Sequences II	HMMs, POS-Tagging	E.7.0-4, JM-3.8	

Week	Date	Theme	Focus	Readings	Assignment
6	2019-11-20		CRFs	E.7.5, E.8.3	A2 due; A3 released
	2019-11-21	Tools / Framings: NNs	NNs I: FF	E.3.0-3, G.1-4	
7	2019-11-27		discussion of A2		
	2019-11-28		NNs II: RNNs	G.10-11	
8	2019-12-04		NNs III: CNNs, Neural CRFs	E.3.4, E.7.6, G9	A3 due; A4 released
	2019-12-05	Structure	CFGs, CKY, PCFG	E.10.0-5, JM-3.12	
9	2019-12-11		discussion of A3		
	2019-12-12		Dependency parsing I	E.11, JM-3.13	
10	2019-12-18		Dependency parsing II		A4 due
	2019-12-19		discussion of A4		

Week	Date	Theme	Focus	Readings	Assignment
11	2020-01-08		pyTorch practical?	TBA	A5 released
	2020-01-09		pyTorch practical?	TBA	
12	2020-01-15	Semantics	Semantics I	E.12	
	2020-01-16		Semantics II, Seq2Seq		
13	2020-01-22		Seq2Seq II: Attentn & Pointers		A5 due; A6 released
	2020-01-23		discussion of A5		
14	2020-01-29	The Real World	Annotation	TBA	
	2020-01-30		Ethics of doing NLP	TBA	
15	2020-02-05		buffer		
	2020-02-06		final projects		

today

- well-formed sequences of words — a statistical approach

Let's play a game

- I will write the start of a sentence on the board.
- Each of you, in turn, gives me a word to continue that sentence, and I will write it down.

Let’s play another game

- You write a word on a piece of paper.
- You get to see the piece of paper of your neighbor, but none of the earlier words.
- In the end, I will read the sentence you wrote.

Statistical models in NLP

- Generative statistical model of language:
  pd P(w) over NL expressions that we can observe.
  - w may be complete sentences or smaller units
  - will later extend this to pd P(w, t) with hidden random variables t

- Assumption: A corpus of observed sentences w is generated by repeatedly sampling from P(w).

- We try to estimate the parameters of the prob dist from the corpus, so we can make predictions about unseen data.

Predictive text models

BOB DYLAN

BOB DYLAN NEWSLETTER

SEARCH & LYRICS

TOUR NEWS ALBUMS SONGS BOOKS ART WHISKEY STORE

Bob On Bob

WRITTEN BY: BOB DYLAN, AMAZON REVIEWS OF BOB DYLAN

The album is another instant classic from a man
Who knows how to be wandering the road
I bought it for my darling but she said it was not for her
Now i know that dylan magic just cant go on blonde
She said his voice is simply horrible
It sounds like a south-bound train
It sounds like every sparrow is blowing through his face

Hey mama is it hopeless?
Is no exception made?
Are dylan's millions of songs all written in vain?

APPEARS ON

Blonde On Blonde
(Original Release)

BUY

http://objectdreams.tumblr.com/

Predictive text models

https://www.theverge.com/2018/5/31/17405042/botnik-studios-predictive-writer-voicebox

Word-by-word random process

- A language model (LM) is a probability distribution P(w) over sentences.
- Think of it as random process that generates sentences word by word:

X_1	X_2	X_3	X_4
$\downarrow$	$\downarrow$	$\downarrow$	$\downarrow$
copper	is	a	metal

Process from our game

- Each of you = a random variable X_t;
event "$X_t = w_t$" means word at position t is w_t.

- When you chose w_t, you could see the outcomes of the previous variables: $X_1 = w_1, \ldots, X_{t-1} = w_{t-1}$.

- Thus, X_t followed a pd

$$P(X_t = w_t \mid X_1 = w_1, \ldots, X_{t-1} = w_{t-1})$$

Process from our game

- Assume that X_t follows some given PD

$$P(X_t = w_t \mid X_1 = w_1, \ldots, X_{t-1} = w_{t-1})$$

- Then probability of the entire corpus (or sentence) $w = w_1 \ldots w_n$ is joint probability

$$\begin{aligned} P(w_1 \ldots w_n) &= P(w_1) \cdot P(w_2 \mid w_1) \cdot P(w_3 \mid w_1, w_2) \\ &\quad \cdot \ldots \cdot P(w_n \mid w_1, \ldots, w_{n-1}) \end{aligned}$$

How do we estimate these?

Statistical models

- We want to use prob theory to estimate a model of a generating process from observations about its outcomes.

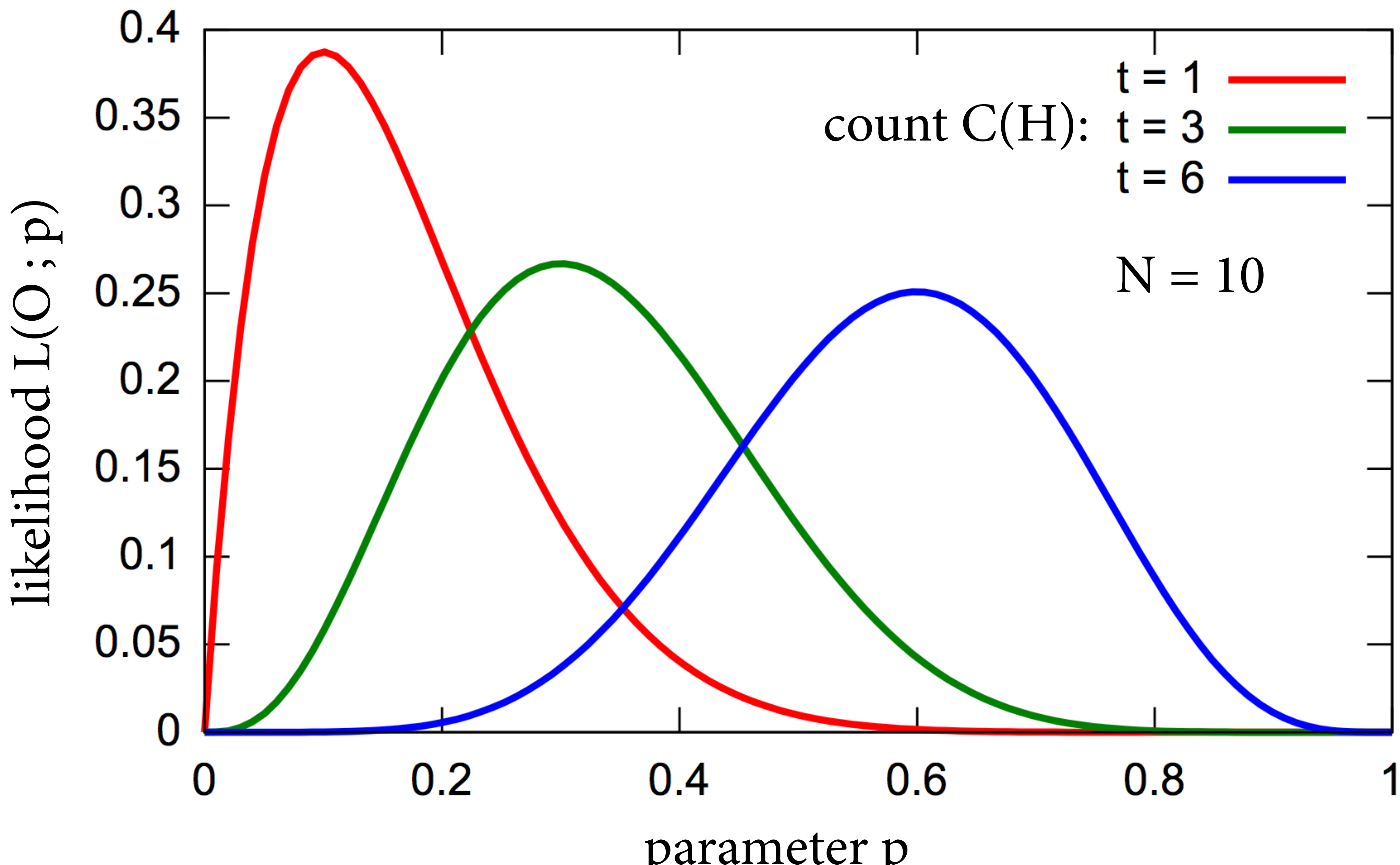
- Simpler case: we flip a coin 100 times and observe H 61 times. Should we believe that it is a fair coin?
  - observation: absolute freq C(H) = 61, C(T) = 39;
    thus relative freq f(H) = 0.61, f(T) = 0.39
  - model: assume rv X follows a Bernoulli distribution,
    i.e. X has two outcomes, and there is a value p such that
    P(X = H) = p and P(X = T) = 1 - p.
  - want to estimate the parameter p of this model

Fit of model and observations

- How do we quantify how well a model fits with the observations we made?
- Out of the many possibilities, easiest is to look at the likelihood: probability P(O ; p) of the observations O given the values p for the model parameters.
- Maximum likelihood estimation: find parameter values for which the likelihood of O is maximal.

Likelihood functions

likelihood L(O ; p) = $p^{C(H)} * (1-p)^{C(T)} *$ binom(N, C(H))



ML Estimation

- Goal: Find value for p that maximizes the likelihood of the observations.
- For Bernoulli models, it is extremely easy to estimate the parameters that maximize the likelihood:
  - $P(X = a) = f(a)$
  - in the coin example above, just take $p = f(H)$
- Can prove that relative frequency is an ML estimator for a lot of different statistical models (Bernoulli, multinomial, etc.; see link on course page).

Parameters of the model

- Our model has one parameter for $P(X_t = w_t \mid w_1, \ldots, w_{t-1})$ for each t and $w_1, \ldots, w_t$.

- Can use maximum likelihood estimation:

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

- Let's say a natural language has 10^5 different words. How many tuples $w_1, \ldots w_t$ of length t?
  - $t = 1$: 10^5
  - $t = 2$: 10^{10} different contexts
  - $t = 3$: 10^{15}; etc.

Sparse data problem

- Typical corpus sizes:
  - Brown corpus: about 10^6 tokens
  - Gigaword corpus: about 10^9 tokens
- Problem exacerbated by Zipf's Law:
  - Order all words by their absolute frequency in corpus (rank 1 = most frequent word).
  - Then log(absolute frequency) falls linearly with log(rank); i.e., most words are really rare.
  - Zipf's Law is very robust across languages and corpora.

Independence assumptions

- Let's pretend that word at position t depends only on the words at positions t-1, t-2, …, t-k for some fixed k (Markov assumption of degree k).

- Then we get an n-gram model, with n = k+1:

  $$P(X_t \mid X_1, \ldots, X_{t-1}) = P(X_t \mid X_{t-k}, \ldots, X_{t-1})$$

  for all t.

- Special names for unigram models (n = 1), bigram models (n = 2), trigram models (n = 3).
  - Thus our second game was a bigram model.

Independence assumptions

- We assume statistical independence of X_t from events that are too far in the past, although we know that this assumption is incorrect.
- Typical tradeoff in statistical NLP:
  - if model is too shallow, it won't represent important linguistic dependencies
  - if model is too complex, its parameters can't be estimated accurately from the available data

low n ⟷ high n

modeling errors ⟷ estimation errors

Bigrams: an example

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

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

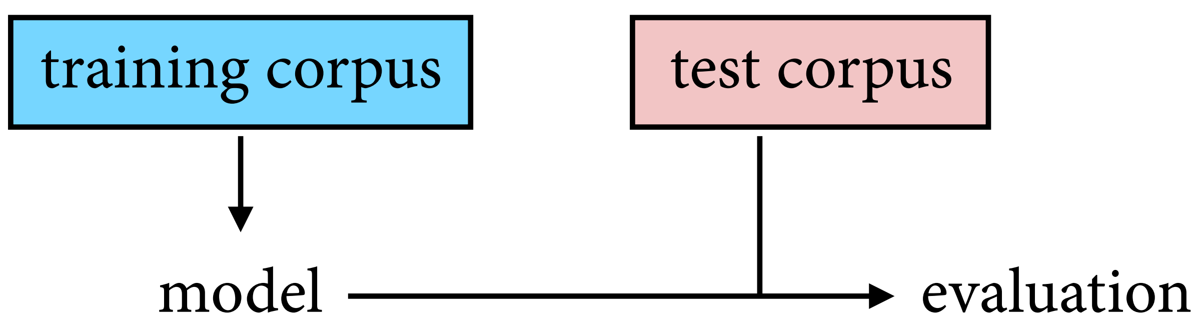
(. is special sentence start and end token)

(Chen & Goodman 98)

n-grams: Evaluation

- Measure quality of n-gram model using perplexity $PP(w) = P(w_1 \ldots w_N)^{-1/N}$ of test data $w = w_1 \ldots w_N$.

- To get honest picture of model's performance, evaluate it on test data that was not used for training.



- Maximum likelihood model for training corpus is not necessarily good for test corpus (overfitting).

Bigrams: a problem

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 98)

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.

- ML estimate predicts probability of 0 for n-grams that were not observed in training. This is a disaster because product with 0 is always 0.

Smoothing techniques

- Basic idea: Replace ML estimate

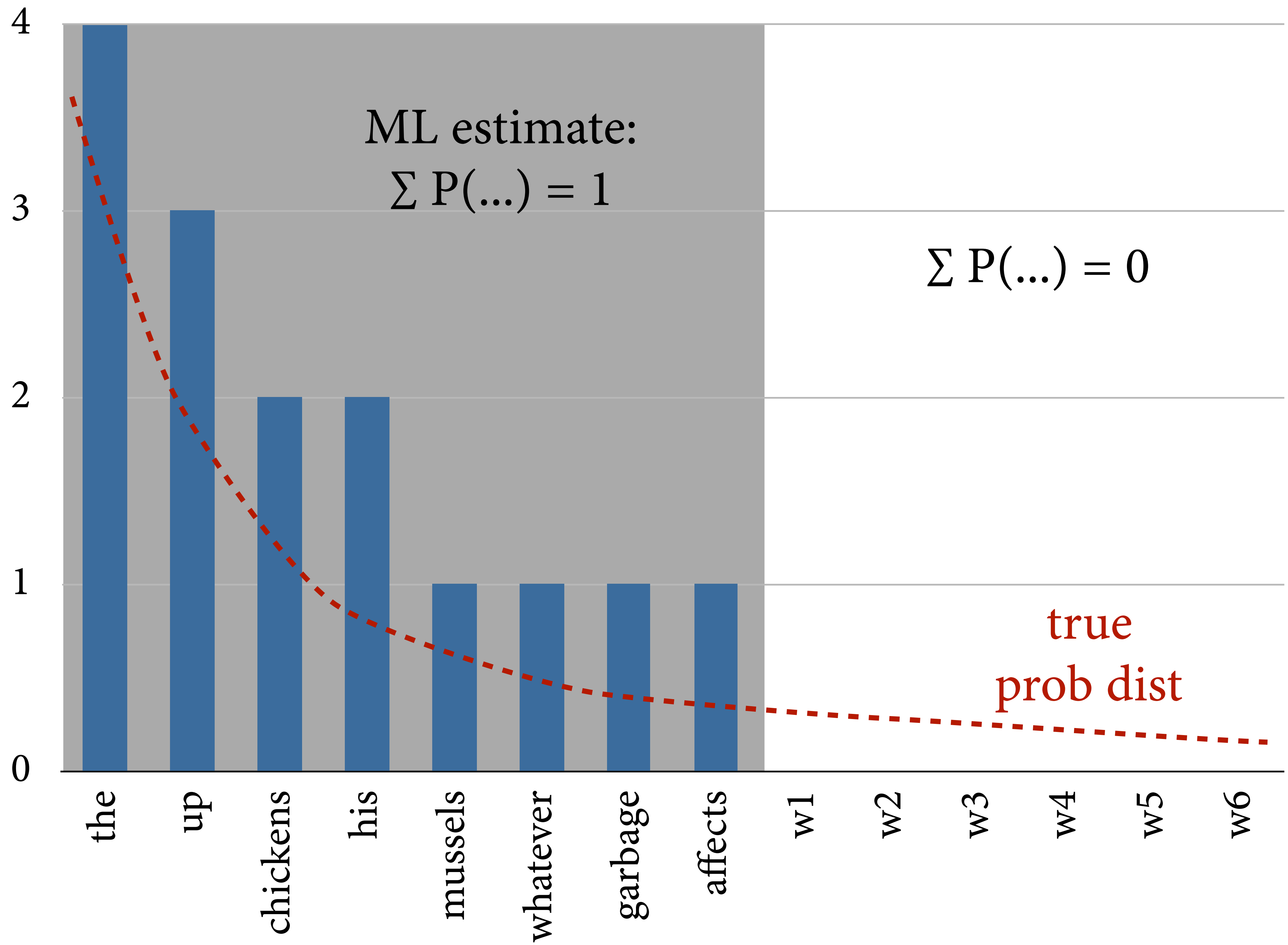
$$P_{\text{ML}}(w_i \mid w_{i-1}) = \frac{C(w_{i-1}w_i)}{C(w_{i-1})}$$

by estimate with adjusted bigram count

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

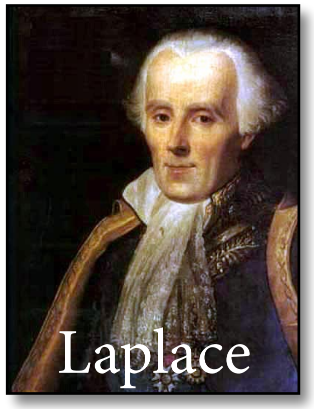
- Redistribute counts from seen to unseen bigrams.
- Generalizes easily to n-gram models with n > 2.

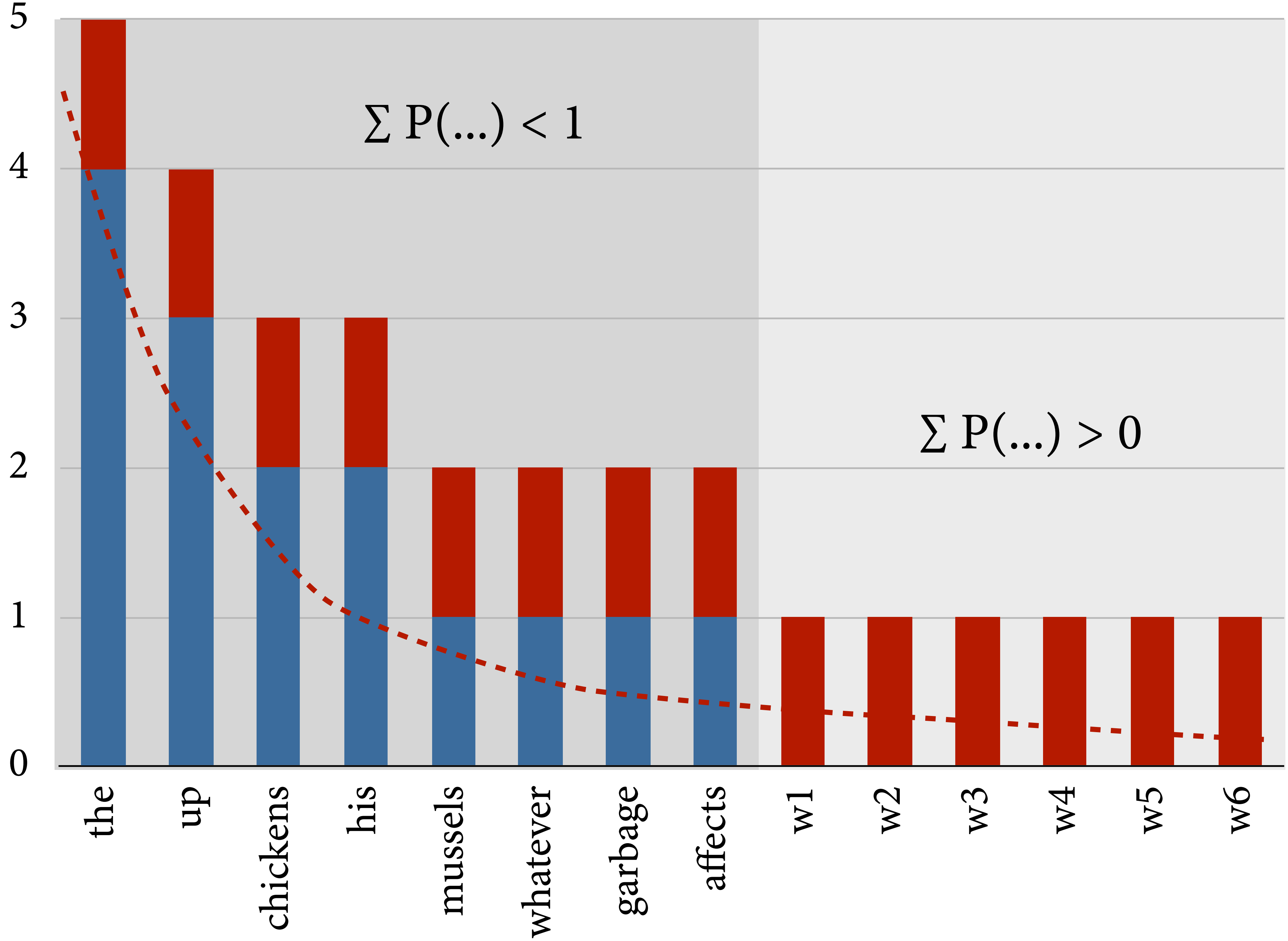
Smoothing



C(eat X) in Brown corpus

Add-one Smoothing





Add-one Smoothing

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

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

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

|V| = 11, |seen bigram types| = 11
⇒ 110 unseen bigrams

p(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(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

Add-one Smoothing

- Easy to implement, but dramatically overestimates probability of unseen events.
  - In the Cher example: $P_{lap}(\text{unseen} \mid w_{i-1}) \geq 1/14$;
    thus "count"(w_{i-1} unseen) $\approx 110 * 1/14 = 7.8$.
  - Compare against 12 bigram tokens in training corpus.

- This has been a very (<u>very</u>) active area of research for many years, and many very sophisticated solutions have been proposed, e.g. using second-order information about the corpus (how expectable are rare events).

- Importance of this reduced thanks to recent different methods for estimating pd. (neural networks).

why do language modelling?

- predictive text input
- important component of many applications:
  - machine translation
  - speech recognition
- common structure: generate many candidates, rank them according to “plausibility” as sentence

Conclusion

- Statistical models of natural language.
- Language models with n-grams.
- The problem of data sparseness.
- Smoothing.

Collaboration on Assignments

Acceptable:

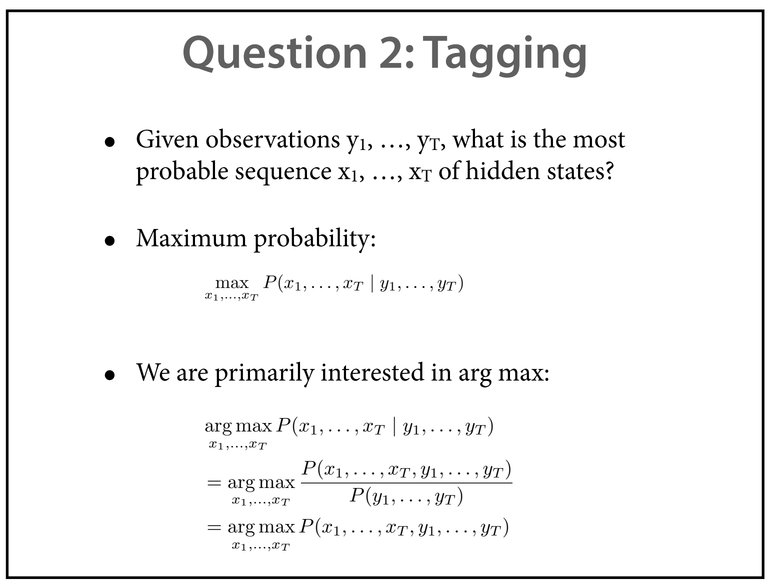
- discussing alternatives on how to do something
- asking someone for a description on how their algorithm works
- explaining on a conceptual level how you overcame an error message
- using a blog post/website for info on how an algorithm works/making your code more efficient

Unacceptable:

- working together on code
- dividing the assignment into parts
- using previous or existing solutions as a starting point
- copying source code from the web (& editing it)
- copying definitions/answers to discussion questions from a textbook or the web

slide credits

slides that look like this



come from

earlier editions of this class (ANLP), given by Alexander Koller

and their use is gratefully acknowledged. I try to make any modifications obvious, but if there are errors on a slide, assume that I added them.