Language and Statistics II

Lecture 20: Contrastive estimation

Noah Smith

Administrivia

- Your drafts: hopefully by Thursday
- Email me a 3-best list for presentation times:
  - 11/28 3:00pm
  - 11/28 3:30pm
  - 11/30 3:00pm
  - 11/30 3:30pm
  - 12/5 3:00pm
  - 12/5 3:30pm
  - 12/7 3:00pm
  - 12/7 3:30pm

Today’s Lecture is a Bit Different

- Adapted from some talks in 2005
- Apologies for the heavy styling

“Red leaves don’t hide blue jays.”

What's a sequence model?

Let ***X*** be a random variable over Σ* (**x** represents a value of ***X***):

x = red | leaves | don't | hide | blue | jays

Markov (*n*-gram) models

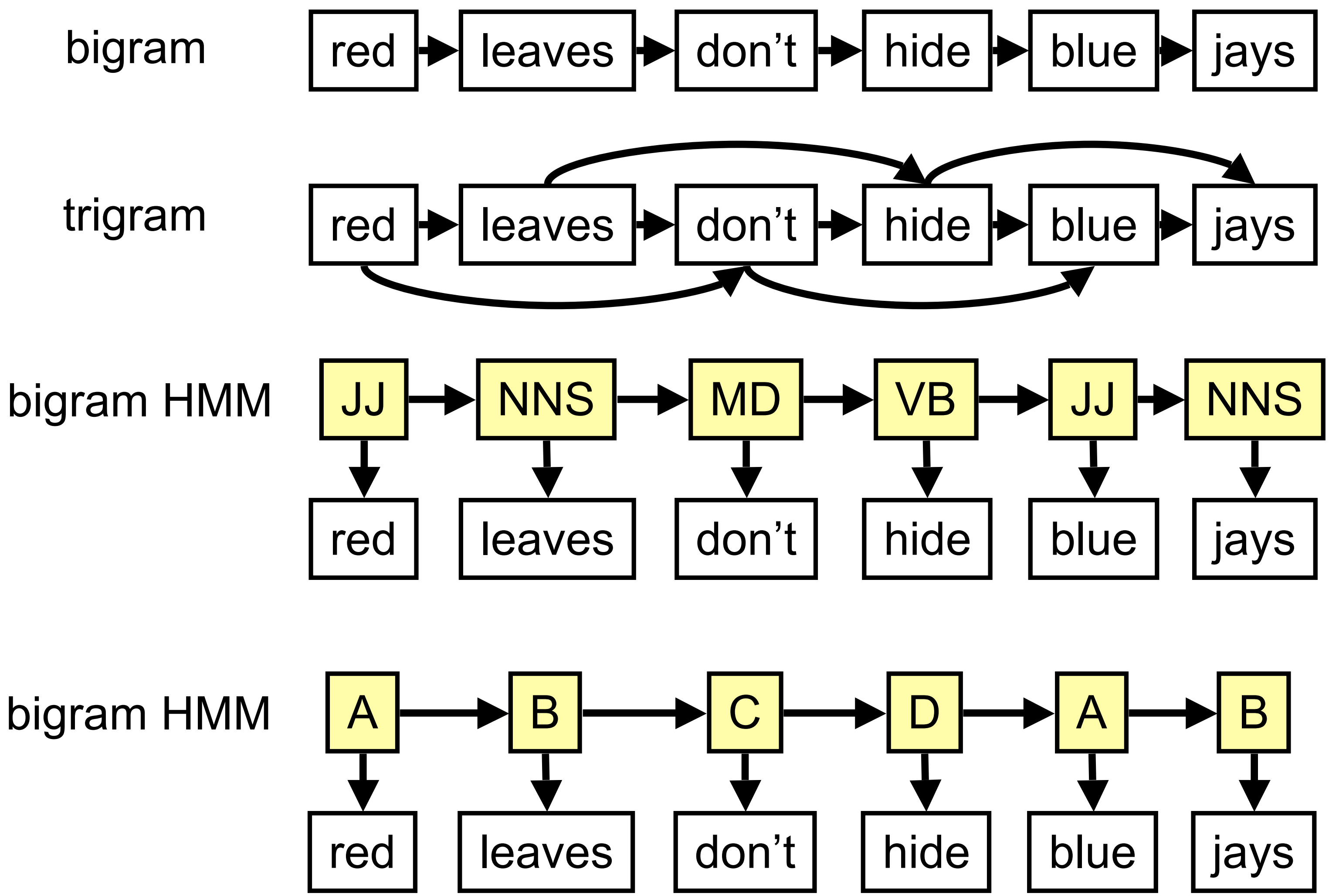
These are all **log-linear** models.

Can add *hidden* variables to the model,
like labels, parse trees, etc.
Call the hidden part ***Y***.

HMMs

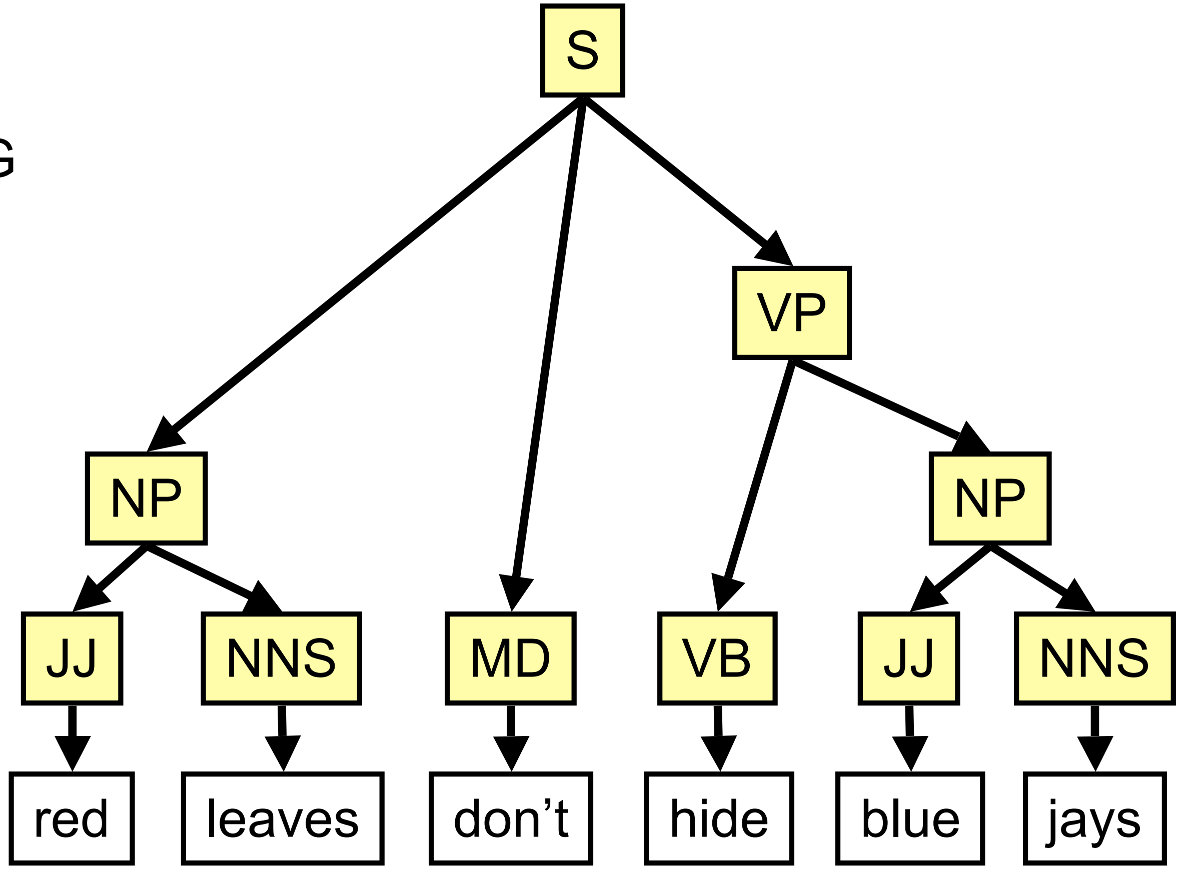
PCFGs

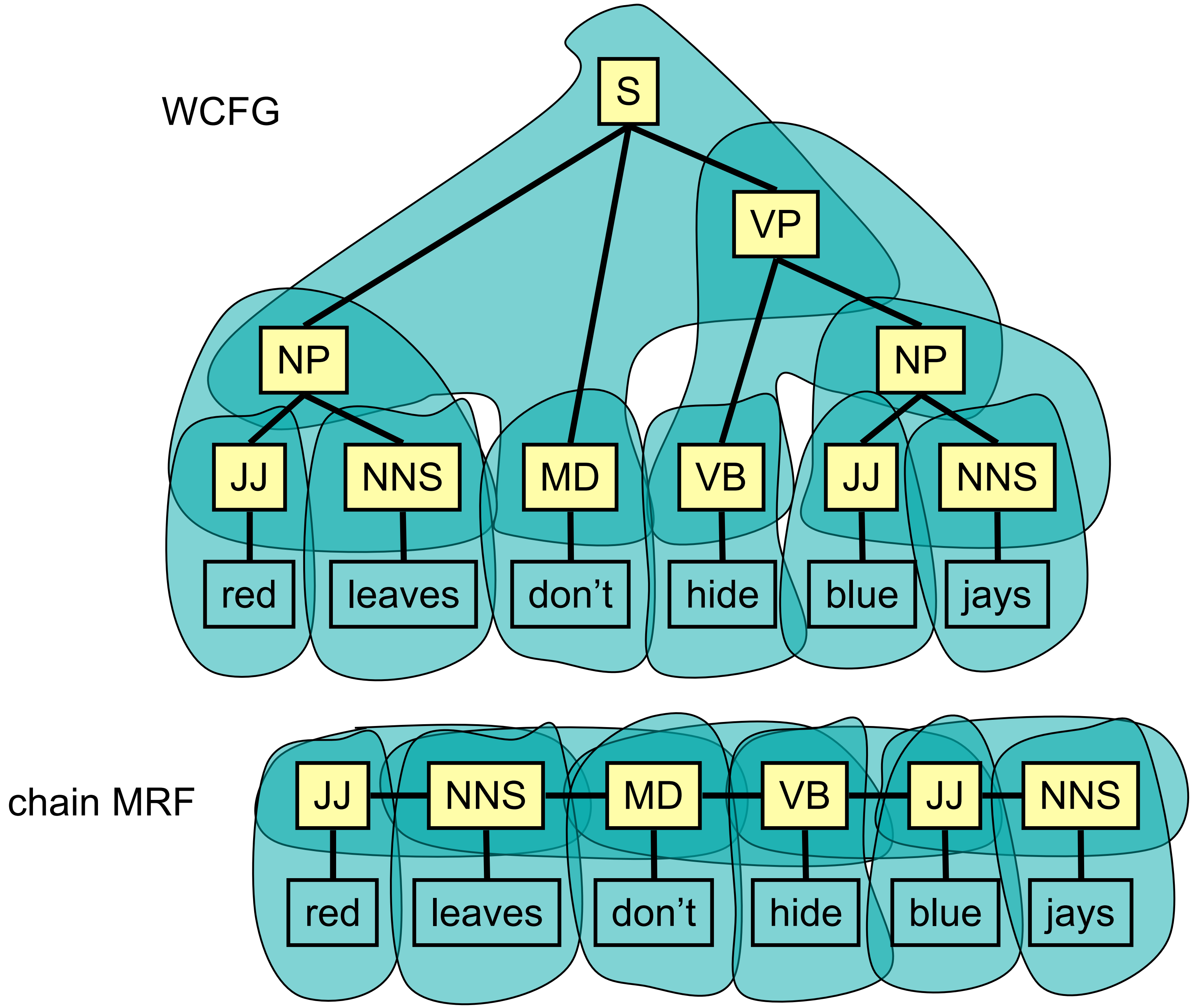
Sequence Models (Finite-State)



Sequence Models (Context-Free)

PCFG



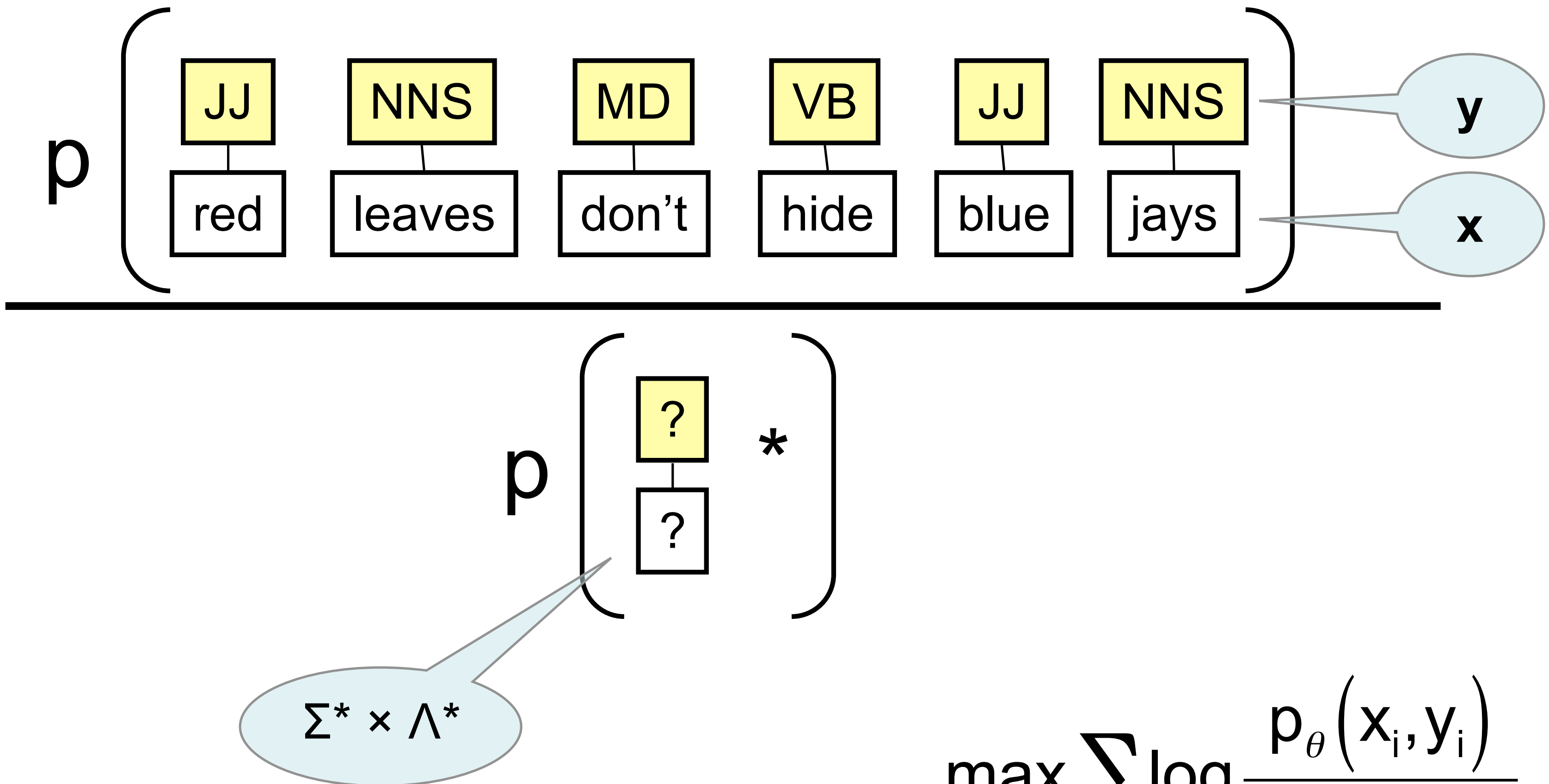

WCFG
S
VP
NP
NP
JJ
NNS
MD
VB
JJ
NNS
red
leaves
don't
hide
blue
jays
chain MRF
JJ
NNS
MD
VB
JJ
NNS
red
leaves
don't
hide
blue
jays

model class $\neq$ estimation method

- *n*-gram models
- HMMs
- "chain" MRFs
- WFSAs
- PCFGs
- WCFGs

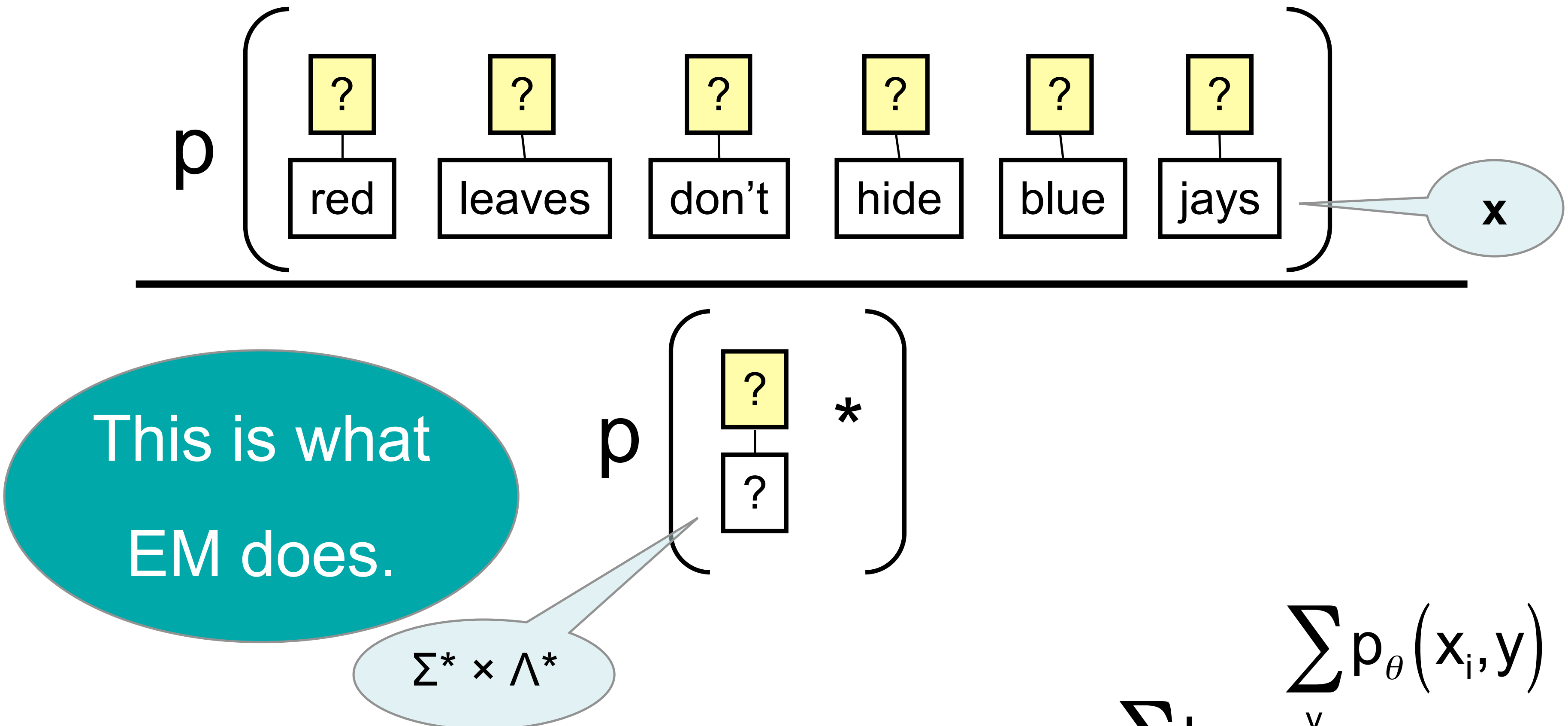
- MLE
- conditional likelihood
- boosting
- perceptron
- maximum margin

Maximum Likelihood Estimation (Supervised)



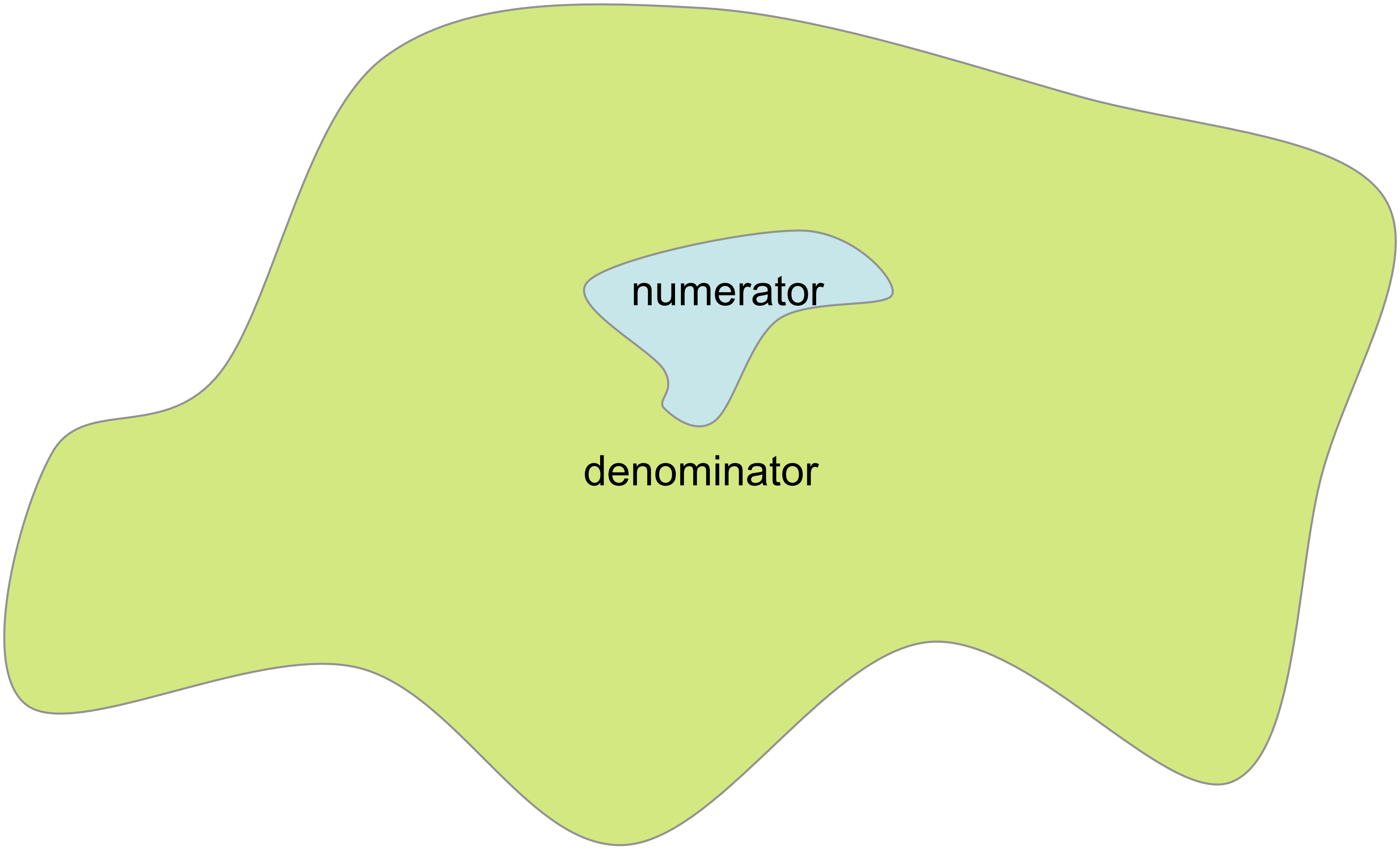
$$\max_{\theta} \sum_{i} \log \frac{p_{\theta}\left(x_i, y_i\right)}{\sum_{x,y} p_{\theta}\left(x, y\right)}$$

Maximum Likelihood Estimation (Unsupervised)

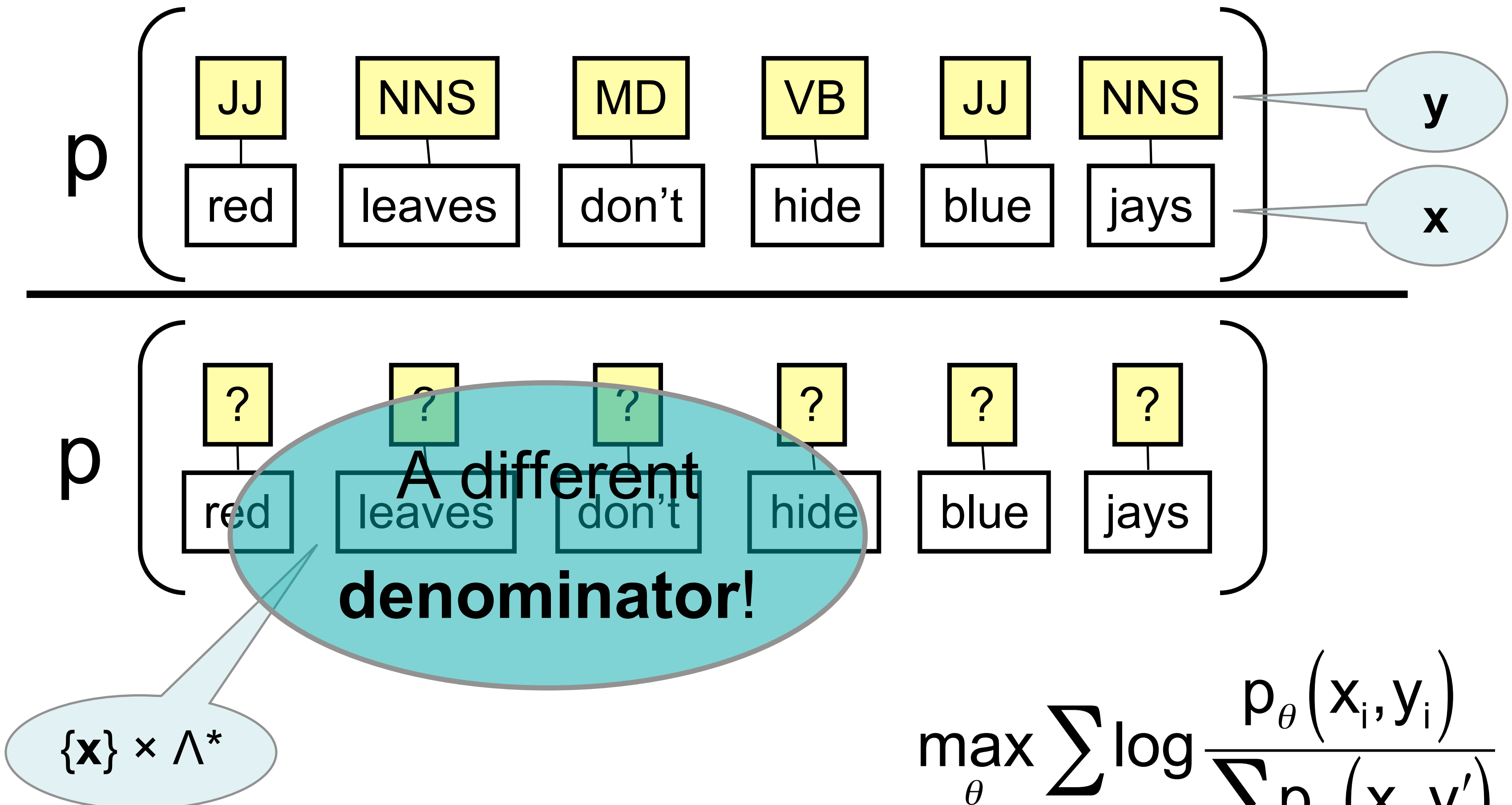


$$\max_{\theta} \sum_{i} \log \frac{\sum_{y} p_{\theta}(x_i, y)}{\sum_{x', y'} p_{\theta}(x', y')}$$

Focusing Probability Mass



Conditional Estimation (Supervised)



$$\max_{\theta} \sum_{i} \log \frac{p_{\theta}\left(x_i, y_i\right)}{\sum_{y'} p_{\theta}\left(x_i, y'\right)}$$

Objective Functions

Objective		Numerator	Denominator
MLE		tags & words	$\Sigma^* \times \Lambda^*$
MLE with hidden variables		words	$\Sigma^* \times \Lambda^*$
Conditional Likelihood		tags & words	(words) $\times \Lambda^*$
Maximum Margin		≈ tags & words	≈ hypothesized tags & words

Objective Functions

Objective	**Optimization Algorithm**	**Numerator**	**Denominator**
MLE	Count & Normalize*	tags & words	$\Sigma^* \times \Lambda^*$
MLE with hidden variables	EM*	words	$\Sigma^* \times \Lambda^*$
Conditional Likelihood	Iterative Scaling	tags & words	(words) $\times \Lambda^*$
Maximum Margin	Perceptron	≈ tags & words	≈ hypothesized tags & words

Objective Functions

Objective	Optimization Algorithm	Numerator	Denominator
Contrastive Estimation	generic numerical solvers (in this talk, LMVM L-BFGS)	observed data (in this talk, raw word sequence, sum over *all possible* values of *Y*)	?

This talk is about **denominators** ...

in the **unsupervised case**.

A good denominator can improve

accuracy

and

tractability.

MLE/EM as a Teacher

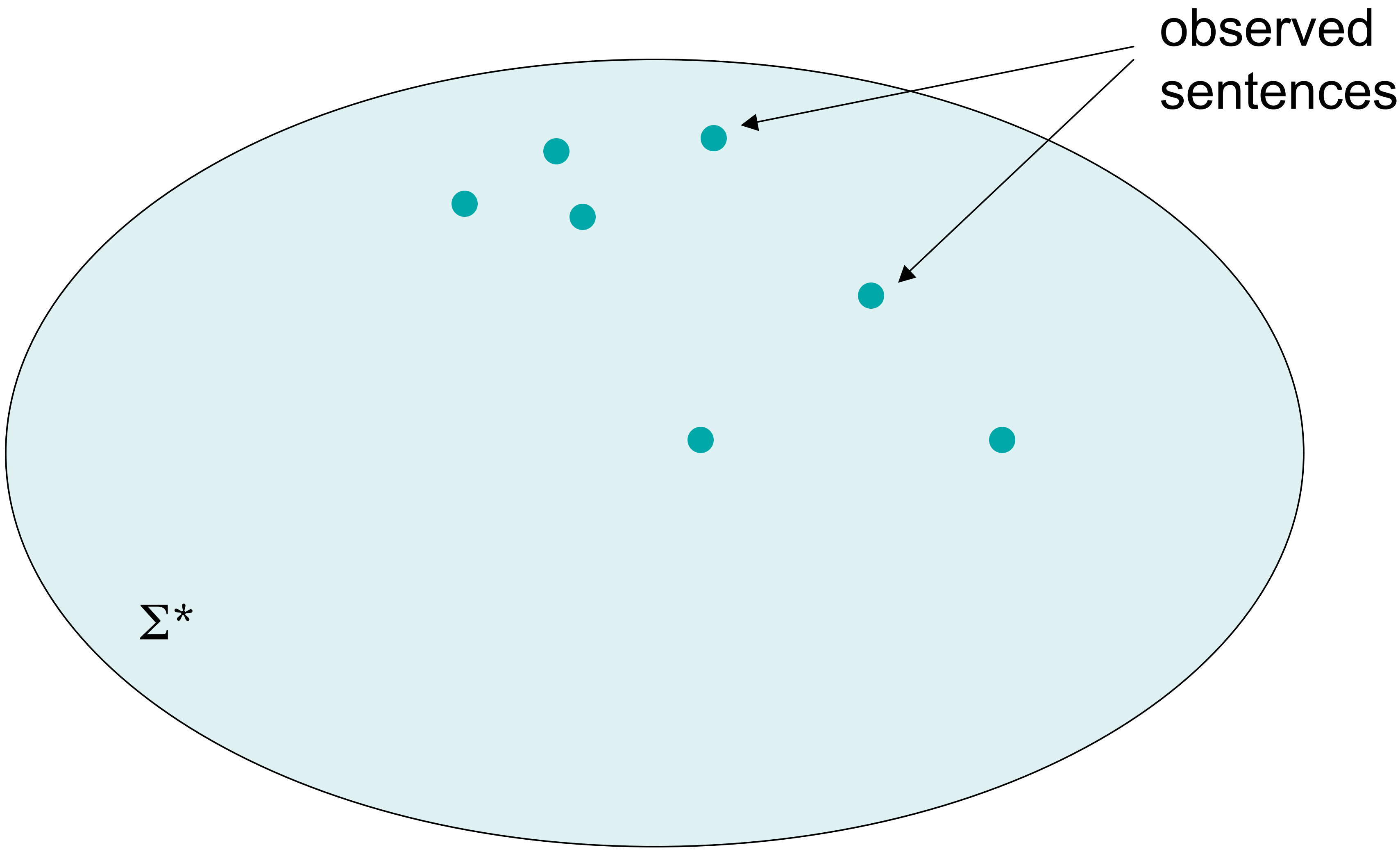
Red leaves don't hide blue jays.

Mommy doesn't love you.

Dishwashers are a dime a dozen.

Dancing granola doesn't hide blue jays.

Probability Allocation



What We'd Like

- Focus on the model on the properties of the data that will lead to an explanation of syntax.

Red leaves don't hide blue jays.

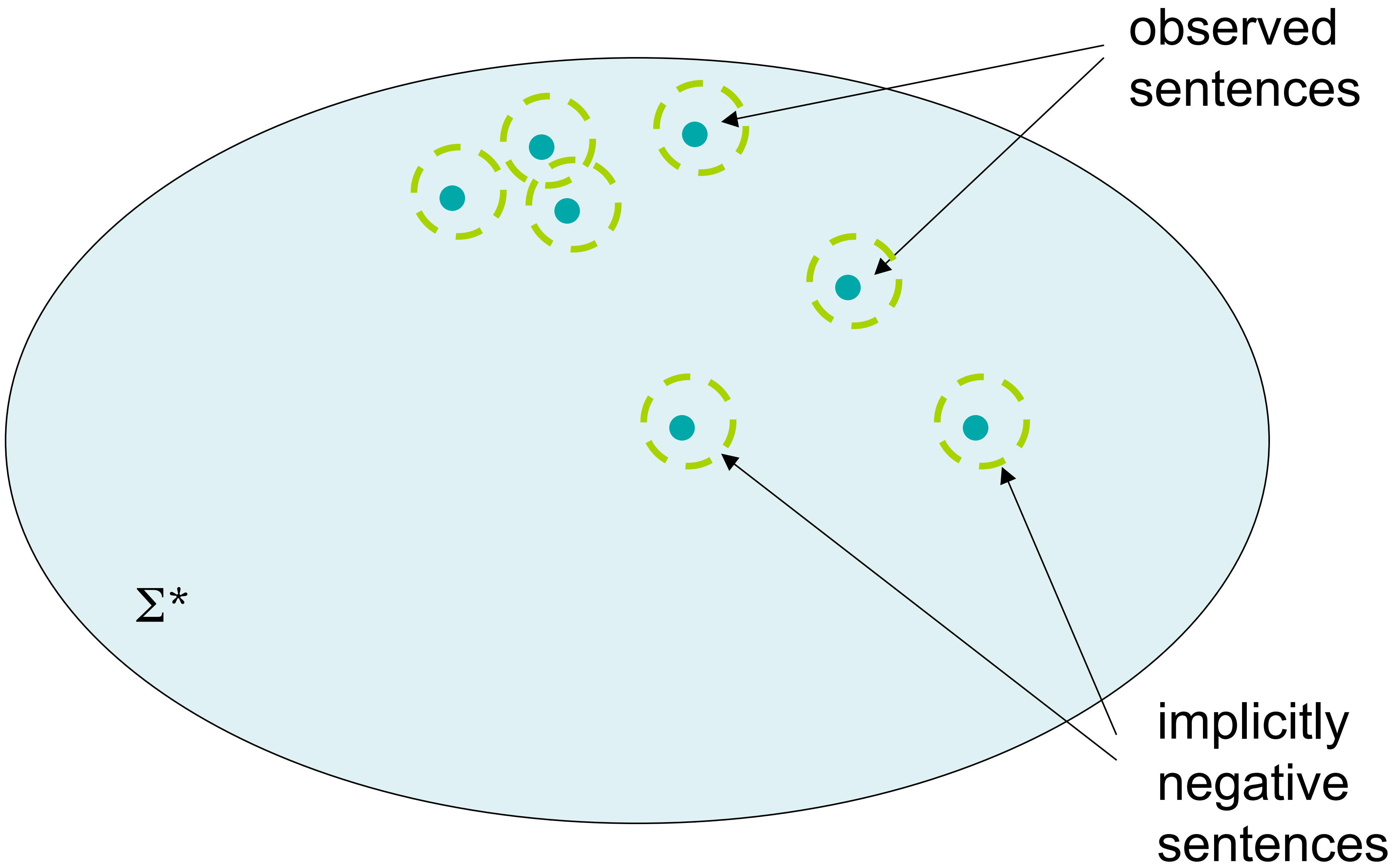
*Jays blue hide don't leaves red.

*Blue don't hide jays leaves red.

*Hide don't blue jays red leaves.

- Idea: train model to explain **order** but not content.

Contrastive Estimation
(Smith & Eisner, 2005)



Maximum Likelihood Estimation vs. Contrastive Estimation

MLE/MAP:
observed data are Sentences,
neighborhood is S*

Require numerical optimization

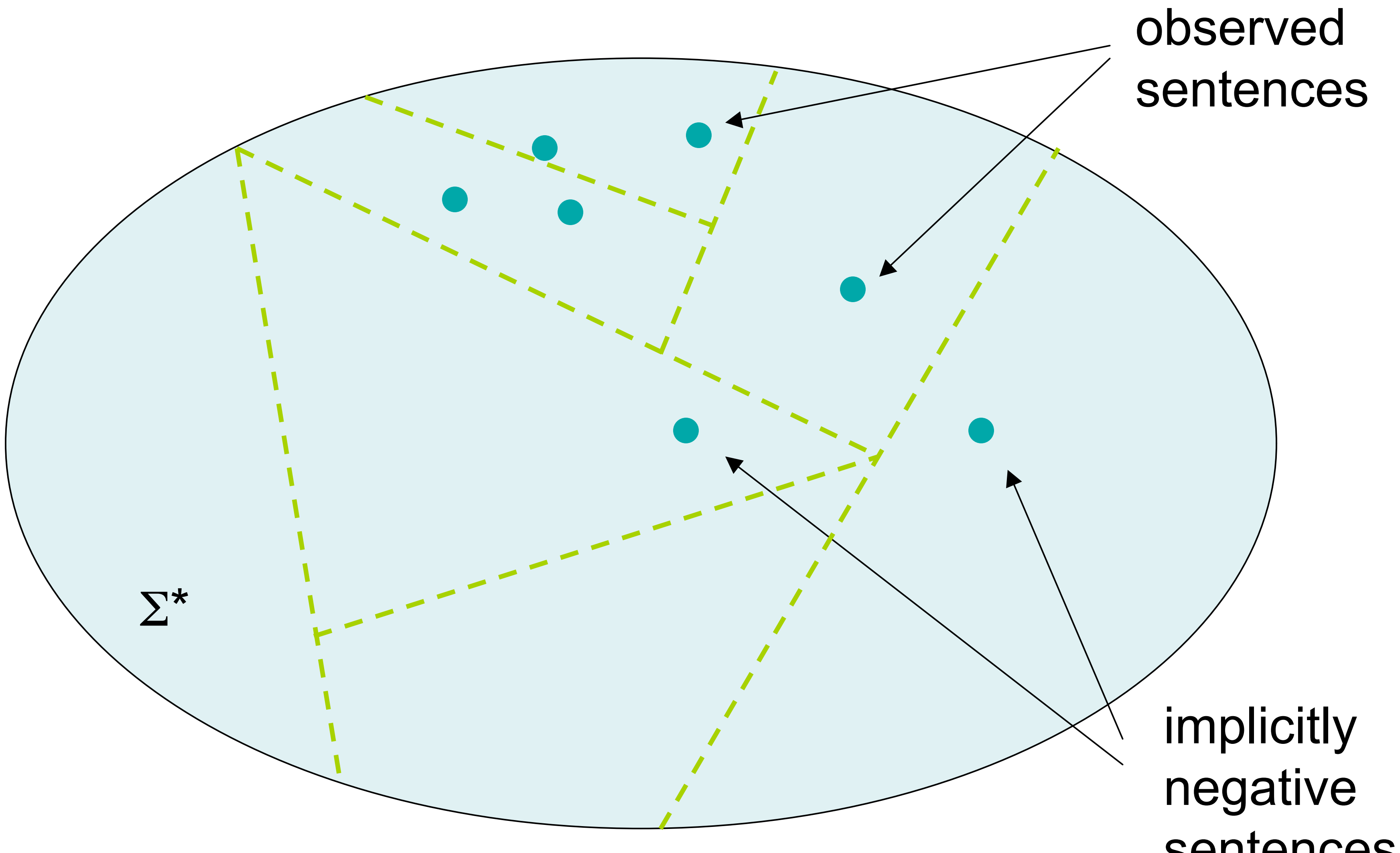
CE:
observed data are sentences,
neighborhood is …?

$$\max_{\vec{\theta}} \left[\prod_{i=1}^{n} \sum_{\mathbf{y}} p_{\vec{\theta}}(\mathbf{x}_i, \mathbf{y}) \right]$$

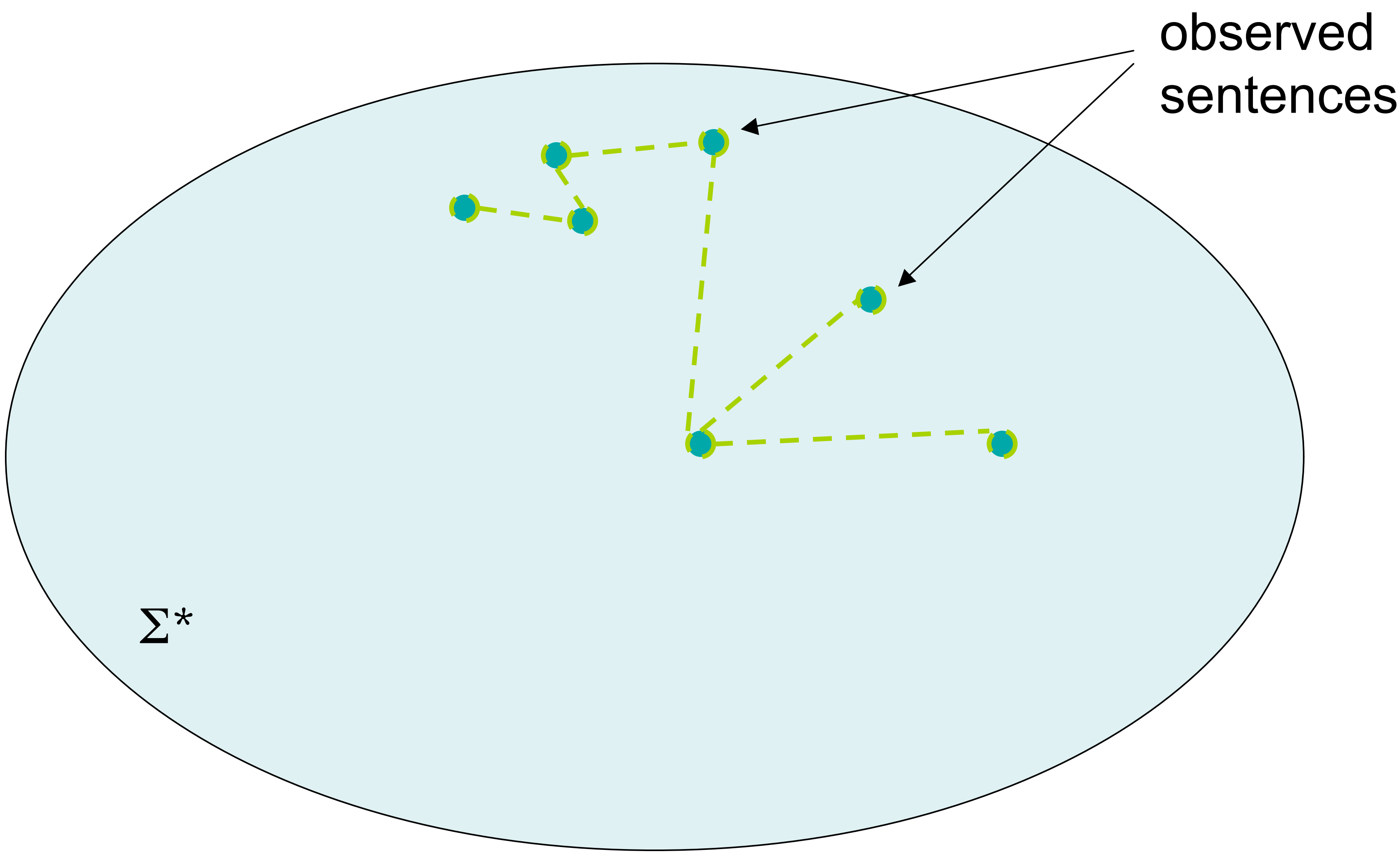
$$\max_{\vec{\theta}} \left[\prod_{i=1}^{n} \frac{\sum_{\mathbf{y}} p_{\vec{\theta}}(\mathbf{x}_i, \mathbf{y})}{\sum_{\mathbf{x} \in \mathcal{N}(\mathbf{x}_i)} \sum_{\mathbf{y}} p_{\vec{\theta}}(\mathbf{x}, \mathbf{y})} \right]$$

$$= \max_{\vec{\theta}} \left[\prod_{i=1}^{n} p_{\vec{\theta}}\left(\mathbf{X} = \mathbf{x}_i \mid \mathbf{X} \in \mathcal{N}(\mathbf{x}_i)\right) \right]$$

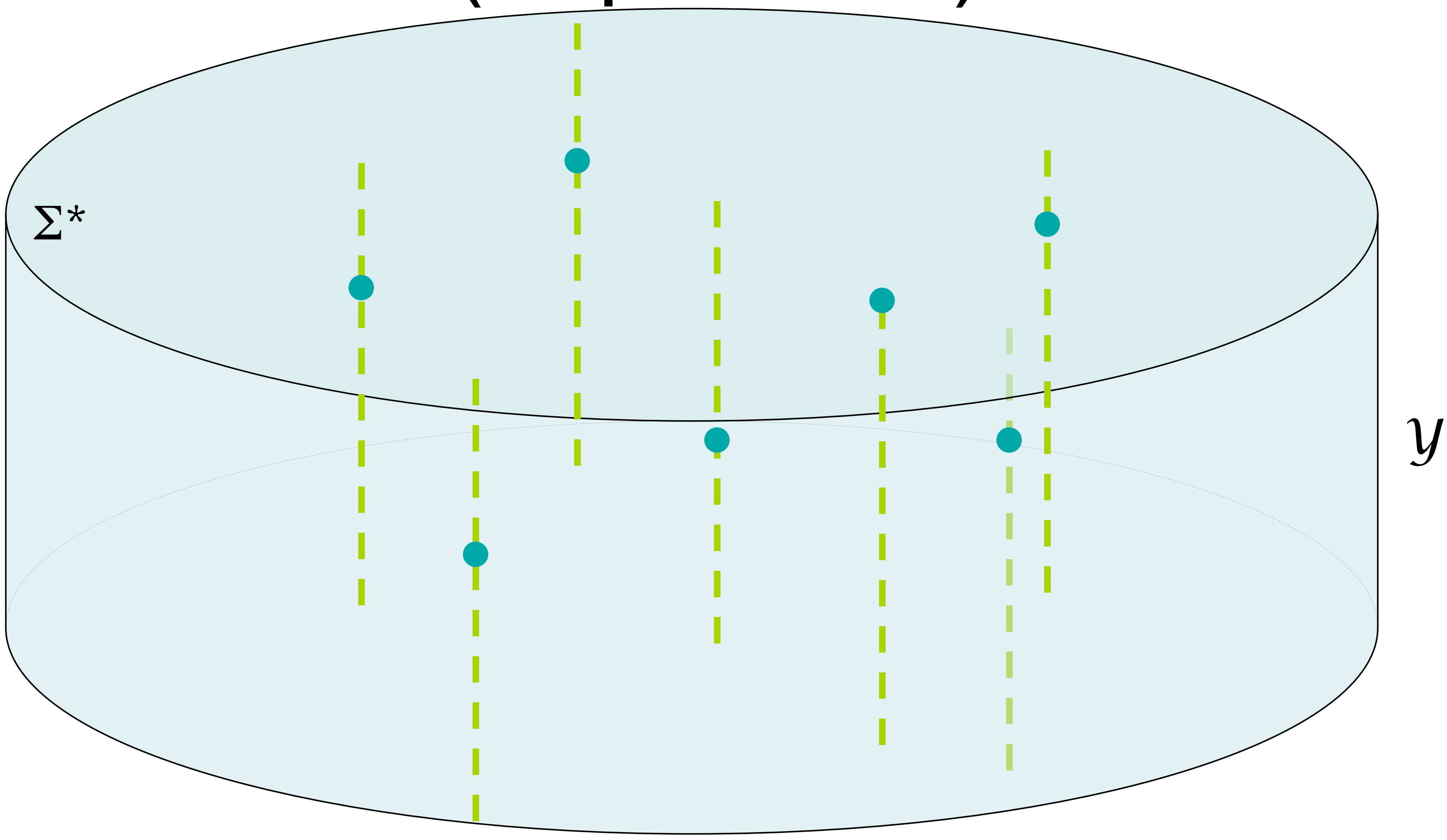
Partition Neighborhood = Conditional EM



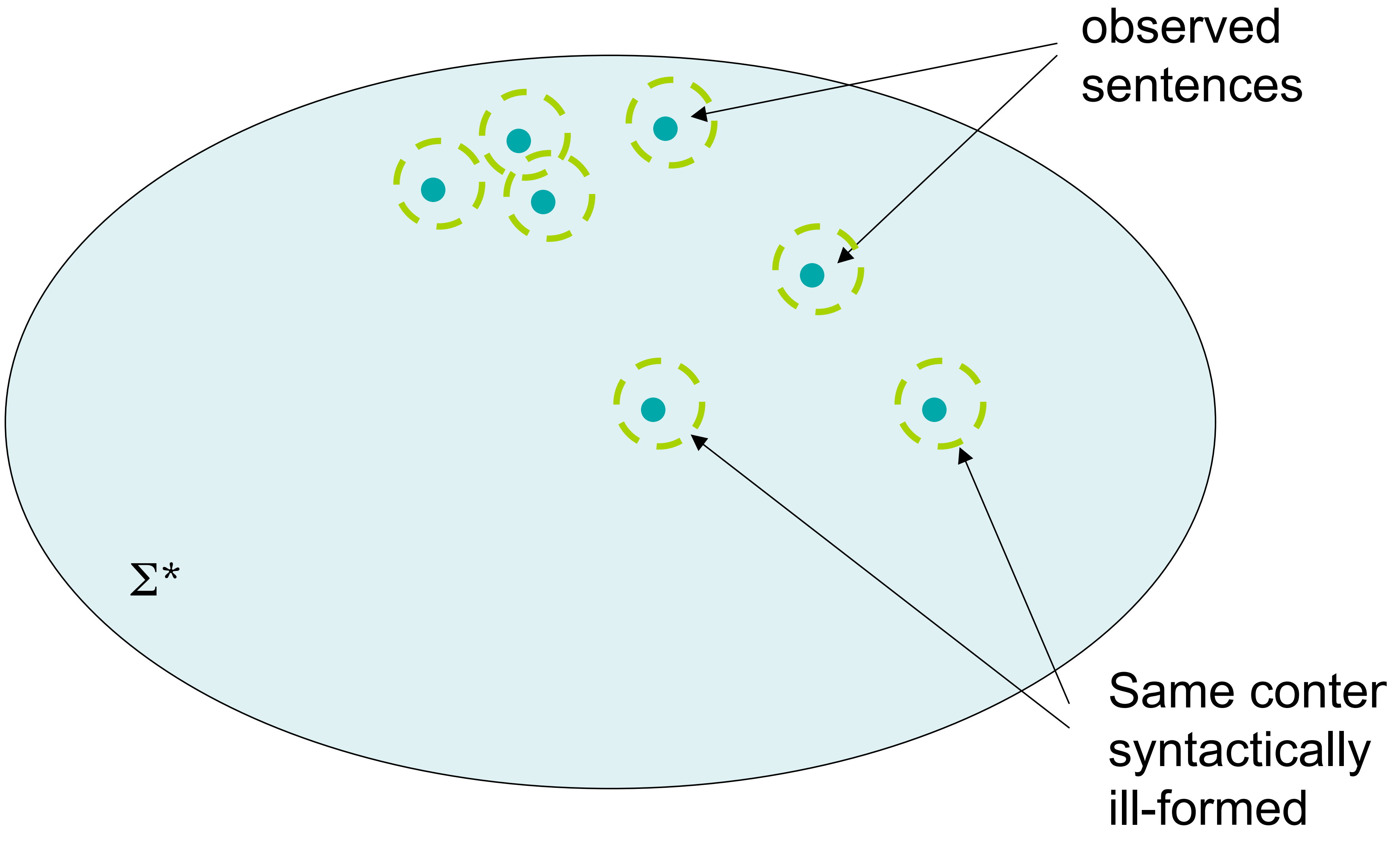
Riezler's (1999) Approximation



Analogy to Conditional Estimation (Supervised)



CE for Syntax



CE as Teacher

Red leaves
don’t
hide blue jays.

Leaves red
don’t hide blue
jays.

Red don’t
leaves hide
blue jays.

Red leaves hide
don’t blue jays.

What is a syntax model supposed to explain?

Each **learning hypothesis**

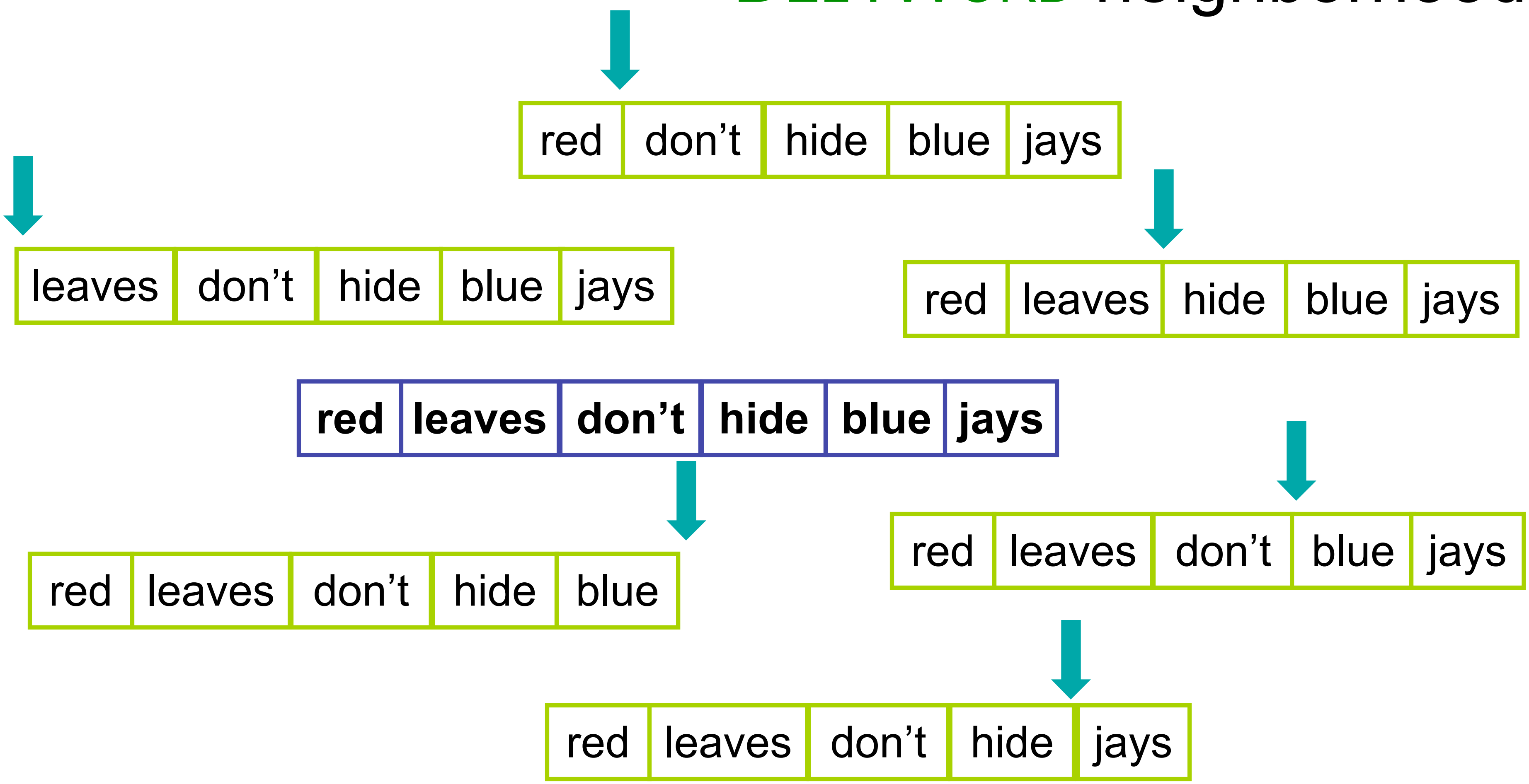
corresponds to

a **denominator / neighborhood**.

The Job of Syntax

"Explain why each word is necessary."

→ DEL1WORD neighborhood



The Job of Syntax

"Explain the (local) order of the words."

→ TRANS1 neighborhood

red | don't | leaves | hide | blue | jays

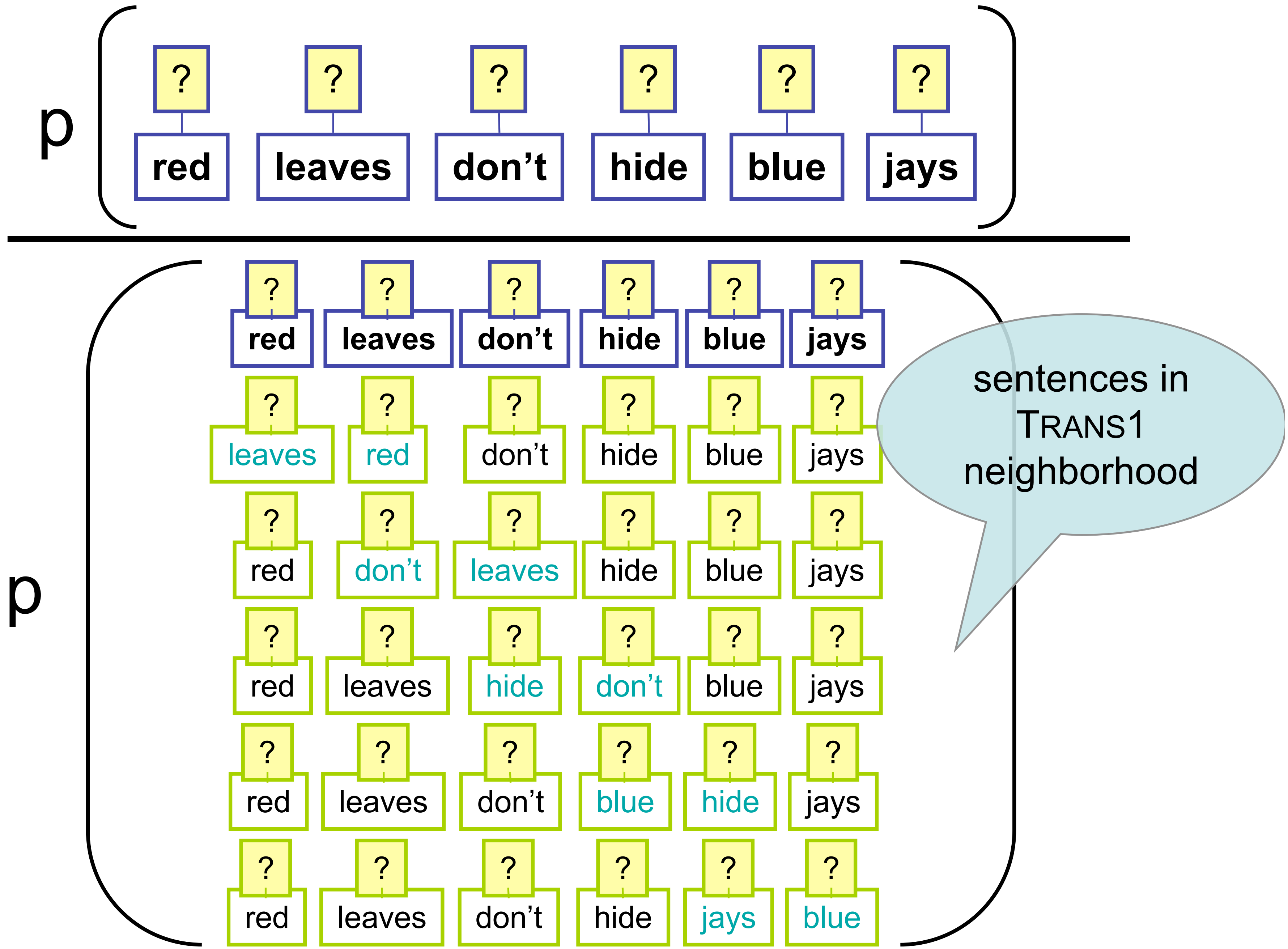
leaves | red | don't | hide | blue | jays

red | leaves | don't | hide | blue | jays

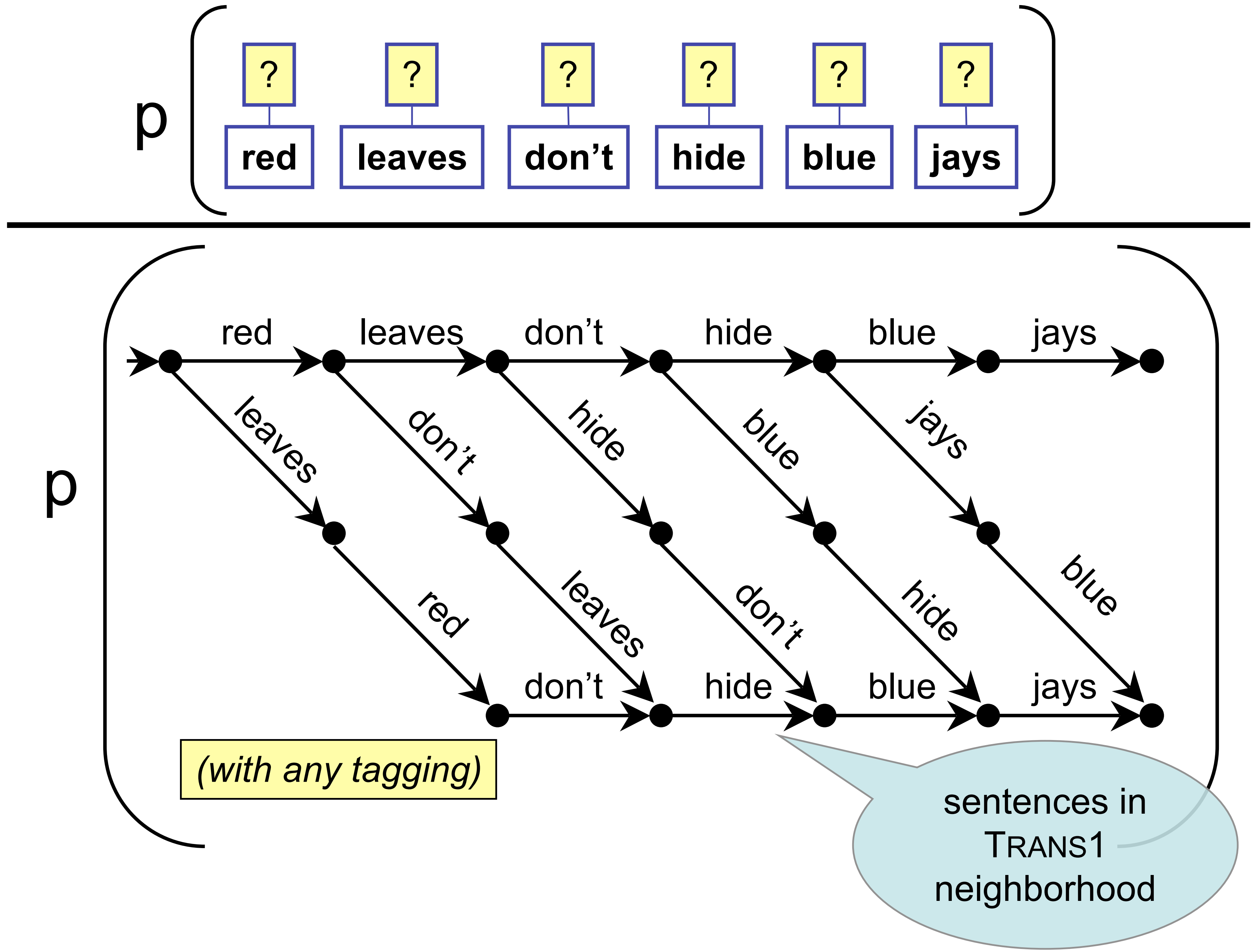
red | leaves | hide | don't | blue | jays

red | leaves | don't | hide | jays | blue

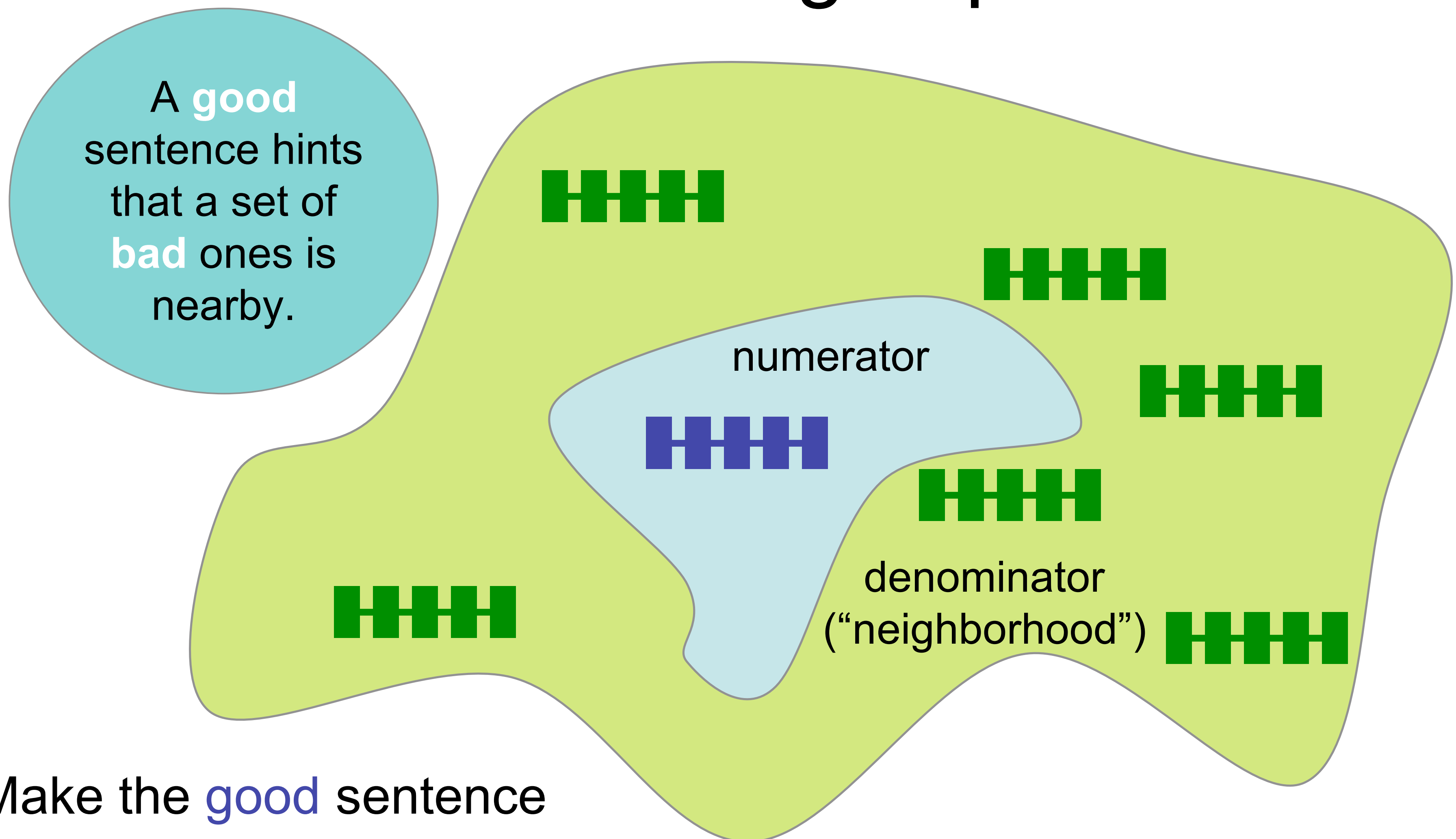
red | leaves | don't | blue | hide | jays



p
?
red
leaves
don't
hide
blue
jays
p
leaves red don't hide blue jays
red don't leaves hide blue jays
red leaves hide don't blue jays
red leaves don't blue hide jays
red leaves don't hide jays blue
sentences in TRANS1 neighborhood


p
?
red
leaves
don't
hide
blue
jays
p
red
leaves
don't
hide
blue
jays
leaves
don't
hide
blue
jays
red
leaves
don't
hide
blue
don't
hide
blue
jays
(with any tagging)
sentences in TRANS1 neighborhood

The New Modeling Imperative



"Make the good sentence likely, at the **expense** of those bad neighbors."

This talk is about **denominators** ...

in the **unsupervised case**.

A good denominator can improve

accuracy

and

tractability.

Log-Linear Models

score of **x**, **y**

$$p(x,y) = \frac{\exp(\mathbf{f}(x,y)\cdot\theta)}{Z(\theta)}$$

partition function

$$Z(\theta) = \sum_x \sum_y \exp(\mathbf{f}(x,y)\cdot\theta)$$

Sums over all possible taggings of all possible sentences!

Z may be infinite for some θ; computing it (if it is finite) may require solving a non-linear system.

Log-Linear Models

score of **x**, **y**

$$p(x,y) = \frac{\exp\left(\mathbf{f}(x,y)\cdot\theta\right)}{Z(\theta)}$$

partition function

$$Z(\theta) = \sum_x \sum_y \exp\left(\mathbf{f}(x,y)\cdot\theta\right)$$

Sums over all possible taggings of all possible sentences!

Computing Z is undesirable!

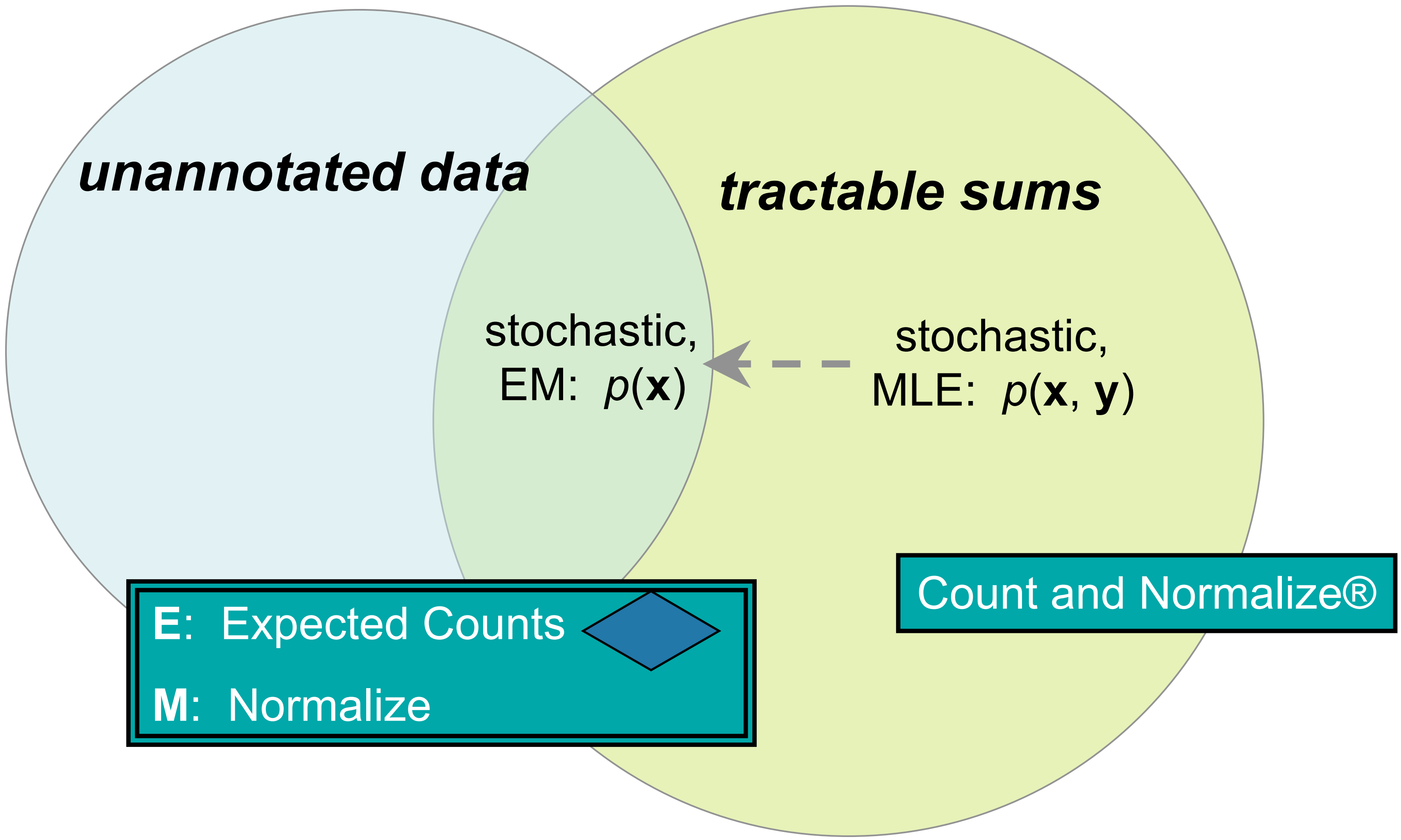
Conditional Estimation (Supervised)

1 sentence: Z(**x**)

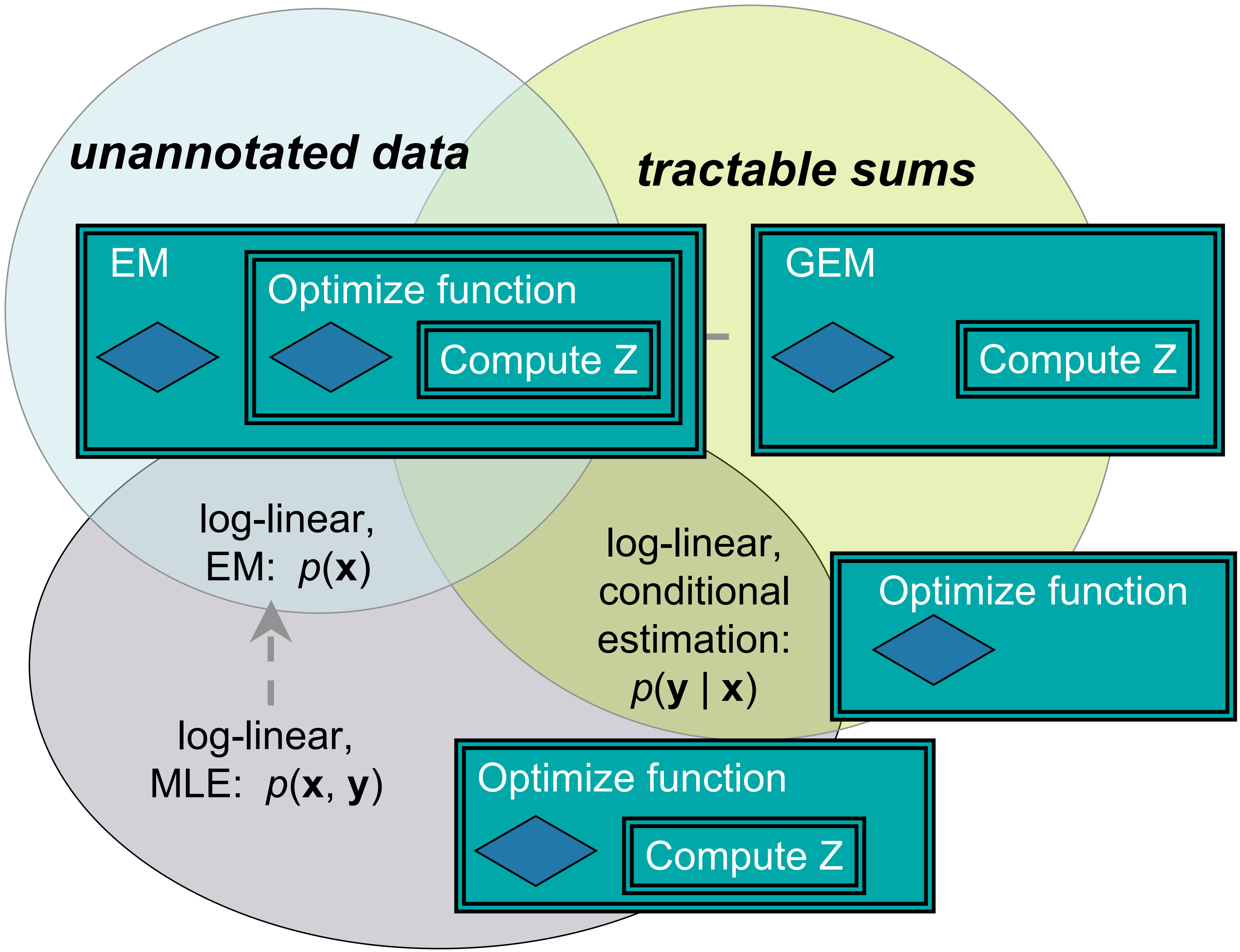
Contrastive Estimation (Unsupervised)

a few sentences: Z(N(**x**))

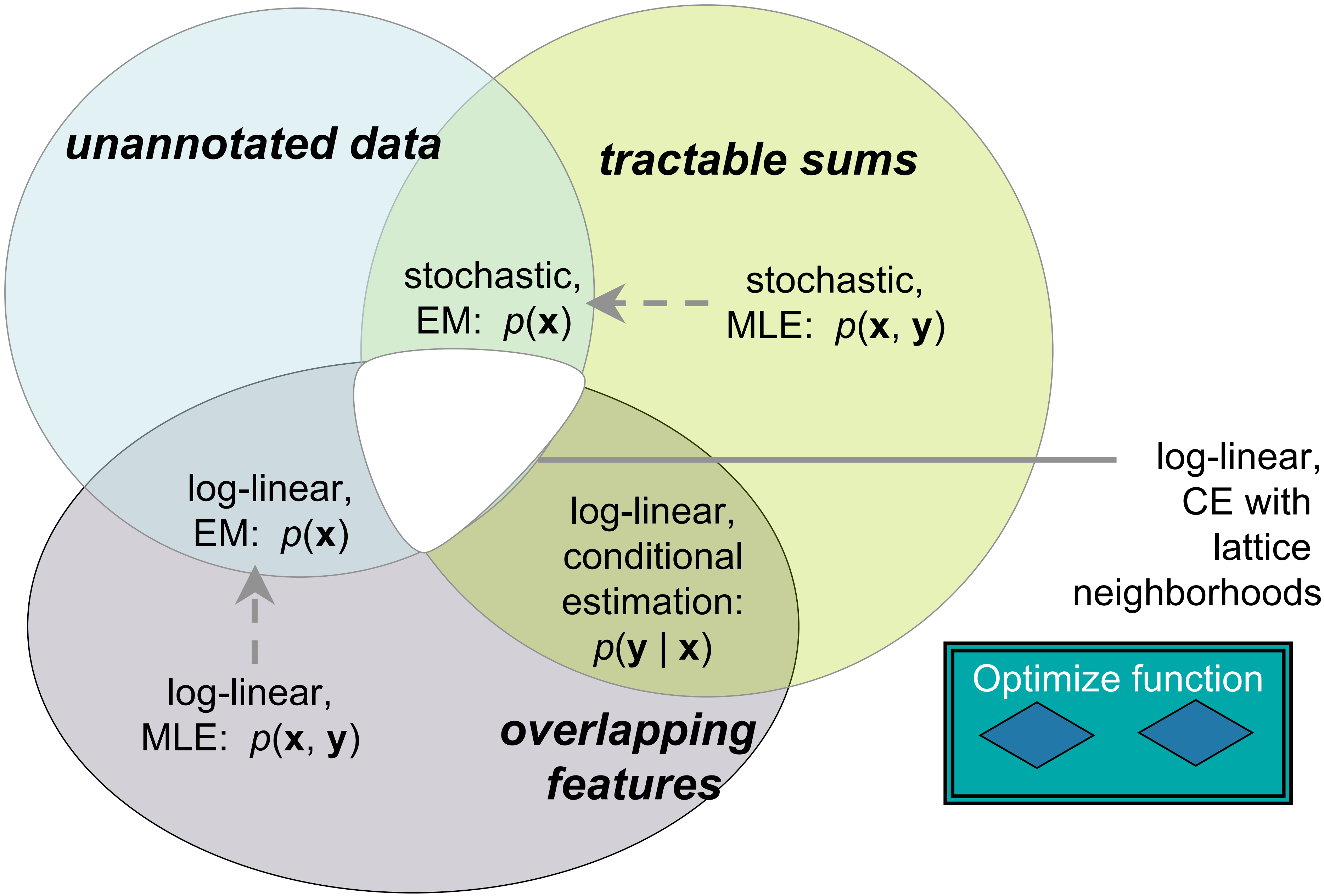
A Big Picture: Sequence Model Estimation



A Big Picture: Sequence Model Estimation



A Big Picture: Sequence Model Estimation



Contrastive Neighborhoods

- **Guide** the learner toward models that do what syntax is **supposed** to do.
- Lattice representation → **efficient** algorithms.

There is an **art** to choosing neighborhood functions.

Neighborhoods

neighborhood	*size*	*lattice arcs*	*perturbations*
DEL1WORD	$n+1$	$O(n)$	delete up to 1 word
TRANS1	n	$O(n)$	transpose any bigram
DELORTRANS1	$O(n)$	$O(n)$	DEL1WORD ∪ TRANS1
DEL1SUBSEQUENCE	$O(n^2)$	$O(n^2)$	delete any contiguous subsequence
Σ^* (MLE)	∞	-	replace each word with anything

Optimizing Contrastive Likelihood

$$F(\vec{\theta}) = \left[\sum_{i=1}^{n} \log p_{\vec{\theta}}\left(\mathbf{X} = \mathbf{x}_i\right) - \log p_{\vec{\theta}}\left(\mathbf{X} \in \mathcal{N}(\mathbf{x}_i)\right)\right]$$

$$\frac{\partial F}{\partial \theta_{\mathbf{r}}} = \left[\sum_{i=1}^{n} \mathbf{E}_{p_{\vec{\theta}}}\left[f_{\mathbf{r}}(\mathbf{x}_i, \mathbf{Y})\right] - \mathbf{E}_{p_{\vec{\theta}}}\left[f_{\mathbf{r}}(\mathbf{X}, \mathbf{Y}) \mid \mathbf{X} \in \mathcal{N}(\mathbf{x}_i)\right]\right]$$

Expected count
Of rule **r** in sentence *i*

Expected count
Of rule **r** in neighborhood *i*

The Merialdo (1994) Task

Given **unlabeled text**

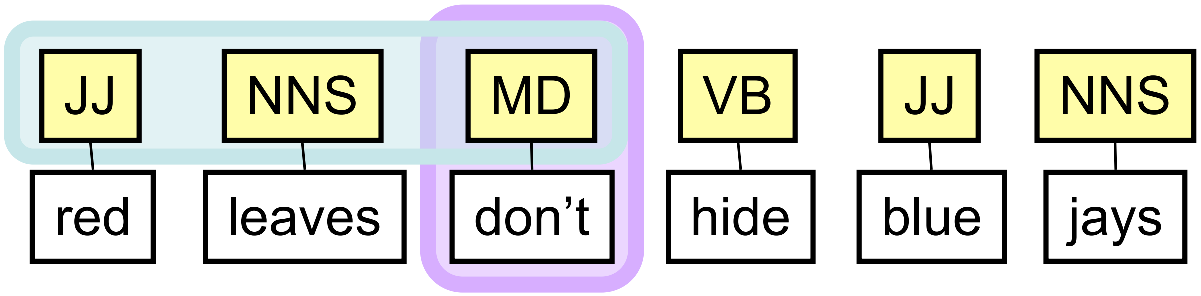
and a **POS dictionary**

(that tells **all** possible tags for **each** word type),

A form of supervision / domain knowledge.

learn to tag.

Trigram Tagging Model



feature set:

tag trigrams

tag/word pairs from a POS dictionary

Tagging Experiment

		12K		24K		48K		96K	
		u-sel.	*oracle*	u-sel.	*oracle*	u-sel.	*oracle*	u-sel.	*oracle*
+	CRF (supervised)		*100.0*		*99.8*		*99.8*		*99.5*
×	HMM (supervised)		*99.3*		*98.5*		*97.9*		*97.2*
△	LENGTH	**74.9**	***77.4***	**78.7**	***81.5***	**78.3**	***81.3***	**78.9**	***79.3***
■	DEL1ORTRANS1	70.8	*70.8*	78.6	*78.6*	**78.3**	*79.1*	75.2	*78.8*
□	TRANS1	72.7	*72.7*	77.2	*77.2*	78.1	*79.4*	74.7	*79.0*
×	EM	49.5	*52.9*	55.5	*58.0*	59.4	*60.9*	60.9	*62.1*
▼	DEL1	55.4	*55.6*	58.6	*60.3*	59.9	*60.2*	59.9	*60.4*
•	DEL1SUBSEQ	53.0	*53.3*	55.0	*56.7*	55.3	*55.4*	57.3	*58.7*
–	random expected		*35.2*		*35.1*		*35.1*		*35.1*
	ambiguous words		6,244		12,923		25,879		51,521

So, why does LENGTH beat EM?

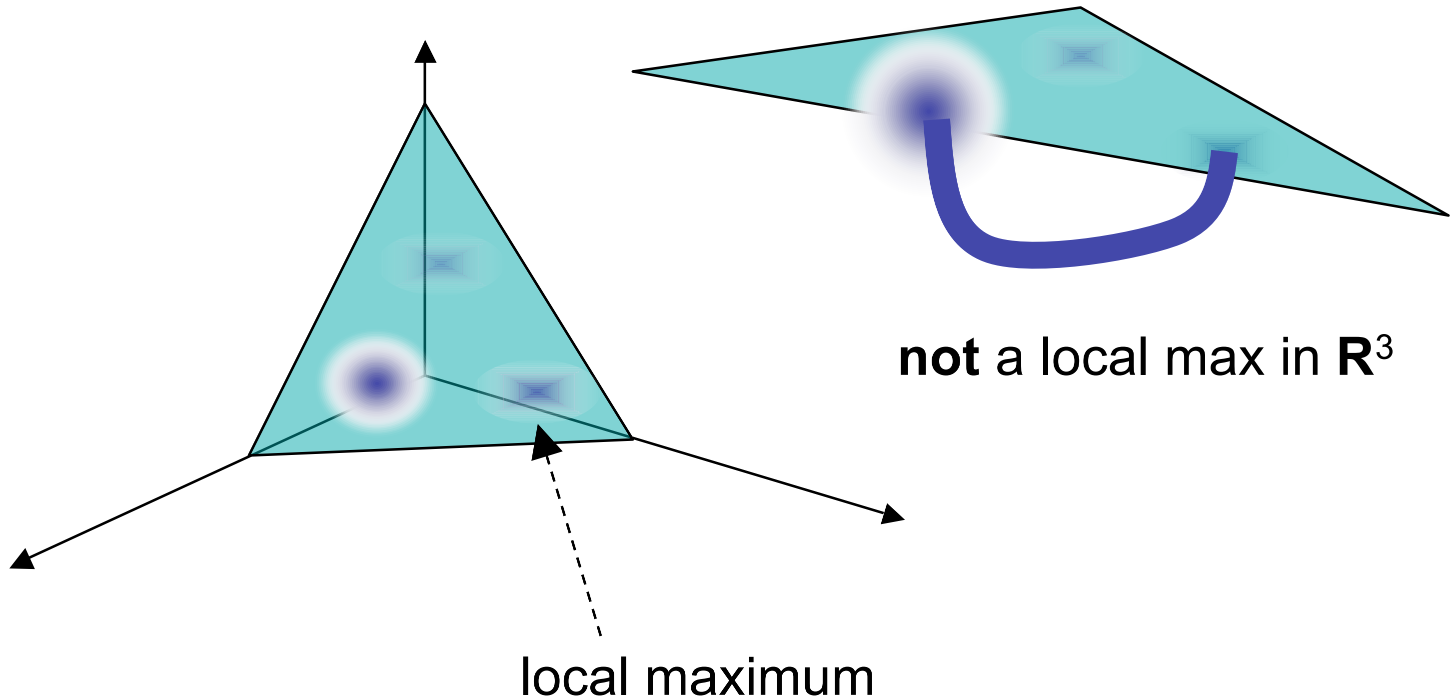
✘ the model is **log-linear**?

the objective function is better?
(don't have to model # words)

functions essentially the same, but better **search**?

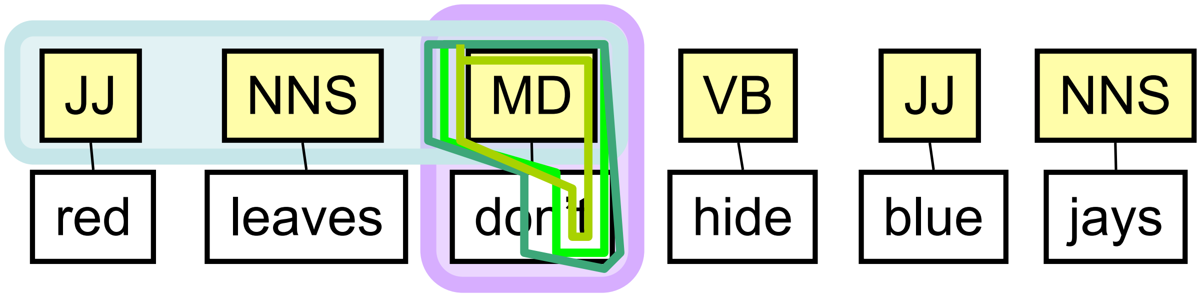
On Local Maxima

- Requiring weights to **sum to one** is simply a numerical **constraint**.



For bumpy functions, it's preferable to have fewer constraints.

Trigram Tagging Model + Spelling



feature set:

- tag trigrams
- tag/word pairs from a POS dictionary
- 1- to 3-character suffixes, contains hyphen, digit

Diluted Dictionary

		tagging dictionary							
		all train & dev.		first 500 sents.		count ≥ 2		count ≥ 3	
estimation	model	u-sel.	*oracle*	u-sel.	*oracle*	u-sel.	*oracle*	u-sel.	*oracle*
MAP/EM	trigram	78.0	*84.4*	77.2	*80.5*	70.1	*70.9*	66.5	*66.5*
CE/DEL1ORTRANS1	trigram	78.3	*90.1*	72.3	*84.8*	69.5	*81.3*	65.0	*77.2*
	+ spelling	80.9	*91.1*	80.2	**90.8**	**79.5**	**90.3**	78.3	**89.8**
CE/TRANS1	trigram	**90.4**	*90.4*	80.8	*82.9*	77.0	*78.6*	71.7	*73.4*
	+ spelling	88.7	*90.9*	**88.1**	*90.1*	78.7	*90.1*	**78.4**	*89.5*
CE/LENGTH	trigram	87.8	*90.4*	68.1	*78.3*	65.3	*75.2*	62.8	*72.3*
	+ spelling	87.1	**91.9**	76.9	*83.2*	73.3	*73.8*	73.2	*73.6*
random expected			69.5		60.5		56.6		51.0
ambiguous words			13,150		13,841		14,780		15,996
ave. tags/token			2.3		3.7		4.4		5.5

(reduced, coarser tag set)

The sequence model need not be finite-state.

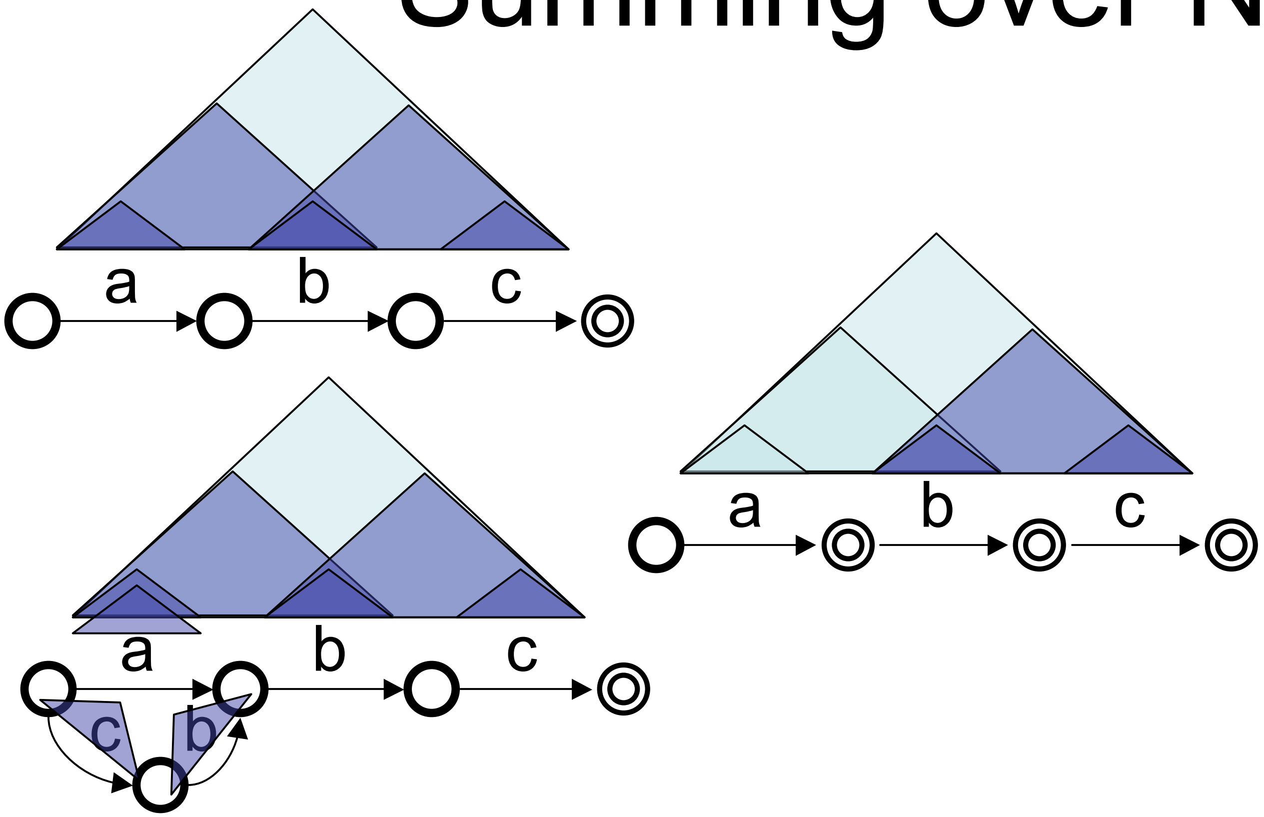
Y can range over trees.

Dependency Parsing

- Features (model from Klein and Manning, 2004):
  - (parent, child, direction) triples
  - “no children on left (right)”
  - “1 child on left (right)”
  - “multiple children on left (right”)

- Dynamic programming:
  - Eisner & Satta (1999) for **inside** algorithm (generalized for lattices)

Summing over N(**x**)



- Dynamic programming saves the day again!
- If the set N(**x**) is represented as a lattice, we can apply the usual Inside-Outside algorithm with a slight change.

	German test accuracy		English test accuracy		Bulgarian test accuracy		Mandarin test accuracy		Turkish test accuracy		Portuguese test accuracy	
	directed	undirected	directed	undirected	directed	undirected	directed	undirected	directed	undirected	directed	undirected
ATTACH-LEFT	8.2	59.1	22.6	62.1	37.2	61.0	13.1	56.1	6.6	68.6	36.2	65.7
ATTACH-RIGHT	47.0	55.2	39.5	62.1	23.8	61.0	42.9	56.1	61.8	68.3	29.5	65.7
Σ^* (MAP/EM)	19.8	55.2	41.6	62.2	44.6	63.1	37.2	56.1	41.2	57.8	37.4	62.2
DEL1	**26.5**	43.7	18.3	33.9	13.1	31.5	25.6	41.4	41.8	45.2	39.9	**67.2**
TRANS1	17.9	53.0	29.4	57.8	23.8	61.0	22.7	56.3	27.7	**59.4**	36.0	**65.5**
DEL1ORTRANS1	**59.3**	**72.6**	**47.3**	**63.6**	24.2	60.0	22.6	58.2	**46.5**	**62.9**	36.0	**65.4**
LENGTH	**49.2**	**64.1**	**45.5**	**64.9**	27.0	60.1	16.5	43.4	34.4	57.6	31.9	59.4
DYNASEARCH	16.0	53.0	39.7	61.9	23.8	61.0	**48.3**	**58.8**	44.9	**62.7**	37.9	62.3

	German test accuracy		English test accuracy		Bulgarian test accuracy		Mandarin test accuracy		Turkish test accuracy		Portuguese test accuracy	
	directed	undirected	directed	undirected	directed	undirected	directed	undirected	directed	undirected	directed	undirected
Attach-Left	8.2	59.1	22.6	62.1	37.2	61.0	13.1	56.1	6.6	68.6	36.2	65.7
Attach-Right	47.0	55.2	39.5	62.1	23.8	61.0	42.9	56.1	61.8	68.3	29.5	65.7
Σ^* (MAP/EM)	54.4	71.9	41.6	62.2	45.6	63.6	50.0	60.9	48.0	59.1	42.3	64.1
Del1	34.4	49.3	39.7	53.5	17.7	33.8	43.4	49.8	42.1	45.1	28.0	43.1
Trans1	45.6	59.0	41.2	62.5	40.1	57.9	41.1	56.1	47.2	**63.4**	35.9	**65.8**
Del1OrTrans1	**63.4**	66.5	**57.6**	**69.0**	40.5	61.5	41.1	56.9	**58.2**	**66.4**	**71.8**	**78.4**
Length	**57.3**	65.1	**45.5**	**64.9**	38.3	63.4	26.2	44.9	**59.0**	**64.9**	33.6	65.3
DynaSearch	45.7	58.6	**47.6**	**65.3**	34.0	58.0	47.9	60.6	44.9	**62.7**	40.9	64.4
s-sel. ($\mathcal{N}$)	**63.4**	66.5	**57.6**	**69.0**	40.5	61.5	41.1	56.1	**59.0**	**64.9**	**71.8**	**78.4**

Summing Up (Ha Ha)

- Contrastive estimation = designing a negative evidence class that keeps part of the data the same (e.g., semantics) but damages the part you want your model to learn (e.g., syntax).
- Idea of “implicit negative evidence” is central.