CS598JHM: Advanced NLP (Spring 2013)

http://courses.engr.illinois.edu/cs598jhm/

Lecture 1: Introduction

Julia Hockenmaier

juliahmr@illinois.edu

3324 Siebel Center

Office hours: by appointment

Class overview

This class

Seminar on (Bayesian) statistical models in NLP:

- Mathematical and algorithmic foundations
- Applications to NLP

Difference to CS498 (Introduction to NLP):

- Focus on current research and state-of-the-art techniques
- Some of the material will be significantly more advanced
- No exams, but a research project
- Lectures (by me) and paper presentations (by you)

Class topics (I)

Modeling text as a bag of words:

- Applications: Text classification, topic modeling
- Methods: Naive Bayes, Probabilistic Latent Semantic Analysis, Latent Dirichlet Allocation

Modeling text as a sequence of words:

- Applications: Language modeling, POS-tagging
- Methods: n-gram models, Hidden Markov Models, Conditional Random Fields

Class topics (II)

Modeling the structure of sentences:

- Applications: syntactic parsing, grammar induction
- Methods: Probabilistic Grammars, Loglinear Models

Modeling correspondences:

- Applications: image annotation/retrieval, machine translation
- Methods: Correspondence LDA, alignment models

Understanding probabilistic models:

- Bayesian vs. frequentist approaches
- Generative vs. discriminative models
- Exact vs. approximate inference
- Parametric vs. nonparametric models

Tentative class outline

Week	Topics
1-4	Lectures: Background and topic models
5-6	Papers: Topic models
7-8	Lectures: Nonparametric models
9	Papers: Nonparametric models
10-11	Lectures: Sequences and trees
11-15	Papers: Sequences and trees

1. Introduction
2. Conjugate priors
3. Text classification: frequentist vs Bayesian approaches
4. The EM algorithm
5. Sampling
6. Probabilistic Latent Semantic Analysis
7. Latent Dirichlet Allocation
8. Variational Inference for LDA
9. Papers: Correlated topic models
10. Papers: Dynamic topic models
11. Papers: Supervised LDA
12. Papers: Correspondence LDA
13. Dirichlet Processes
14. Hierarchical Dirichlet Processes
15. Hierarchical Dirichlet Processes
16. **Project proposals**
17. Papers: Unsupervised coreference resolution with HDPs
18. Papers: Nonparametric language modeling

————**Spring break** ————

19. Hidden Markov Models
20. Probabilistic Context Free Grammars
21. Conditional random fields
22. Papers: The infinite HMM
23. Papers: Nonparametric PCFGs
24. **Project updates**
25. Papers: Grammar induction
26. Papers: Grammar induction
27. Papers: Language evolution
28. Papers: Multilingual POS tagging
29. Papers: Synchronous grammar induction

Paper presentations

About half the lectures, you will present research papers in class

Goals:

- Get familiar with current work
- Read and learn to present and critique research papers

Paper presentations: procedure

Presenter:

- **Meet with me** at least two days before your presentation

We want to make sure you understand the paper

- **Slides** are recommended, but: please **make your own**, even when the authors make theirs available

You don't actually learn much by regurgitating somebody else's slides.

- Send me a **PDF of your slides before class**
- Bring your laptop (or let me know in advance if you need to use mine)

Everybody else:

- Before class: submit a **one-page summary of the paper**

I won't grade what you write, but I want you to engage with the material

- During/after class: **critique** the presentation

This is merely for everybody's benefit, and not part of the grade.
In fact, I won't even see what you write.

Research projects

Goal: Write a research paper of publishable quality on a topic that is related to this class

Requires **literature review** and **implementation**
Previous projects have been published in good conferences

Research projects: milestones

Week 4: Initial **project proposal** due (1-2 pages)

What project are you going to work on? What resources do you need? Why is this interesting/novel? List related work

Week 8: Fleshed out proposal due (3-4 pages)
First **in-class spotlight presentation**

Add initial literature review, and present preliminary results

Week 12: **Status update** report due;
Second in-class spotlight presentation

Make sure things are moving along

Finals week: **Final report** (8-10 pages); **poster + talk**

Include detailed literature review, describe your results

Grading policies

50% Research project
30% Paper presentations
20% In-class participation and paper summaries

A quick review of probability theory

Probability theory: terminology

Trial:
picking a shape, predicting a word

Sample space Ω:
the set of all possible outcomes
(all shapes; all words in *Alice in Wonderland*)

Event $\omega \subseteq \Omega$:
an actual outcome (a subset of Ω)
(predicting '*the*', picking a triangle)

The probability of events

Kolmogorov axioms:

1) Each event has a probability between 0 and 1.
2) The null event has probability 0.
   The probability that any event happens is 1.
3) The probability of all disjoint events sums to 1.

$$0 \leq P(\omega \subseteq \Omega) \leq 1$$

$$P(\emptyset) = 0 \text{ and } P(\Omega) = 1$$

$$\sum_{\omega_i \subseteq \Omega} P(\omega_i) = 1 \quad \text{if } \forall j \neq i : \omega_i \cap \omega_j = \emptyset \text{ and } \bigcup_i \omega_i = \Omega$$

Random variables

A random variable X is a function from the sample space to a set of outcomes.

In NLP, the sample space is often the set of all possible words or sentences

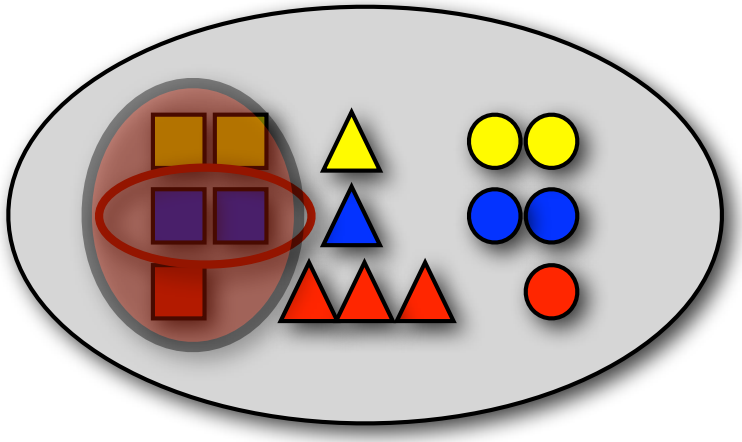
Random variables may be:

- categorical (discrete): the word; its part of speech
- boolean: is the word capitalized?
- integer-valued: how many letters are in the word?
- continuous/real-valued
- vectors (e.g. a probability distribution)

Joint and Conditional Probability

The conditional probability of X given Y, $P(X|Y)$, is defined in terms of the probability of Y, $P(Y)$, and the joint probability of X and Y, $P(X,Y)$:

$$P(X|Y) = \frac{P(X,Y)}{P(Y)}$$



P(blue | ■) = 2/5

The chain rule

The joint probability $P(X,Y)$ can also be expressed in terms of the conditional probability $P(X|Y)$

$$P(X,Y) = P(X|Y)P(Y)$$

This leads to the so-called chain rule:

$$\begin{aligned} P(X_1, X_2, \ldots, X_n) &= P(X_1)P(X_2|X_1)P(X_3|X_2, X_1)....P(X_n|X_1, ...X_{n-1}) \\ &= P(X_1)\prod_{i=2}^{n} P(X_i|X_1 \ldots X_{i-1}) \end{aligned}$$

Independence

Two random variables X and Y are independent if

$$P(X,Y) = P(X)P(Y)$$

If X and Y are independent, then $P(X/Y) = P(X)$:

$$\begin{aligned} P(X|Y) &= \frac{P(X,Y)}{P(Y)} \\ &= \frac{P(X)P(Y)}{P(Y)} \quad (X,\ Y \text{ independent}) \\ &= P(X) \end{aligned}$$

Probability models

Building a probability model consists of two steps:

- defining the model
- estimating the model's parameters

Using a probability model requires inference

Models (almost) always make
independence *assumptions.*

That is, even though X and Y are not actually independent, our model may treats them as independent.

This reduces the number of model parameters we need to estimate (e.g. from n^2 to $2n$)

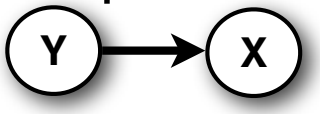
Graphical models

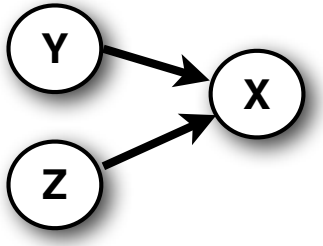
Graphical models are a **notation for probability models**.

Nodes represent distributions over random variables:

$P(X)$ = 

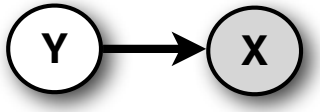
Arrows represent dependencies:

$P(Y)\ P(X \mid Y)$ = 

$P(Y)\ P(Z)\ P(X \mid Y, Z)$ = 

Shaded nodes represent observed variables.

White nodes represent hidden variables

$P(Y)\ P(X \mid Y)$ with Y hidden and X observed = 

Discrete probability distributions: Throwing a coin

Bernoulli distribution:

Probability of success (=head,yes) in single yes/no trial

- The probability of *head* is p.
- The probability of *tail* is $1-p$.

Binomial distribution:

Prob. of the number of heads in a sequence of yes/no trials

The probability of getting exactly k heads in n independent yes/no trials is:

$$P(k \text{ heads},\ n-k \text{ tails}) = \binom{n}{k} p^k (1-p)^{n-k}$$

Discrete probability distributions: Rolling a die

Categorical distribution:

Probability of getting one of *N* outcomes in a single trial.

The probability of category/outcome c_i is p_i $(\sum p_i = 1)$

Multinomial distribution:

Probability of observing each possible outcome c_i exactly X_i times in a sequence of n trials

$$P(X_1 = x_i, \ldots, X_N = x_N) = \frac{n!}{x_1! \cdots x_N!} p_1^{x_1} \cdots p_N^{x_N} \quad \text{if} \sum_{i=1}^{N} x_i = n$$

Multinomial variables

- In NLP, X is often a **discrete** random variable that can take one of K states.

- We can represent such Xs as **K-dimensional vectors** in which one $x_k = 1$ and all other elements are 0
  $x = (0,0,1,0,0)^T$

- Denote probability of $x_k = 1$ as μ_k with $0 \leq \mu_k \leq 1$ and $\sum_k \mu_k = 1$
  Then the probability of x is:

$$P(x|\boldsymbol{\mu}) = \prod_{k=1}^{K} \mu_k^{x_k}$$

Probabilistic models for natural language: Language modeling

Unigram (bag of word) language models

Alice was beginning to get very tired of sitting by her sister on the bank, and of having nothing to do: once or twice she had peeped into the book her sister was reading, but it had no pictures or conversations in it, 'and what is the use of a book,' thought Alice 'without pictures or conversation?'

P(of) = 3/66
P(Alice) = 2/66
P(was) = 2/66
P(to) = 2/66

P(her) = 2/66
P(sister) = 2/66
P(,) = 4/66
P(') = 4/66

N-gram models

Unigram model	$P(w_1)P(w_2)...P(w_i)$
Bigram model	$P(w_1)P(w_2\|w_1)...P(w_i\|w_{i-1})$
Trigram model	$P(w_1)P(w_2\|w_1)...P(w_i\|w_{i-2}w_{i-1})$
N-gram model	$P(w_1)P(w_2\|w_1)...P(w_i\|w_{i-n-1}...w_{i-1})$

N-gram models assume each word (event) depends only on the previous n-1 words (events). Such independence assumptions are called Markov assumptions (of order n-1).

$$P(w_i|w_1^{i-1}) :\approx P(w_i|w_{i-n}^{i-1})$$

Estimating N-gram models

1. Bracket each sentence by special start and end symbols:

   <s> Alice was beginning to get very tired… </s>

   (We only assign probabilities to strings <s>...</s>)

2. Count the frequency of each n-gram….

   C(<s> Alice) = 1, C(Alice was) = 1,….

3. …. and normalize to get the probability:

$$P(w_n|w_{n-1}) = \frac{C(w_{n-1}w_n)}{C(w_{n-1})}$$

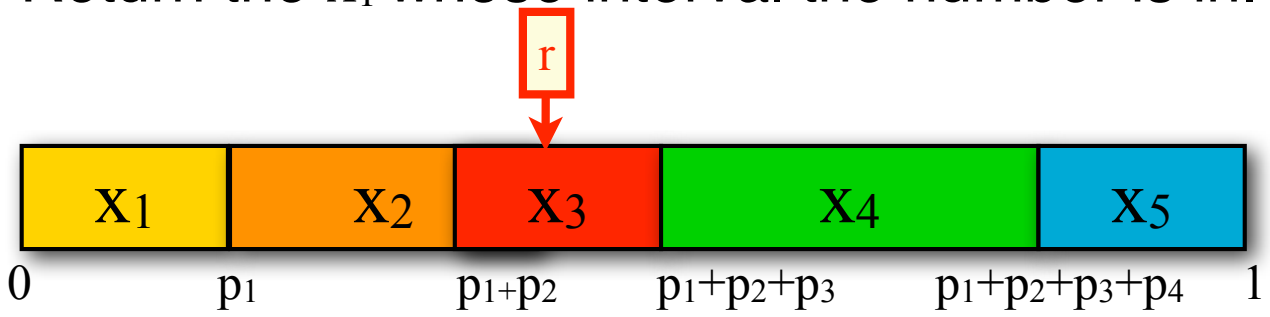
This is the relative frequency estimate

Generating from a distribution

How do you generate text from an *n*-gram model?

That is, how do you sample from a distribution $P(X \mid Y=y)$?

- Assume X has n possible outcomes (values): $\{x_1,....,x_n\}$ and $P(x_i \mid Y=y) = p_i$
- Divide the interval [0,1] into n smaller intervals according to the probabilities of the outcomes
- Generate a random number r between 0 and 1.
- Return the x_1 whose interval the number is in.



Generating Shakespeare

Unigram	• To him swallowed confess hear both. Which. Of save on trail for are ay device and rote life have • Every enter now severally so, let • Hill he late speaks; or! a more to leg less first you enter • Are where exeunt and sighs have rise excellency took of.. Sleep knave we. near; vile like
Bigram	• What means, sir. I confess she? then all sorts, he is trim, captain. •Why dost stand forth thy canopy, forsooth; he is this palpable hit the King Henry. Live king. Follow. •What we, hath got so she that I rest and sent to scold and nature bankrupt, nor the first gentleman? •Enter Menenius, if it so many good direction found'st thou art a strong upon command of fear not a liberal largess given away, Falstaff! Exeunt
Trigram	• Sweet prince, Falstaff shall die. Harry of Monmouth's grave. • This shall forbid it should be branded, if renown made it empty. • Indeed the duke; and had a very good friend. • Fly, and will rid me these news of price. Therefore the sadness of parting, as they say, 'tis done.
Quadrigram	• King Henry. What! I will go seek the traitor Gloucester. Exeunt some of the watch. A great banquet serv'd in; • Will you not tell me who I am? • It cannot be but so. • Indeed the short and the long. Marry, 'tis a noble Lepidus.

Shakespeare as corpus

The Shakespeare corpus consists of N=884,647 word **tokens** and a vocabulary of V=29,066 word **types**

Shakespeare produced 300,000 bigram types out of V^2= 844 million possible bigrams…
99.96% of the possible bigrams were never seen

Quadrigrams look like Shakespeare because they are Shakespeare

Unseen events matter

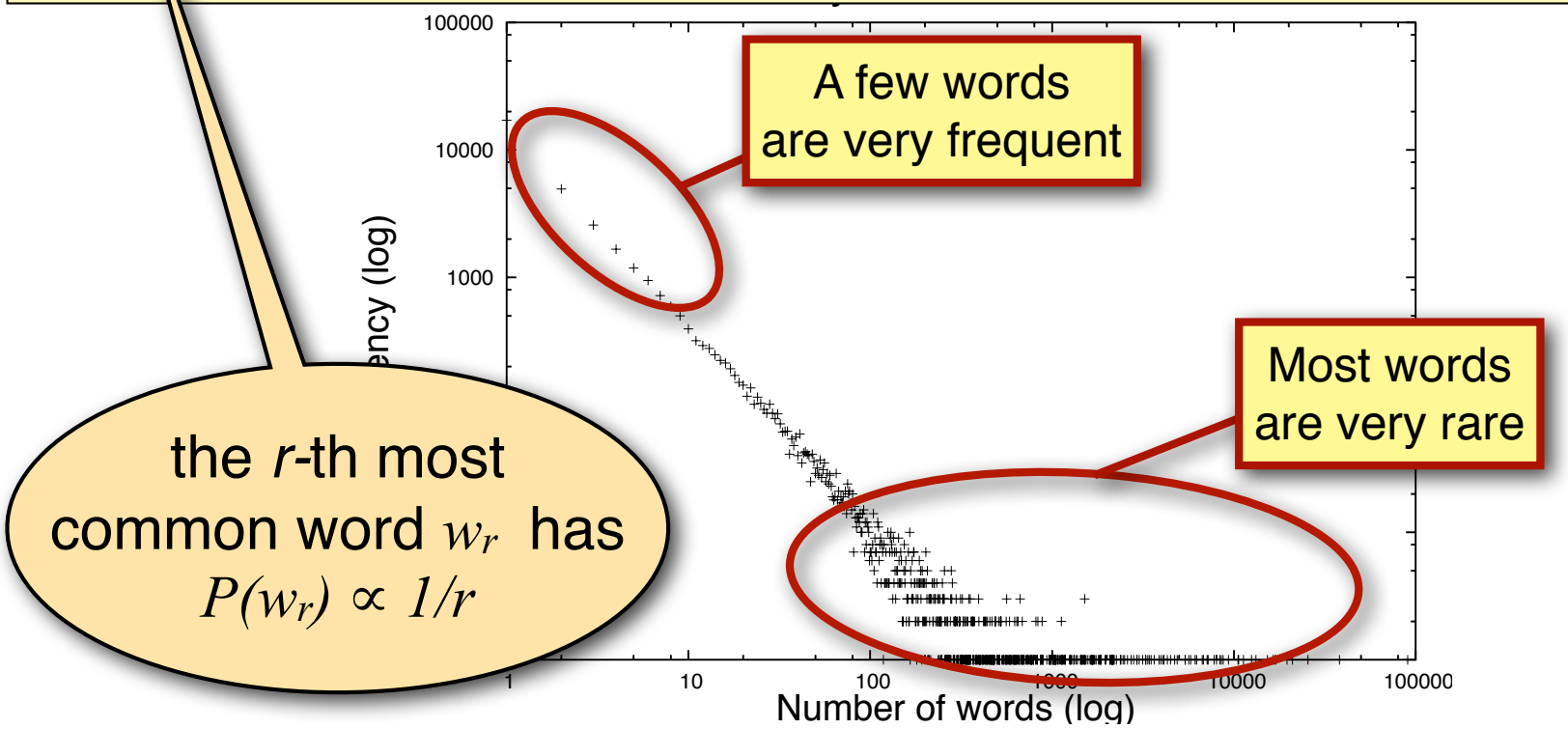
We estimated a model on 440K word tokens, but:

Only 30,000 word types occurred.
Any word that does not occur in the training data has zero probability!

Only 0.04% of all possible bigrams occurred.
Any bigram that does not occur in the training data has zero probability!

Zipf's law: the long tail

How many words occur once, twice, 100 times, 1000 times?



In natural language:

- A small number of events (e.g. words) occur with high frequency
- A large number of events occur with very low frequency

Dealing with unseen events

Relative frequency estimation assigns all probability mass to events in the training corpus

But we need to reserve *some* probability mass to events that don't occur in the training data

- Unseen events = new words, new bigrams

Important questions:

- What possible events are there?
- How much probability mass should they get?

Probabilistic models for natural language: Text classification

Naive Bayes for text classification

The task:

Assign (sentiment) label $L_i \in \{+, -\}$ to a document $\boldsymbol{W}_i$.

$\boldsymbol{W}_1$= *"This is an amazing product: great battery life, amazing features and it's cheap."*
$\boldsymbol{W}_2$= *"How awful. It's buggy, saps power and is way too expensive."*

The model:

- Use Bayes' Rule:

$$L_i = argmax_L\, P(L \mid \boldsymbol{W}_i) = argmax_L\, P(\boldsymbol{W}_i \mid L) P(L)$$

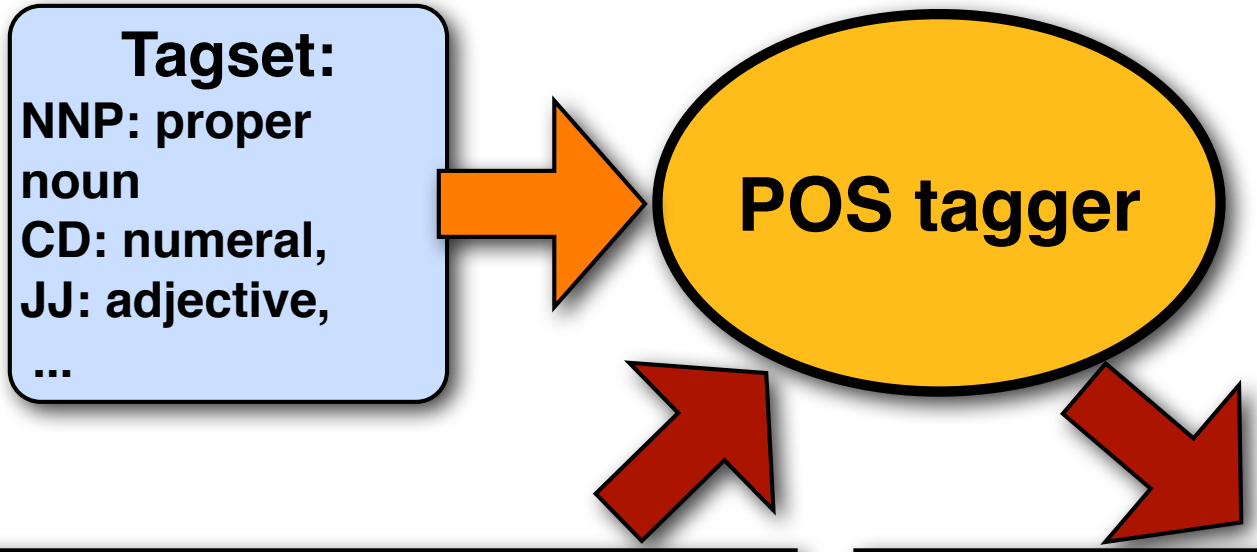
- Assume $\boldsymbol{W}_i$ is a "bag of words":

$\boldsymbol{W}_1$= {an:1, and: 1, amazing: 2, battery: 1, cheap: 1, features: 1, great: 1, ...}
$\boldsymbol{W}_2$= {awful: 1, and: 1, buggy: 1, expensive: 1, ...}

- $P(\boldsymbol{W}_i \mid L)$ is a multinomial distribution: $\boldsymbol{W}_i \sim Multinomial(\boldsymbol{\theta}_L)$
  We have a vocabulary of V words. Thus: $\boldsymbol{\theta}_L = (\theta_1, ..., \theta_V)$
- $P(L)$ is a Bernoulli distribution: $L \sim Bernoulli(\pi)$

Probabilistic models for natural language: Sequence labeling

POS tagging



Raw text

```
Pierre Vinken , 61 years
old , will join the
board as a nonexecutive
director Nov. 29 .
```

Tagged text

```
Pierre_NNP Vinken_NNP ,_, 61_CD
years_NNS old_JJ ,_, will_MD
join_VB the_DT board_NN as_IN
a_DT nonexecutive_JJ director_NN
Nov._NNP 29_CD ._.
```

Statistical POS tagging

What is the **most likely** sequence of tags ***t*** for the given sequence of words ***w*** ?

$$\begin{aligned}
\underset{\mathbf{t}}{\operatorname{argmax}} P(\mathbf{t}|\mathbf{w}) &= \underset{\mathbf{t}}{\operatorname{argmax}} \frac{P(\mathbf{t},\mathbf{w})}{P(\mathbf{w})} \\
&= \underset{\mathbf{t}}{\operatorname{argmax}} P(\mathbf{t},\mathbf{w}) \\
&= \underset{\mathbf{t}}{\operatorname{argmax}} P(\mathbf{t})P(\mathbf{w}|\mathbf{t})
\end{aligned}$$

$P(\mathbf{t},\mathbf{w})$ is a **generative** (joint) model.

Hidden Markov Models are generative models which decompose $P(\mathbf{t},\mathbf{w})$ as $P(\mathbf{t})P(\mathbf{w}|\mathbf{t})$

Hidden Markov Models

$$\underset{\mathbf{t}}{\operatorname{argmax}} P(\mathbf{t}|\mathbf{w}) \quad = \quad \underset{\mathbf{t}}{\operatorname{argmax}} P(\mathbf{t})P(\mathbf{w}|\mathbf{t})$$

$$:=_{def} \quad \underset{\mathbf{t}}{\operatorname{argmax}} \prod_{i=1}^{n} P(t_i|t_{i-N}\ldots t_{i-1}) \prod_{i} P(w_i|t_i)$$

HMM models are **generative** models of $P(w,t)$
(because they model $P(w|t)$ rather than $P(t|w)$)

They make two **independence assumptions**:

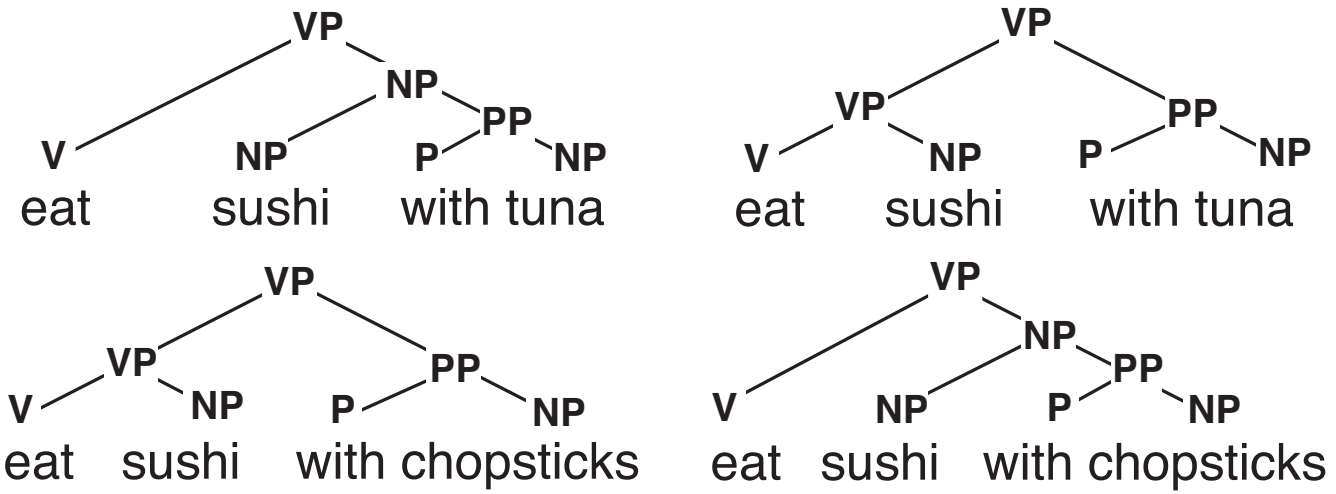
a) approximate $P(t)$ with an N-gram model

b) assume that each word depends only on its POS tag

Probabilistic models for natural language: Grammars

Grammars are ambiguous

A grammar might generate multiple trees for a sentence:



What's the most likely parse τ for sentence S ?

We need a model of $P(\tau \mid S)$

Probabilistic Context-Free Grammars

For every nonterminal X, define a probability distribution $P(X \rightarrow \alpha \mid X)$ over all rules with the same LHS symbol X:

S	$\rightarrow$	NP VP	0.8
S	$\rightarrow$	S conj S	0.2
NP	$\rightarrow$	Noun	0.2
NP	$\rightarrow$	Det Noun	0.4
NP	$\rightarrow$	NP PP	0.2
NP	$\rightarrow$	NP conj NP	0.2
VP	$\rightarrow$	Verb	0.4
VP	$\rightarrow$	Verb NP	0.3
VP	$\rightarrow$	Verb NP NP	0.1
VP	$\rightarrow$	VP PP	0.2
PP	$\rightarrow$	P NP	1.0