Part-of-Speech Tagging & Parsing

Chloé Kiddon

(slides adapted and/or stolen outright from Andrew McCallum, Christopher Manning, and Julia Hockenmaier)

Announcements

- We do have a Hadoop cluster!
  - It's offsite. I need to know all groups who want it!
- You all have accounts for MySQL on the cubist machine (cubist.cs.washington.edu)
  - Your folder is /projects/instr/cse454/a-f
- I'll have a better email out this afternoon I hope
- Grading HW1 should be finished by next week.

Timely warning

- POS tagging and parsing are two large topics in NLP
- Usually covered in 2-4 lectures
- We have an hour and twenty minutes. ☺

Part-of-speech tagging

- Often want to know what part of speech (POS) or word class (noun,verb,…) should be assigned to words in a piece of text
- **Part-of-speech tagging** assigns POS labels to words

JJ	JJ	NNS	VBP	RB
Colorless	**green**	**ideas**	**sleep**	**furiously.**

Why do we care?

- Parsing (come to later)
- Speech synthesis
  - INsult or inSULT, overFLOW or OVERflow, REad or reAD
- Information extraction: entities, relations
  - Romeo loves Juliet vs. lost loves found again
- Machine translation

Penn Treebank Tagset

1.	CC	Coordinating conjunction	20.	RB	Adverb
2.	CD	Cardinal number	21.	RBR	Adverb, comparative
3.	DT	Determiner	22.	RBS	Adverb, superlative
4.	EX	Existential there	23.	RP	Particle
5.	FW	Foreign word	24.	SYM	Symbol
6.	IN	Preposition or subordinating conjunction	25.	TO	to
			26.	UH	Interjection
7.	JJ	Adjective	27.	VB	Verb, base form
8.	JJR	Adjective, comparative	28.	VBD	Verb, past tense
9.	JJS	Adjective, superlative	29.	VBG	Verb, gerund or present participle
10.	LS	List item marker			
11.	MD	Modal	30.	VBN	Verb, past participle
12.	NN	Noun, singular or mass	31.	VBP	Verb, non-3rd person singular present
13.	NNS	Noun, plural			
14.	NP	Proper noun, singular	32.	VBZ	Verb, 3rd person singular present
15.	NPS	Proper noun, plural			
16.	PDT	Predeterminer	33.	WDT	Wh-determiner
17.	POS	Possessive ending	34.	WP	Wh-pronoun
18.	PP	Personal pronoun	35.	WP$	Possessive wh-pronoun
19.	PP$	Possessive pronoun	36.	WRB	Wh-adverb

Ambiguity

Buffalo buffalo buffalo.

How many words are ambiguous?

		87-tag Original Brown	45-tag Treebank Brown
Unambiguous (1 tag)		**44,019**	**38,857**
Ambiguous (2–7 tags)		**5,490**	**8844**
Details:	2 tags	4,967	6,731
	3 tags	411	1621
	4 tags	91	357
	5 tags	17	90
	6 tags	2 (*well, beat*)	32
	7 tags	2 (*still, down*)	6 (*well, set, round, open, fit, down*)
	8 tags		4 (*'s, half, back, a*)
	9 tags		3 (*that, more, in*)

Hockenmaier

Naïve approach!

- Pick the most common tag for the word

Word	POS listings in Brown		
heat	noun/89	**verb/5**	
oil	**noun/87**		
in	**prep/20731**	noun/1	adv/462
a	**det/22943**	noun/50	noun-proper/30
large	**adj/354**	noun/2	adv/5
pot	**noun/27**		

- 91% success rate!

Andrew McCallum

We have more information

- We are not just tagging words, we are tagging *sequences* of words

For a sequence of words **W**:

$$\mathbf{W} = \mathbf{w_1 w_2 w_3 \ldots w_n}$$

We are looking for a sequence of tags **T**:

$$\mathbf{T} = \mathbf{t_1\, t_2\, t_3 \ldots t_n}$$

where P(**T**|**W**) is maximized

Andrew McCallum

In an ideal world…

- Find all instances of a sequence in the dataset and pick the most common sequence of tags
  - Count("heat oil in a large pot") = 0 ????
  - Uhh…
- Spare data problem
- Most sequences will never occur, or will occur too few times for good predictions

Bayes' Rule

- To find P(T|W), use Bayes' Rule:

$$P(T \mid W) = \frac{P(W \mid T) \times P(T)}{P(W)} \qquad P(T \mid W) \propto P(W \mid T) \times P(T)$$

- We can maximize P(T|W) by maximizing P(W|T)*P(T)

Andrew McCallum

Finding P(T)

- Generally,

$$P(t_1 t_2 \dots t_n) = P(t_1) \times P(t_2 \dots t_n \mid t_1)$$

$$P(t_1 t_2 \dots t_n) = P(t_1) \times P(t_2 \mid t_1) \times P(t_3 \dots t_n \mid t_1 t_2)$$

$$P(t_1 t_2 \dots t_n) = \prod_i P(t_i \mid t_1 t_2 \dots t_{i-1})$$

- Usually not feasible to accurately estimate more than tag bigrams (possibly trigrams)

Markov assumption

- Assume that the probability of a tag only depends on the tag that came **directly before** it

$$P(t_i \mid t_1 t_2 \dots t_{i-1}) = P(t_i \mid t_{i-1})$$

- Then,

$$P(t_1 t_2 \dots t_n) = \prod_i P(t_i \mid t_{i-1})$$

$$P(t_1 t_2 \dots t_n) = P(t_1) \times P(t_2 \mid t_1) \times P(t_3 \mid t_2) \times \dots \times P(t_n \mid t_{n-1})$$

- Only need to count tag bigrams.

Putting it all together

- We can similarly assume

$$P(w_i \mid t_1 \dots t_n) = P(w_i \mid t_i)$$

- So:

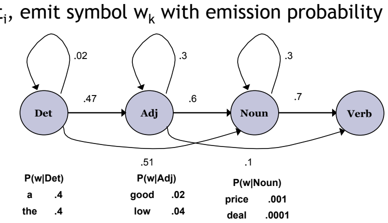
$$P(w_1 \dots w_n \mid t_1 \dots t_n) = P(w_1 \mid t_1) \times P(w_2 \mid t_2) \times \dots \times P(w_n \mid t_n)$$

- And the final equation becomes:

$$P(W \mid T) \times P(T) = P(w_1 \mid t_1) \times P(w_2 \mid t_2) \times \dots \times P(w_n \mid t_n) \times P(t_1) \times P(t_2 \mid t_1) \times P(t_3 \mid t_2) \times \dots \times P(t_n \mid t_{n-1})$$

Process as an HMM

- Start in an initial state t_0 with probability $\pi(t_0)$
- Move from state t_i to t_j with transition probability $a(t_j \mid t_i)$
- In state t_i, emit symbol w_k with emission probability $b(w_k \mid t_i)$



Three Questions for HMMs

1. **Evaluation** - Given a sequence of words **W** = $w_1w_2w_3…w_n$ and an HMM model Θ, what is P(**W**|Θ)
2. **Decoding** - Given a sequence of words **W** and an HMM model Θ, find the most probable parse **T** = $t_1\,t_2\,t_3 … t_n$
3. **Learning** - Given a tagged (or untagged) dataset, find the HMM Θ that maximizes the data

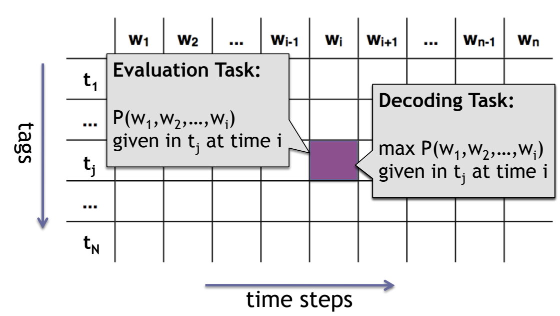
Three Questions for HMMs

1. **Evaluation** - Given a sequence of words **W** = $w_1w_2w_3…w_n$ and an HMM model Θ, what is P(**W**|Θ)
2. **Tagging** - Given a sequence of words **W** and an HMM model Θ, find the most probable parse **T** = $t_1\,t_2\,t_3 … t_n$
3. **Learning** - Given a tagged (or untagged) dataset, find the HMM Θ that maximizes the data

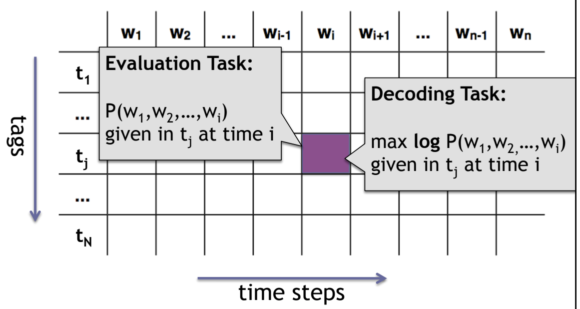
Tagging

- Need to find the **most likely** tag sequence given a sequence of words
  - maximizes P(W|T)*P(T) and thus P(T|W)
- Use Viterbi!

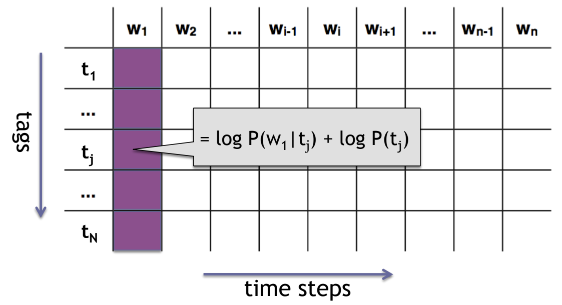
Trellis



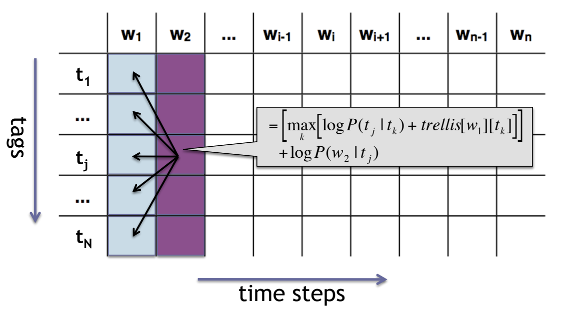
Trellis



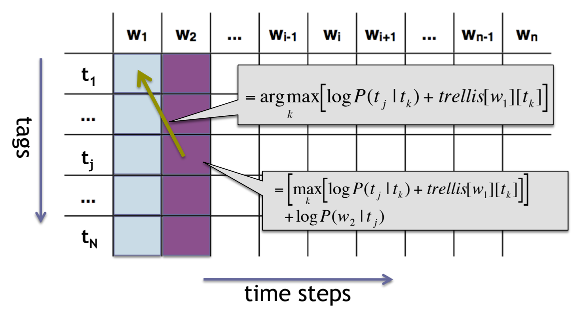
Tagging initialization



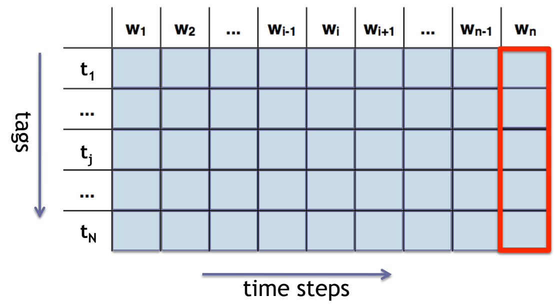
Tagging recursive step



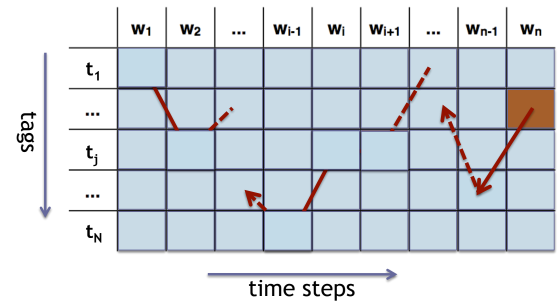
Tagging recursive step



Pick best trellis cell for last word



Use back pointers to pick best sequence



Learning a POS-tagging HMM

- Estimate the parameters in the model using counts

$$P(t_i \mid t_{i-1}) \rightarrow$$

$$P(w_i \mid t_i) \rightarrow$$

- With smoothing, this model can get 95-96% correct tagging

Problem with supervised learning

- Requires a large hand-labeled corpus
  - Doesn't scale to new languages
  - Expensive to produce
  - Doesn't scale to new **domains**
- **Instead**, apply unsupervised learning with Expectation Maximization (EM)
  - Expectation step: calculate probability of all sequences using set of parameters
  - Maximization step: re-estimate parameters using results from E-step

Lots of other techniques!

- Trigram models (more common)
- Text normalization
- Error-based transformation learning ("Brill learning")
  - Rule-based system
    - Calculate initial states: proper noun detection, tagged corpus
    - Acquire transformation rules
      - Change VB to NN when prev word was adjective
      - The long **race** finally ended
- Minimally supervised learning
  - Unlabeled data but have a dictionary

Seems like POS-tagging is solved

- Penn Treebank POS-tagging accuracy ≈ human ceiling
  - Human agreement 97%
- In other languages, not so much

So now we are HMM Masters

- We can use HMMs to…
  - Tag words in a sentence with their parts of speech
  - Extract entities and other information from a sentence
- Can we use them to determine syntactic categories?

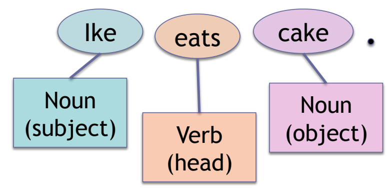
Syntax

- Refers to the study of the way words are arranged together, and the relationship between them.
- Prescriptive vs. Descriptive
- **Goal of syntax is to model the knowledge of that people unconsciously have about the grammar of their native language**
- Parsing extracts the syntax from a sentence

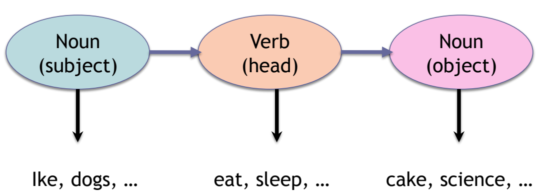
Parsing applications

- High-precision Question-Answering systems
- Named Entity Recognition (NER) and information extraction
- Opinion extraction in product reviews
- Improved interaction during computer applications/games

Basic English sentence structure



Can we build an HMM?

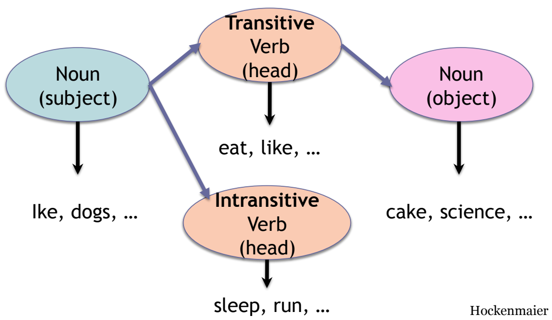


Words take arguments

I eat cake. ☺
I sleep cake. ☹
I give you cake. ☺
I give cake. Hmm…
I eat you cake??? ☹☹

- Subcategorization
  - **Intransitive verbs**: take only a subject
  - **Transitive verbs**: take a subject and an object
  - **Ditransitive verbs**: take a subject, object, and indirect object
- Selectional preferences
  - The object of *eat* should be edible

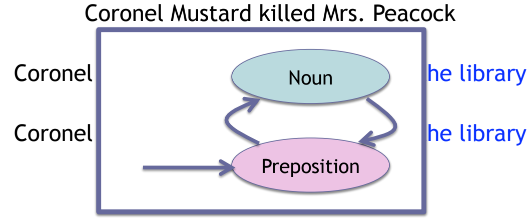
A better model



Hockenmaier

Language has recursive properties

Coronel Mustard killed Mrs. Peacock



Coronel Mustard killed Mrs. Peacock in the library with the candlestick at midnight

HMMs can't generate hierarchical structure

Coronel Mustard killed Mrs. Peacock in the library with the candlestick at midnight.

- Does Mustard have the candlestick?
- Or is the candlestick just sitting in the library?
- Memoryless
  - Can't make long range decisions about attachments
- Need a better model

Words work in groups

- **Constituents** – words or groupings of words that function as single units
  - Noun phrases (NPs)
    - The computer science class
    - Peter, Paul, and Mary
    - PAC10 Schools, such as UW,
    - He
    - The reason I was late

Words work in groups

- **Constituents** – words or groupings of words that function as single units
  - Noun phrases (NPs)
    - The computer science class **listened** …
    - **NPs can appear before a verb.**
    - PAC10 Schools, such as UW, **dominate** …
    - He **juggled** …
    - The reason I was late **was** …
    - *the listened
    - *such sing
    - *late was

Many different constituents

1. S - simple declarative clause
2. SBAR - Clause introduced by a (possibly empty) subordinating conjunction
3. SBARQ - Direct question introduced by a wh-word or a wh-phrase
4. SINV - Inverted declarative sentence
5. SQ - Inverted yes/no question, or main clause of a wh-question
6. ADJP - Adjective Phrase.
7. ADVP - Adverb Phrase.
8. CONJP - Conjunction Phrase.
9. FRAG - Fragment.
10. INTJ - Interjection.
11. LST - List marker.
12. NAC - Not a Constituent; used to show the scope of certain prenominal modifiers within an NP.
13. NP - Noun Phrase.
14. NX - Used within certain complex NPs to mark the head of the NP. Corresponds very roughly to N-bar level but used quite differently.
15. PP - Prepositional Phrase.
16. PRN - Parenthetical.
17. PRT - Particle.
18. QP - Quantifier Phrase (i.e. complex measure/amount phrase); used within NP.
19. RRC - Reduced Relative Clause.
20. UCP - Unlike Coordinated Phrase.
21. VP - Vereb Phrase.
22. WHADJP - Wh-adjective Phrase.
23. WHAVP - Wh-adverb Phrase.
24. WHNP - Wh-noun Phrase.
25. WHPP - Wh-prepositional Phrase.
26. X - Unknown, uncertain, or unbracketable. X is often used for bracketing typos and in bracketing the...the-constructions.

Many different constituents

1.	CC	Coordinating conjunction	20.	RB	Adverb
2.	CD	Cardinal number	21.	RBR	Adverb, comparative
3.	DT	Determiner	22.	RBS	Adverb, superlative
4.	EX	Existential there	23.	RP	Particle
5.	FW	Foreign word	24.	SYM	Symbol
6.	IN	Preposition or subordinating conjunction	25.	TO	to
			26.	UH	Interjection
7.	JJ	Adjective	27.	VB	Verb, base form
8.	JJR	Adjective, comparative	28.	VBD	Verb, past tense
9.	JJS	Adjective, superlative	29.	VBG	Verb, gerund or present participle
10.	LS	List item marker			
11.	MD	Modal	30.	VBN	Verb, past participle
12.	NN	Noun, singular or mass	31.	VBP	Verb, non-3rd person singular present
13.	NNS	Noun, plural			
14.	NP	Proper noun, singular	32.	VBZ	Verb, 3rd person singular present
15.	NPS	Proper noun, plural			
16.	PDT	Predeterminer	33.	WDT	Wh-determiner
17.	POS	Possessive ending	34.	WP	Wh-pronoun
18.	PP	Personal pronoun	35.	WP$	Possessive wh-pronoun
19.	PP$	Possessive pronoun	36.	WRB	Wh-adverb

Attachment ambiguities

- Teacher Strikes Idle Kids
- Squad Helps Dog Bite Victim
- Complaints About NBA Referees Getting Ugly
- Soviet Virgin Lands Short of Goal Again
- Milk Drinkers are Turning to Powder

Attachment ambiguities

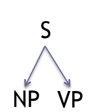
- The key parsing decision: How do we 'attach' various kinds of constituents – PPs, adverbial or participial phrases, coordinations, etc.
- Prepositional phrase attachment
  - *I saw the man with the telescope.*
- What does with *a telescope* modify?
  - The verb *saw*?
  - The noun *man*?
- Very hard problem. AI Complete.

Parsing

- We want to run a grammar backwards to find possible structures for a sentence
- Parsing can be viewed as a **search problem**
- Parsing is a **hidden data problem**

Context-free grammars (CFGs)

- Specifies a set of tree structures that capture constituency and ordering in language
  - A noun phrase can come before a verb phrase
    - S → NP VP



Phrase structure grammars = Context-free grammars

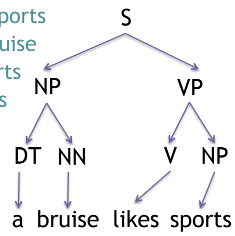
- G = (T, N, S, R)
  - T is the set of terminals (i.e. words)
  - N is the set of non-terminals
    - Usually separate the set P of preterminals (POS tags) from the rest of the non-terminals
    - S is the start symbol
    - R is the set of rules/productions of the form X → γ where X is a nonterminal and γ is a sequence of terminals and nonterminals (possibly empty)
- A grammer G generates a language L

Manning

A phrase structure grammar

- By convention, S is the start symbol
  - S → NP VP
  - NP → DT NN
  - NP → NNS
  - VP → V NP
  - VP → V
  - …

NN → boy
NNS → sports
NN → bruise
V → sports
V → likes
DT → a



But since a sentence can have more than one parse…

Probabilistic context-free grammars (PCFGs)

- G = (T, N, S, R, P)
  - T is the set of terminals (i.e. words)
  - N is the set of non-terminals
    - Usually separate the set P of preterminals (POS tags) from the rest of the non-terminals
    - S is the start symbol
    - R is the set of rules/productions of the form X → γ where X is a nonterminal and γ is a sequence of terminals and nonterminals (possibly empty)
    - P(R) gives the probability of each rule

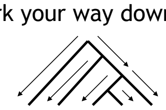
$$\forall X \in N, \quad \sum_{X \to \gamma \in R} P(X \to \gamma) = 1$$

- A grammer G generates a language L

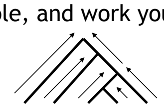
Manning

How to parse

- Top-down: Start at the top of the tree with an S node, and work your way down to the words.



- Bottom-up: Look for small pieces that you know how to assemble, and work your way up to larger pieces.



Given a sentence S…

- We want to find the most likely parse τ

$$\arg\max_{\tau} P(\tau \mid S) = \arg\max_{\tau} \frac{P(\tau, S)}{P(S)}$$
$$= \arg\max_{\tau} P(\tau, S)$$
$$= \arg\max_{\tau} P(\tau) \quad \text{If S = yield}(\tau)$$

- How are we supposed to find P(τ)?
- Infinitely many trees in the language!

Finding P(τ)

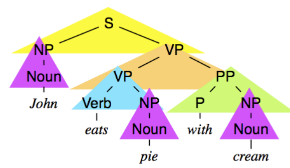
- Define probability distributions over the rules in the grammar
- Context free!

S	→	NP VP	0.8
S	→	S conj S	0.2
NP	→	Noun	0.2
NP	→	Det Noun	0.4
NP	→	NP PP	0.2
NP	→	NP conj NP	0.2
VP	→	Verb	0.4
VP	→	Verb NP	0.3
VP	→	Verb NP NP	0.1
VP	→	VP PP	0.2
PP	→	P NP	1.0

Hockenmaier

Finding P(τ)

- The probability of a tree is the product of the probability of the rules that created it



S	→	NP VP	0.8
S	→	S conj S	0.2
NP	→	Noun	0.2
NP	→	Det Noun	0.4
NP	→	NP PP	0.2
NP	→	NP conj NP	0.2
VP	→	Verb	0.4
VP	→	Verb NP	0.3
VP	→	Verb NP NP	0.1
VP	→	VP PP	0.2
PP	→	P NP	1.0

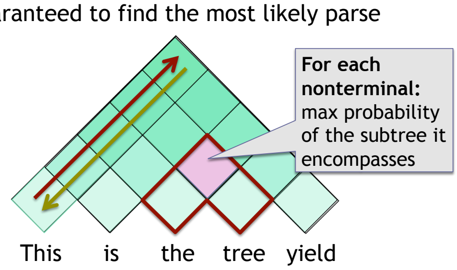
$$P(\tau) = 0.8 \times 0.3 \times 0.2 \times 1.0 \times 0.2^3$$

$$= \mathbf{0.00384}$$

Hockenmaier

Parsing – Cocke-Kasami-Younger (CKY)

- Like Viterbi but for trees
- Guaranteed to find the most likely parse



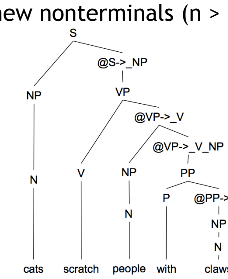
Chomsky Normal Form

- All rules are of the form X → Y Z or X → w.
- *n*-ary rules introduce new nonterminals (n > 2)
  - VP → V NP PP

  becomes:

  VP → V @VP-V and

  @VP-V → NP PP



Manning

CKY Example

Input: POS-tagged sentence
`John_N eats_V pie_N with_P cream_N`

John	eats	pie	with	cream	
N NP 0.2	S 0.8*0.2*0.4	S 0.8*0.2*0.08		S 0.2*0.0024*0.8	John
	V VP 0.4	VP 0.3*0.2		VP max(0.008*0.2, 0.06*0.2*0.2)	eats
		N NP 0.2		NP 0.2*0.2*0.2	pie
			P	PP 1*0.2	with
				N NP 0.2	cream

S	→ NP VP	0.8
S	→ S conj S	0.2
NP	→ Noun	0.2
NP	→ Det Noun	0.4
NP	→ NP PP	0.2
NP	→ NP conj NP	0.2
VP	→ Verb	0.4
VP	→ Verb NP	0.3
VP	→ Verb NP NP	0.1
VP	→ VP PP	0.2
PP	→ P NP	1.0

Hockenmaier

Estimating P(X→α)

- Supervised
  - Relative frequency estimation
  - Count what is seen in a treebank corpus

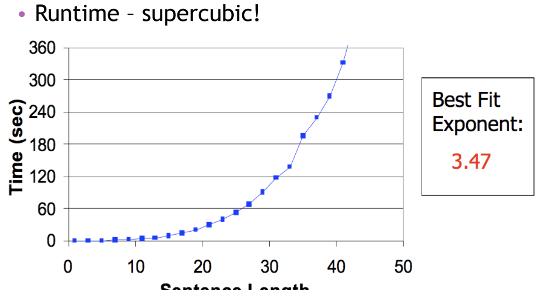
$$P(X \to \alpha) = \frac{C(X \to \alpha)}{C(X)}$$

- Unsupervised
  - Expected relative frequency estimation
  - Use Inside-Outside Algorithm (EM variant)

$$P(X \to \alpha) = \frac{E[C(X \to \alpha)]}{E[C(X)]}$$

How well do PCFGs perform?

- Runtime – supercubic!



Manning

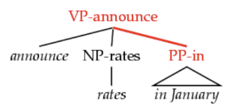
How well do PCFGs perform?

\+ Robust to variations in language
\- Strong independence assumptions
? WSJ parsing accuracy: about 73% LP/LR F1

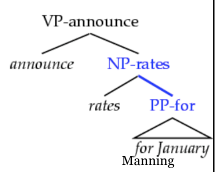
- **Lack of lexicalization**
  - A PCFG uses the actual words only to determine the probability of parts-of-speech (the preterminals)
    - I like to eat cake with white frosting.
    - I like to eat cake with a spork.

Lexicalization

- Lexical heads are important for certain classes of ambiguities (e.g., PP attachment):



- Lexicalizing grammar creates a much larger grammar.
  - Sophisticated smoothing needed
  - Smarter parsing algorithms needed
  - More DATA needed



Manning

Huge area of research

- Coarse-to-fine parsing
  - Parse with a simpler grammar
  - Refine with a more complex one
- Dependency parsing
  - A sentence is parsed by relating each word to other words in the sentence which depend on it.
- Discriminative parsing
  - Given training examples, learn a function that classifies a sentence with its parse tree
- and more!

The good news!

- Part of speech taggers and sentence parsers are freely available!

- So why did we sit through this lecture?
  - Maybe you'll be interested in this area
  - Useful ideas to be applied elsewhere
    - Write a parser to parse web tables
    - PCFGs for information extraction
  - Like to know how things work

It's over!

- Thanks!