Lecture 3:
n-gram Language Models (cont.)
& Word Embedding

Instructor: Xiang Ren

USC CSCI 444 NLP

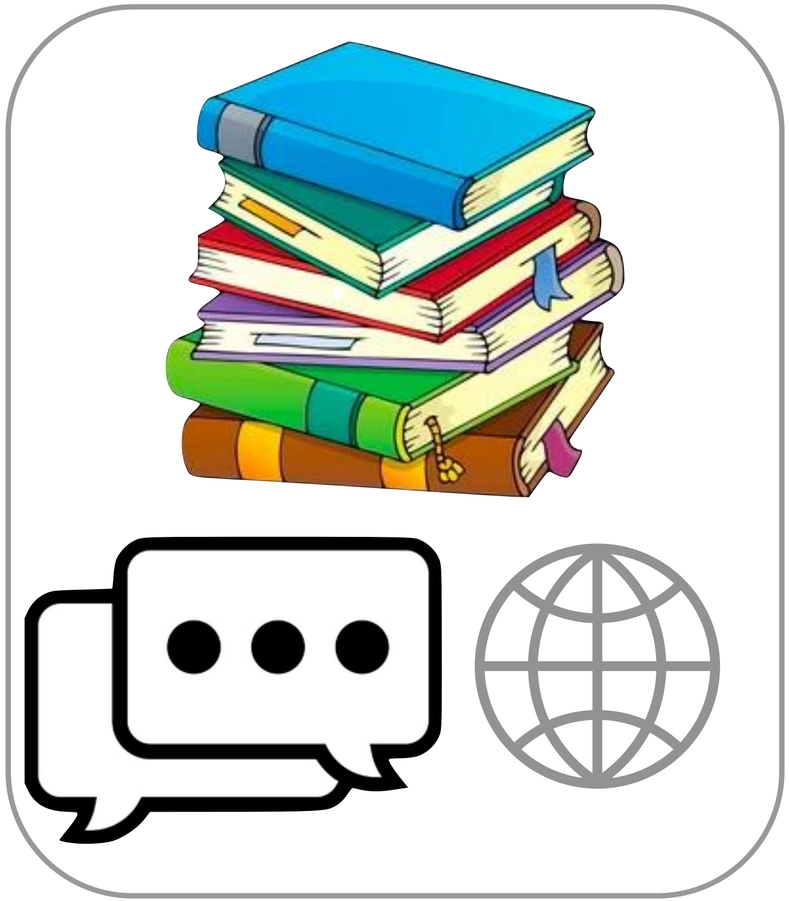
2026 Spring

Slides mostly adapted from Dan Jurafsky, some from Mohit Iyyer

Announcements

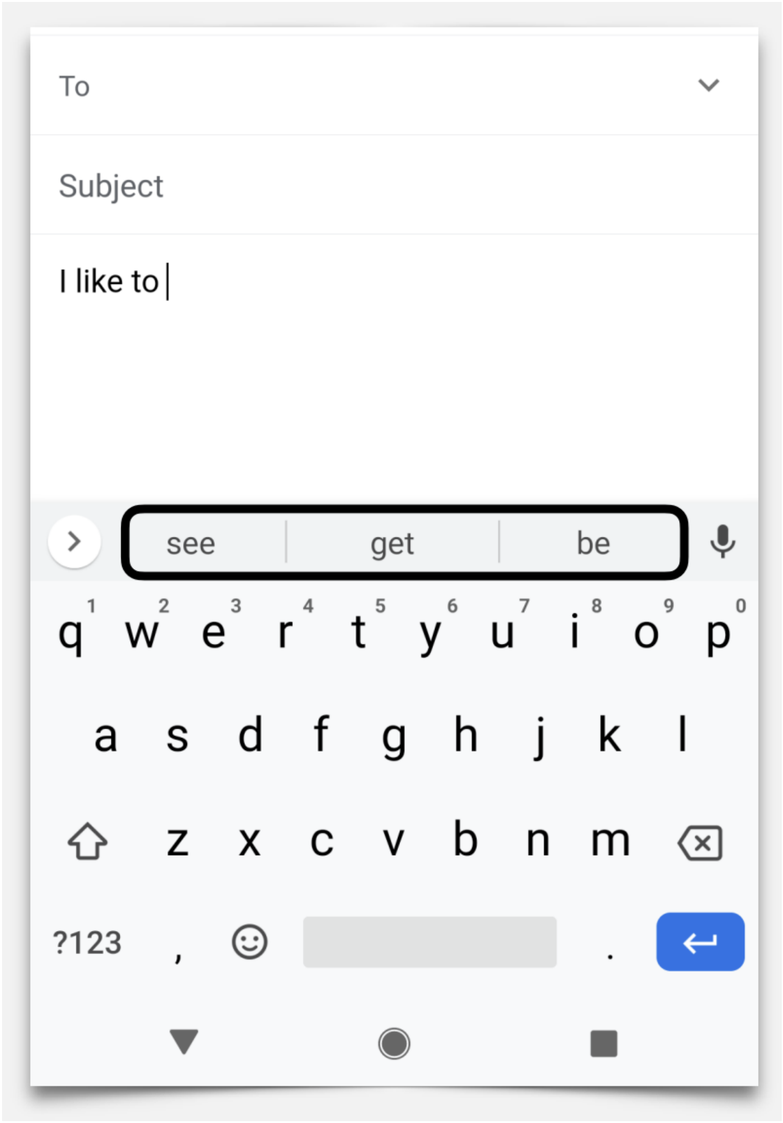
Announcements + Logistics

- HW1 is due by Feb 4, 11:59 PM PT
- Project pitch on Jan 26 → find your project team ASAP! (if not individual project)
  - List of suggested project ideas from my lab: [LINK]
  - Project team finalized by end of Jan → submit your team by Feb 2 and no more adjustment!
- Project proposal is due by Feb 11, 11:59 PM PT
  - Please use Slack to find teammates
- Classes will not be recorded, but under extreme circumstances, we might allow folks to join over zoom
- Sharing slides after class…



Lots and lots of text data

2-gram LM probabilities



Larger Example: Berkeley Restaurant Project (BRP)

- can you tell me about any good cantonese restaurants close by
- mid priced thai food is what i'm looking for
- tell me about chez panisse
- can you give me a listing of the kinds of food that are available
- i'm looking for a good place to eat breakfast
- when is caffe venezia open during the day

Total: 9222 similar sentences

BRP: Raw Counts

Out of 9222 sentences

Unigrams

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

Bigrams

Next Word

History

	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

BRP: Bigram Probabilities

Bigram Probabilities: Raw bigram counts normalized by unigram counts

$$P(w_i|w_{i-1}) = \frac{c(w_{i-1}, w_i)}{c(w_{i-1})}$$

w_i

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

Perplexity

The best language model is one that best predicts an unseen test set

- Gives the highest $P(sentence)$

Perplexity is the inverse probability of the test set, normalized by the number of words

$$PPL(\mathbf{w}) = P(w_1 w_2 \ldots w_N)^{-\frac{1}{N}}$$

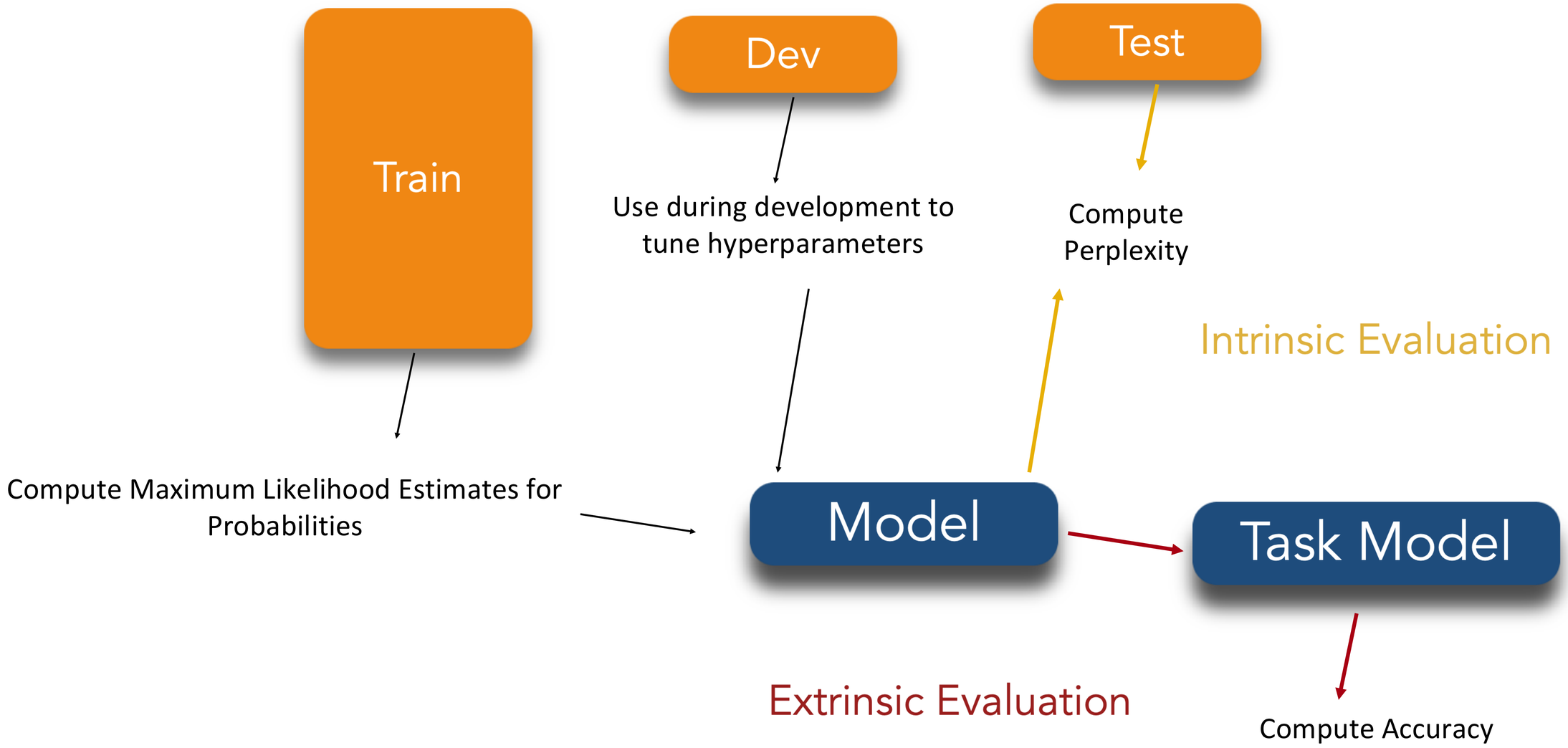
Lower perplexity = better model!

Training 38 million words, test 1.5 million words, from the Wall Street Journal

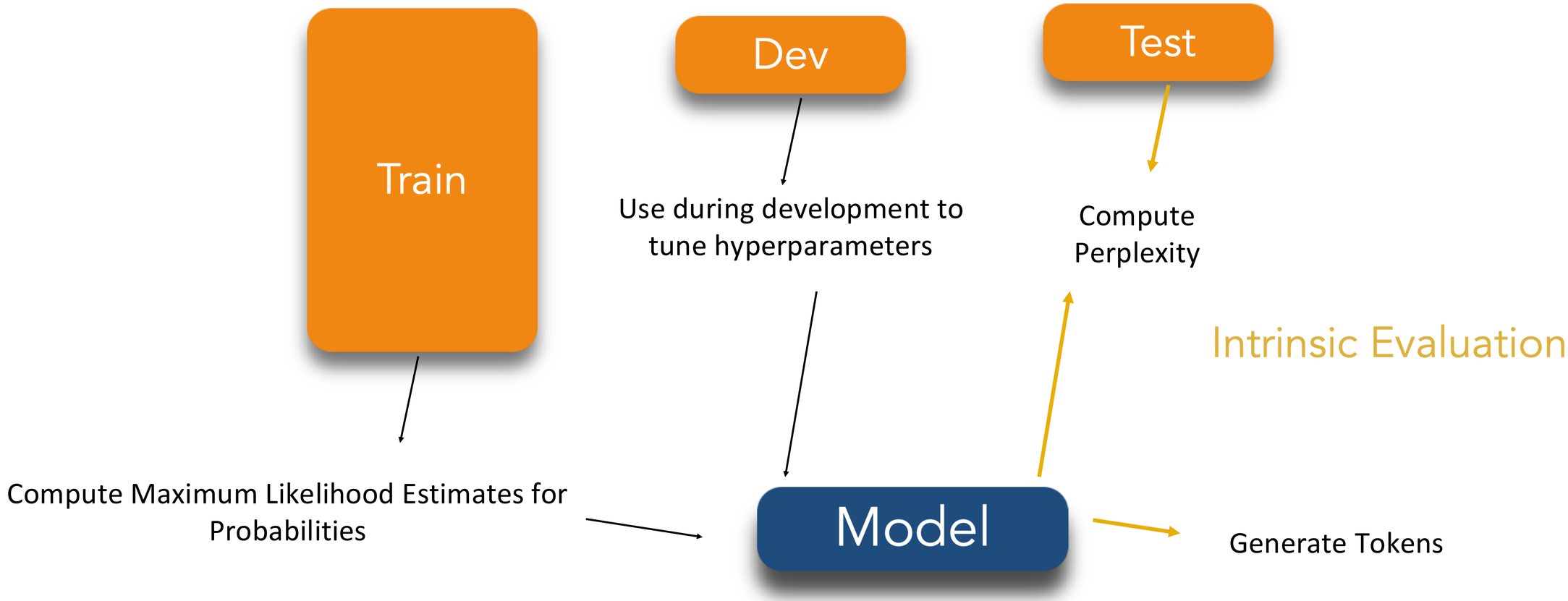
N-gram Order	Unigram	Bigram	Trigram
Perplexity	962	170	109

What are the two things that might affect perplexity?

Language Model Development



Language Model Development



Shakespearean n-grams

1 gram	–To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have –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.

The WSJ is no Shakespeare!

1 gram	Months the my and issue of year foreign new exchange's september were recession exchange new endorsed a acquire to six executives
2 gram	Last December through the way to preserve the Hudson corporation N. B. E. C. Taylor would seem to complete the major central planners one point five percent of U. S. E. has already old M. X. corporation of living on information such as more frequently fishing to keep her
3 gram	They also point to ninety nine point six billion dollars from two hundred four oh six three percent of the rates of interest stores as Mexico and Brazil on market conditions

So why not just sample from very high order n-gram models? Do we even need ChatGPT?

Can only produce n-grams from training data!

Shakespearean corpus cannot produce WSJ vocabulary and vice versa

1 gram	Months the my and issue of year foreign new exchange's september were recession exchange new endorsed a acquire to six executives
2 gram	Last December through the way to preserve the Hudson corporation N. B. E. C. Taylor would seem to complete the major central planners one point five percent of U. S. E. has already old M. X. corporation of living on information such as more frequently fishing to keep her
3 gram	They also point to ninety nine point six billion dollars from two hundred four oh six three percent of the rates of interest stores as Mexico and Brazil on market conditions

The successes we are seeing here are due to a phenomena commonly known as overfitting

Overfitting bad!

N-grams only work well for word prediction if the test corpus looks like the training corpus

- In real life, it often doesn't
- We need to train robust models that generalize!
  - Technical terms for "doing well on the test data" or "doing well on any test data"

- One kind of generalization: **Zeros**!
  - Things that don't ever occur in the training set
    - But occur in the test set

N-gram models: Common Issues that need handling

Token Type Vocabulary

At test time, we might encounter:

- Token never seen in context (i.e. n-gram with 0 frequency)
- Token never seen (unigram with 0 frequency)
- More severe!
  - Problem: Many words like "***Petrichor***" won't appear in most training sets!
  - These are known as OOV for "out of vocabulary", or unknown tokens

Missing n-grams

Training set:

… denied the allegations

… denied the reports

… denied the claims

… denied the request

Test set

… denied the offer

… denied the loan

$$P(offer|denied\ the) = 0$$

will assign 0 probability to the test set!

And hence we cannot compute perplexity

- No one can divide by 0!

What happens to perplexity??

Missing Unigrams: the <UNK> token

One way to handle OOV tokens is by adding a pseudo-word called <UNK>

Closed Vocabulary: Only allow a list of predetermined tokens, everything else (in the training data) is <UNK>

Closed Vocabulary

We can replace all words that occur fewer than n times in the training set, where n is some small number, by <UNK> and re-estimate the counts and probabilities

Open Vocabulary

When not done carefully, may lead to artificially lower perplexity


Smoothing

Intuition for Smoothing

I like to **eat** cake but I want to **eat** pizza right now. Mary told her brother to **eat** pizza too.

→

P(next word = *pizza* | previous word = *eat*) = 2/3
P(next word = *cake* | previous word = *eat*) = 1/3
All other next words = 0 probability

- All other vocabulary tokens getting 0 probability just doesn't seem right. We want to assign some probability to other words
- We want to smooth the distribution from our counts

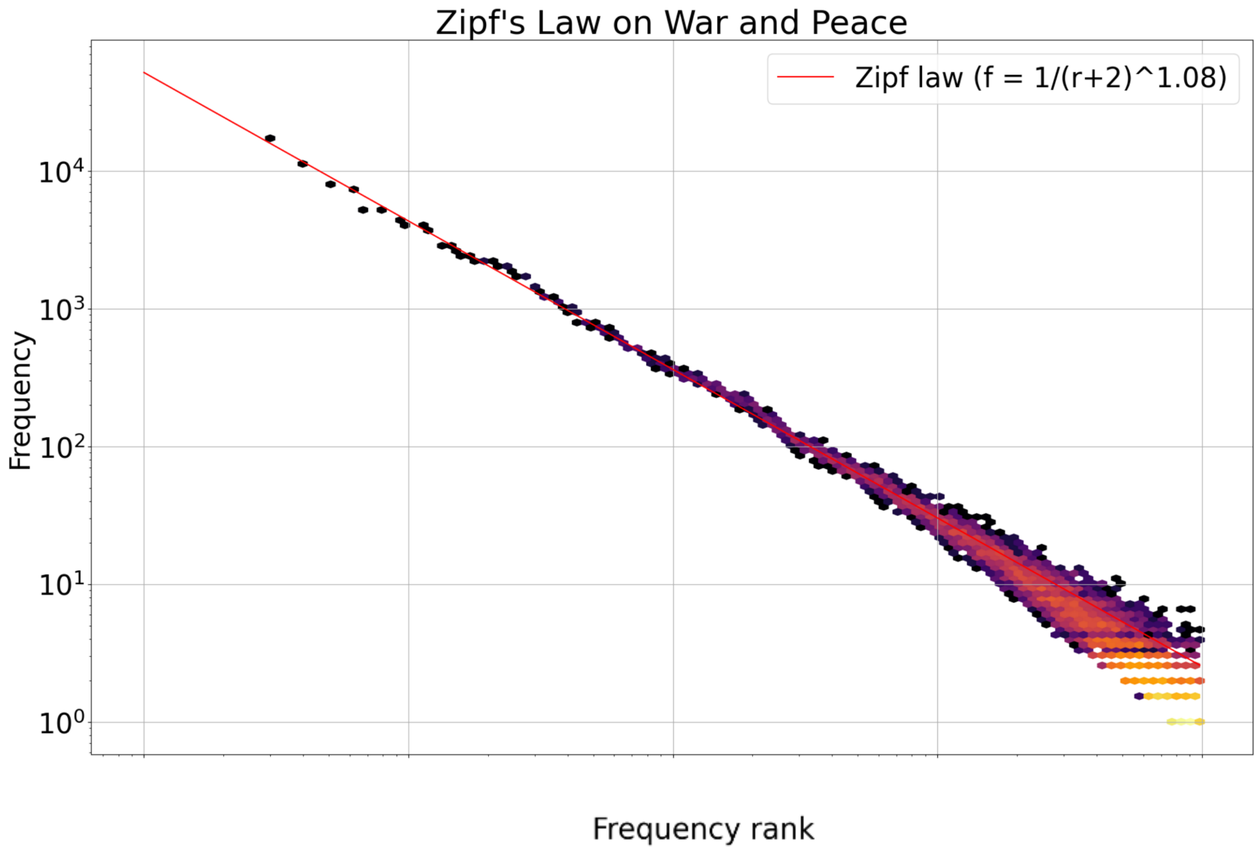
What does a count distribution look like?

Zipf's Law

The distribution over words resembles that of a power law:

- there will be a few words that are very frequent, and a long tail of words that are rare
- $freq_w(r) \approx r^{-s}$

NLP algorithms must be especially robust to observations that do not occur or rarely occur in the training data



Zipf, G. K. (1949). Human behavior and the principle of least effort.

Smoothing ~ Massaging Probability Masses

When we have sparse statistics: $Count(w|denied\ the)$

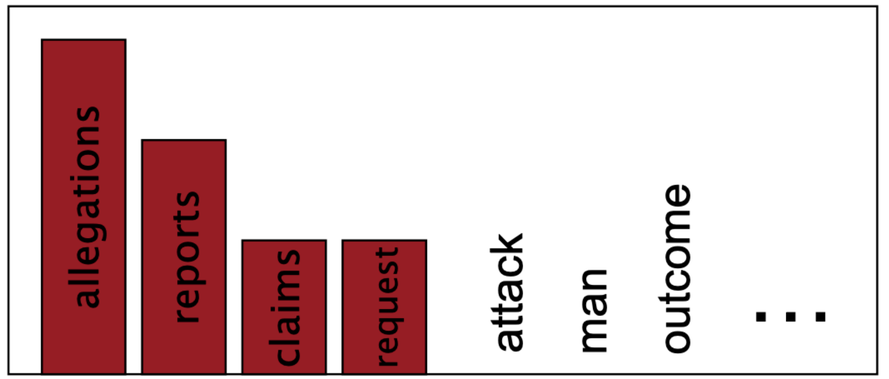
3 allegations

2 reports

1 claims

1 request

7 total



Steal probability mass to generalize better: $Count(w|denied\ the)$

2.5 allegations

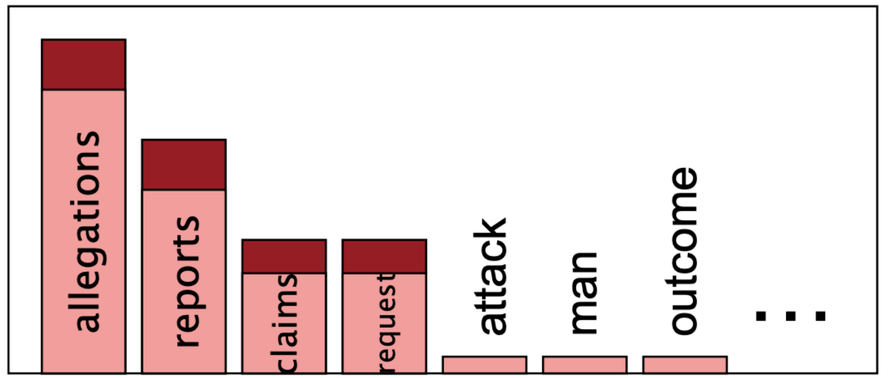
1.5 reports

0.5 claims

0.5 request

2 other (or, **<UNK>**)

7 total



Add-One Estimation

MLE estimate

$$P_{MLE}(w_i) = \frac{c(w_i)}{\sum_w c(w)}$$

Pretend we saw each word one more time than we did

Just add one to all the counts!

Laplace smoothing

75 year old method!

All the counts that used to be zero will now have a count of 1…

Add-1 estimate

$$P_{Add-1}(w_i) = \frac{c(w_i)+1}{\sum_w (c(w)+1)} = \frac{c(w_i)+1}{V + \sum_w c(w)}$$

What happens to our P values if we don't increase the denominator?

Add-1 Estimation Bigrams

MLE estimate

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

Pretend we saw each bigram one more time than we did

Add-1 estimate

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

What does this do to the unigram counts?

Keep the same denominator as before and reconstruct bigram counts

$$= \frac{c^*(w_{i-1}w_i)}{c(w_{i-1})}$$

Recall: BRP Corpus

- can you tell me about any good cantonese restaurants close by
- mid priced thai food is what i'm looking for
- tell me about chez panisse
- can you give me a listing of the kinds of food that are available
- i'm looking for a good place to eat breakfast
- when is caffe venezia open during the day

Unigrams

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

Bigrams

w_i

w_{i-1}	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

Laplace-smoothed bigram counts

Just add one to all the counts!

w_i

w_{i-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

Laplace-smoothed bigram probabilities

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

	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

Reconstituted Counts

$$c * (w_{i-1}w_i) = \frac{[c(w_{i-1}w_i) + 1]c(w_{i-1})}{c(w_{i-1}) + V}$$

	i	want	to	eat	chinese	food	lunch	spend
i	3.8	527	0.64	6.4	0.64	0.64	0.64	1.9
want	1.2	0.39	238	0.78	2.7	2.7	2.3	0.78
to	1.9	0.63	3.1	430	1.9	0.63	4.4	133
eat	0.34	0.34	1	0.34	5.8	1	15	0.34
chinese	0.2	0.098	0.098	0.098	0.098	8.2	0.2	0.098
food	6.9	0.43	6.9	0.43	0.86	2.2	0.43	0.43
lunch	0.57	0.19	0.19	0.19	0.19	0.38	0.19	0.19
spend	0.32	0.16	0.32	0.16	0.16	0.16	0.16	0.16

Compare with raw bigram counts

Original, Raw

	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

Big change to the counts!

Reconstructed

	i	want	to	eat	chinese	food	lunch	spend
i	3.8	527	0.64	6.4	0.64	0.64	0.64	1.9
want	1.2	0.39	238	0.78	2.7	2.7	2.3	0.78
to	1.9	0.63	3.1	430	1.9	0.63	4.4	133
eat	0.34	0.34	1	0.34	5.8	1	15	0.34
chinese	0.2	0.098	0.098	0.098	0.098	8.2	0.2	0.098
food	6.9	0.43	6.9	0.43	0.86	2.2	0.43	0.43
lunch	0.57	0.19	0.19	0.19	0.19	0.38	0.19	0.19
spend	0.32	0.16	0.32	0.16	0.16	0.16	0.16	0.16

Perhaps 1 is too much, add a fraction?

Add-k smoothing

Add-1 Estimation: Last thoughts

So add-1 isn't used for n-grams, being something of a blunt instrument

- One-size-fits-all

Add-1 is used to smooth other NLP models though…

- For text classification
- In domains where the number of zeros isn't so huge

Interpolation

Perhaps use some pre-existing evidence

- Condition on less context for contexts you haven't learned much about

Interpolation

- mix unigram, bigram, trigram probabilities for a trigram LM
- mix n-gram, (n-1)gram, … unigram probabilities for an n-gram LM

Interpolation works better than Add-1 / Laplace

Linear Interpolation

Simple Interpolation

$$\hat{P}(w_i|w_{i-2}w_{i-1}) = \lambda_1 P(w_i) + \lambda_2 P(w_i|w_{i-1}) + \lambda_3 P(w_i|w_{i-2}w_{i-1})$$

$$\sum_k \lambda_k = 1$$

Hyperparameters!

Context-Conditional Interpolation

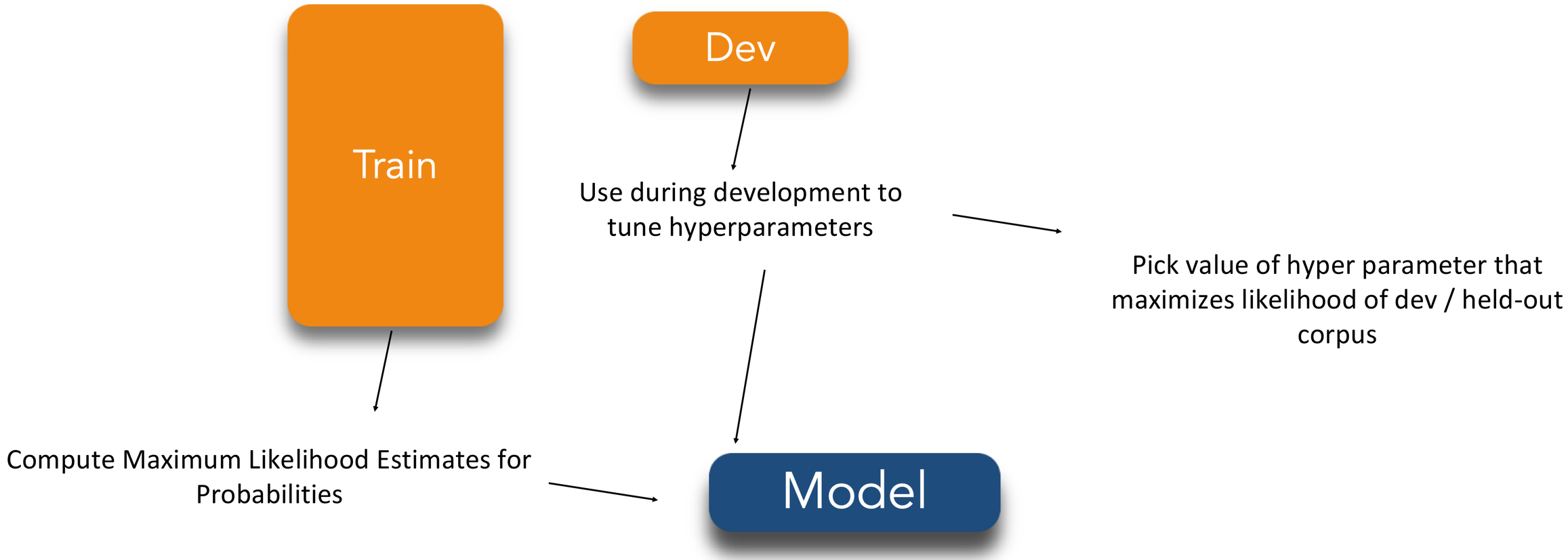
$$\hat{P}(w_i|w_{i-2}w_{i-1}) = \lambda_3(w_{i-2}^{i-1}) P(w_i|w_{i-2}w_{i-1}) + \lambda_2(w_{i-2}^{i-1}) P(w_i|w_{i-1}) + \lambda_1(w_{i-2}^{i-1}) P(w_i)$$

Reconstituted Counts

Different for every unique context

How to set the λs?

Language Model Development



How to set the λs?

Choose λs to maximize the probability of held-out data:

- Fix the n-gram probabilities (on the training data)
- Then search for λs that give largest probability to held-out set:

$$logP(w_1 \ldots w_n | M(\lambda_1 \ldots \lambda_k)) = \sum_i logP_{M(\lambda_1 \ldots \lambda_k)}(w_i | w_{i-1})$$

Backoff and Discounting

Backoff

- use trigram if you have good evidence,
- otherwise bigram, otherwise unigram

Still need a correct probability distribution!

- Discount higher order n-grams by d to save some probability mass for the lower order n-grams
- need a function α to distribute this probability mass to the lower order n-grams

Stupid Backoff

No discounting, just use relative frequencies

Don't care about a valid language model

Not a probability distribution (usually denoted as P)

$$S(w_i \mid w_{i-k+1}^{i-1}) = \begin{cases} \dfrac{\text{count}(w_{i-k+1}^{i})}{\text{count}(w_{i-k+1}^{i-1})} & \text{if } \text{count}(w_{i-k+1}^{i}) > 0 \\ 0.4\, S(w_i \mid w_{i-k+2}^{i-1}) & \text{otherwise} \end{cases}$$

$$S(w_i) = \frac{\text{count}(w_i)}{N}$$

Hyperparameter!

Brants *et al.* 2007

Absolute discounting: just subtract a little from each count

Consider an n-gram with count of 4. Suppose we wanted to subtract a little from this to save probability mass for the zeros

- How much to subtract ?

Church and Gale (1991)'s clever idea

- Divide up 22 million words of AP Newswire
- Training and held-out set
  - for each bigram in the training set
    - see the actual (averaged) count in the held-out set!

It sure looks like $c^* \approx (c - .75)$

Bigram count in training	Bigram count in heldout set
0	.0000270
1	0.448
2	1.25
3	2.24
4	3.23
5	4.21
6	5.23
7	6.21
8	7.21
9	8.26

Absolute Discounting Interpolation

Save ourselves some time and just subtract 0.75 (or some d, such that $0 \leq d < 1$)!

- Maybe keeping a couple extra values of d (for counts 1 and 2 and 3)

$$P_{AbsoluteDiscounting}(w_i|w_{i-1}) = \frac{c(w_{i-1}w_i) - d}{\sum_w c(w_{i-1}w)} + \lambda(w_{i-1})P(w_i)$$

Discounted Bigram

But should we really just use the regular unigram $P(w_i)$?

Smoothing ~ Massaging Probability Masses

When we have sparse statistics: $Count(w|denied\ the)$

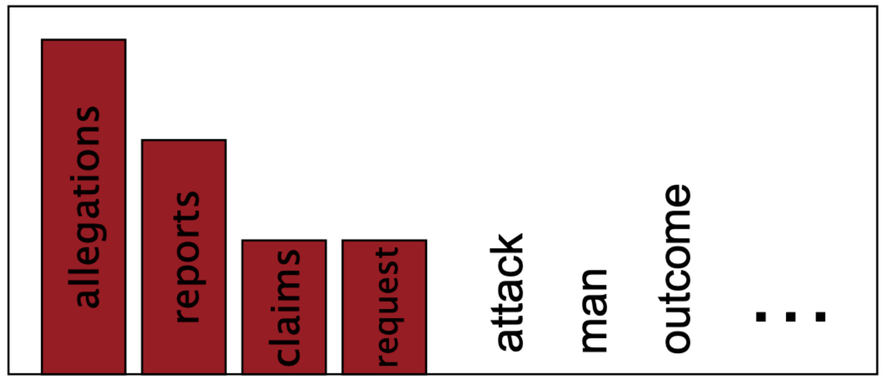
3 allegations

2 reports

1 claims

1 request

7 total



Steal probability mass to generalize better: $Count(w|denied\ the)$

2.5 allegations

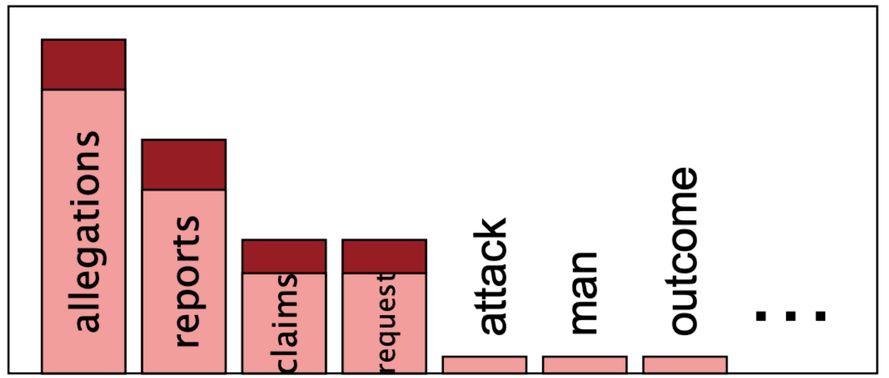
1.5 reports

0.5 claims

0.5 request

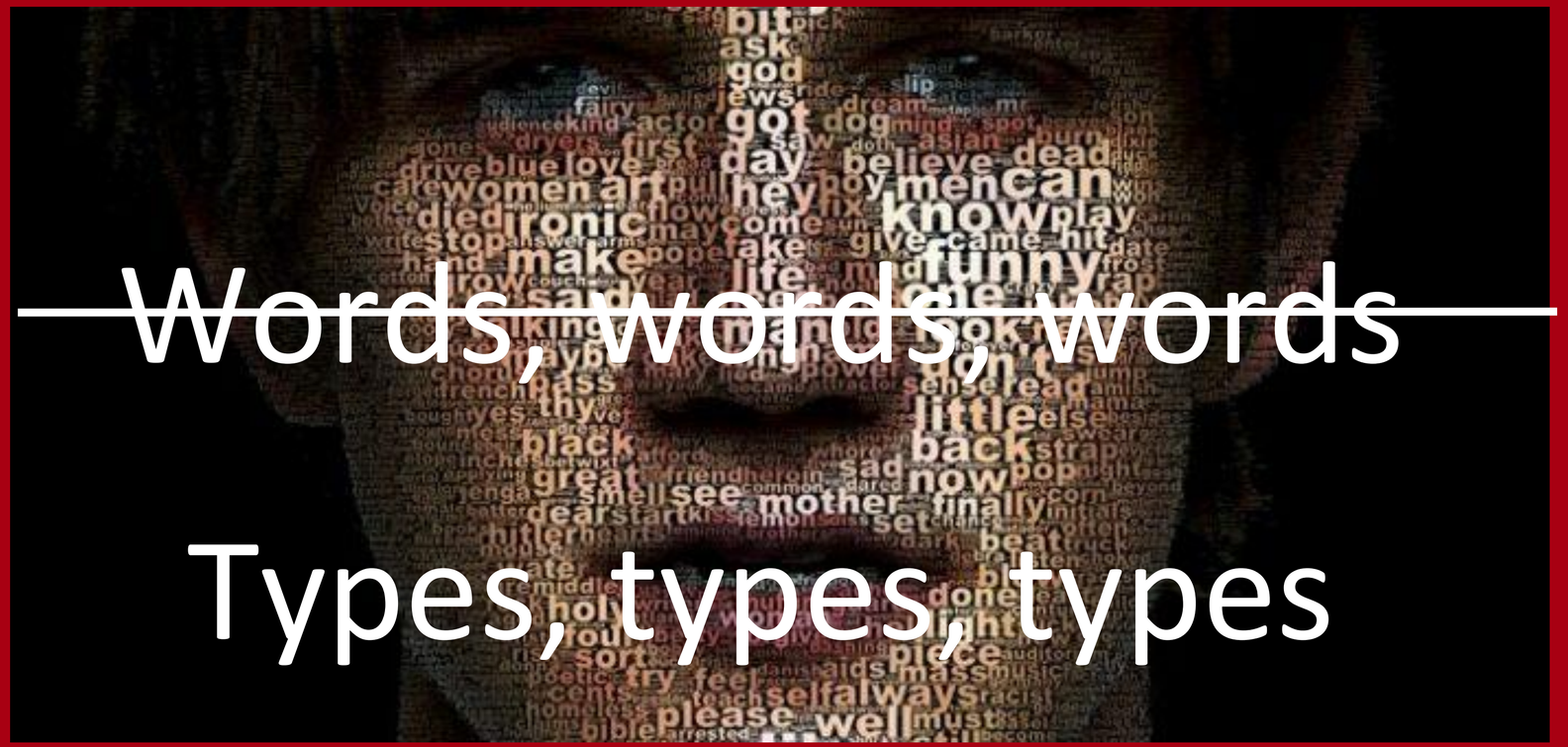
2 other (or, **<UNK>**)

7 total



Early Neural Language Models

This class: Word Embeddings — the most important component of a neural LM


~~Words, words, words~~
Types, types, types

Word: Types & Tokens

"The cat chased the cat."

- Word types: *the*, *cat*, *chased* → **3 types**
- Word tokens: *the*, *cat*, *chased*, *the*, *cat* → **5 tokens**

So:

- **Word** = the linguistic item
- **Type** = the category representing identical items

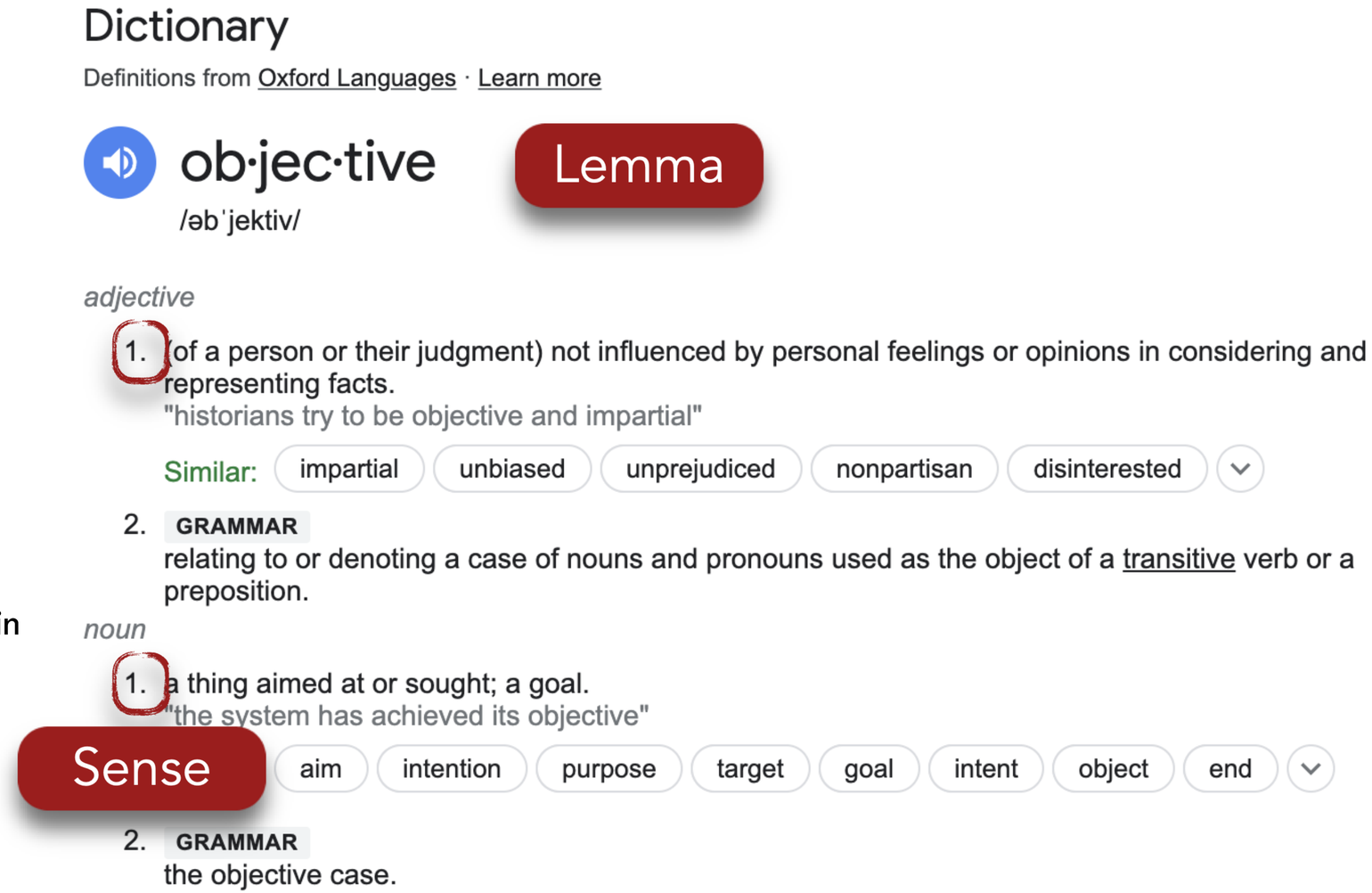
What do words mean?

A sense or "concept" is the ***meaning*** component of a word

Lemmas

- Canonical form
- For example,
  break, breaks, broke, broken and breaking all share the lemma "break"

Can be polysemous (have multiple senses)



Synonymy

Synonyms: words that have the same ***meaning*** in some or all contexts

- couch / sofa
- automobile / car
- water / H_20

Is perfect synonymy possible?

- Even if many aspects of meaning are identical
- Still may differ based on politeness, slang, register, genre, etc.
  - e.g. cannot use H_20 in a surfing guide!

Similarity

Words with similar meanings. Not synonyms, but sharing some element of meaning

Human assessment of word similarity

Simlex-999 dataset (Hill et al., 2015)

word1	word2	similarity
vanish	disappear	9.8
behave	obey	7.3
belief	impression	5.95
muscle	bone	3.65
modest	flexible	0.98
hole	agreement	0.3

Not to be confused with word association / relatedness:

- couch / sofa vs. couch / pillow

Antonymy

Senses that are opposites with respect to only one feature of meaning

Otherwise, they are very similar!

- e.g. dark/light, short/long, fast/slow, rise/fall, hot/cold, up/down, in/out

More formally: antonyms can

- define a binary opposition or be at opposite ends of a scale
    - e.g. long/short, fast/slow
- be reversives: denote opposing processes
    - rise/fall, up/down

WordNet

- WordNet® is a large lexical database of English
- Nouns, verbs, adjectives and adverbs are grouped into sets of synonyms (synsets), each expressing a distinct concept
- Relations between synsets:
  - Super-subordinate relations (hyperonymy, hyponymy or ISA relation)
    - an armchair is a kind of chair, chair is a kind of furniture
  - Meronymy (part-of)
    - chair has legs
  - Antonymy

n-grams and Semantics

- n-grams do not represent meaning well
  - Do not tell us that the word “rancor” is close in meaning to the word “hatred”
  - Or that “Rise” and “Fall” have opposite meanings
  - Let alone more complex is-a or part-of relations

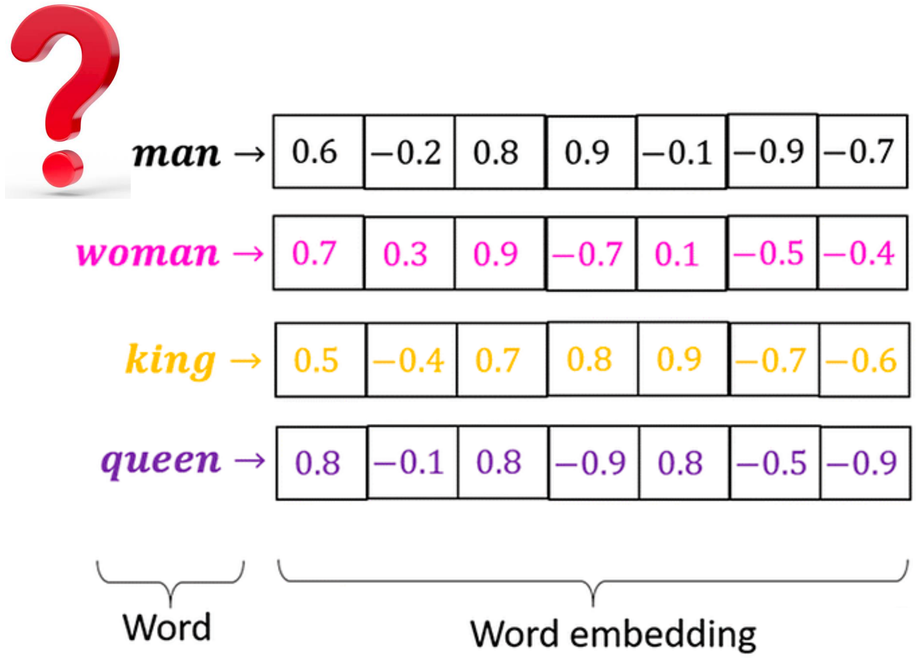
n-grams and Semantics

- n-grams do not represent meaning well
  - Do not tell us that the word “rancor” is close in meaning to the word “hatred”
  - Or that “Rise” and “Fall” have opposite meanings
  - Let alone more complex is-a or part-of relations
- Discrete representations of meaning!
- Next: feature representations which are continuous

Words as Vectors

In NLP, we commonly represent word types with vectors!

- But why?
- Very useful in capturing similarity between words, and other forms of lexical semantics (e.g. synonymy, hypernyms, antonymy)
- Computing the similarity between two words (or phrases, or documents) is extremely useful for many NLP tasks
  - Q: How tall is Mount Everest?
  - A: The official height of Mount Everest is 29029 ft



- Similarity for plagiarism detection
- Word similarity can lead to sentence and document similarity

enough scale for companies to make profit from it. In order to be competitive with new technologies, the challenge of today's large companies is to create new business within their business (Garvin & Levesque, 2006). Furthermore, the two researchers emphasize a switch from downsizing and cost cutting to the creation, development and assistance of innovative new businesses. For existing companies the implementation of corporate entrepreneurship, in order to develop innovative businesses, is risky. Are the three types of entrepreneurship linked together over time? How long does it take to change behavior of the firm as a whole? If the five attributes are created, do all grow together equally, or do some grow faster and earlier than others? How do the importance and intensities of the attributes differ both absolutely and relatively in each type? These are the questions that a longitudinal study such as this can attempt to answer to shed light on the nature of organizations' adjustments to hostile environments. According to Garvin and Levesque (2006) implementing new ventures face several barriers, and can only be successful if a blend of old and new organizational traits is done. To achieve a blend of old and new, an organization needs to rely on employee innovative behavior in order to succeed in dynamic business environments (Yuan & Woodman, 2010).

The downside is potentially very damaging to a startup's lifespan: if a startup lands a pilot or POC with the corporation running the accelerator, they have very little bartering power or time to find other partners to test their solution with. The transition from manufacturing economy to service economy has led to a shift in business agenda from

1 Original source
onlinelibrary.wiley.com/stor...

...all grow together equally, or do some grow faster and earlier than others? These are the questions that a longitudinal study such as this can attempt to answer to shed light on the nature of organiza-tions' adjustments to hostile environments. Of the many ways to adjust, two stand out at

- Visualizing semantic change over time

- New words: `dank`, `cheugy`, `rizz`, `shook`, `situationship`

~30 million books, 1850-1990, Google Books data

Discrete representations

- So far, words are regarded as atomic symbols: hotel, conference, walk
- Missed nuances: synonyms, antonyms
- WordNet taxonomy

Words as Vectors

"You shall know a word by the company it keeps."

- Firth (1957)

Word Meaning via Language Use

- The meaning of a word can be given by its distribution in language usage:
  - One way to define "usage": words are defined by their environments
    - Neighboring words or grammatical environments
- Intuitions: Zellig Harris (1954):
  - "oculist and eye-doctor … occur in almost the same environments"
  - "If A and B have almost identical environments we say that they are synonyms."

A bottle of tesgüino is on the table
Everybody likes tesgüino
Tesgüino makes you drunk
We make tesgüino out of corn.

Two words are similar if they have similar word contexts

Word Meanings via Language Properties

- Meaning of a word can be determined by some properties of the word
- Point in space (Osgood et al., 1957)
- Example Properties: Affective Dimensions

	Word	Score		Word	Score
Valence	love	1.000		toxic	0.008
	happy	1.000		nightmare	0.005
Arousal	elated	0.960		mellow	0.069
	frenzy	0.965		napping	0.046
Dominance	powerful	0.991		weak	0.045
	leadership	0.983		empty	0.081

NRC VAD Lexicon (Mohammad 2018)

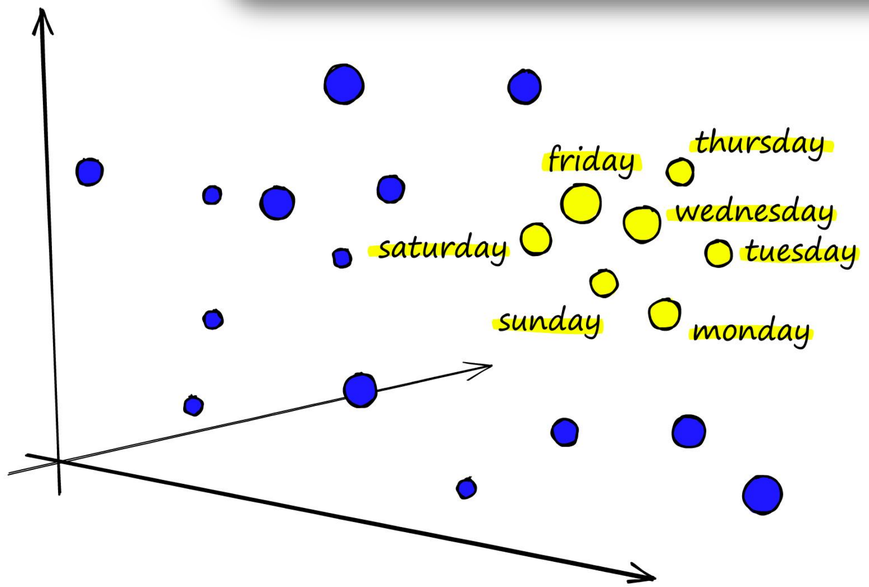
Defining meaning as a point in space based on distribution

- Each word = a vector
  - not just "good" or "word#45"
- Similar words are "nearby in semantic space"
- We build this space automatically by seeing which words are nearby in text
- 2-D representation

Word Embeddings

Vector Semantics

- Represent a word as a point in a multidimensional semantic space
  - Space itself constructed from distribution of word neighbors
- Called an "embedding" because it's embedded into a space
- Fine-grained model of meaning for similarity



Every modern NLP algorithm uses embeddings as the representation of word meaning

Image Credit: Pinecone

Announcements + Logistics

- HW1 is due by Feb 4, 11:59 PM PT
- Project pitch on Jan 26 → find your project team ASAP! (if not individual project)
  - List of suggested project ideas from my lab: [LINK]
  - Project team finalized by end of Jan → submit your team by Feb 2 and no more adjustment!
- Project proposal is due by Feb 11, 11:59 PM PT
  - Please use Slack to find teammates
- Classes will not be recorded, but under extreme circumstances, we might allow folks to join over zoom
- Sharing slides after class…