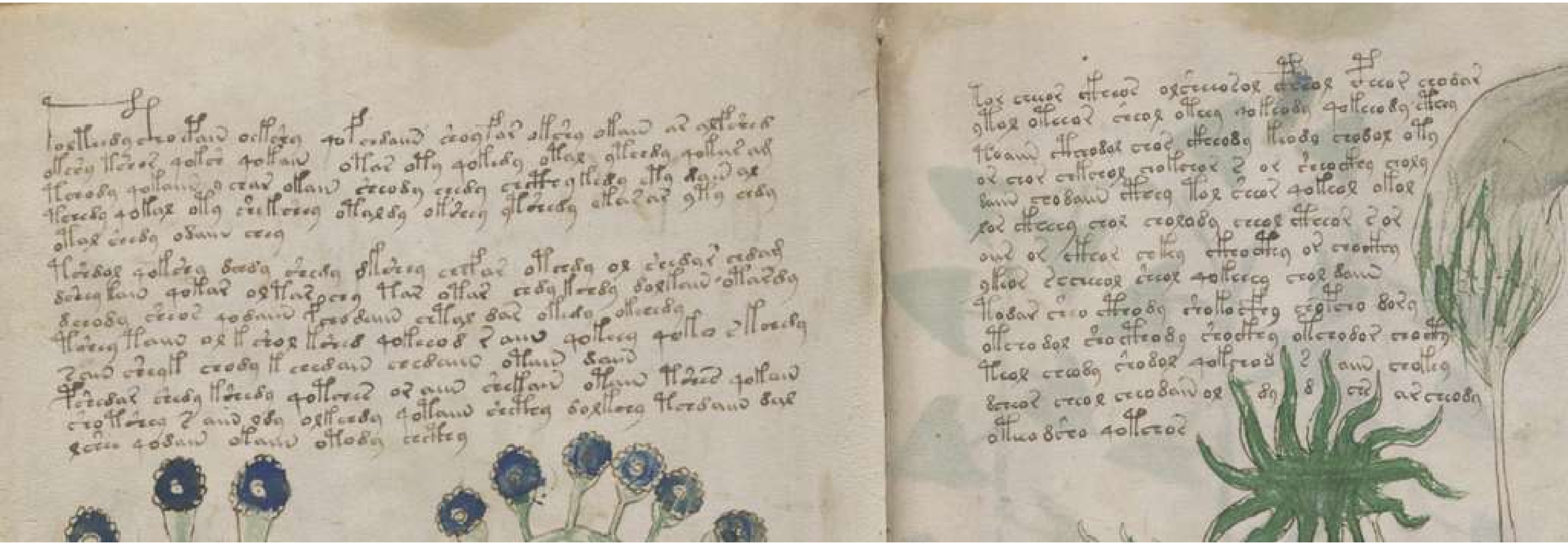


Natural Language Processing

Info 159/259

Lecture 6: Language models 1 (Feb 6, 2020)

David Bamman, UC Berkeley

Language Model

- Vocabulary $\mathcal{V}$ is a finite set of discrete symbols (e.g., words, characters); V = | $\mathcal{V}$ |
- $\mathcal{V}^+$ is the infinite set of sequences of symbols from $\mathcal{V}$; each sequence ends with STOP
- $x \in \mathcal{V}^+$

Language Model

$$P(w) = P(w_1, \ldots, w_n)$$

P("Call me Ishmael") =
P(w_1 = "call", w_2 = "me", w_3 = "Ishmael") x P(STOP)

$$\sum_{w \in V^+} P(w) = 1$$

$$0 \leq P(w) \leq 1$$

over all sequence lengths!

Language Model

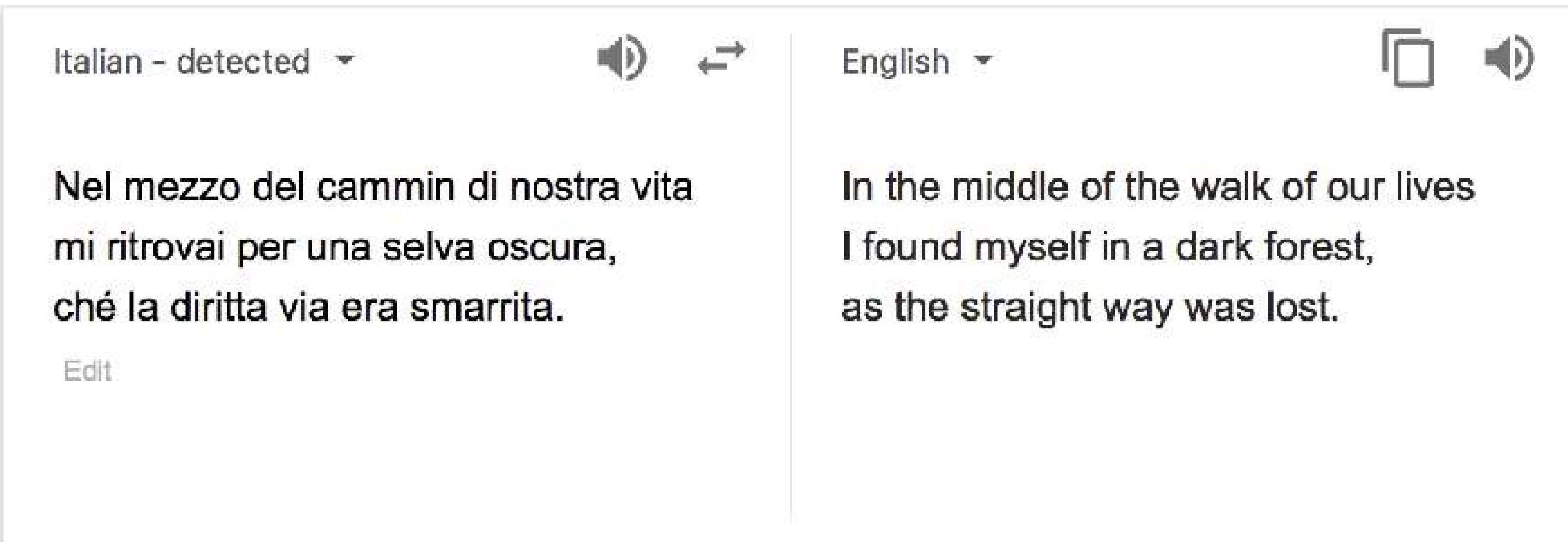
- Language models provide us with a way to quantify the likelihood of a sequence — i.e., plausible sentences.

OCR

To ſee great *Pompey* paſſe the ſtreets of Rome:
And when you ſaw his Chariot but appeare,
Haue you not made an Vniuerſall ſhout,
That Tyber trembled vnderneath her bankes
To heare the replication of your ſounds,
Made in her Concaue Shores?

- to fee great Pompey paffe the Areets of Rome:
- to see great Pompey passe the streets of Rome:

Machine translation



- Fidelity (to source text)
- Fluency (of the translation)


natural lan
natural language processing
natural language understanding
natural language processing with python
natural language generation
Google Search
I'm Feeling Lucky
Report inappropriate predictions

Speech Recognition

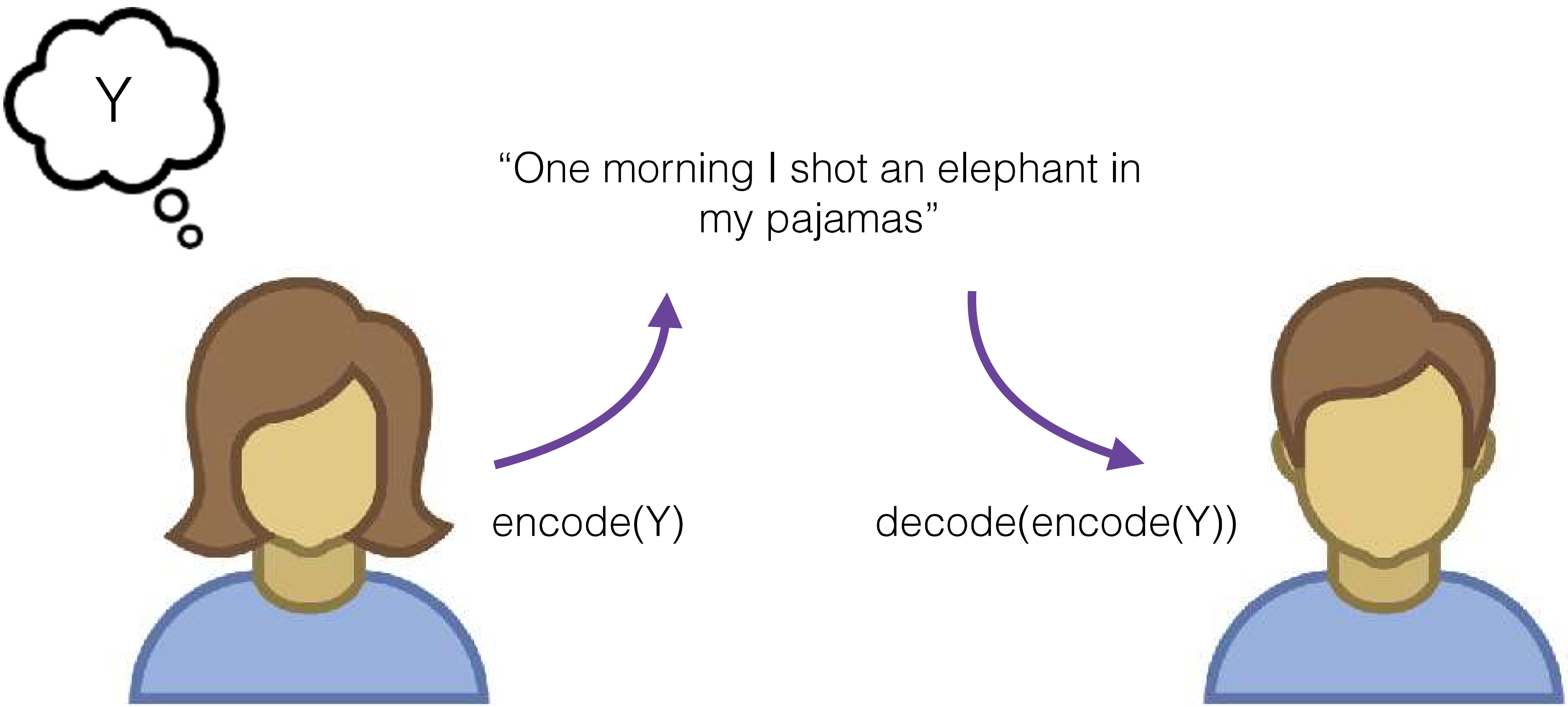
- 'Scuse me while I kiss the sky.
- 'Scuse me while I kiss this guy
- 'Scuse me while I kiss this fly.
- 'Scuse me while my biscuits fry

Dialogue generation

Baseline mutual information model (Li et al. 2015)	Proposed reinforcement learning model
A: Where are you going? (1) B: I'm going to the restroom. (2) A: See you later. (3) B: See you later. (4) A: See you later. (5) B: See you later. (6)	A: Where are you going? (1) B: I'm going to the police station. (2) A: I'll come with you. (3) B: No, no, no, no, you're not going anywhere. (4) A: Why? (5) B: I need you to stay here. (6) A: I don't know what you are talking about. (7) ...
A: how old are you? (1) B: I'm 16. (2) A: 16? (3) B: I don't know what you are talking about. (4) A: You don't know what you are saying. (5) B: I don't know what you are talking about . (6) A: You don't know what you are saying. (7) ...	A: How old are you? (1) B: I'm 16. Why are you asking? (2) A I thought you were 12. (3) B: What made you think so? (4) A: I don't know what you are talking about. (5) B: You don't know what you are saying. (6) ... ...

Li et al. (2016), "Deep Reinforcement Learning for Dialogue Generation" (EMNLP)

Information theoretic view



Shannon 1948

Noisy Channel

	X	Y
ASR	speech signal	transcription
MT	target text	source text
OCR	pixel densities	transcription

$$P(Y \mid X) \propto \underbrace{P(X \mid Y)}_{\text{channel model}} \underbrace{P(Y)}_{\text{source model}}$$

Language Model

- Language modeling is the task of estimating P(w)
- Why is this hard?

P("It was the best of times, it was the worst of times")

Chain rule (of probability)

$$
\begin{aligned}
P(x_1, x_2, x_3, x_4, x_5) = P(x_1)& \\
\times P(x_2 \mid x_1)& \\
\times P(x_3 \mid x_1, x_2)& \\
\times P(x_4 \mid x_1, x_2, x_3)& \\
\times P(x_5 \mid x_1, x_2, x_3, x_4)&
\end{aligned}
$$

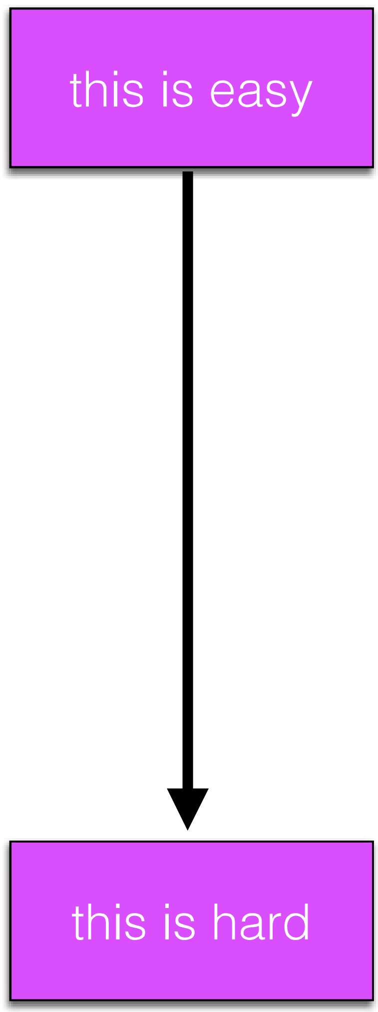
Chain rule (of probability)

P("It was the best of times, it was the worst of times")

Chain rule (of probability)

P("It")

P("was" | "It")



$$P(w_1)$$

$$P(w_2 \mid w_1)$$

$$P(w_3 \mid w_1, w_2)$$

$$P(w_4 \mid w_1, w_2, w_3)$$

$$P(w_n \mid w_1, \ldots, w_{n-1})$$

P("times" | "It was the best of times, it was the worst of")

Markov assumption

first-order

$$P(x_i \mid x_1, \ldots x_{i-1}) \approx P(x_i \mid x_{i-1})$$

second-order

$$P(x_i \mid x_1, \ldots x_{i-1}) \approx P(x_i \mid x_{i-2}, x_{i-1})$$

Markov assumption

bigram model
(first-order markov)

$$\prod_{i}^{n} P(w_i \mid w_{i-1}) \times P(\text{STOP} \mid w_n)$$

trigram model
(second-order markov)

$$\prod_{i}^{n} P(w_i \mid w_{i-2}, w_{i-1})$$

$$\times P(\text{STOP} \mid w_{n-1}, w_n)$$

"It was the best of times, it was the worst of times"

$$P(It \mid \mathrm{START}_1, \mathrm{START}_2)$$

$$P(was \mid \mathrm{START}_2, It)$$

$$P(the \mid It, was)$$

$$\ldots$$

$$P(times \mid worst, of)$$

$$P(\mathrm{STOP} \mid of, times)$$

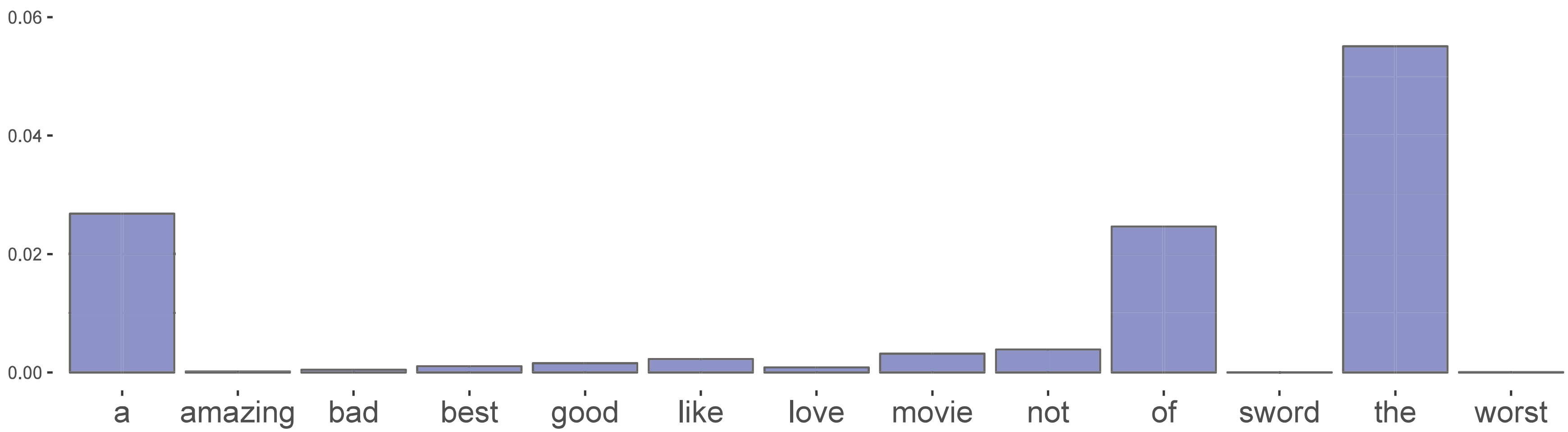
Estimation

unigram	bigram	trigram
$\prod_i^n P(w_i)$	$\prod_i^n P(w_i \mid w_{i-1})$	$\prod_i^n P(w_i \mid w_{i-2}, w_{i-1})$
$\times P(STOP)$	$\times P(STOP \mid w_n)$	$\times P(STOP \mid w_{n-1}, w_n)$

Maximum likelihood estimate

$$\frac{c(w_i)}{N} \qquad \frac{c(w_{i-1}, w_i)}{c(w_{i-1})} \qquad \frac{c(w_{i-2}, w_{i-1}, w_i)}{c(w_{i-2}, w_{i-1})}$$

Generating



- What we learn in estimating language models is P(word | context), where context — at least here — is the previous n-1 words (for ngram of order n)

- We have one multinomial over the vocabulary (including STOP) for each context

Generating

- As we sample, the words we generate form the new context we condition on

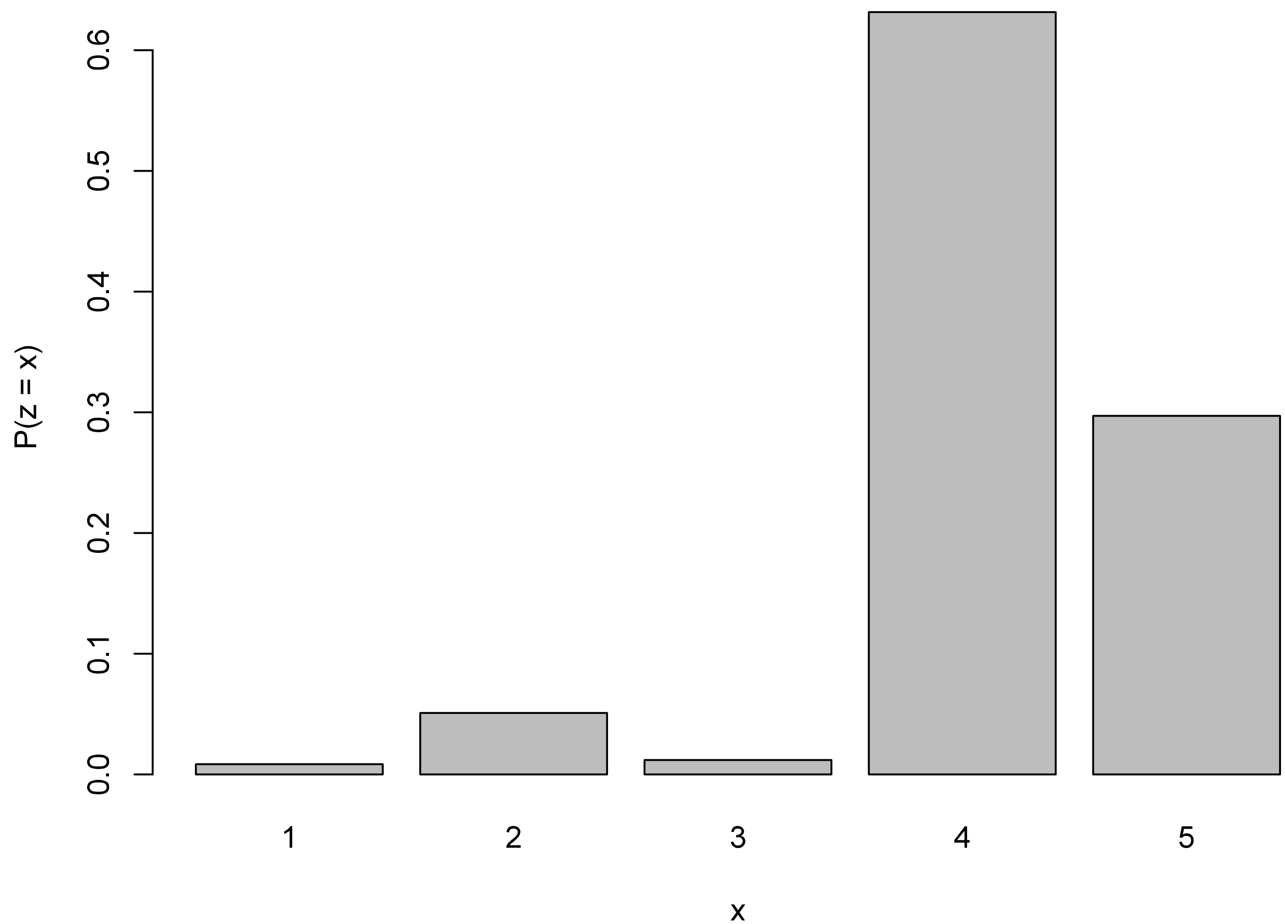
context1	context2	generated word
START	START	The
START	The	dog
The	dog	walked
dog	walked	in

Aside: sampling?

Sampling from a Multinomial

Probability mass function (PMF)

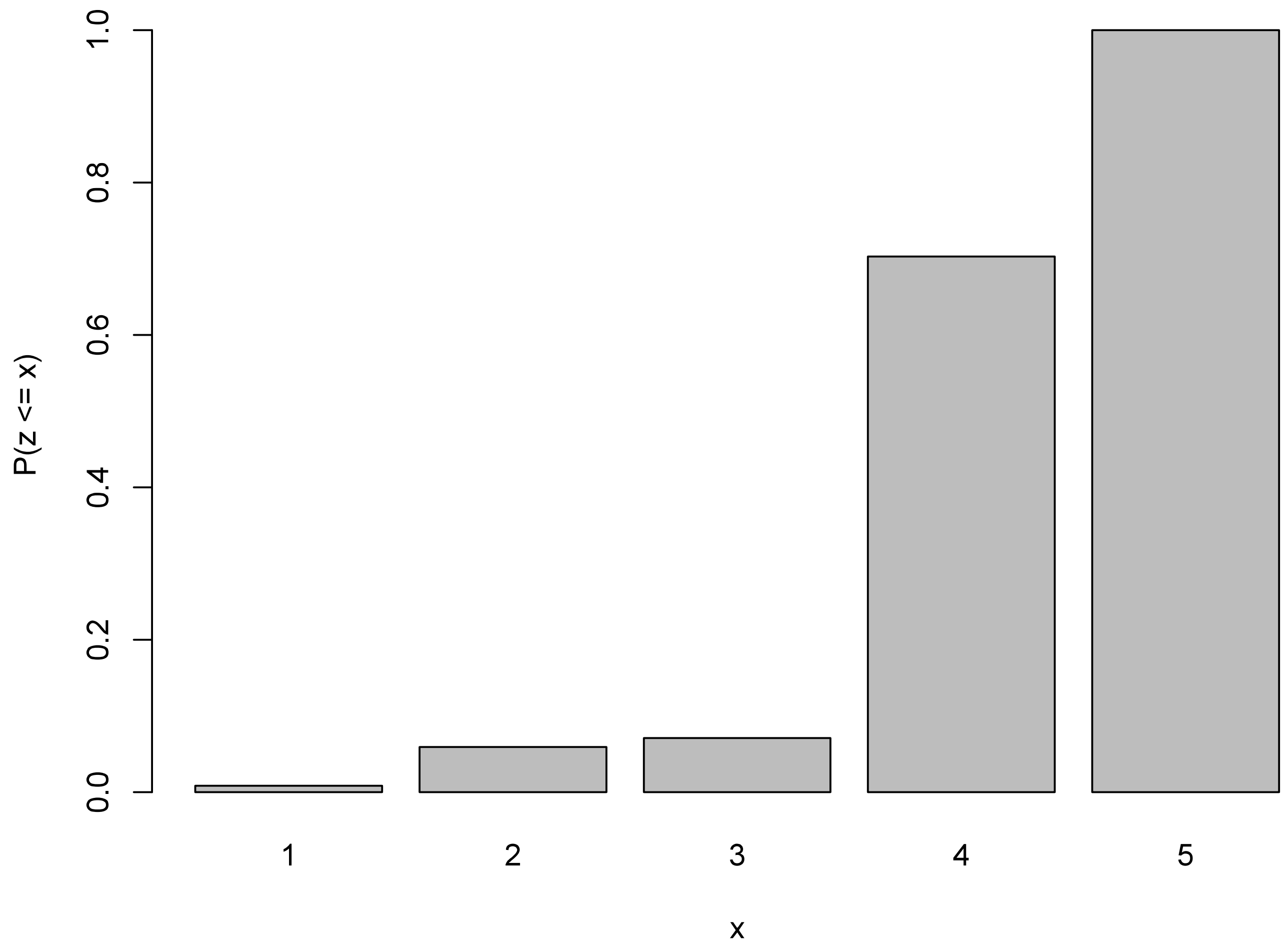
P(z = x) exactly



Sampling from a Multinomial

Cumulative density function (CDF)

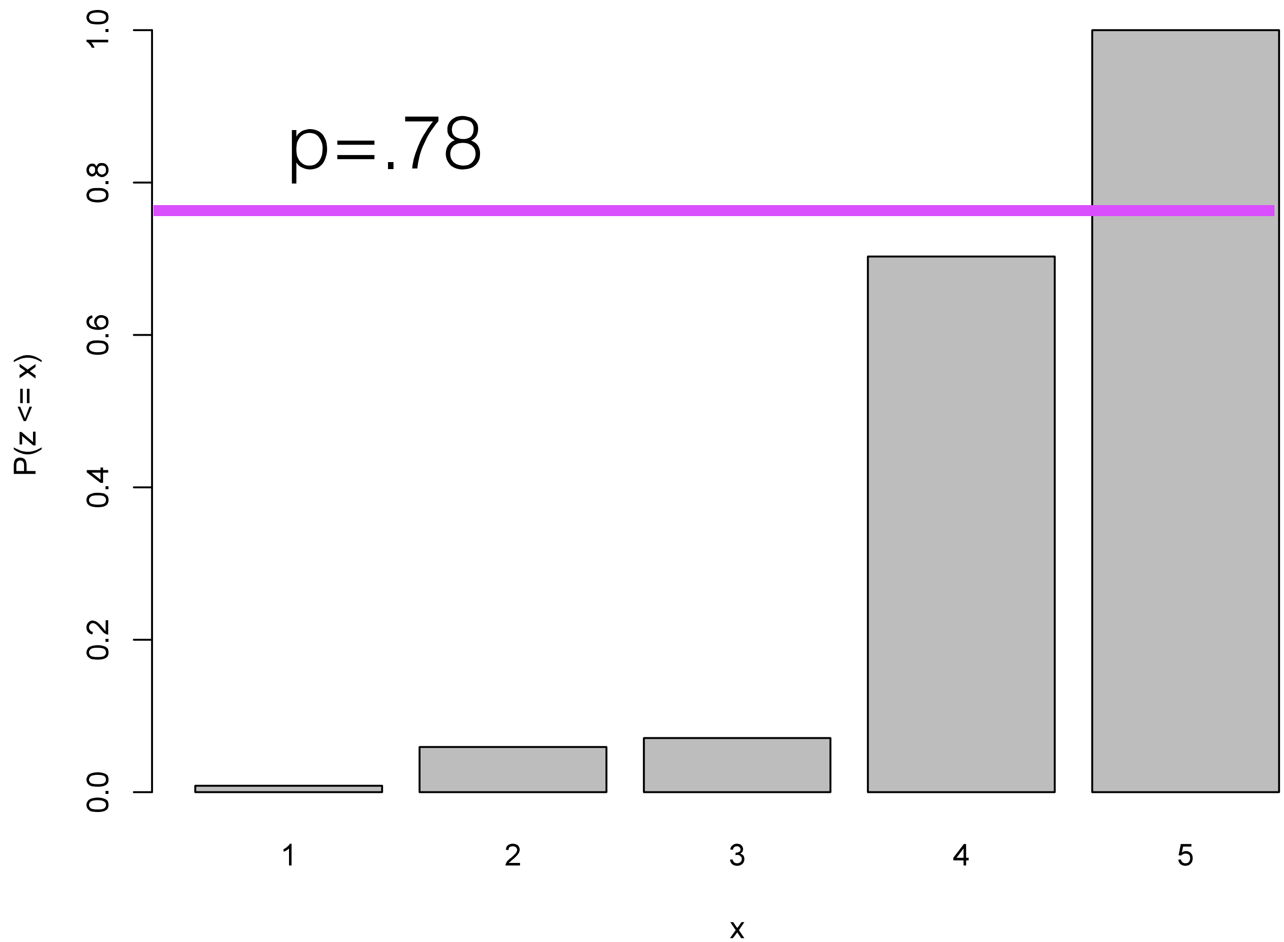
$P(z \leq x)$



Sampling from a Multinomial

Sample *p* uniformly in [0,1]

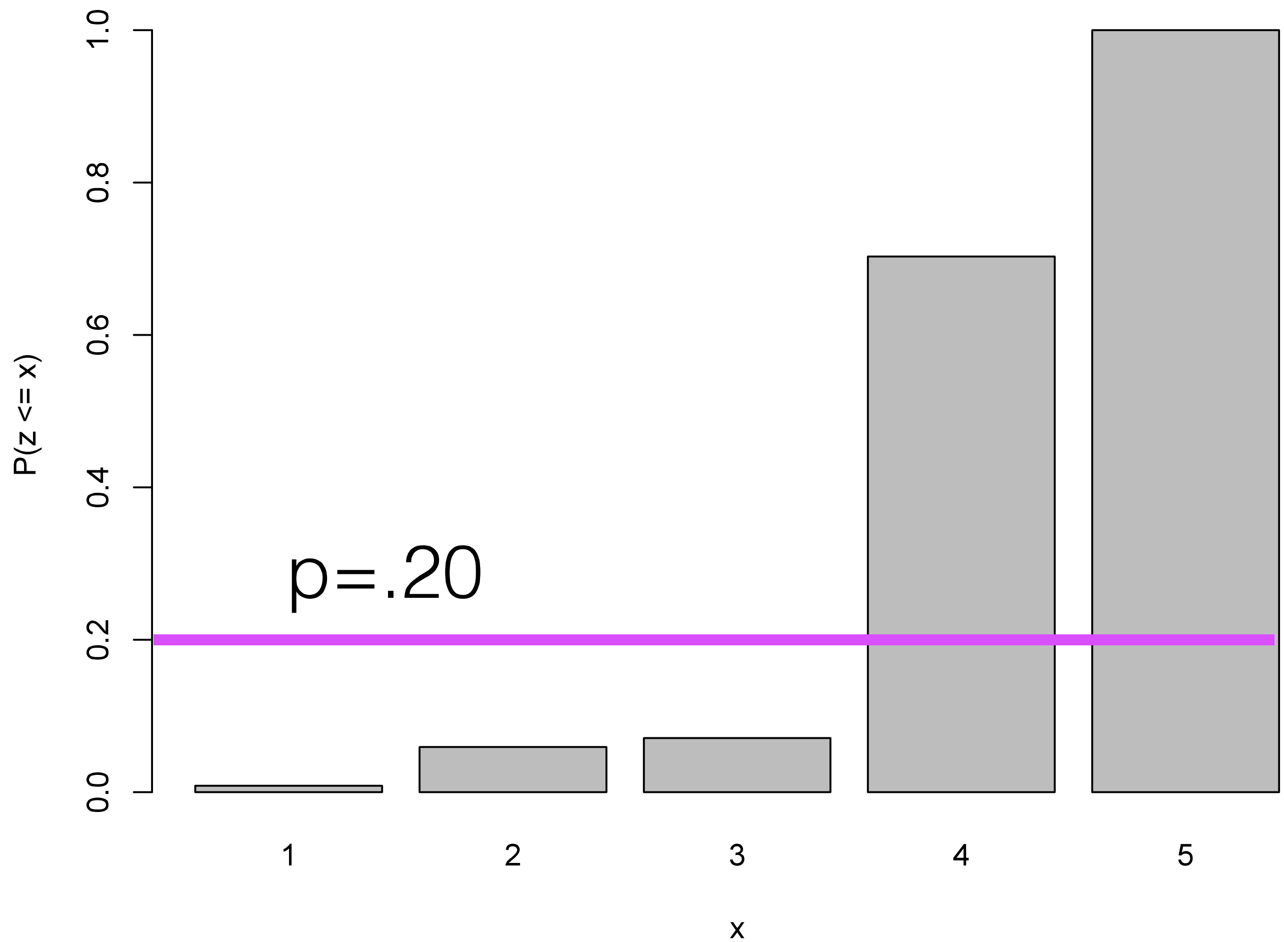
Find the point $CDF^{-1}(p)$



Sampling from a Multinomial

Sample *p* uniformly in [0,1]

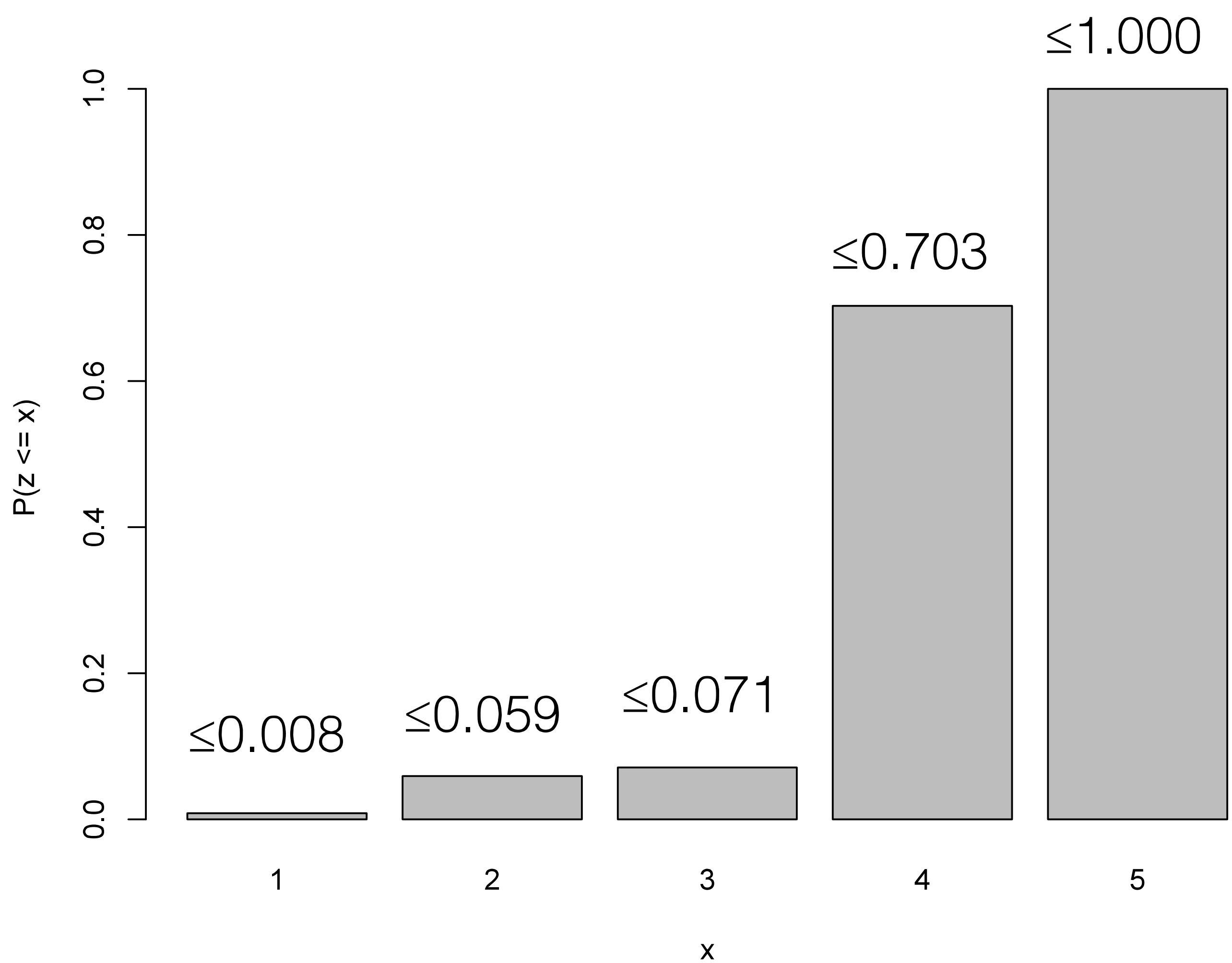
Find the point $CDF^{-1}(p)$



Sampling from a Multinomial

Sample p uniformly in [0,1]

Find the point $CDF^{-1}(p)$



Unigram model

- the around, she They I blue talking “Don’t to and little come of
- on fallen used there. young people to Lázaro
- of the
- the of of never that ordered don't avoided to complaining.
- words do had men flung killed gift the one of but thing seen I plate Bradley was by small Kingmaker.

Bigram Model

- “What the way to feel where we’re all those ancients called me one of the Council member, and smelled Tales of like a Korps peaks.”
- Tuna battle which sold or a monocle, I planned to help and distinctly.
- “I lay in the canoe ”
- She started to be able to the blundering collapsed.
- “Fine.”

Trigram Model

- “I’ll worry about it.”
- Avenue Great-Grandfather Edgeworth hasn’t gotten there.
- “If you know what. It was a photograph of seventeenth-century flourishin’ To their right hands to the fish who would not care at all. Looking at the clock, ticking away like electronic warnings about wonderfully SAT ON FIFTH
- Democratic Convention in rags soaked and my past life, I managed to wring your neck a boss won’t so David Pritchet giggled.
- He humped an argument but her bare He stood next to Larry, these days it will have no trouble Jay Grayer continued to peer around the Germans weren’t going to faint in the

4gram Model

- Our visitor in an idiot sister shall be blotted out in bars and flirting with curly black hair right marble, wallpapered on screen credit.”

- You are much instant coffee ranges of hills.

- Madison might be stored here and tell everyone about was tight in her pained face was an old enemy, trading-posts of the outdoors watching Anyog extended On my lips moved feebly.

- said.

- “I’m in my mind, threw dirt in an inch,’ the Director.

Evaluation

- The best evaluation metrics are external — how does a better language model influence the application you care about?
- Speech recognition (word error rate), machine translation (BLEU score), topic models (sensemaking)

Evaluation

- A good language model should judge unseen real language to have high probability
- Perplexity = inverse probability of test data, averaged by word.
- To be reliable, the test data must be truly unseen (including knowledge of its vocabulary).

$$\text{perplexity} = \sqrt[N]{\frac{1}{P(w_1, \ldots, w_n)}}$$

Experiment design

	training	development	testing
size	80%	10%	10%
purpose	training models	model selection; hyperparameter tuning	evaluation; never look at it until the very end

$$\sqrt[N]{\frac{1}{\prod_i^N P(w_i)}} = \left(\prod_i^N P(w_i)\right)^{-\frac{1}{N}}$$

$$= \exp\log\left(\prod_i^N P(w_i)\right)^{-\frac{1}{N}}$$

$$= \exp\left(-\frac{1}{N}\log\prod_i^N P(w_i)\right)$$

perplexity $= \exp\left(-\frac{1}{N}\sum_i^N \log P(w_i)\right)$

Perplexity

bigram model
(first-order markov)

$$= \exp\left(-\frac{1}{N}\sum_{i}^{N} \log P(w_i \mid w_{i-1})\right)$$

trigram model
(second-order markov)

$$= \exp\left(-\frac{1}{N}\sum_{i}^{N} \log P(w_i \mid w_{i-2}, w_{i-1})\right)$$

Perplexity

Model	Unigram	Bigram	Trigram
Perplexity	962	170	109

SLP3 4.3

Smoothing

- When estimating a language model, we're relying on the data we've observed in a training corpus.
- Training data is a small (and biased) sample of the creativity of language.

Data sparsity

	i	want	to	eat	chinese	food	lunch	spend
i	5	827	0	9	0	0	0	2
want	2	0	608	1	6	6	5	1
to	2	0	4	686	2	0	6	211
eat	0	0	2	0	16	2	42	0
chinese	1	0	0	0	0	82	1	0
food	15	0	15	0	1	4	0	0
lunch	2	0	0	0	0	1	0	0
spend	1	0	1	0	0	0	0	0

Figure 4.1 Bigram counts for eight of the words (out of $V = 1446$) in the Berkeley Restaurant Project corpus of 9332 sentences. Zero counts are in gray.

SLP3 4.1

$$\prod_{i}^{n} P(w_i \mid w_{i-1}) \times P(\text{STOP} \mid w_n)$$

- As in Naive Bayes, $P(w_i) = 0$ causes $P(w) = 0$. (Perplexity?)

Smoothing in NB

- One solution: add a little probability mass to every element.

maximum likelihood estimate

$$P(x_i \mid y) = \frac{n_{i,y}}{n_y}$$

$n_{i,y}$ = count of word i in class y
n_y = number of words in y
V = size of vocabulary

smoothed estimates

$$P(x_i \mid y) = \frac{n_{i,y} + \alpha}{n_y + V\alpha}$$

same α for all x_i

$$P(x_i \mid y) = \frac{n_{i,y} + \alpha_i}{n_y + \sum_{j=1}^{V} \alpha_j}$$

possibly different α for each x_i

Additive smoothing

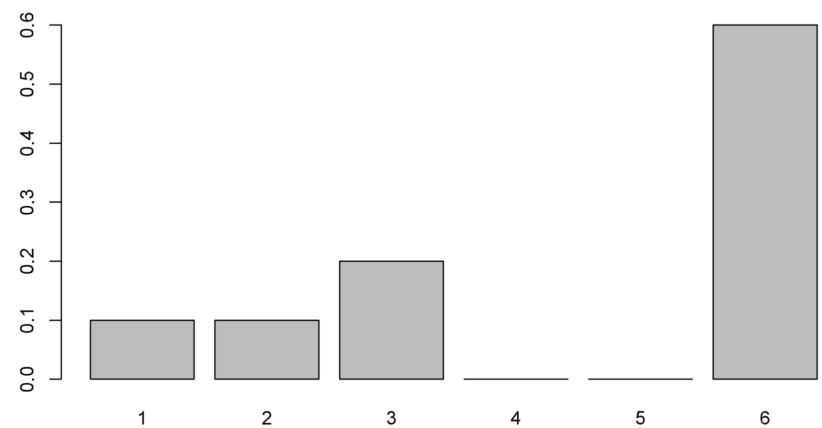
Laplace smoothing:
α = 1

$$P(w_i) = \frac{c(w_i) + \alpha}{N + V\alpha}$$

$$P(w_i \mid w_{i-1}) = \frac{c(w_{i-1}, w_i) + \alpha}{c(w_{i-1}) + V\alpha}$$

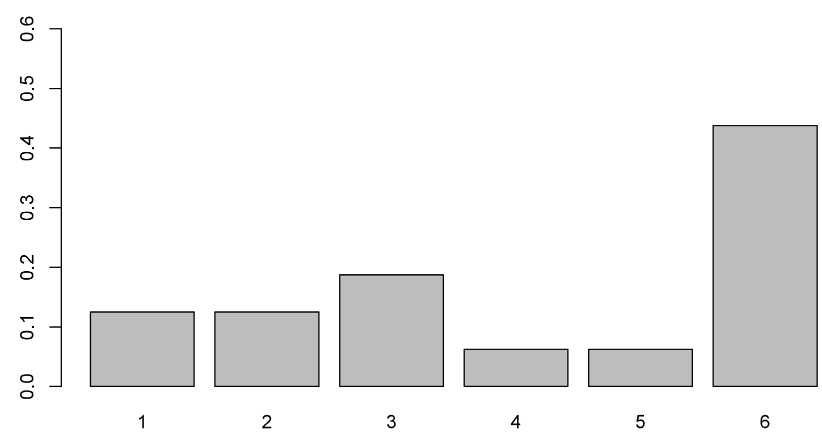
Smoothing

MLE



Smoothing is the re-allocation of probability mass

smoothing with $\alpha = 1$



Smoothing

- How can best re-allocate probability mass?

Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Center for Research in Computing Technology, Harvard University, 1998.

Interpolation

- As ngram order rises, we have the potential for higher precision but also higher variability in our estimates.
- A linear interpolation of any two language models p and q (with $\lambda \in [0,1]$) is also a valid language model.

$$\lambda p + (1 - \lambda) q$$

p = the web

q = political speeches

Interpolation

- We can use this fact to make higher-order language models more robust.

$$
\begin{aligned}
P(w_i \mid w_{i-2}, w_{i-1}) = {} & \lambda_1 P(w_i \mid w_{i-2}, w_{i-1}) \\
& + \lambda_2 P(w_i \mid w_{i-1}) \\
& + \lambda_3 P(w_i)
\end{aligned}
$$

$$\lambda_1 + \lambda_2 + \lambda_3 = 1$$

Interpolation

- How do we pick the best values of λ?
  - Grid search over development corpus
  - Expectation-Maximization algorithm (treat as missing parameters to be estimated to maximize the probability of the data we see).

Kneser-Ney smoothing

- Intuition: When backing off to a lower-order ngram, maybe the overall ngram frequency is not our best guess.

I can't see without my reading ___________

P("Francisco") > P("glasses")

- *Francisco* is more frequent, but shows up in fewer unique bigrams ("San Francisco") — so we shouldn't expect it in new contexts; *glasses*, however, does show up in many different bigrams

Kneser-Ney smoothing

- Intuition: estimate how likely a word is to show up in a new continuation?
- How many different bigram types does a word type *w* show up in (normalized by all bigram types that are seen)

continuation probability: of all bigram types in training data, how many is w the suffix for?

$$\frac{|v \in \mathcal{V} : c(v, w) > 0|}{|v', w' \in \mathcal{V} : c(v', w') > 0|}$$

$P_{CONTINUATION}$(Francisco)

$$\approx \frac{149}{10000000}$$

$P_{CONTINUATION}$(dog)

$$\approx \frac{1391}{10000000}$$

$$P_{CONTINUATION}(w) = \frac{|v \in \mathcal{V} : c(v, w) > 0|}{|v', w' \in \mathcal{V} : c(v', w') > 0|}$$

$P_{CONTINUATION}(w)$ is the continuation probability for the unigram w (the frequency with which it appears as the suffix in distinct bigram types)

Kneser-Ney smoothing

discounted mass

$$\frac{\max\{c(w_{i-1}, w_i) - d, 0\}}{c(w_{i-1})} + \lambda(w_{i-1})\, P_{CONTINUATION}(w_i)$$

discounted bigram probability

continuition probability

Kneser-Ney smoothing

$$\frac{\max\{c(w_{i-1}, w_i) - d, 0\}}{c(w_{i-1})}$$

discounted bigram probability

d is a discount factor (usually between 0 and 1 — how much we discount the observed counts by

Kneser-Ney smoothing

prefix types

$$\lambda(w_{i-1}) = d\frac{|w_i \in \mathscr{V} : c(w_{i-1}, w_i) > 0|}{c(w_{i-1})}$$

prefix tokens

λ here captures the discounted mass we're reallocating from prefix w_{i-1}

Kneser-Ney smoothing

w_{i-1}	w_i	$C(w_{i-1}, w_i)$	$C(w_{i-1}, w_i) - d(1)$
red	hook	3	2
red	car	2	1
red	watch	10	9
sum		15	12

$$\lambda(\text{red}) = 1 \times \frac{3}{15}$$

12/15 of the probability mass stays with the original counts; 3/15 is reallocated

$$P_{CONTINUATION}(w) = \frac{|v \in \mathcal{V} : c(v, w) > 0|}{|v', w' \in \mathcal{V} : c(v', w') > 0|}$$

discounted mass

$$\frac{\max\{c(w_{i-1}, w_i) - d, 0\}}{c(w_{i-1})} + \lambda(w_{i-1})\, P_{CONTINUATION}(w_i)$$

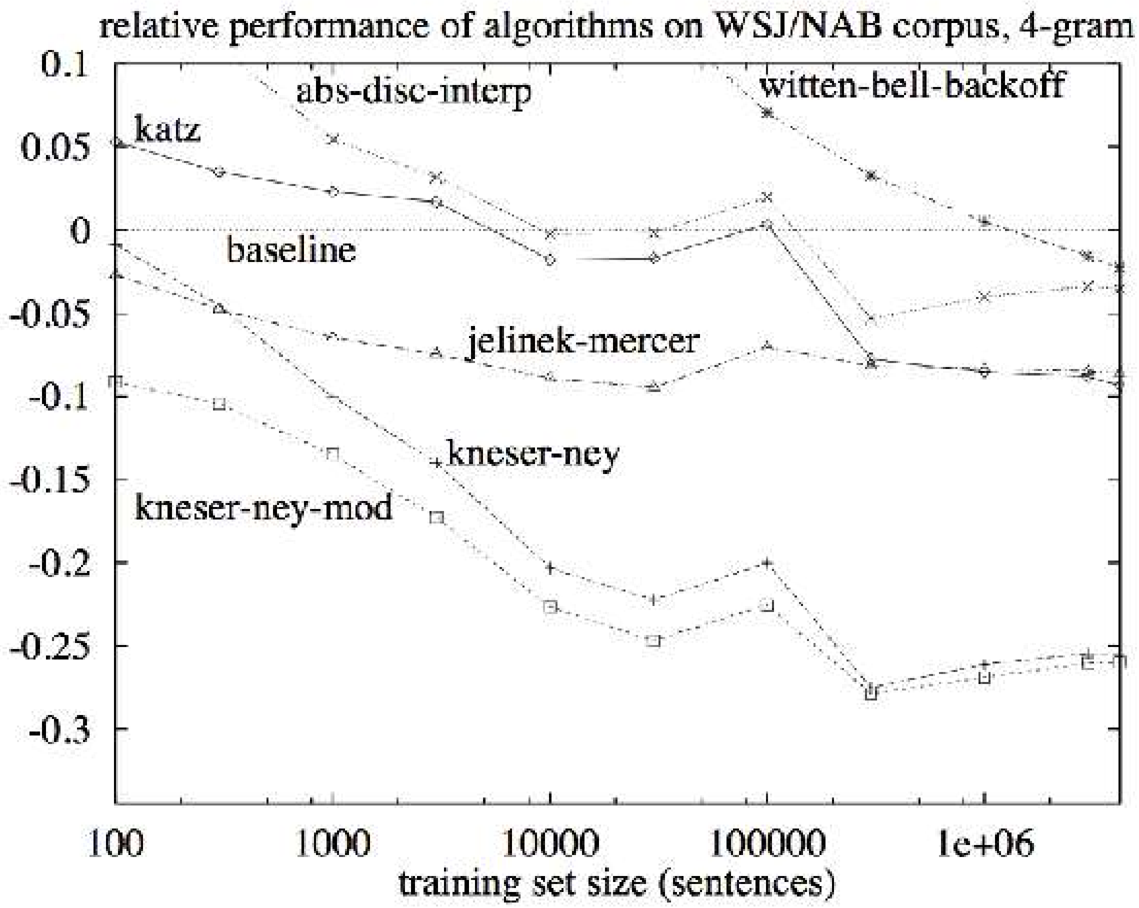
discounted bigram probability

continuition probability

we'll move all of the mass we subtracted here over to this side

$$\frac{\max\{c(w_{i-1}, w_i) - d, 0\}}{c(w_{i-1})} + \lambda(w_{i-1})\, P_{CONTINUATION}(w_i)$$

and distribute it according to the continuation probability



Stanley F. Chen and Joshua Goodman. An empirical study of smoothing techniques for language modeling. Technical Report TR-10-98, Center for Research in Computing Technology, Harvard University, 1998.

"Stupid backoff"

if full sequence observed

$$S(w_i \mid w_{i-k+1}, \ldots, w_{i-1}) = \frac{c(w_{i-k+1}, \ldots, w_i)}{c(w_{i-k+1}, \ldots, w_{i-1})}$$

No discounting here, just back off to lower order ngram if the higher order is not observed.

Cheap to calculate; works almost as well as KN when there is a lot of data

otherwise

$$= \lambda S(w_i \mid w_{i-k+2}, \ldots, w_{i-1})$$

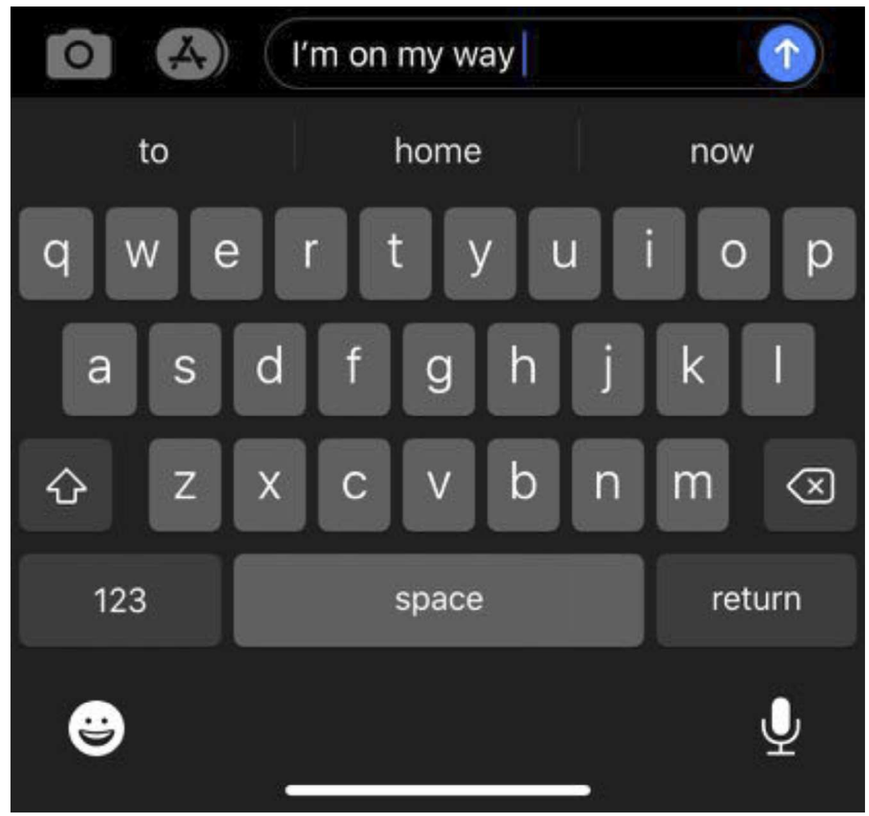
Brants et al. (2007), "Large Language Models in Machine Translation"

Why?

- Language models give us an estimate for the probability of a sequence, which is directly useful for applications that are deciding between different sentences) as viable outputs:
  - Machine translation
  - Speech recognition
  - OCR
  - Dialogue agents

Why?

- Language models directly allow us to predict the next word in a sequence (useful for autocomplete).



Why?

- Language models can directly encode knowledge present in the training corpus.

The director of *The Irishman* is ______________

Why?

- Language models can directly encode knowledge present in the training corpus.

Query	Answer	Generation
Francesco Bartolomeo Conti was born in ____.	Florence	Rome [-1.8] , **Florence [-1.8]** , Naples
Adolphe Adam died in ____.	Paris	**Paris [-0.5]** , London [-3.5] , Vienna
English bulldog is a subclass of ____.	dog	dogs [-0.3] , breeds [-2.2] , **dog [-2.4]**
The official language of Mauritius is ____.	English	**English [-0.6]** , French [-0.9] , Arabic
Patrick Oboya plays in ____ position.	midfielder	centre [-2.0] , center [-2.2] , **midfielder**
Hamburg Airport is named after ____.	Hamburg	Hess [-7.0] , Hermann [-7.1] , Schmidt

Petroni et al. (2019), "Language Models as Knowledge Bases?" (ACL)

Why?

- Language modeling turns out to be a good proxy task for learning about linguistic structure.
- See contextual word embeddings (BERT/ELMo), in class 2/18.

You should feel comfortable:

- Calculate the probability of a sentence given a trained model
- Estimating (e.g., trigram) language model
- Evaluating perplexity on held-out data
- Sampling a sentence from a trained model

Tools

- SRILM
  http://www.speech.sri.com/projects/srilm/

- KenLM
  https://kheafield.com/code/kenlm/

- Berkeley LM
  https://code.google.com/archive/p/berkeleylm/