

Probability and Statistics in NLP

Niranjana Balasubramanian

Jan 28th, 2016





Mechanism for communicating thoughts, ideas, emotions, and more.

What is NLP?

Building natural language interfaces to computers (devices more generally).

Building intelligent machines requires knowledge, a large portion of which is textual.

Building tools to understand how humans learn, use, and modify language.

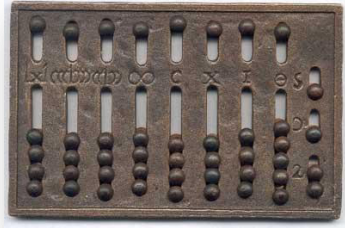
- Help **linguists** test theories about language.
- Help **cognitive scientists** understand how children acquire language.
- Help **sociologists** and **psychologists** model human behavior from language.

Natural Language Interfaces to Computing



A brief history of computing

2000 BC



1800-1930



1950-ish



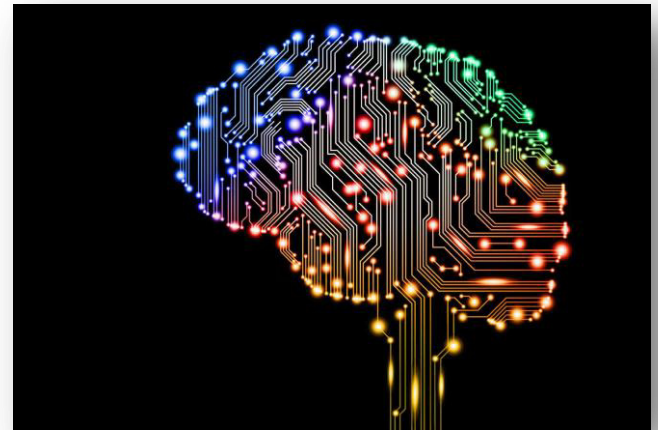
1980s



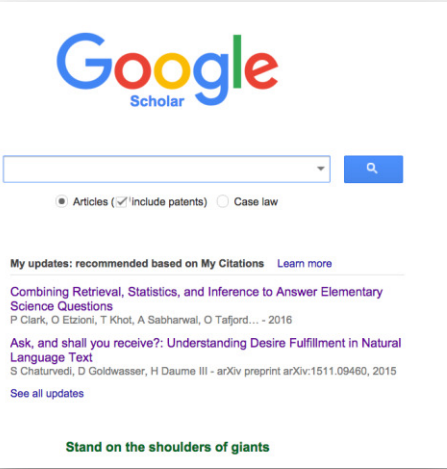
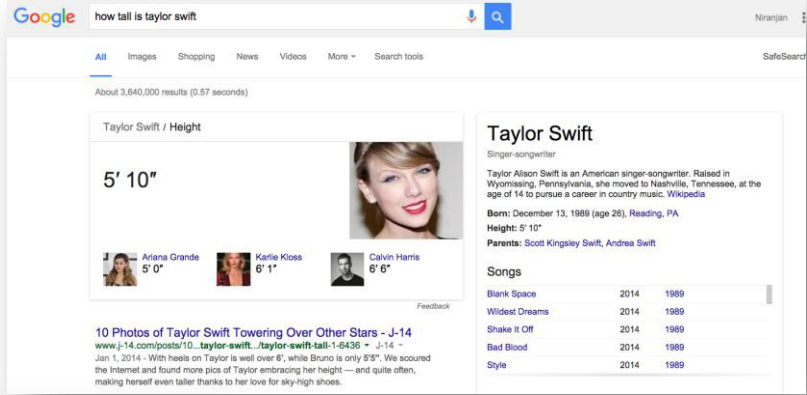
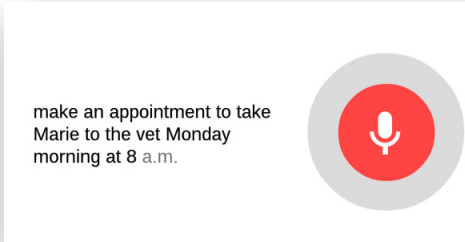
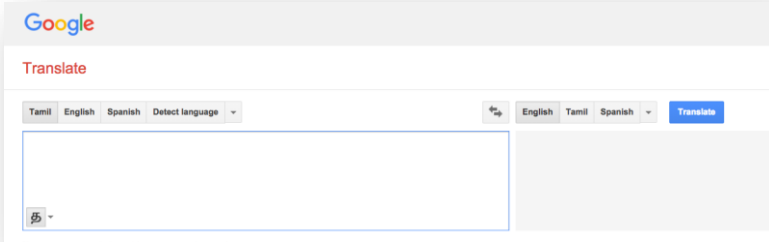
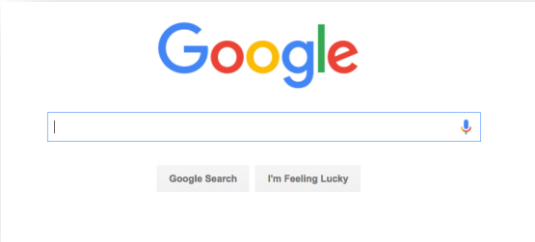
2016 – We need to be able to talk to our devices!



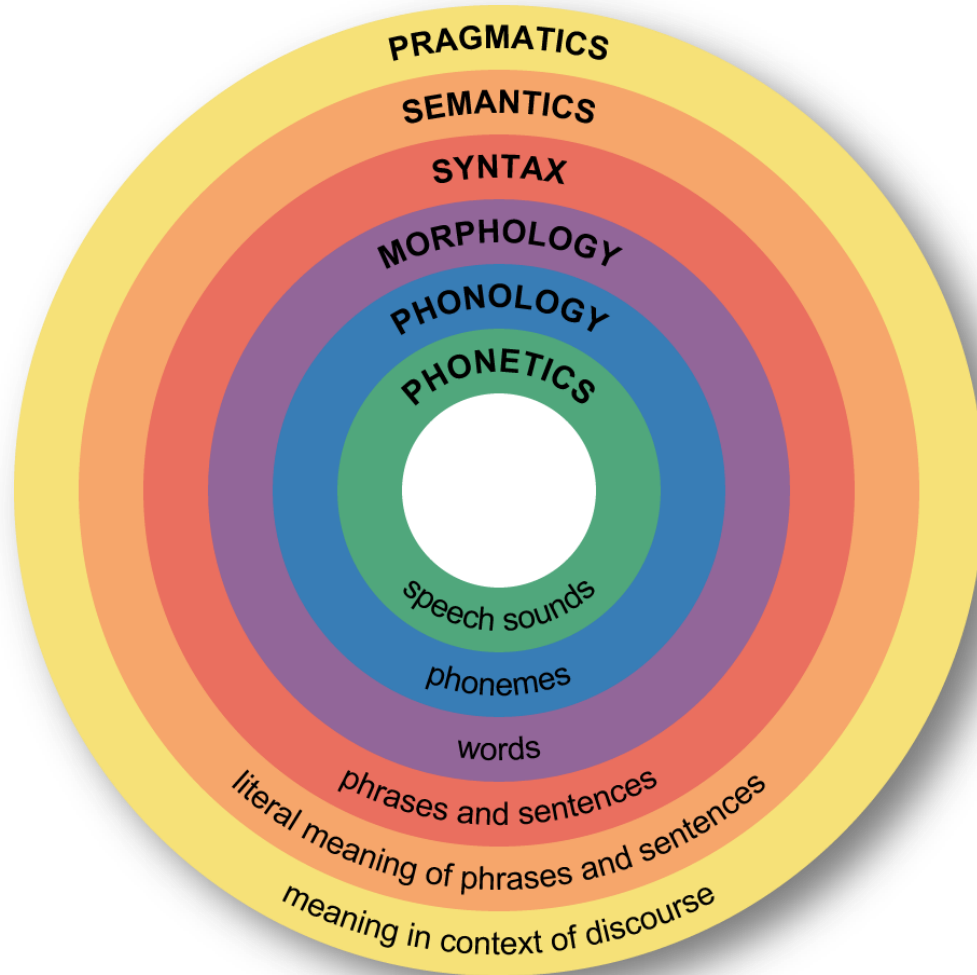
Artificial Intelligence needs our knowledge!



NLP Applications



What aspects of language do we need to worry about?



Why is NLP hard?

- Ambiguity
- Meaning is context dependent
- Requires background knowledge

Ambiguity (and consequently, uncertainty).

I saw a man with a telescope.

I saw a bird flying over a mountain.

Exists in all kinds of NLP tasks.

Ambiguity compounds explosively (Catalan numbers)

e.g., I saw a man with a telescope on a hill

Context Dependence and Background Knowledge

Rachel **ran** to the **bank**.

vs.

Rachel **swam** to the **bank**.

John drank some wine at the table. **It** was red.

vs.

John drank some wine at the table. **It** was wobbly.

Language Modeling

Niranjn Balasubramanian

Slide Credits:

Chris Manning, Dan Jurafsky, Mausam



Today's Plan

- What is a language model?
- Basic methods for estimating language models from text.
- How does one evaluate how good a model is?

What is language modeling?

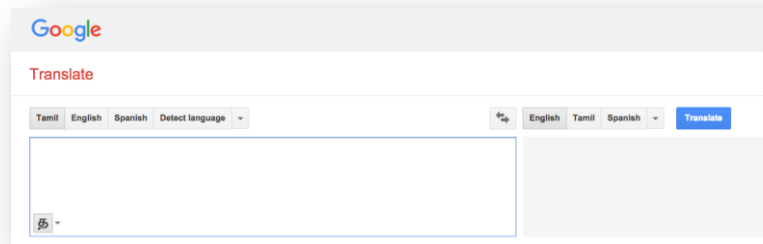
- Task of building a predictive model of language.
- A language model is used to predict two types of quantities.
 1. Probability of observing a sequence of words from a language.
e.g., $\Pr(\text{Colorless green ideas sleep furiously}) = ?$
 2. Probability of observing a word having observed a sequence.
e.g. $\Pr(\text{furiously} \mid \text{Colorless green ideas}) = ?$

Why model language?

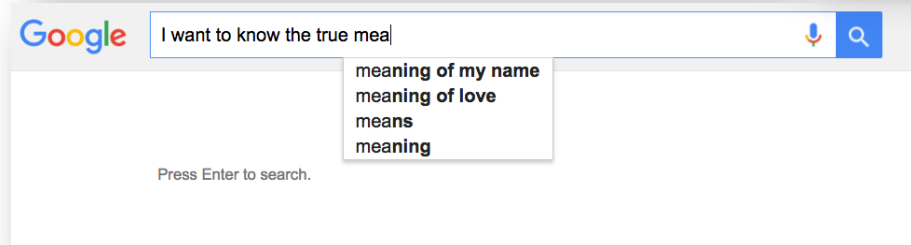
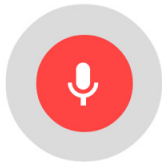
- The probability of observing a sequence is a measure of *goodness*.
- If a system outputs some piece of text, I can assess its goodness.
 - Many NLP applications output text.

- Example Applications

- Speech recognition
- OCR
- Spelling correction
- Machine translation
- Authorship detection



make an appointment to take Marie to the vet Monday morning at 8 a.m.



Are roti and chapati the same?

Chrome File Edit View History Bookmarks People Window Help

are roti and chapati the same

are roti and chapati the same

Google


are roti and chapati the same

Niranjan

SafeSearch on

About 123,000 results (0.39 seconds)

Roti is a different name for **Chapati**. Different people call it by different names. But the method to prepare it is the **same**. Phulka is smaller round prepared from wheat flour but is rolled first, then one side is cooked on a pan(Tawa) and the other side directly on heat.



What is the difference between paratha, roti, chapathi and ...
<https://www.quora.com/.../What-is-the-difference-between-paratha-ro...> Quora

Feedback

What is the difference between a roti, naan, paratha and ...
<https://www.quora.com/What-is-the-difference-between-a-roti-naan-...> Quora

PARATHA- also known as parantha, parauntha, is an Unleavened flat bread made from same flour as that of Roti and filled with different fillings made using potato, Fenugreek, Radish, cheese,spices etc.

What is the difference between paratha, roti, chapathi and ...
<https://www.quora.com/.../What-is-the-difference-between-paratha-ro...> Quora

Roti is a different name for Chapati. Different people call it by different names. But the method to prepare it is the same. Phulka is smaller round prepared from wheat flour but is rolled first, then one side is cooked on a pan(Tawa) and the other side directly on heat.

what is difference between roti, paratha and chapati? | Yahoo ...
<https://answers.yahoo.com/question/index?qid...>

Oct 13, 2008 - Roti and Chapati are one and same. These are about 7" round flat breads made of whole-grain wheat flour without leavening and cooked on a ...

What is the difference between Roti, Chapathi and Parathas ... Oct 24, 2009

ShakespearePlaysPlus.zip language-modeling-...pptx digpath-experimentati...pdf Digital Pathology.zip Show All

bin
spock

61% [E] Thu 9:17 AM

Language Model Corrections

Google

are roti and chapati the same



Language Modeling: Formal Problem Definition

A language model is something that specifies the following two quantities, for all words in the vocabulary (of a language).

1. Probability of a sentence or sequence

$$\Pr(w_1, w_2, \dots, w_n)$$

2. Probability of the next word in a sequence

$$\Pr(w_{k+1} \mid w_1, \dots, w_k)$$

Note on notation:

$\Pr(w_1, w_2, \dots, w_n)$ is short for $\Pr(W_1 = w_1, W_2 = w_2, \dots, W_n = w_n)$
Random variable W_1 taking on value w_1 and so on.

e.g., $\Pr(\text{I, love, fish}) = \Pr(W_1 = \text{I}, W_2 = \text{love}, W_3 = \text{fish})$

How to model language?

- Count! (and normalize).
 - Need some source text – corpora.
- Main Issues

Issue 1: We can generate infinitely many *new* sequences.

e.g., “Colorless green ideas sleep furiously” is not a frequent sequence.
[Thanks to Chomsky, this sequence is now popular.]

Issue 2: We generate new words all the time.

e.g., Truthiness, #letalonethehashtags,

Pr(W): Assumptions are key to modeling.

- We are free to model the probabilities however we want to.
 - Usually means that you have to make assumptions.
- If you make no *independence* assumptions about the sequence, then one way to estimate is the fraction of times you see it.

$$\Pr(w_1, w_2, \dots, w_n) = \#(w_1, w_2, \dots, w_n) / N$$

where N is the total number of sequences.

- [White board] Markov assumption and n-gram definitions.

Issues with Direct Estimation

- How many times would you have seen the particular sentence?

$$\Pr(w_1, w_2, \dots, w_n) = \#(w_1, w_2, \dots, w_n) / N$$

- Estimating from sparse observations is unreliable.
- Also, we don't have a solution for a new sequence.

- Use chain rule to decompose joint into a product of conditionals

$$\Pr(w_1, w_2, \dots, w_n) = \Pr(w_1 | w_2, \dots, w_n) \\ \times \Pr(w_2, \dots, w_n)$$

$$\Pr(w_1, w_2, \dots, w_n) = \Pr(w_1 | w_2, \dots, w_n) \\ \times \Pr(w_2 | w_3, \dots, w_n) \\ \times \Pr(w_3, \dots, w_n)$$

...

$$\Pr(w_1, w_2, \dots, w_n) = ?$$

- Estimating conditional probabilities with long contexts is difficult!
 - Conditioning on 4 or more words itself is hard.

Markov Assumption

- Next event in a sequence depends only on its immediate past (context).

$$\Pr(w_{k+1} | w_{i-k}, w_{i-k+1}, \dots, w_k)$$



- n -grams

- Unigrams $\Pr(w_{k+1} | w_k)$
- Bigrams $\Pr(w_{k+1} | w_{k-1}, w_k)$
- Trigrams $\Pr(w_{k+1} | w_{k-2}, w_{k-1}, w_k)$
- 4-grams $\Pr(w_{k+1} | w_{k-2}, w_{k-2}, w_{k-1}, w_k)$

- Note:

- Other contexts are possible and in many cases preferable.
- Models tend to be more complex.

Unigrams

- Next event in a sequence is *independent* of the past.

$$\Pr(W) = \prod_i \Pr(w_i)$$

- An extreme assumption but can be useful nonetheless.
- Issue
 - Non-sensical phrases or sentences can get high probability.

$\Pr(\text{the a an the a an the an a}) > \Pr(\text{The dog barks})$

Bigrams and higher-order N-grams

- Bi-grams: Next word is dependent on the previous word alone.

$$Pr(W) = \prod_i Pr(w_i | w_{i-1})$$

Widely used.

- N-grams: Next word dependent on the previous n words.

$$Pr(W) = \prod_i Pr(w_i | w_{i-2}, w_{i-1})$$

$$Pr(W) = \prod_i Pr(w_i | w_{i-3}, w_{i-2}, w_{i-1})$$

...

Reliable Estimation vs. Generalization

- We can estimate unigrams quite reliably but they are often not a good model.
- Higher order n-gram require large amounts of data but are better models.
 - However, they have a tendency to overfit the data.
- Example sentences generated from Shakespeare language models:

Unigram Every enter now severally so, let.

Bigram then all sorts, he is trim, captain.

Trigram Indeed the duke; and had a very good friend.

4-gram It cannot be but so.

Shakespeare and Sparsity

Shakespeare's works have about 800K tokens (words) in all with a vocabulary of 30K.

The number of unique bigrams turn out to be around 300K.

What is the total space of possible bigrams?

Sparse!

-- Many bigrams are unseen.

Are these well-defined distributions? Proof for unigrams.

[Work out on board.]

What is a good language model? An ideal perspective!

- To be a bit recursive, a good language model should model the language well.
 - If you ask questions of the model it should provide reasonable answers.
- Well formed English sentences should be more probable.
- Words that the model predicts as the next in a sequence should “fit”
 - Grammatically
 - Semantically
 - Contextually
 - Culturally
- These are too much to ask for, given how mind-numbingly simple these models are!

What is a good language model? An utilitarian perspective.

- Does well on the task we want to use it for.
 - Difficult to do because of the time it takes.
- Want models that *assign* high probabilities to samples from language.
 - Can't use the samples used for estimation. [Why?]

Many choices in modeling: How to pick a language model?

- Machine Learning Paradigm
 - A model is good if it predicts a test set of sentences.
 - Reserve some portion of you data for estimating parameters.
 - Use the remainder for testing your model.
- A good model assigns high probabilities to the test sentences.
 - Probability of each sentence is normalized for length.

$$\prod_{W^{(i)} \in Test} \left(Pr(W^{(i)}) \right)^{-\frac{1}{N_i}}$$

Perplexity

- An alternative that measures how well the test samples are predicted.
- Models that minimize perplexity, also maximize the probability.
 - Take inverse of the probability and apply a log-transform.
 - [Work out on board.]

Perplexity of a Probability Distribution

- Perplexity is a measure of surprise in random choices.
- Distributions with high uncertainty have high perplexity.
 - A uniform distribution has high perplexity because it is *hard* to predict a random draw from it.
 - A peaked distribution has low perplexity because it is *easy* to predict the outcome of a random draw from it.

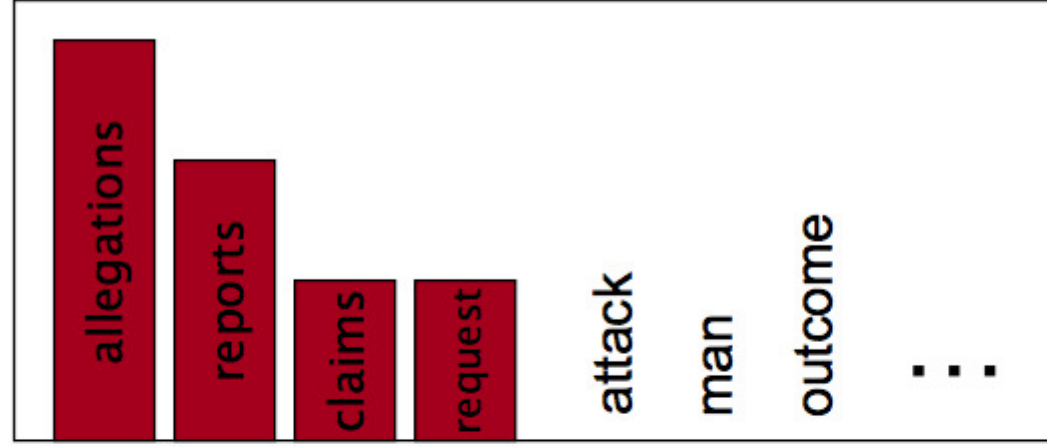
Generalization Issues

- New words
- Rare events

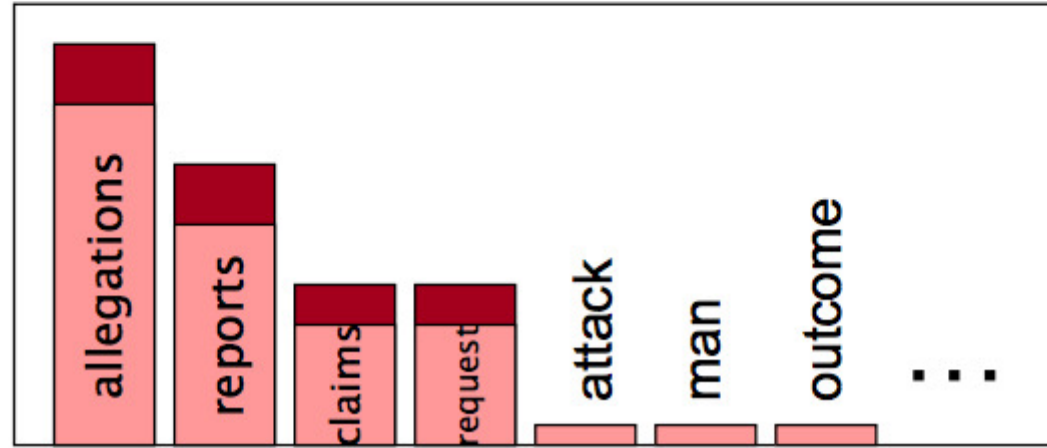
Discounting

$\Pr(w \mid \text{denied, the})$

MLE



Discounting



Add One / Laplace Smoothing

- Assume that there were some additional documents in the corpus, where every possible sequence of words was seen exactly once.
 - Every bigram sequence was seen one more time.
- For bigrams, this means that every possible bi-gram was seen at least once.
 - Zero probabilities go away.

$$Pr_{MLE}(w_i|w_{i-1}) = \frac{\text{count}(w_{i-1}, w_i)}{\text{count}(w_{i-1})}$$

$$Pr_L(w_i|w_{i-1}) = \frac{\text{count}(w_{i-1}, w_i) + 1}{\text{count}(w_{i-1}) + V}$$

Add One Smoothing

	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

Figure 4.5 Add-one smoothed bigram counts for eight of the words (out of $V = 1446$) in the Berkeley Restaurant Project corpus of 9332 sentences. Previously-zero counts are in gray.

Add One Smoothing

	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

Figure 4.2 Bigram probabilities for eight words in the Berkeley Restaurant Project corpus of 9332 sentences. Zero probabilities are in gray.

	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

Figure 4.6 Add-one smoothed bigram probabilities for eight of the words (out of $V = 1446$) in the BeRP corpus of 9332 sentences. Previously-zero probabilities are in gray.

Add One Smoothing

	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

Figure 4.5 Add-one smoothed bigram counts for eight of the words (out of $V = 1446$) in the Berkeley Restaurant Project corpus of 9332 sentences. Previously-zero counts are in gray.

	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

Figure 4.7 Add-one reconstituted counts for eight words (of $V = 1446$) in the BeRP corpus of 9332 sentences. Previously-zero counts are in gray.

Add- k smoothing

- Adding partial counts could mitigate the huge discounting with Add-1.

$$Pr_{L_k}(w_i|w_{i-1}) = \frac{\text{count}(w_{i-1}, w_i) + k}{\text{count}(w_{i-1}) + kV}$$

- How to choose a good k ?
 - Use training/held out data.
- While Add- k is better, it still has issues.
 - Too much mass is stolen from observed counts.

Good-Turing Discounting

- Chance of seeing a new (unseen) bigram
= Chance of seeing a bigram that has occurred only once (singleton)

Chance of seeing a singleton = $\# \text{singletons} / \# \text{ of bigrams}$

- Probabilistic world falls a little ill.
 - We just gave some non-zero probability to new bigrams.
 - Need to steal some probability from the *seen* singletons.
- Recursively discount probabilities of higher frequency bins.
 - $\Pr_{\text{GT}}(w_i^1) = 2 \cdot N_2 / N_1$
 - $\Pr_{\text{GT}}(w_i^2) = 3 \cdot N_3 / N_2$
- Exercise: Can you prove that this forms a valid probability distribution?

Absolute and Kneser-Ney

[Don't make text heavy slides like these.]

- Empirically one finds that GT reduces a fixed amount for bigrams that occur two or more times, typically around 0.75.
- This suggests a more direct method, which is to simply discount a fixed amount for this bigrams that occur 2 or more times, and keep GT method for 0 and 1 bigrams.
 - Absolute discounting.
- Kneser-Ney method extends the absolute discounting idea. For instance for bigrams:
 - Discount counts by a fixed amount and interpolate with unigram probability.
 - However, the raw unigram probability is not such a good measure to use.
 - $\Pr(\text{Francisco}) > \Pr(\text{glasses})$ but $\Pr(\text{glasses} \mid \text{reading})$ should be $> \Pr(\text{Francisco} \mid \text{reading})$
 - glasses is better because it follow many word types compared to Francisco which only typically follows San.
 - Interpolate with the continuation probability of the word, rather than the unigram.
 - Interpolation weight is chosen so that the discounted mass is spread over each possible bigram.
- Commonly used in Speech Recognition and Machine Translation.

Back-off

- Conditioning on longer context is useful if counts are not sparse.
- When counts are sparse, back-off to smaller contexts.
 - If estimating trigrams, use bigram probabilities instead.
 - If estimating bigrams, use unigram probabilities instead.

Interpolation

- Instead of backing off *some* times, interpolate estimates from various contexts.
- Requires a way to combine the estimates.
- Use training/dev set.

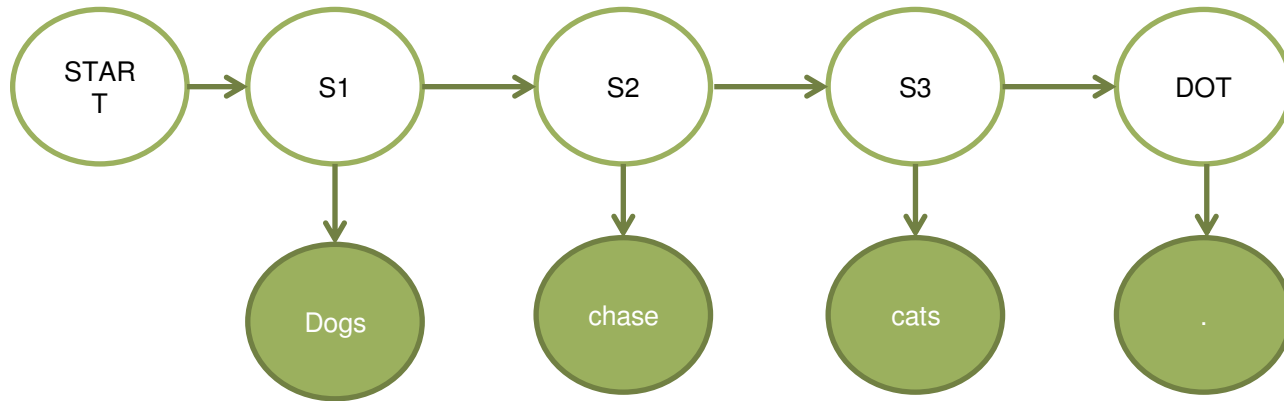
Summary

- Language modeling is the task of building predictive models.
 - Predict the next word in a sequence.
 - Predict the probability of observing a sequence in a language.
- Difficult to directly estimate probabilities for large sequences.
- Markov independence assumptions help deal with this.
 - Leads to various n-gram models.
- Careful chosen estimation techniques are critical for effective application.
 - Smoothing

A (not so) random sample of NLP tasks.

- Language Modeling
- POS Tagging
- Syntactic Parsing
- Topic Modeling

One Slide Injustice to POS Tagging



Sequence Modeling using Hidden Markov Models

A finite state machine goes through a sequence of states and produces the sentence.

Tagging is the task of figuring out the states that the machine went through.

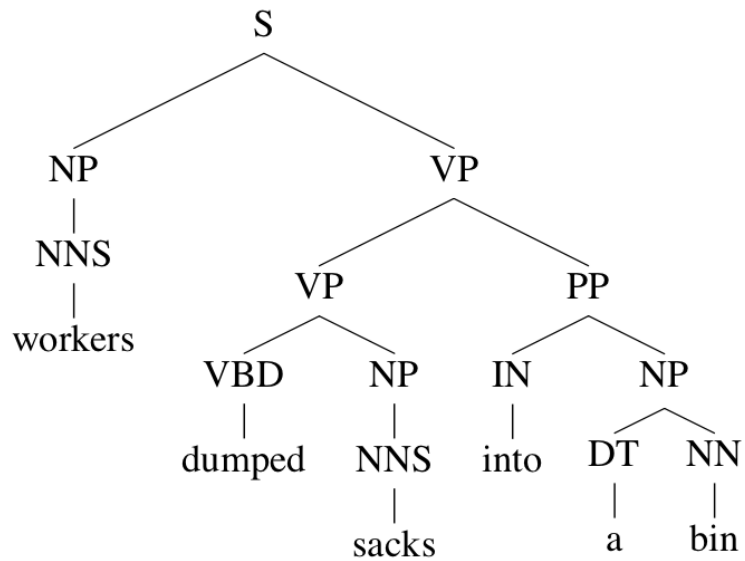
State transitions are conditioned on previous state.

Word emissions are conditioned on current state.

Given training data, we can learn (estimate) the transition, and emission probabilities.

It is also possible to learn the probabilities with unlabeled data using EM.

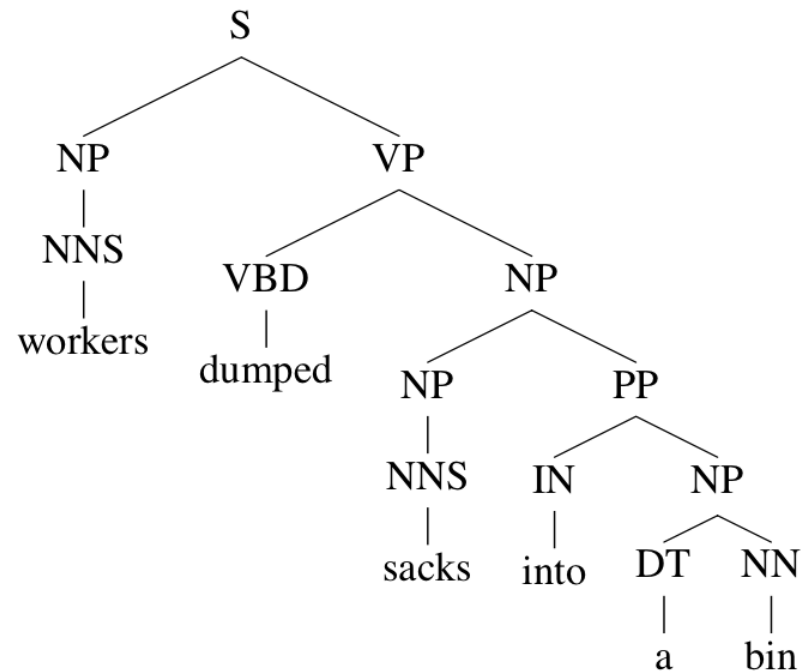
One Slide Injustice to Syntactic Parsing



RULES

S → NP VP
NP → NNS
DT →
N → d
VP → VBD NP PP
VB →

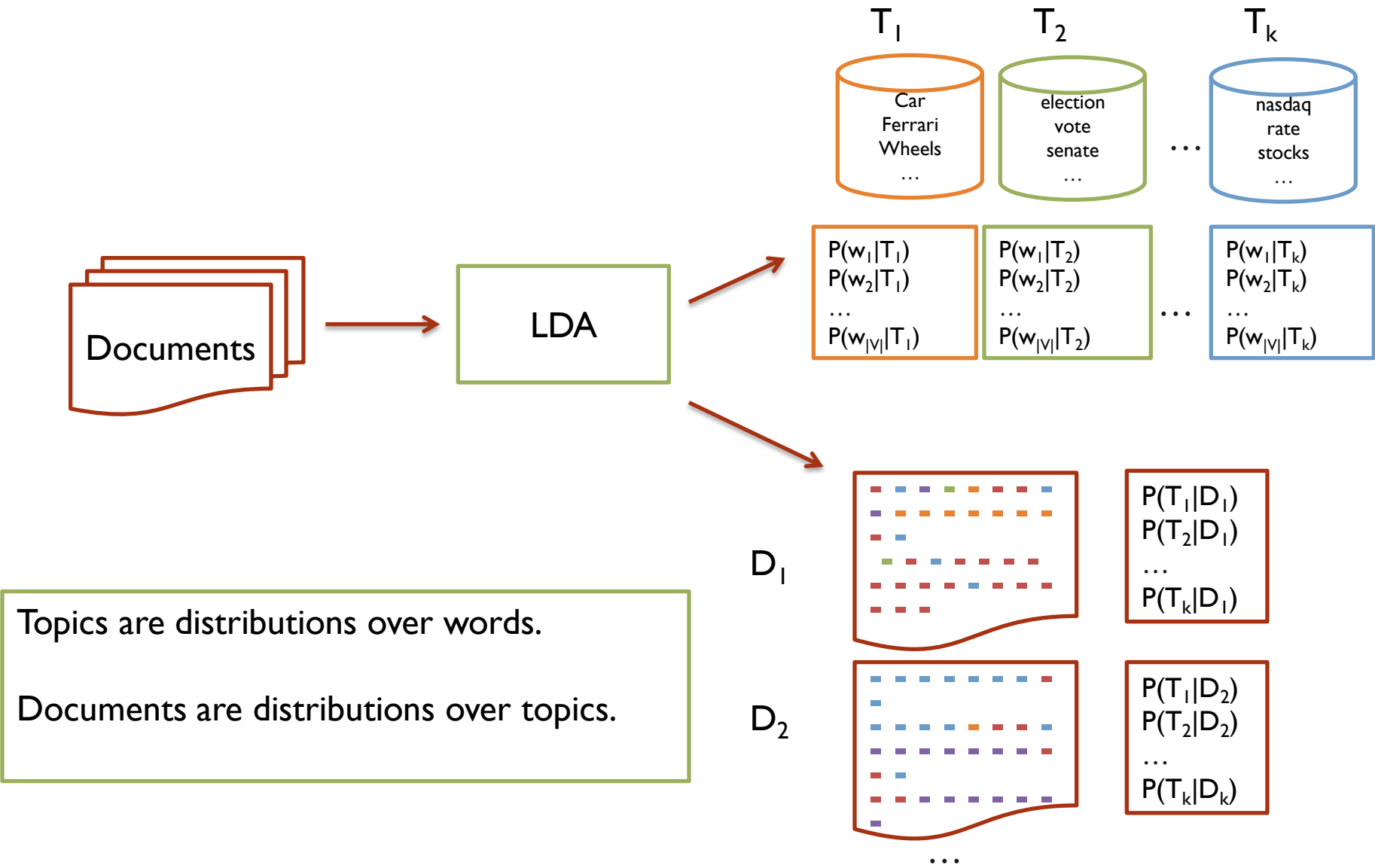
ILITY =



Trade-off: Adding lexical information and fine-grained categories:

- Increases sparsity -- Need appropriate smoothing.
- Adds more rules – Can affect parsing speed.

One Slide Injustice to Topic Modeling



Topics are distributions over words.
Documents are distributions over topics.

Why probability ~~and statistics~~ are relevant for NLP?

A *don't-quote-me-on-this* answer:

All of NLP can reduce to the estimation of uncertainty of various aspects of interpretation and balancing them to draw inferences.

This reduction isn't as far fetched as I made it sound.

We probably do the same.

We interpret sentences into some form of meaning (or a call to action).