Foundational Models of Language II

IPAM Graduate Summer School:
Probabilistic Models of Cognition

Sharon Goldwater

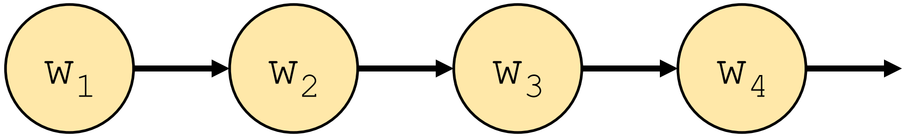
School of Informatics
University of Edinburgh
sgwater@inf.ed.ac.uk

Latent variable models

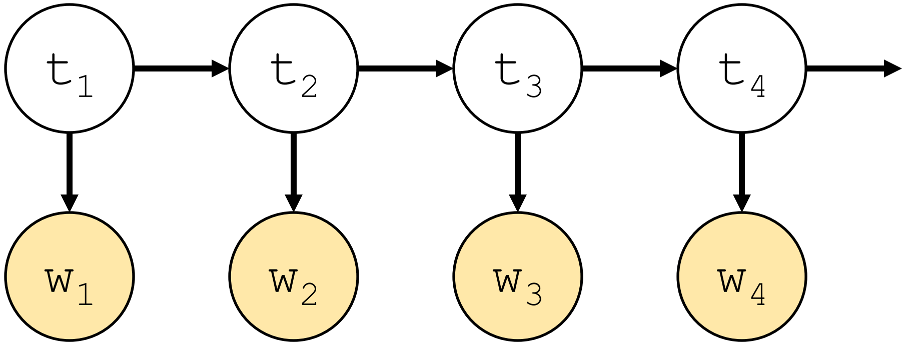
- Previous lecture: Bag-of-words and *n*-gram models.
  - Useful for many purposes, but don't capture the hidden structure language seems to contain.
- This lecture: two common models that do include hidden structure.
  - Hidden Markov model (HMM): sequence model.
  - Probabilistic context-free grammar (PCFG): tree-structured model.

Hidden Markov models

- *N*-gram model: dependencies between observed variables.



- HMM: dependencies only between latent variables representing different classes or categories.



Example uses for HMMs

- Modeling speech.
  - Latent variables represent phonemes.
  - Observed variables are real-valued acoustic data.
- Modeling syntax.
  - Latent variables represent syntactic categories (parts of speech).
  - Observed variables are words.
  - Less accurate model of syntax than PCFG, but often useful in NLP when full parses aren't needed.
  - Open question: do human learners go through a stage where they know some syntactic categories, but have no hierarchical structure?

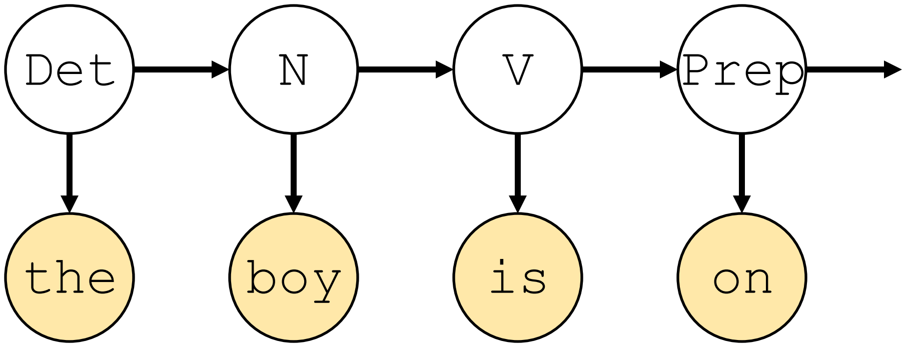
HMM for part-of-speech tagging

- Generative model assigns a probability distribution over sequences of POS tags and words.

$$P(\mathbf{t}, \mathbf{w}) = \prod_{i=1}^{n} P(t_i \mid t_{i-1}) P(w_i \mid t_i)$$

tags $\mathbf{t} = t_1 \ldots t_n$

words $\mathbf{w} = w_1 \ldots w_n$

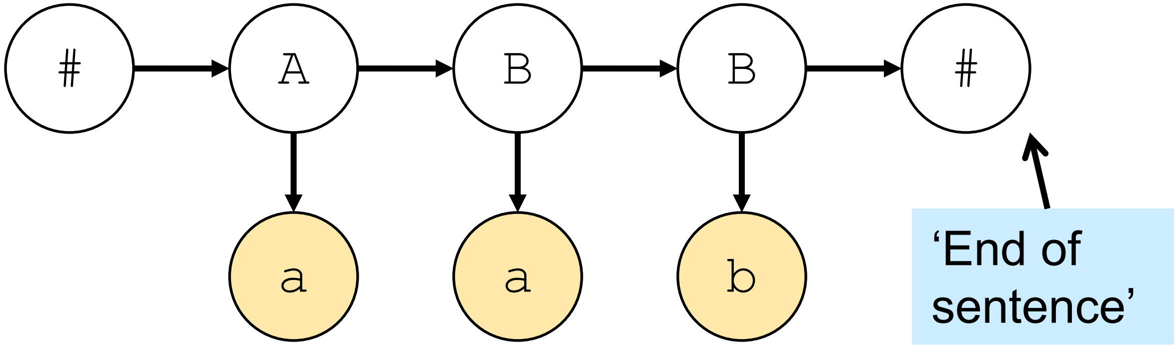


Example

- Assume we know the parameters of the model.

Transitions, $P(t_i|t_j)$:

	A	B	#
A	0.4	0.6	0
B	0.6	0.3	0.1
#	0.5	0.5	0



Outputs, $P(w_i|t_i)$:

	a	b
A	0.7	0.3
B	0.2	0.8

$$P(\mathbf{t}|\theta) = (.5)(.6)(.3)(.1)$$

$$P(\mathbf{w}|\mathbf{t},\theta) = (.7)(.2)(.8)$$

$$P(\mathbf{t},\mathbf{w}|\theta) = P(\mathbf{t}|\theta)\ P(\mathbf{w}|\mathbf{t},\theta)$$

What can we do now?

- Ambiguity resolution: what's the right POS sequence?

Pro	**N/V**	**Det**	**N/V**
I	saw	the	man

  - $P(\mathbf{t} \mid \mathbf{w}, \theta) \propto P(\mathbf{w}, \mathbf{t} \mid \theta)$, so compute $P(\mathbf{t},\mathbf{w}|\theta)$ for each sequence and choose the best.
- Language modeling: what's the probability of the sequence of words?
  - Compute $P(\mathbf{w} \mid \theta) = \sum_{\mathbf{t}} P(\mathbf{t}, \mathbf{w} \mid \theta)$

Parameter estimation

- Supervised learning: training data is annotated with correct POS tags.
  - Can use MLE + smoothing or Bayesian methods, similar to learning *n*-gram model.
- Unsupervised learning: training data is words only.
  - More similar to language acquisition in children.

MLE for HMMs

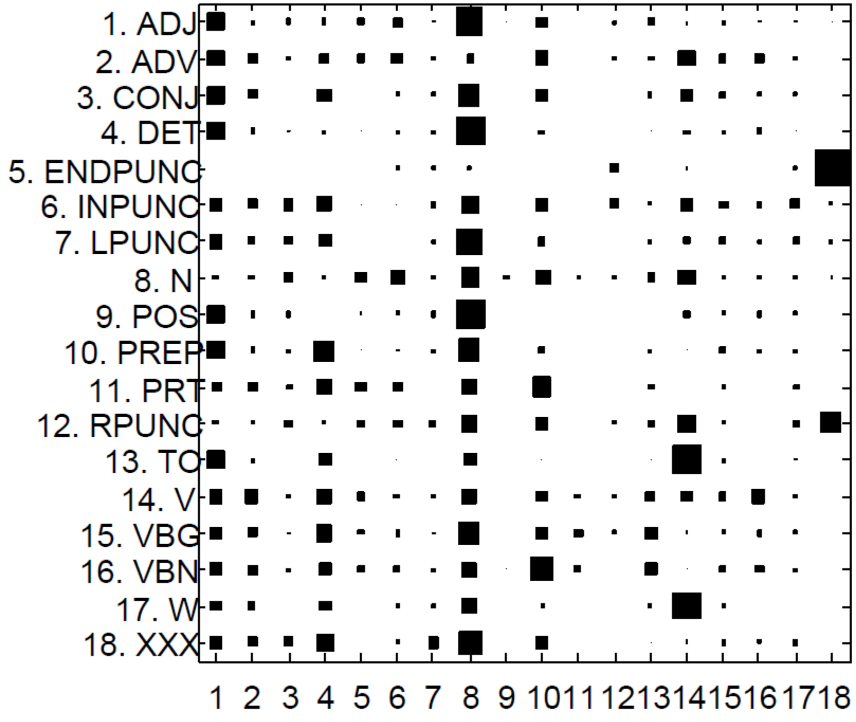
- Choose params θ to maximize $P(\mathbf{w} \mid \theta) = \sum_{\mathbf{t}} P(\mathbf{t}, \mathbf{w} \mid \theta)$.
- Can do this using Expectation-Maximization (EM):
  - Initialize parameters at random.
  - E-step: use current estimate of θ to compute $\mathrm{P}(\mathbf{t}|\mathbf{w}, \theta)$ and expected counts of hidden events, $\mathrm{E}[C(t_i,t_j)]$ and $\mathrm{E}[C(t_i,w_i)]$.
    - non-trivial; uses the forward-backward (Baum-Welch) algorithm.
  - M-step: use expected counts to update θ to maximize $\mathrm{P}(\mathbf{w}| \theta)$.
  - Iterate E and M: converges to local maximum of likelihood.

For more on HMMs and the forward-backward algorithm, see Manning and Schuetze (1999).

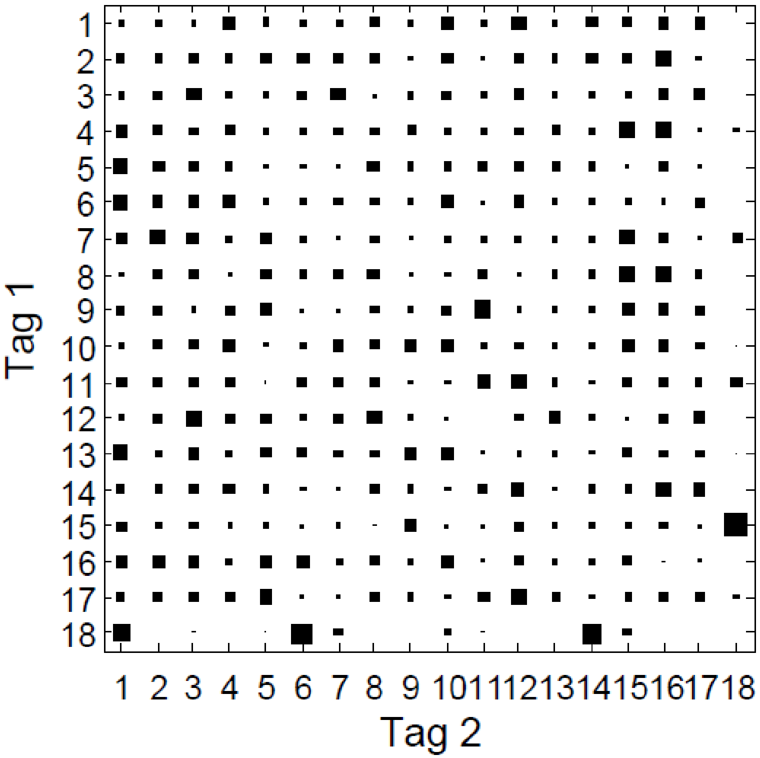
MLE doesn't work

- True POS transition matrix is sparse, MLE matrix isn't.

True:



MLE:



Can being Bayesian help?

- MLE estimates θ in order to
  - Predict next words (language modeling): $P(w_{n+1}|\theta)$.
  - Infer hidden structure (POS tagging): $P(\mathbf{t}|\theta,\mathbf{w})$.
- But actually, what we really want is
  - Predict next words (language modeling): $P(w_{n+1}|\mathbf{w})$.
  - Infer hidden structure (POS tagging): $P(\mathbf{t}|\mathbf{w})$.
- By integrating out θ, we can compute just that.

$$P(w_{n+1} \mid \mathbf{w}) = \int_{\Delta} P(w_{n+1} \mid \theta) P(\theta \mid \mathbf{w}) d\theta$$

$$P(\mathbf{t} \mid \mathbf{w}) = \int_{\Delta} P(\mathbf{t} \mid \theta, \mathbf{w}) P(\theta \mid \mathbf{w}) d\theta$$

Bayesian HMM

- Notation: assume T distinct tags and W distinct words; split θ into two parts, (τ, ω).

$$\omega = \omega^{(1)} \ldots \omega^{(T)} : \text{the output distributions for each tag}$$

$$\omega^{(t)} = \omega_1^{(t)} \ldots \omega_W^{(t)} : \text{the output distribution from tag } t$$

$$\tau = \tau^{(1)} \ldots \tau^{(T)} : \text{the transition distributions for each tag}$$

$$\tau^{(t)} = \tau_1^{(t)} \ldots \tau_T^{(t)} : \text{the transition distribution from tag } t$$

$$\omega^{(\mathtt{N})} : \text{the output distribution of } \mathtt{N} \text{ tag}$$

$$\omega^{(\mathtt{N})} = (\omega_{\mathtt{cat}}^{(\mathtt{N})}, \omega_{\mathtt{dog}}^{(\mathtt{N})}, \omega_{\mathtt{the}}^{(\mathtt{N})}, \ldots)$$

$$= (.012, .004, .00001, \ldots)$$

Bayesian HMM

- Bayesian HMM augments standard HMM with symmetric Dirichlet priors:

$$
\begin{aligned}
t_i \mid t_{i-1} = t, \tau^{(t)} &\sim \mathrm{Multinomial}(\tau^{(t)}) \\
w_i \mid t_i = t, \omega^{(t)} &\sim \mathrm{Multinomial}(\omega^{(t)}) \\
\tau^{(t)} \mid \alpha &\sim \mathrm{Dirichlet}(\alpha) \\
\omega^{(t)} \mid \beta &\sim \mathrm{Dirichlet}(\beta)
\end{aligned}
$$

Predictive distributions

- If we integrate out parameters, we get

$$P(t_{n+1} \mid \mathbf{t}, \alpha) = \frac{n_{(t_n, t_{n+1})} + \alpha}{n_{(t_n)} + T\alpha}$$

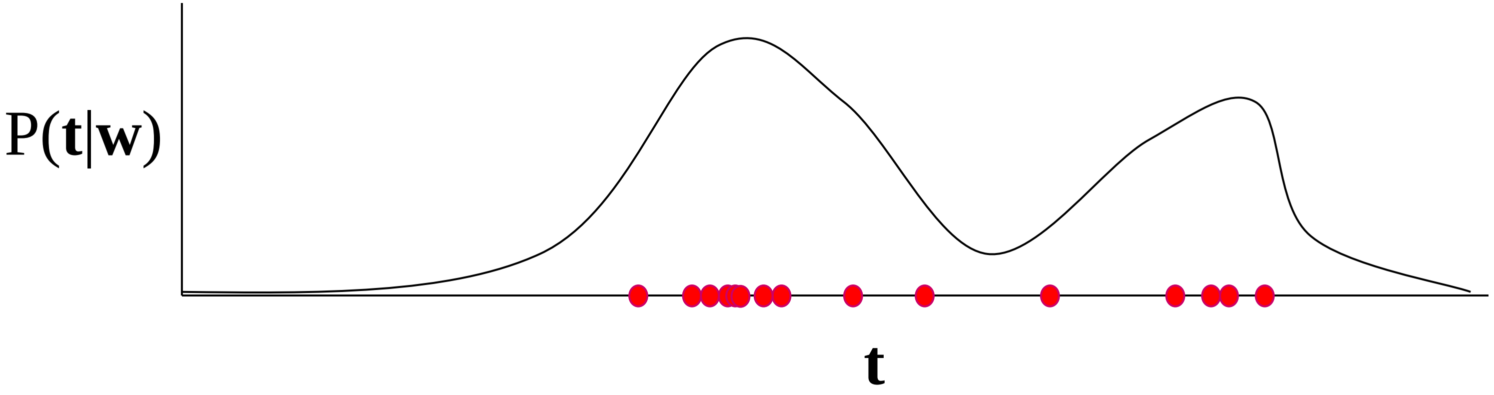
$$P(w_{n+1} \mid t_{n+1}, \mathbf{t}, \mathbf{w}, \beta) = \frac{n_{(t_{n+1}, w_{n+1})} + \beta}{n_{(t_{n+1})} + W_{t_{n+1}}\beta}$$

  - assuming T possible tags and W_t possible words with tag t.
- We can use these distributions to find $\mathrm{P}(\mathbf{t}|\mathbf{w})$ using Markov Chain Monte Carlo, specifically Gibbs sampling.*

*We could instead use Variational Bayes, which is a generalization of EM and, like EM, uses the forward-backward algorithm. See Johnson (2007).

Gibbs sampling

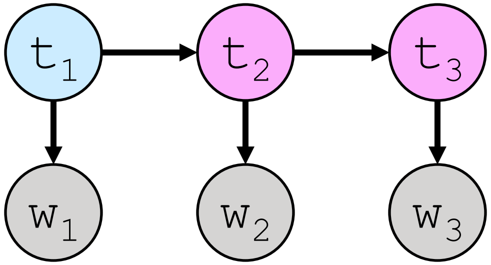
- Like EM, a general technique (algorithm family), specific versions for specific models.
- A randomized algorithm.
  - Guaranteed to converge.
  - After convergence, each iteration produces a sample from the posterior distribution of interest (here, $P(\mathbf{t}|\mathbf{w})$).



For more on Gibbs sampling and parameter integration as applied to NLP problems, see Resnik and Hardisty (2009).

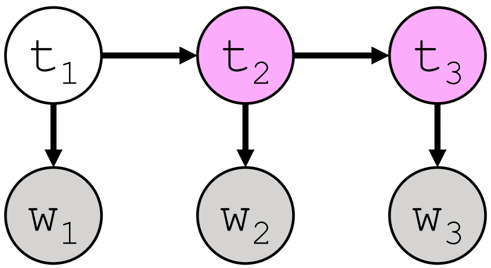
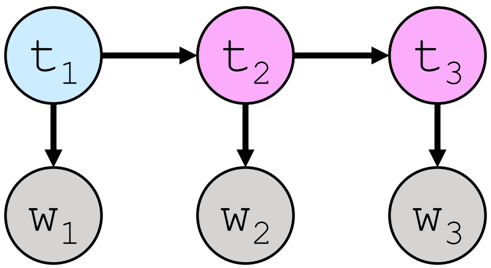
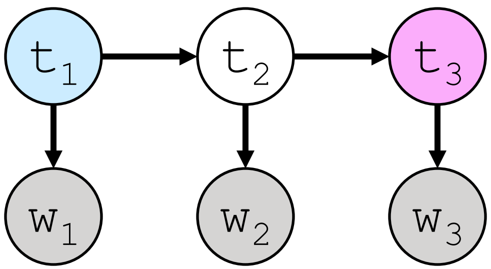
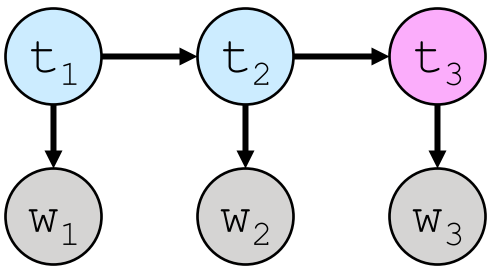
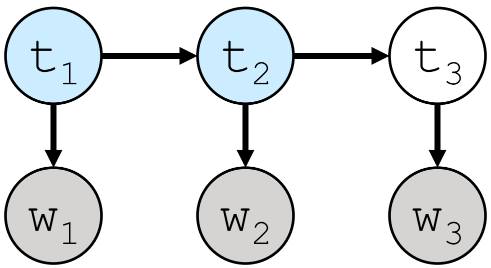
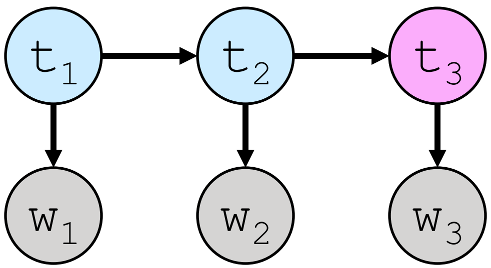
Gibbs sampling

1. Initialize hidden variables (tags) at random:



2. On each iteration, sample a new value for each tag from its distribution conditioned on the current values of all other variables (both tags and words).
   - Basically, pretend all the other tags are correct, treat them as training data, and sample a new value for current tag.

One iteration

	Sample from:	Result:
1. 	$P(t_1 \mid t_2, t_3, \mathbf{w}, \alpha, \beta)$	
2. 	$P(t_2 \mid t_1, t_3, \mathbf{w}, \alpha, \beta)$	
3. 	$P(t_3 \mid t_1, t_2, \mathbf{w}, \alpha, \beta)$	

Computing conditional distributions

- By Bayes' rule, we have

$$
\begin{aligned}
&P(t_1 \mid t_2, t_3, w_1, w_2, w_3, \alpha, \beta) \\
&\quad \propto P(w_1 \mid t_1, t_2, t_3, w_2, w_3, \alpha, \beta) P(t_1 \mid t_2, t_3, w_2, w_3, \alpha, \beta) \\
&\quad = P(w_1 \mid t_1, t_2, t_3, w_2, w_3, \beta) P(t_1 \mid t_2, t_3, \alpha)
\end{aligned}
$$

- Now pretend that t_i and w_i are the last tag/word in the sequence, and use our formulas from earlier to compute the two factors*:

$$
P(w_{n+1} \mid t_{n+1}, \mathbf{t}, \mathbf{w}, \beta) = \frac{n_{(t_{n+1}, w_{n+1})} + \beta}{n_{(t_{n+1})} + W_{t_{n+1}} \beta} \qquad P(t_{n+1} \mid \mathbf{t}, \alpha) = \frac{n_{(t_n, t_{n+1})} + \alpha}{n_{(t_n)} + T\alpha}
$$

*We are only allowed to pretend this because the model is exchangeable: probabilities are the same for any ordering of the variables, e.g. P(a,b,c) = P(a,c,b).

Bayesian HMM in practice

- Goldwater and Griffiths (2007) compared BHMM+Gibbs to standard HMM+MLE (= MLHMM).
- Two scenarios:
  - Constrained: provide dictionary listing possible tags for each word (e.g. ‘run’ is a noun or verb).

    53.6% of words have multiple possible tags.
    Average number of tags per word = 2.3.

  - Unconstrained: any word can have any tag (17 possibilities).
- Train and test on unlabeled corpus (24k words of WSJ).
  - Results evaluated against gold standard POS tags.

Accuracy results (full dictionary)

MLHMM	74.7%
BHMM ($\alpha = 1, \beta = 1$)	83.9%
BHMM (best: $\alpha = .003, \beta = 1$)	86.8%
CRF/CE	90.1%

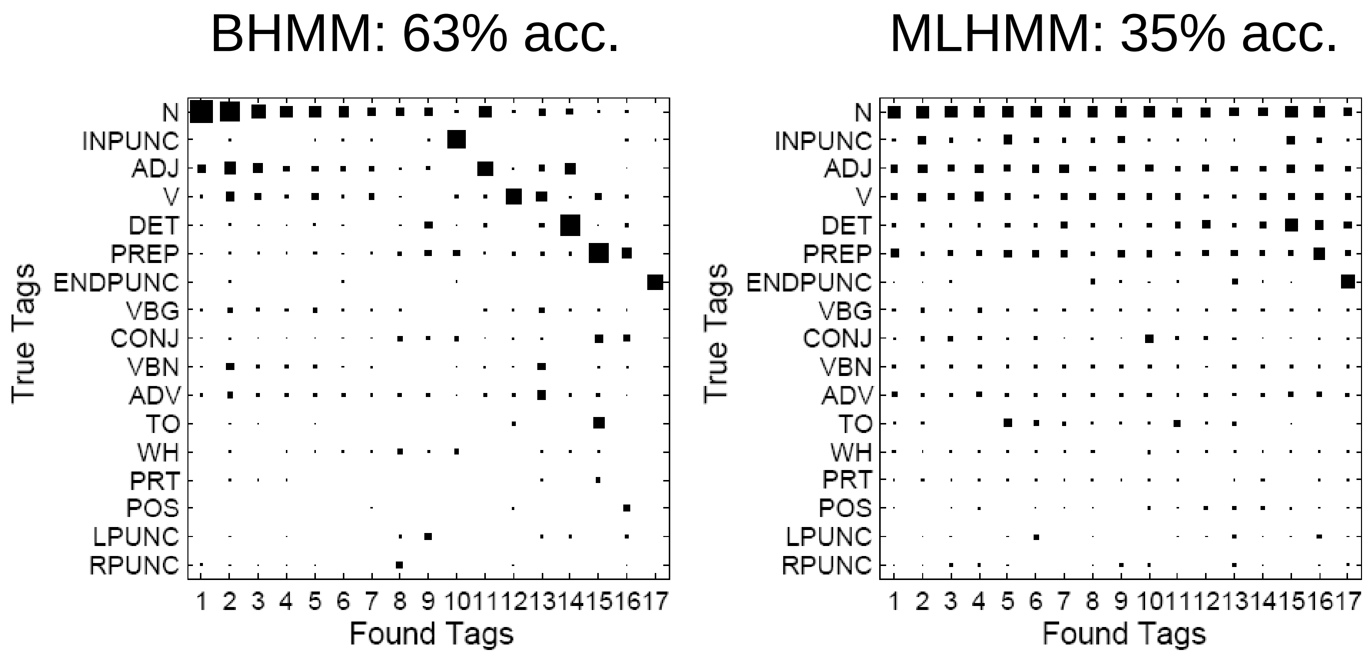
State-of-the-art system (Smith and Eisner, 2005) →

- Integrating over parameters is useful in itself, even with uninformative priors ($\alpha = \beta = 1$).
- Better priors can help even more, though not quite to state-of-the-art.

Syntactic clustering

- No tag dictionary, any word can have any tag.
- Hyperparameters (α, β) are inferred automatically using Metropolis-Hastings sampler.
- Standard accuracy measure requires labeled classes, so measure using best matching of classes.

Clustering results

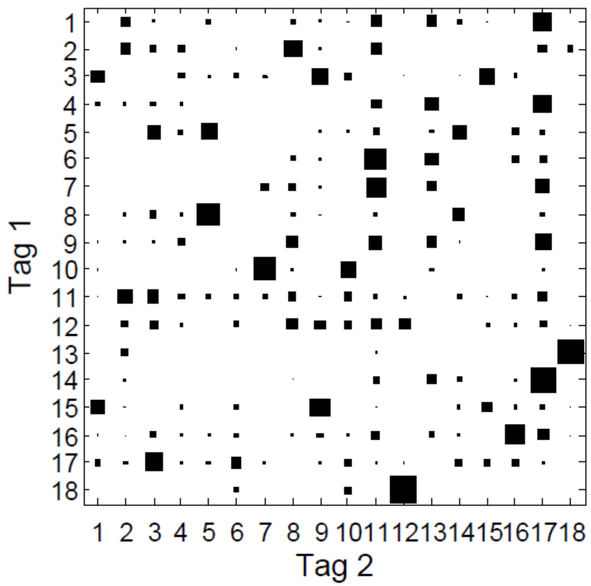


- MLHMM groups instances of the same lexical item together.
- BHMM clusters are more coherent, more variable in size. Errors are often sensible (e.g. separating common nouns/proper nouns, confusing determiners/adjectives, prepositions/participles).

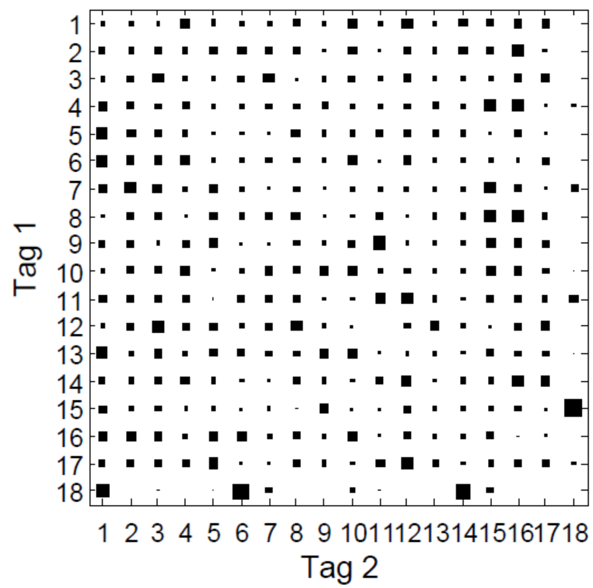
Learned distributions

- BHMM transition matrix is sparse, MLHMM is not.

BHMM



MLHMM

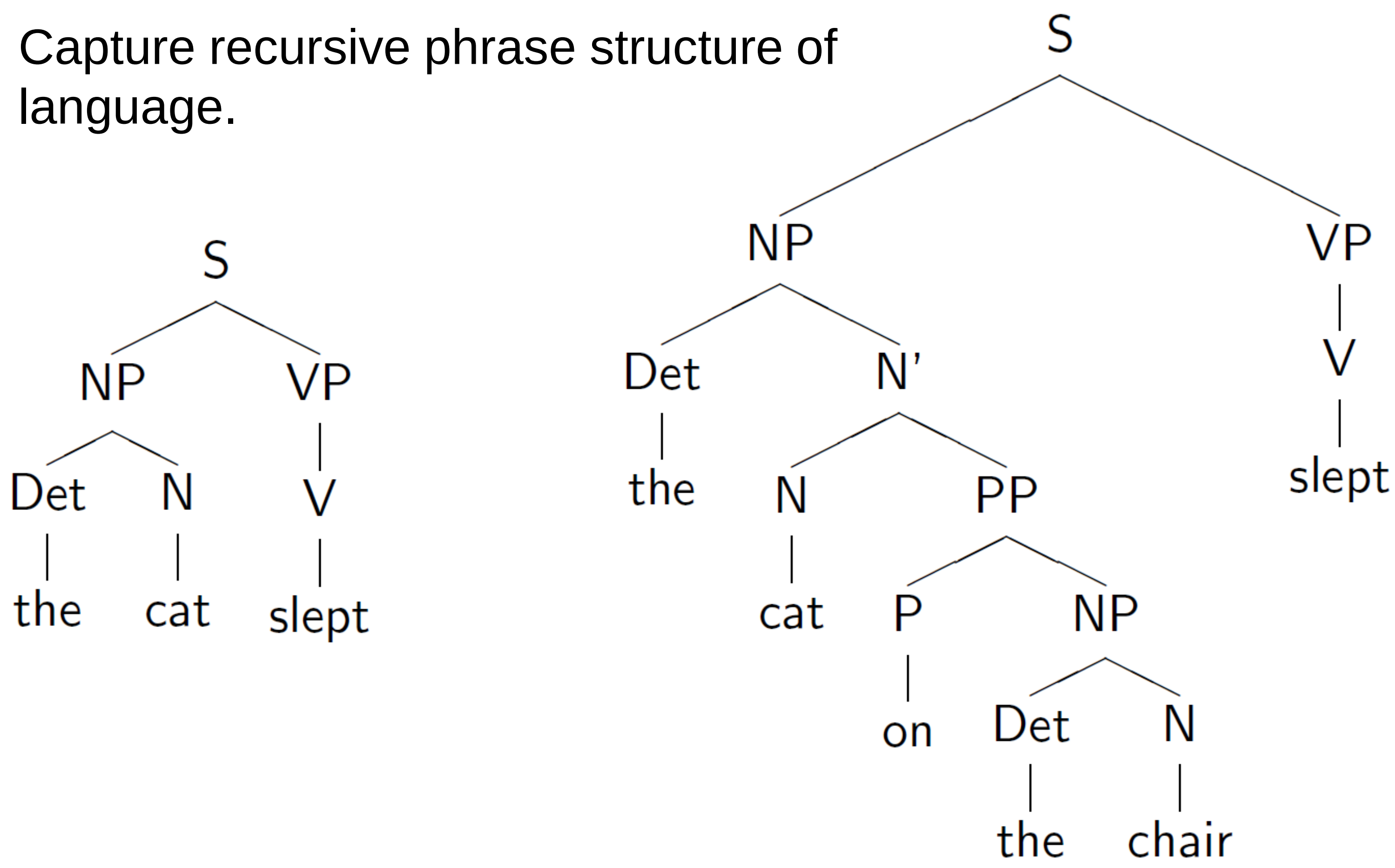


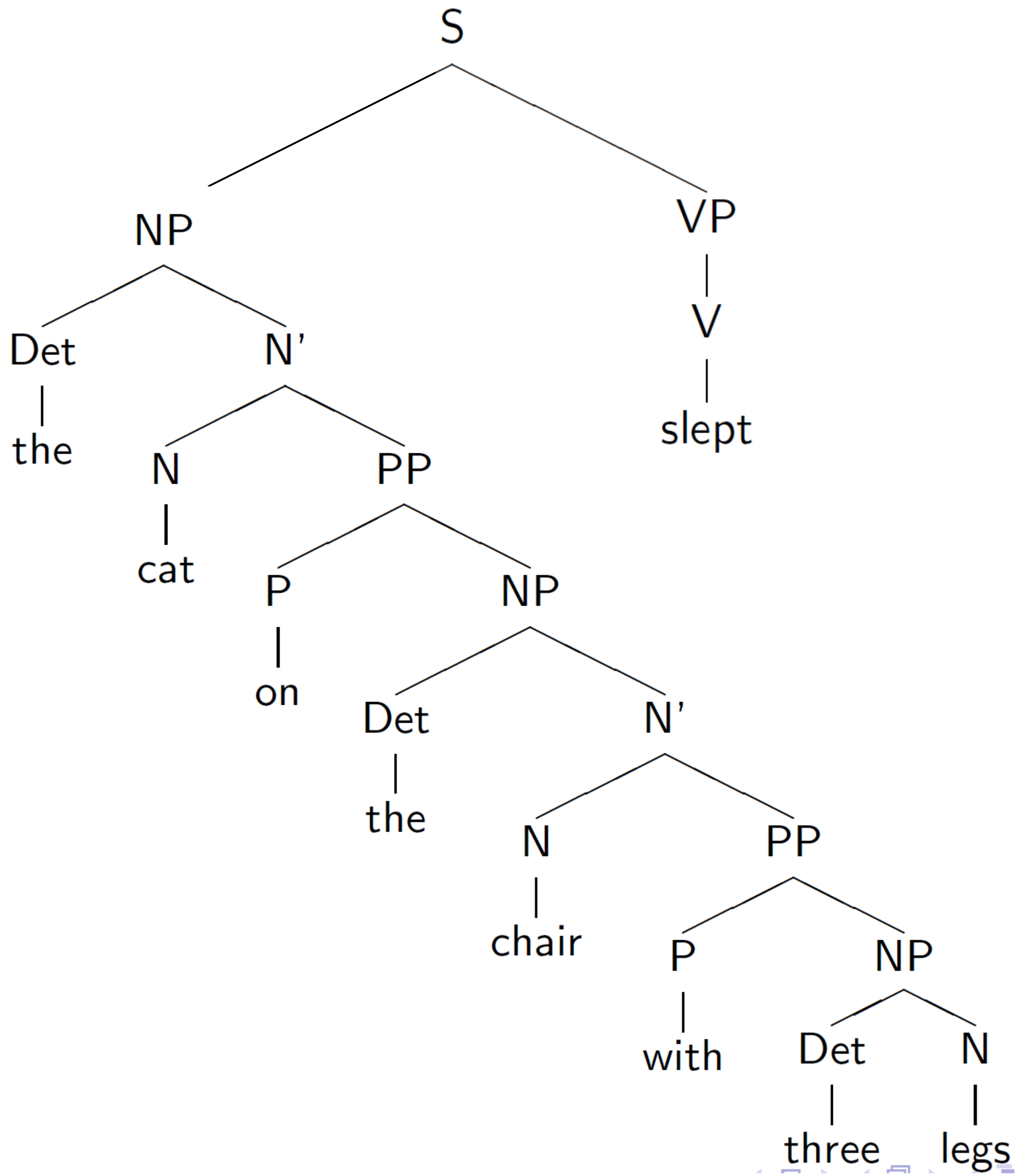
HMM summary

- Sequence model with latent variables representing classes, widely used in language (and elsewhere).
- Can be used for language modeling: $\mathrm{P}(\mathbf{w})$.
- Normally used for inferring hidden structure: $\mathrm{P}(\mathbf{t}|\mathbf{w})$, in POS tagging, speech recognition, etc.
- For unsupervised learning of latent variables, Bayesian methods work better than MLE.

Probabilistic context-free grammars

- Tree-structured latent variable models.
  - Capture recursive phrase structure of language.




S
NP
VP
Det
N'
V
the
N
PP
slept
cat
P
NP
on
Det
N'
the
N
PP
chair
P
NP
with
Det
N
three
legs

Example PCFG

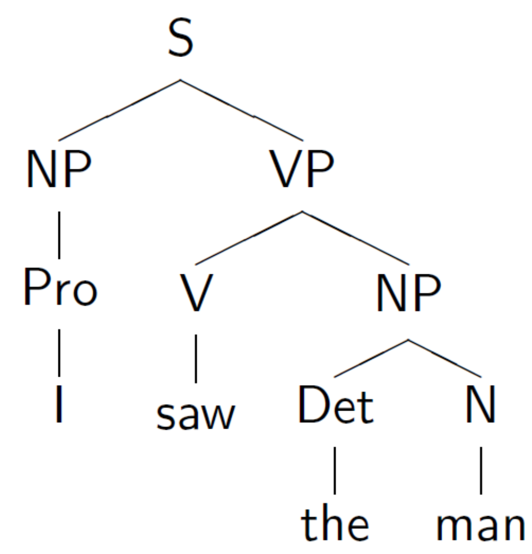
Rule	Prob	Rule	Prob
S $\rightarrow$ NP VP	1.0	V $\rightarrow$ saw	0.5
VP $\rightarrow$ V NP	0.6	V $\rightarrow$ slept	0.5
VP $\rightarrow$ VP PP	0.4	N $\rightarrow$ saw	0.1
NP $\rightarrow$ Pro	0.3	N $\rightarrow$ man	0.3
NP $\rightarrow$ Det N	0.5	N $\rightarrow$ glasses	0.6
NP $\rightarrow$ NP PP	0.2	Det $\rightarrow$ the	1.0
PP $\rightarrow$ P NP	1.0	P $\rightarrow$ with	1.0
		Pro $\rightarrow$ I	1.0

- $\mathrm{P}(\mathrm{X} \rightarrow \mathrm{Y}) = z$ means $\mathrm{P}(\mathrm{RHS\ is\ Y} \mid \mathrm{LHS\ is\ X}) = z$.
- Parameters θ are the rule probabilites.

Probability of a parse

- For parse tree t, $P(t, \mathbf{w} \mid \theta) = \prod_{rule \in t} P(rule \mid \theta)$

Rule	Prob	Rule	Prob
S $\rightarrow$ NP VP	1.0	V $\rightarrow$ saw	0.5
VP $\rightarrow$ V NP	0.6	V $\rightarrow$ slept	0.5
VP $\rightarrow$ VP PP	0.4	N $\rightarrow$ saw	0.1
NP $\rightarrow$ Pro	0.3	N $\rightarrow$ man	0.3
NP $\rightarrow$ Det N	0.5	N $\rightarrow$ glasses	0.6
NP $\rightarrow$ NP PP	0.2	Det $\rightarrow$ the	1.0
PP $\rightarrow$ P NP	1.0	P $\rightarrow$ with	1.0
		Pro $\rightarrow$ I	1.0

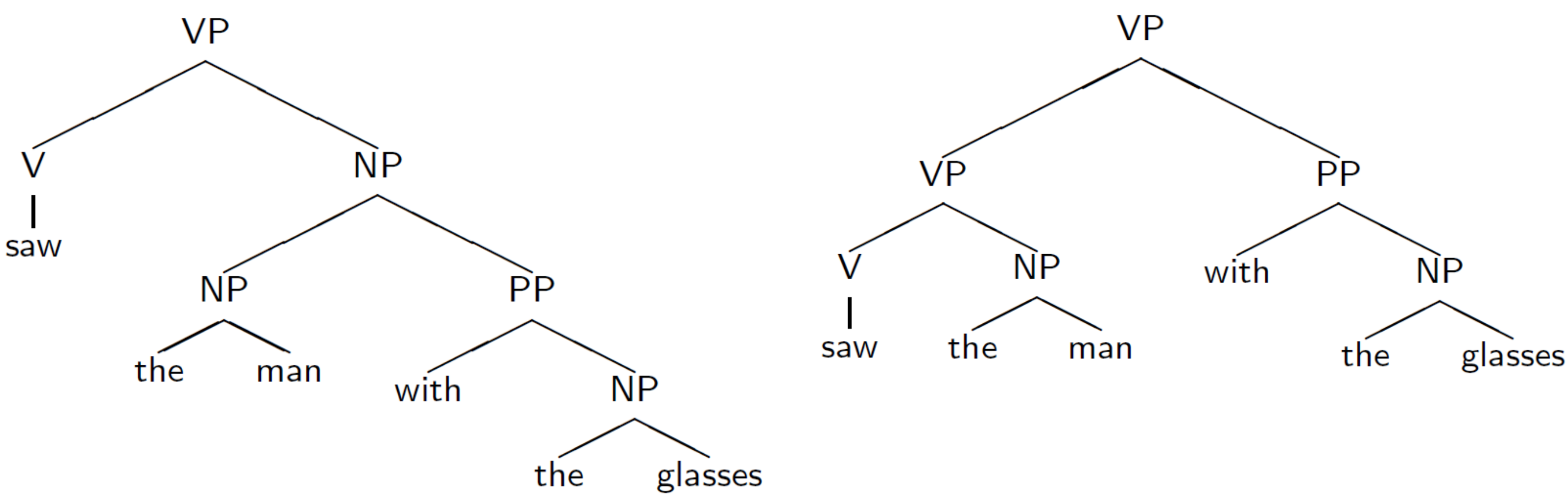


$$P(t,\mathbf{w}|\theta) = (1.0)(0.3)(1.0)(0.6)(0.5)(0.5)(1.0)(0.3)$$

Language modeling

- A string of words may have more than one parse, so

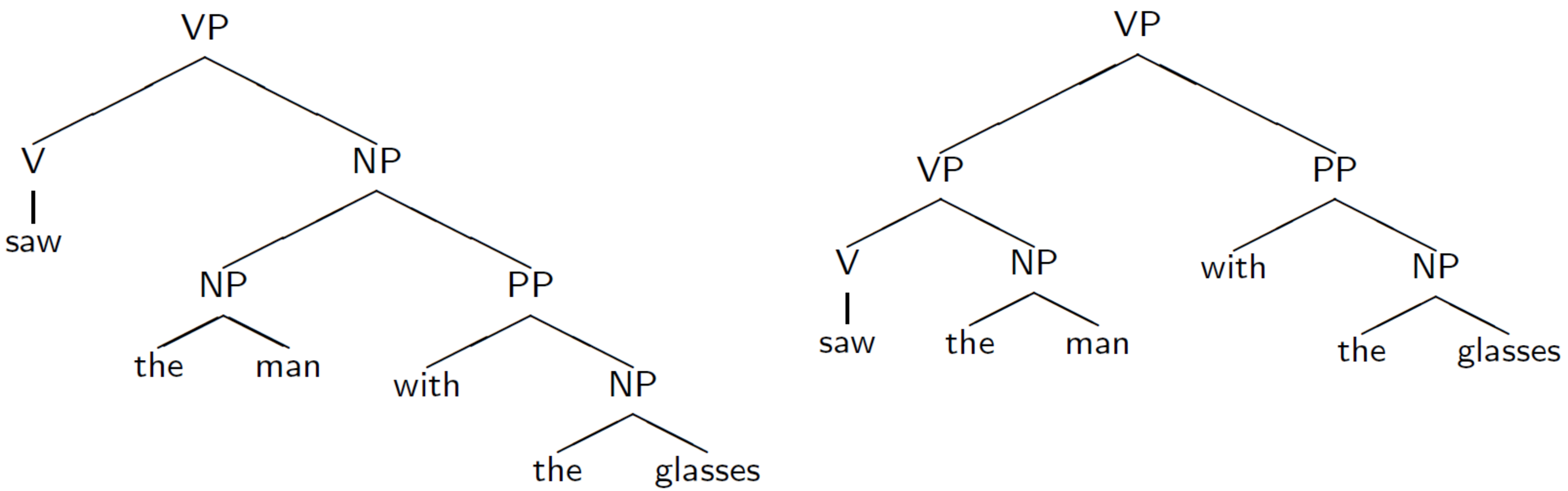
$$P(\mathbf{w} \mid \theta) = \sum P(t, \mathbf{w} \mid \theta)$$



Disambiguation

- Can also compute which parse is more probable:

$$P(t \mid \mathbf{w}, \theta) \propto P(t, \mathbf{w} \mid \theta)$$



- Assumption: different parses indicate different meanings.

Local ambiguity

- Previous examples showed global ambiguity:

> I saw the man with the glasses.
> Look at the cat on the chair with three legs.

- Many psychologists are interested in how humans resolve local ambiguity:

> The chief … [problem is…]
> [said that…]
>
> The player tossed the ball … [to her teammate]
> [by her teammate stumbled]

  - More on this next week.

Learning a PCFG

- Like HMMs, PCFGs can be trained from annotated or unannotated data.
  - Methods are generalizations of HMM training procedures.
  - MLE again uses EM (inside-outside algorithm), again not very effective.
  - Bayesian training uses sampling, but more involved than HMM Gibbs sampler. (probably more on this next week).

For more on PCFGs and the inside-outside algorithm, see Manning and Schuetze (1999). For more on Bayesian PCFGs and PCFG sampling methods, see Johnson et al. (2007).

Issues with PCFGs

- Standard categories don't encode enough information.

VP $\rightarrow$ V	V $\rightarrow$ slept
VP $\rightarrow$ V NP	V $\rightarrow$ chased
VP $\rightarrow$ V NP PP	V $\rightarrow$ put

The girl slept
The girl slept the dog
The girl slept the book on the table

- Consider using subcategories, e.g., V_{IT}, V_T, V_{IP}. But how many/which subcategories to choose?
  - Nonparametric Bayesian models can help with this problem.*

*See Liang et al. (2007), Finkel et al. (2007).

Issues with PCFGs

- *N*-gram models are lexicalized, capturing some semantic information. PCFGs aren't – should they be?

The girl walked the dog	[??] The girl read the dog
The girl read the book	[??] The girl walked the book

- For years, grammars used in NLP have included partially lexicalized rules.
  - More complex lexicalized models have more sparse data and more complex smoothing methods.
  - Nonparametric Bayesian models can help with this too!*

*See Cohn et al. (2010).

PCFG summary

- Tree-structured model with latent variables representing phrase types.
- Like HMM, can be used for language modeling but normally used for inferring hidden structure.
  - In NLP: part of the pipeline for tasks like translation, question answering, many others.
  - In Cog Sci: basis of many human sentence processing models.
- Recent work using Bayesian methods (esp. nonparametric) extends PCFGs in interesting and useful ways.

References

- Trevor Cohn, Phil Blunsom, and Sharon Goldwater. Inducing Tree-Substitution Grammars. *Journal of Machine Learning Research* 11(Nov), pp. 3053-3096, 2010.
- Jenny Rose Finkel, Trond Grenager and Christopher D. Manning. 2007. The infinite tree. *Proceedings of ACL*.
- Sharon Goldwater and Thomas L. Griffiths. 2007. A fully Bayesian approach to unsupervised part-of-speech tagging. *Proceedings of ACL.*
- Mark Johnson. 2007. Why doesn't EM find good HMM POS-taggers? *Proceedings of EMNLP-CoNLL.*
- Mark Johnson, Thomas L. Griffiths, and Sharon Goldwater. 2007. Bayesian inference for PCFGs via Markov chain Monte Carlo. *Proceedings of ACL*.
- Percy Liang, Slav Petrov, Michael I. Jordan, and Dan Klein. 2007. The infinite PCFG using hierarchical Dirichlet processes. *Proceedings of EMNLP-CoNLL*.
- Christopher D. Manning and Hinrich Schütze. 1999. *Foundations of Statistical Natural Language Processing.* Cambridge, MA: MIT Press.
- Philip Resnik and Eric Hardisty. 2009. Gibbs sampling for the uninitiated. Version 0.3. Available at http://www.umiacs.umd.edu/~resnik/pubs/gibbs.pdf
- Noah Smith and Jason Eisner. 2005. Contrastive estimation: Training log-linear models on unlabeled data. *Proceedings of ACL.*