Lecture 2: N-gram Language Models

Claire Cardie, Tanya Goyal

CS 4740 (and crosslists): Introduction to Natural Language Processing

Administrivia

- HW0 due on Friday, 11.59 p.m.
- HW1 will be released next Monday, Feb 3.
  - We will post a mega-thread on ed to find a partner.
  - Optional partner-matching service.

What is a Language Model?

- A model that computes a probability distribution over any sequence of words:

$$P(w_1 w_2 w_3 \ldots w_n)$$

legacy example from Cornell NLP course.

e.g.

P(Mayenne ate my shoes today.) $= 10^{-12}$

P(Mayenne ate my) $= 10^{-9}$

P(I ate dinner in Collegetown.) $= 2 \times 10^{-10}$

P(Collegetown Bagels slaps.) $= 10^{-14}$

Q: Why would we ever want to do this?

Language Models' Use

- Grammar Error Correction

  $P(\textit{You're nice.}) \gg P(\textit{Your nice.})$

- Automatic Speech Recognition (ASR)
  - **Input**: Audio, **Output**: Text

$P(\textit{I saw a van}) >>>> P(\textit{Eyes awe of an})$

What else?

Credit: *Yoav Artzi's LM-Class*

Language Models' Use

- ASR Noisy Channel System
  - **Input**: Audio a, **Output**: Text $\mathbf{w}$
- We want to decode $\mathbf{w}$ from given acoustics a:

$$\mathbf{w}^* = \arg\max_{\mathbf{w}} P(\mathbf{w} \mid a)$$

Bayes Rule

$$P(A \mid B) = \frac{P(B \mid A)P(A)}{P(B)}$$

$$\arg\max_{\mathbf{w}} P(\mathbf{w} \mid a) = \arg\max_{\mathbf{w}} \frac{P(a \mid \mathbf{w})P(\mathbf{w})}{P(a)}$$

$$= \arg\max_{\mathbf{w}} P(a \mid \mathbf{w}) \, P(\mathbf{w})$$

Acoustic Model: Distribution over acoustic waves given a sentence

Language Model: Distribution over word sequences

Language Models' Use

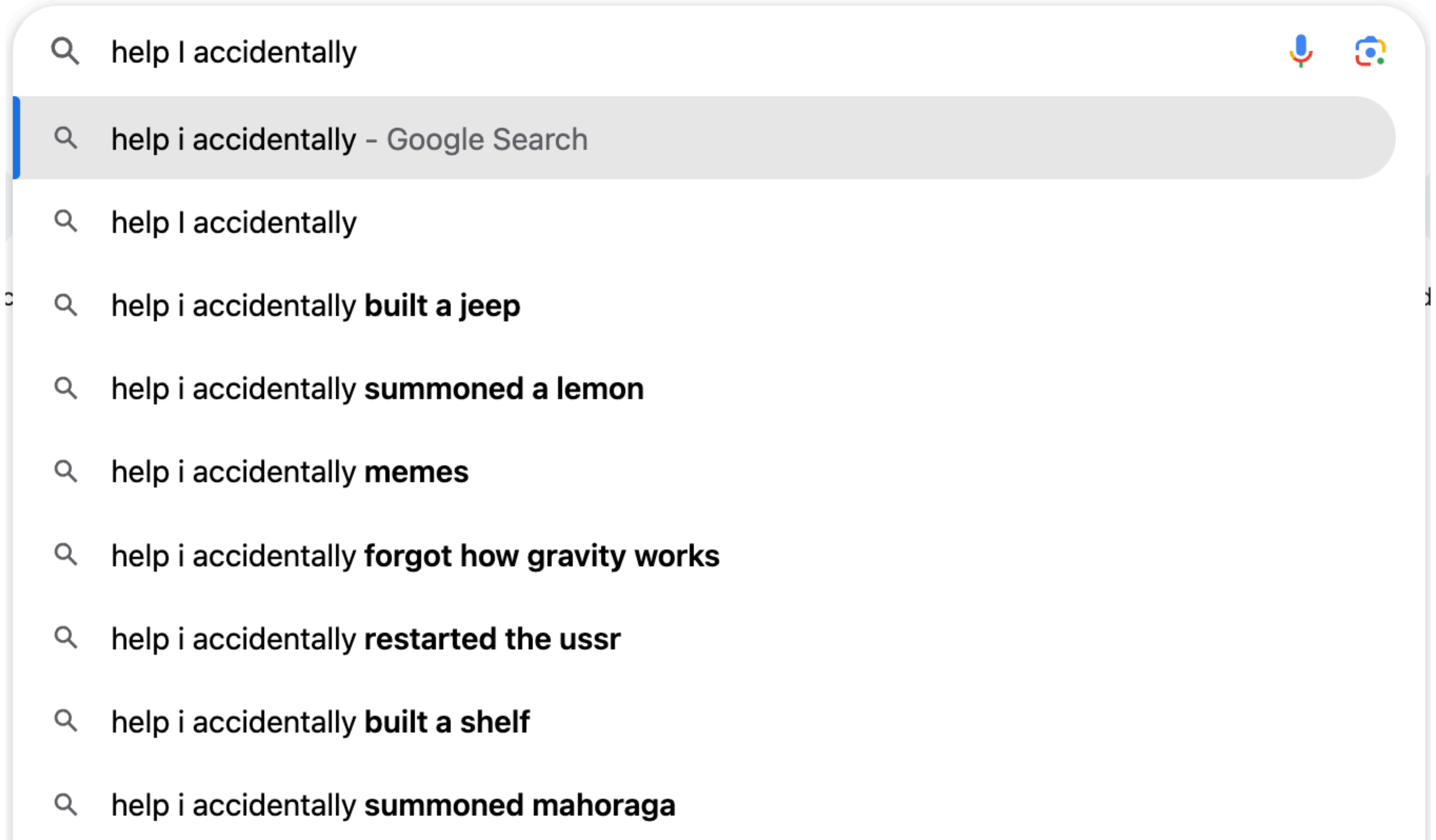
$$\arg\max_{\mathbf{w}} P(a \mid \mathbf{w})\, P(\mathbf{w})$$

the station signs are in deep in english	-14732
the stations signs are in deep in english	-14735
the station signs are in deep into english	-14739
the station 's signs are in deep in english	-14740
the station signs are in deep in the english	-14741
the station signs are indeed in english	-14757
the station 's signs are indeed in english	-14760
the station signs are indians in english	-14790
the station signs are indian in english	-14799
the stations signs are indians in english	-14807
the stations signs are indians and english	-14815

Language Models' Use

Where else are language models used?

Google



Language Models can be powerful

FEBRUARY 14, 2019

Better Language Models and Their Implications

We've trained a large-scale unsupervised language model which generates coherent paragraphs of text, achieves state-of-the-art performance on many language modeling benchmarks, and performs rudimentary reading comprehension, machine translation, question answering, and summarization — all without task-specific training.

VIEW CODE

READ PAPER

READ MORE

If any language task can be described as a text-to-text problem…

Sentiment Analysis:

What is the sentiment of I loved the movie? Very positive.

Language Models can be powerful

FEBRUARY 14, 2019

Better Language Models and Their Implications

We've trained a large-scale unsupervised language model which generates coherent paragraphs of text, achieves state-of-the-art performance on many language modeling benchmarks, and performs rudimentary reading comprehension, machine translation, question answering, and summarization — all without task-specific training.

VIEW CODE

READ PAPER

READ MORE

If any language task can be described as a text-to-text problem…

Machine Translation:

What is the translation of "J'aime Lucy" in English? I love Lucy.

…then conceptually, we can solve it by just generating the answer as a continuation of a "prompt"

It would need to be a very powerful LM though!

Language Modeling Problem

- Let $\mathscr{V}$ be a finite vocabulary of words.

$$\mathscr{V} = \{\text{ the, a, man, telescope, Madrid, two, } \ldots\}$$

- We can construct (infinite) word sequences $\mathbf{w}$

$$\mathscr{V}^{\dagger} = \{\text{ the, a, the a, the fan, the man, the man with a telescope}\}$$

- **Input**: a dataset of sentences $\mathscr{D} = \{\mathbf{w}\}_{i=1}^{M}$
- **Goal**: estimate a probability distribution over **all** word sequences:

$$P(\mathbf{w}), \quad P(\mathbf{w}) \geq 0 \text{ for all } \mathbf{w} \in \mathscr{V}^{\dagger}$$

Language Modeling Problem

- **Use**: estimate $P(\mathbf{w})$, where $\mathbf{w}$ is a sentence.
- **Learning Input:** M observations of raw sentences $\mathbf{w}$
- **Learning Output:** model that computes $P(\mathbf{w})$ over any $\mathbf{w}$

- Probabilities should broadly indicate plausibility of sentences:
  - P(I saw a van) > P(eyes awe of an)
  - Not *only* grammaticality: P(artichokes intimidate zippers) ~0
  - Plausibility depends on the context.

Language Modeling Problem

- **Use**: estimate $P(\mathbf{w})$, where $\mathbf{w}$ is a sentence.
- **Learning Input:** M observations of raw sentences $\mathbf{w}$
- **Learning Output:** model that computes $P(\mathbf{w})$ over any $\mathbf{w}$

So, how do we estimate $P(\mathbf{w})$?

Naive option: empirical distribution over the training data.

$$P(\mathbf{w}) = \frac{c(\mathbf{w})}{M}$$

Problem? Does not generalize to unseen sentences!

Language Modeling Problem

First, let's decompose $P(\mathbf{w})$

$$P(\mathbf{w}) = P(w_1 w_2 w_3 \ldots w_n)$$

applying chain rule

$$= P(w_1)\ P(w_2 \,|\, w_1)\ P(w_3 \,|\, w_2 w_1) \ldots P(w_n \,|\, w_1 \ldots w_{n-1})$$

assumption: probability of a word depends on previous words only

$$= \prod_{i=1}^{n} P(w_i \,|\, w_1 \ldots w_{i-1})$$

$P(\text{I saw a man}) = P(\text{I})\ P(\text{saw} \,|\, \text{I})\ P(\text{a} \,|\, \text{I saw})\ P(\text{man} \,|\, \text{I saw a})$

Language Modeling Problem

$$P(\mathbf{w}) = P(w_1 w_2 w_3 \ldots w_n) \;\; = \prod_{i=1}^{n} P(w_i \,|\, w_1 \ldots w_{i-1})$$

Can we now use count based estimates?

$$= P(w_1)\; P(w_2 \,|\, w_1)\; P(w_3 \,|\, w_2 w_1) \ldots P(w_n \,|\, w_1 \ldots w_{n-1})$$

If a test sentence **w** is unseen in the training data, this will again be zero!

Language Modeling Problem

$$P(\mathbf{w}) = P(w_1 w_2 w_3 \ldots w_n) \; = \prod_{i=1}^{n} P(w_i \,|\, w_1 \ldots w_{i-1})$$

Key idea: Markov Assumption

Probability of each word in a continuation only depends on a fixed number of previous words

$$\approx \prod_{i=1}^{n} P(w_i \,|\, w_{i-k+1} \ldots w_{i-1})$$

N-gram language models: Probability of each word depends on N-1 previous words.

Unigram Language Models

Assumption: Each word w_i is sampled from a i.i.d. distribution.

$$P(\mathbf{w}) = P(w_1 w_2 w_3 \ldots w_n) \; = \prod_{i=1}^{n} P(w_i)$$

where $w_i \in \mathcal{V} \cup \mathrm{STOP}$

Does this solve this sparsity problem?

To a large extent, yes. We can compute probability of an unseen sentence by multiplying probability of words.

Unigram Language Models

Assumption: Each word w_i is sampled from a i.i.d. distribution.

$$P(\mathbf{w}) = P(w_1 w_2 w_3 \dots w_n) \;= \prod_{i=1}^{n} P(w_i)$$

where $w_i \in \mathscr{V} \cup \mathrm{STOP}$

How do we learn this?

Parameter of a unigram LM are probabilities of each word in $\mathscr{V}$.

$$P(w) = \frac{c(w)}{c()}$$

Unigram Language Models

Assumption: Each word w_i is sampled from a i.i.d. distribution.

$$P(\mathbf{w}) = P(w_1 w_2 w_3 \ldots w_n) = \prod_{i=1}^{n} P(w_i)$$

where $w_i \in \mathcal{V} \cup \mathrm{STOP}$

Note: In addition to assigning a probability distribution to some sentence, we can also generate/decode a sentence!

$i = 0$

repeat

  $i++$

  $w_i \sim P(w)$

until $w_i = \mathrm{STOP}$

return $< w_1 w_2 \ldots w_i >$

Unigram Language Models

Assumption: Each word w_i is sampled from a i.i.d. distribution.

$$P(\mathbf{w}) = P(w_1 w_2 w_3 \dots w_n) = \prod_{i=1}^{n} P(w_i)$$

where $w_i \in \mathscr{V} \cup \mathrm{STOP}$

More frequent words will have higher prob.

P(the the) > P(ice cream)

Let's generate!

- [thrift, did, eighty, said, hard, 'm, july, bullish]
- []
- [after, any, on, consistently, hospital, lake, of, of, other, and, factors, raised, analyst, too, allowed, mexico, never, consider, fall, bungled, davison, that, obtain, price, lines, the, to, sass, the, the, further, board, a, details, machinists, between, nasdaq]

Bi-gram Language Models

N=2

$$P(\mathbf{w}) = P(w_1 w_2 w_3 \ldots w_n) \;= \prod_{i=1}^{n} P(w_i \mid w_{i-1})$$

where $w_i \in \mathcal{V} \cup \{\mathrm{STOP}\}$ and $w_0 =$ <s>

Let's generate!

- [texaco, rose, one, in, this, issue, is, pursuing, growth, in, a, boiler, house, said, mr., gurria, mexico, 's, motion, control, proposal, without, permission, from, five, hundred, fifty, five, yen]
- [although, common, shares, rose, forty, six, point, four, hundred, dollars, from, thirty, seconds, at, the, greatest, play, disingenuous, to, be, reset, annually, the, buy, out, of, american, brands, vying, for, mr., womack, currently, sharedata, incorporated, believe, chemical, prices, undoubtedly, will, be, as, much, is, scheduled, to, conscientious, teaching]
- [this, would, be, a, record, november]

N-gram Language Models

Can be extended to any N.

$$P(\mathbf{w}) = P(w_1 w_2 w_3 \ldots w_n) \;=\; \prod_{i=1}^{n} P(w_i \mid w_{i-(N-1)} \ldots w_{i-1})$$

But how do we **learn** this?

Remember, parameters are these probabilities.

Maximum likelihood estimates has a closed-form solution: relative frequencies.

E.g. for bi-gram models: $q_{MLE}(u \mid v) = \dfrac{c(u,v)}{c(v)}$

Training Counts

198015222 the first
194623024 the same
168504105 the following
158562063 the world
…
14112454 the door

23135851162 the *

$$P_{ML}(\text{door} \mid \text{the}) = \frac{14{,}112{,}454}{2{,}313{,}581{,}162} = 0.0006$$

N-gram Language Models

- **Learning Input:** M observations of raw sentences $\mathbf{w}$
- **Learning Output:** model that computes $P(\mathbf{w})$ over any $\mathbf{w}$

$$P(\mathbf{w}) = P(w_1 w_2 w_3 \ldots w_n) \;= \prod_{i=1}^{n} P(w_i \,|\, w_{i-(N-1)} \ldots w_{i-1})$$

Compute ML estimates using the M observations.

$$q_{MLE}(u \,|\, v \ldots) = \frac{c(u, v \ldots)}{c(v \ldots)}$$

Use it to assign probabilities to any test sentence or generate

N-gram Models on Shakespeare

- **1-gram**
  - To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have gram
  - Hill he late speaks; or! a more to leg less first you enter
- **2-gram**
  - Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow.
  - What means, sir. I confess she? then all sorts, he is trim, captain.
- **3-gram**
  - Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, ’tis done.
  - This shall forbid it should be branded, if renown made it empty.
- **4-gram**
  - King Henry. What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv’d in;
  - It cannot be but so.

N-gram Models on Shakespeare

- **Corpus statistics**
  - 884,647 tokens, vocabulary size of =29,066
  - Shakespeare produced 300,000 bigram types out of = 844M possible bigrams
    - So 99.96% of the possible bigrams were never seen (have zero entries in the table)

N-gram Language Models

- How should we choose N?

Because it was a sunny day, I should take a ___________.

Suppose N=2:

P (raincoat | Because it was a sunny day, I should take a) = P (raincoat | a)

P (hat | Because it was a sunny day, I should take a) = P (hat | a)

Suppose N=3:

P (raincoat | Because it was a sunny day, I should take a) = P (raincoat | take a)

P (hat | Because it was a sunny day, I should take a) = P (hat | take a)

N-gram Language Models

- How should we choose N?

rainy
Because it was a sunny day, I should take a ___________.

Suppose N=2:

P (raincoat | Because it was a sunny day, I should take a) = P (raincoat | a)

P (hat | Because it was a sunny day, I should take a) = P (hat | a)

Suppose N=3:

P (raincoat | Because it was a sunny day, I should take a) = P (raincoat | take a)

P (hat | Because it was a sunny day, I should take a) = P (hat | take a)

N-gram Language Models

- How should we choose N?

Solution: Increase N?

We run into the previous sparsity problem! 

Sparsity in LMs

- What happens if we encounter zero counts in the training data?

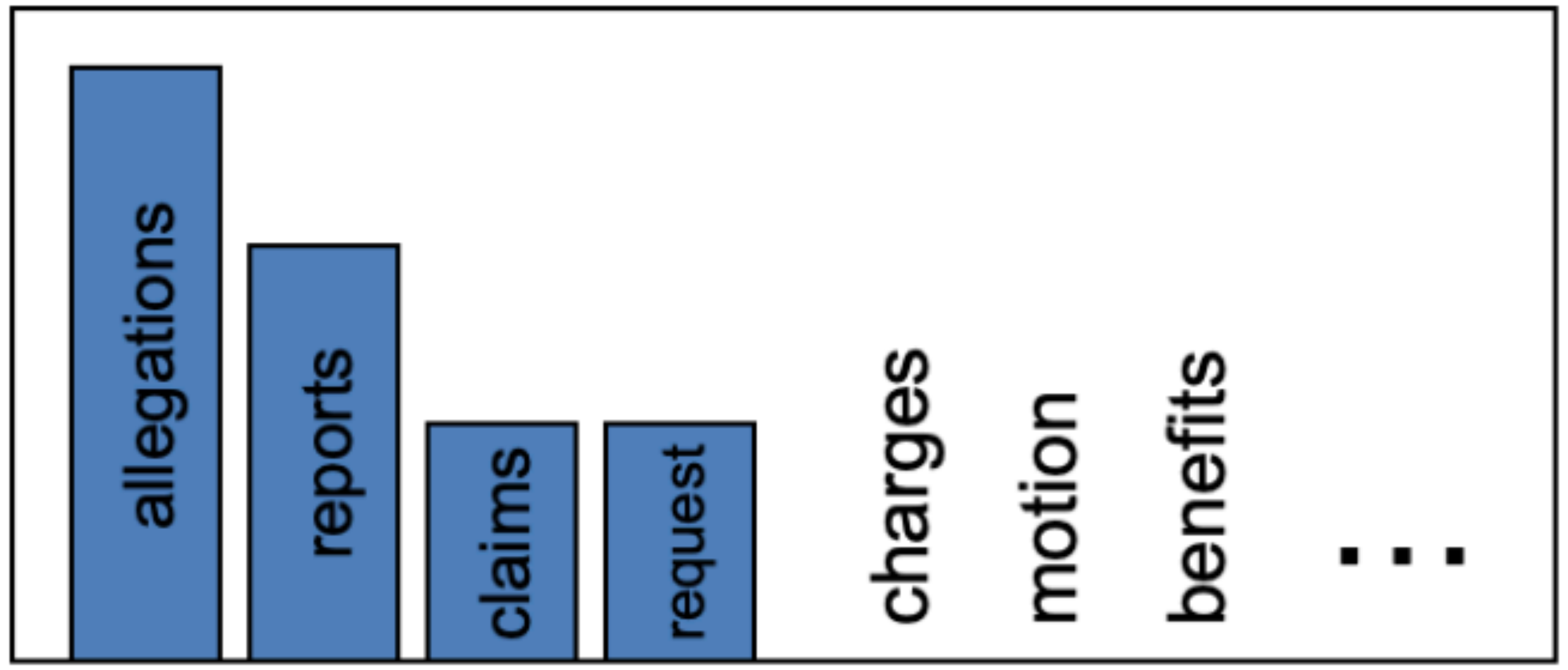
Training Set	Test Set
… denied the allegations	… denied the offer
… denied the reports	… denied the loan
… denied the claims	
… denied the request	

$$p(\text{offer} \mid \text{denied the}) = 0$$

- A single n-gram with zero probability –> probability of the entire sequence is 0.

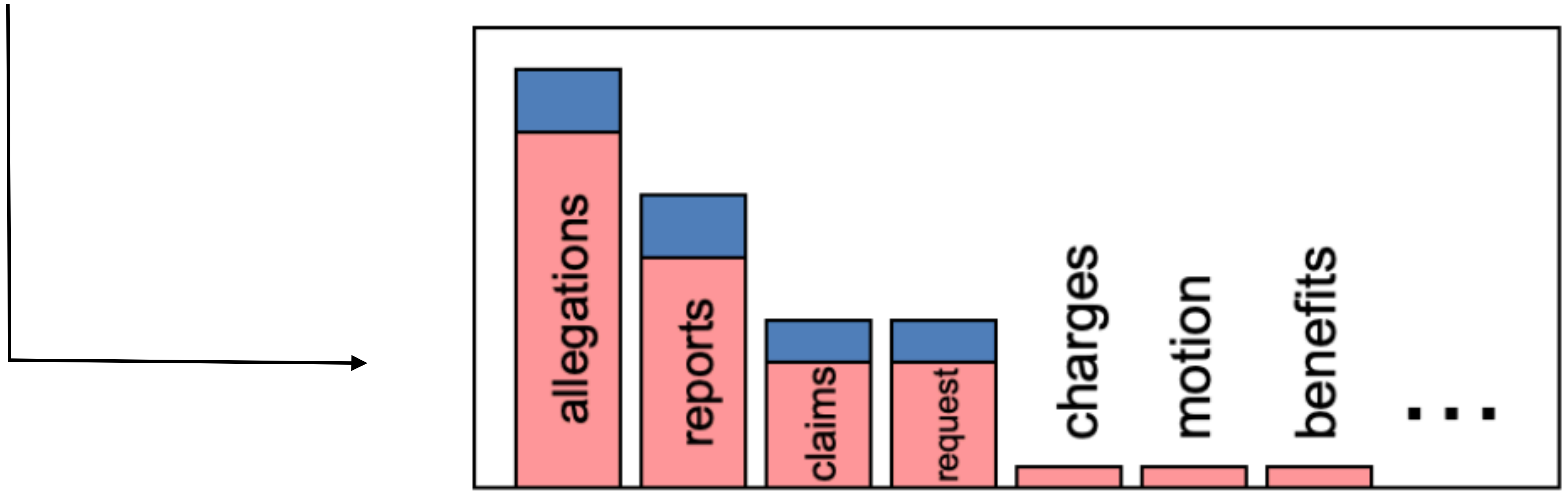
Smoothing

- Goal: Estimating statistics from sparse data.
- Idea: **Steal** some probability mass from seen data.



P(w | denied the)
3 allegations
2 reports
1 claims
1 request

7 total



P(w | denied the)
2.5 allegations
1.5 reports
0.5 claims
0.5 request
2 other

7 total

Smoothing

- **Add-one smoothing**
- Pretend we saw each word one more time that we did (even unseen ones). For 2-gram:

$$P_{MLE} = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})} \rightarrow P_{MLEAdd-1} = \frac{c(w_{i-1}, w_i) + 1}{c(w_{i-1}) + |\mathcal{V}|}$$

- Called Laplace Smoothing.
- Can be generalized to Add-K

$$P_{MLEAdd-K} = \frac{c(w_{i-1}, w_i) + K}{c(w_{i-1}) + K.|\mathcal{V}|}$$

Berkeley Restaurant Corpus

Raw counts: 9222 sentences

- Bigrams

	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

- Unigram

i	want	to	eat	chinese	food	lunch	spend
2533	927	2417	746	158	1093	341	278

Berkeley Restaurant Corpus

Bi-gram probabilities

$$P_{MLE}(w_i \mid w_{i-1}) = \frac{c(w_i w_{i-1})}{c(w_{i-1})}$$

	i	want	to	eat	chinese	food	lunch	spend
i	0.002	0.33	0	0.0036	0	0	0	0.00079
want	0.0022	0	0.66	0.0011	0.0065	0.0065	0.0054	0.0011
to	0.00083	0	0.0017	0.28	0.00083	0	0.0025	0.087
eat	0	0	0.0027	0	0.021	0.0027	0.056	0
chinese	0.0063	0	0	0	0	0.52	0.0063	0
food	0.014	0	0.014	0	0.00092	0.0037	0	0
lunch	0.0059	0	0	0	0	0.0029	0	0
spend	0.0036	0	0.0036	0	0	0	0	0

Berkeley Restaurant Corpus

Smoothed counts (Add-1)

	i	want	to	eat	chinese	food	lunch	spend
i	6	828	1	10	1	1	1	3
want	3	1	609	2	7	7	6	2
to	3	1	5	687	3	1	7	212
eat	1	1	3	1	17	3	43	1
chinese	2	1	1	1	1	83	2	1
food	16	1	16	1	2	5	1	1
lunch	3	1	1	1	1	2	1	1
spend	2	1	2	1	1	1	1	1

Berkeley Restaurant Corpus

Smoothed bigram probs (Add-1)

$$P_{MLEAdd-1} = \frac{c(w_{i-1}, w_i) + 1}{c(w_{i-1}) + |\mathcal{V}|}$$

	i	want	to	eat	chinese	food	lunch	spend
i	0.0015	0.21	0.00025	0.0025	0.00025	0.00025	0.00025	0.00075
want	0.0013	0.00042	0.26	0.00084	0.0029	0.0029	0.0025	0.00084
to	0.00078	0.00026	0.0013	0.18	0.00078	0.00026	0.0018	0.055
eat	0.00046	0.00046	0.0014	0.00046	0.0078	0.0014	0.02	0.00046
chinese	0.0012	0.00062	0.00062	0.00062	0.00062	0.052	0.0012	0.00062
food	0.0063	0.00039	0.0063	0.00039	0.00079	0.002	0.00039	0.00039
lunch	0.0017	0.00056	0.00056	0.00056	0.00056	0.0011	0.00056	0.00056
spend	0.0012	0.00058	0.0012	0.00058	0.00058	0.00058	0.00058	0.00058

Other smoothing options

- **Back-off smoothing:** use lower-order n-gram
  - For tri-gram, use tri-gram if you have good evidence, otherwise use bi-gram, otherwise unigram
- **Linear interpolation**: mix lower-order n-grams
  - For tri-gram, mix with with bi-gram and unigram probabilities

$$P_\lambda(x_i \mid x_{i-1}, x_{i-2}) = \lambda_3 p_{\text{MLE}}(x_i \mid x_{i-1}, x_{i-2}) + \lambda_2 p_{\text{MLE}}(x_i \mid x_{i-1}) + \lambda_1 p_{\text{MLE}}(x_i)$$

$$\sum \lambda_i = 1$$

Slide Acknowledgements

- Earlier versions of this course offerings including materials from Marten van Schijndel, Lillian Lee.
- Yoav Artzi's LM-class.