10-418/10-618 Machine Learning for Structured Data

Machine Learning Department
School of Computer Science
Carnegie Mellon University

Variational EM
+
Hidden State CRFs

Matt Gormley
Lecture 19
Nov. 9, 2022

Reminders

- **Homework 5: Variational Inference**
  - Out: Fri, Nov 4
  - Due: Wed, Nov 16 at 11:59pm

EXPECTATION MAXIMIZATION

Hard Expectation-Maximization

- Initialize **parameters** **randomly**
- **while** not converged

  1. **E-Step:**
     Set the **latent variables** to the the values that maximizes likelihood, treating parameters as observed

     Estimate unobserved variables

  2. **M-Step:**
     Set the **parameters** to the values that maximizes likelihood, treating latent variables as observed

     MLE given the estimated values of unobserved variables

(Soft) Expectation-Maximization

- Initialize **parameters randomly**
- **while** not converged
  1. **E-Step:**
     Create one training example for each possible value of the **latent variables**
     Weight each example according to model's confidence
     Treat parameters as observed
  2. **M-Step:**
     Set the **parameters** to the values that maximizes likelihood
     Treat pseudo-counts from above as observed

Estimate unobserved variables

MLE given the estimated values of unobserved variables

Hard EM vs. Soft EM

Algorithm 1 Hard EM for GMMs

procedure HARDEM($\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$)
  Randomly initialize parameters, $\boldsymbol{\phi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$
  while not converged **do**
    E-Step:

$$z^{(i)} \leftarrow \operatorname*{argmax}_z \log p(\mathbf{x}^{(i)}|z; \boldsymbol{\mu}, \boldsymbol{\Sigma}) + \log p(z; \boldsymbol{\phi})$$

    M-Step:

$$\phi_k \leftarrow \frac{1}{N}\sum_{i=1}^N \mathbb{I}(z^{(i)} = k), \forall k$$

$$\boldsymbol{\mu}_k \leftarrow \frac{\sum_{i=1}^N \mathbb{I}(z^{(i)} = k)\mathbf{x}^{(i)}}{\sum_{i=1}^N \mathbb{I}(z^{(i)} = k)}, \forall k$$

$$\boldsymbol{\Sigma}_k \leftarrow \frac{\sum_{i=1}^N \mathbb{I}(z^{(i)} = k)(\mathbf{x}^{(i)} - \boldsymbol{\mu}_k)(\mathbf{x}^{(i)} - \boldsymbol{\mu}_k)^T}{\sum_{i=1}^N \mathbb{I}(z^{(i)} = k)}, \forall k$$

  return $(\boldsymbol{\phi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$

Algorithm 1 Soft EM for GMMs

procedure SOFTEM($\mathcal{D} = \{\mathbf{x}^{(i)}\}_{i=1}^N$)
  Randomly initialize parameters, $\boldsymbol{\phi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$
  while not converged **do**
    E-Step:

$$c_k^{(i)} \leftarrow p(z^{(i)} = k|\mathbf{x}^{(i)}; \boldsymbol{\phi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$$

    M-Step:

$$\phi_k \leftarrow \frac{1}{N}\sum_{i=1}^N c_k^{(i)}, \forall k$$

$$\boldsymbol{\mu}_k \leftarrow \frac{\sum_{i=1}^N c_k^{(i)}\mathbf{x}^{(i)}}{\sum_{i=1}^N c_k^{(i)}}, \forall k$$

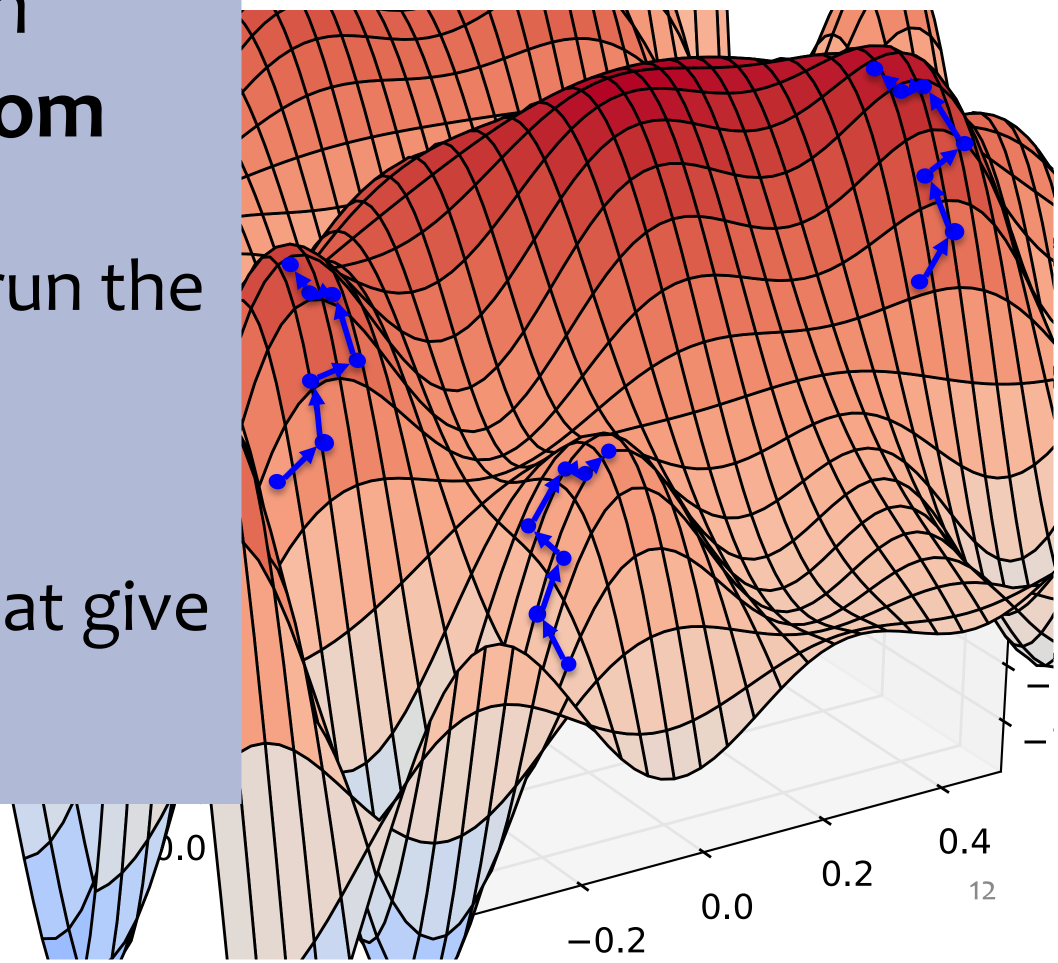
$$\boldsymbol{\Sigma}_k \leftarrow \frac{\sum_{i=1}^N c_k^{(i)}(\mathbf{x}^{(i)} - \boldsymbol{\mu}_k)(\mathbf{x}^{(i)} - \boldsymbol{\mu}_k)^T}{\sum_{i=1}^N c_k^{(i)}}, \forall k$$

  return $(\boldsymbol{\phi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})$

PROPERTIES OF EM

Properties of (Variational) EM

- EM is *trying* to optimize a **nonconvex** function
- But EM is a **local** optimization algorithm
- Typical solution: **Random Restarts**
  - Just like K-Means, we run the algorithm many times
  - Each time initialize parameters randomly
  - Pick the parameters that give highest likelihood



Variants of EM

- **Generalized EM**: Replace the M-Step by a single gradient-step that improves the likelihood
- **Monte Carlo EM**: Approximate the E-Step by sampling
- **Sparse EM:** Keep an "active list" of points (updated occasionally) from which we estimate the expected counts in the E-Step
- **Incremental EM / Stepwise EM**: If standard EM is described as a *batch* algorithm, these are the *online* equivalent
- **etc.**

A Report Card for EM

- Some good things about EM:
  - no learning rate (step-size) parameter
  - automatically enforces parameter constraints
  - very fast for low dimensions
  - each iteration guaranteed to improve likelihood

- Some bad things about EM:
  - can get stuck in local minima
  - can be slower than conjugate gradient (especially near convergence)
  - requires expensive inference step
  - is a maximum likelihood/MAP method

VARIATIONAL EM

Variational EM

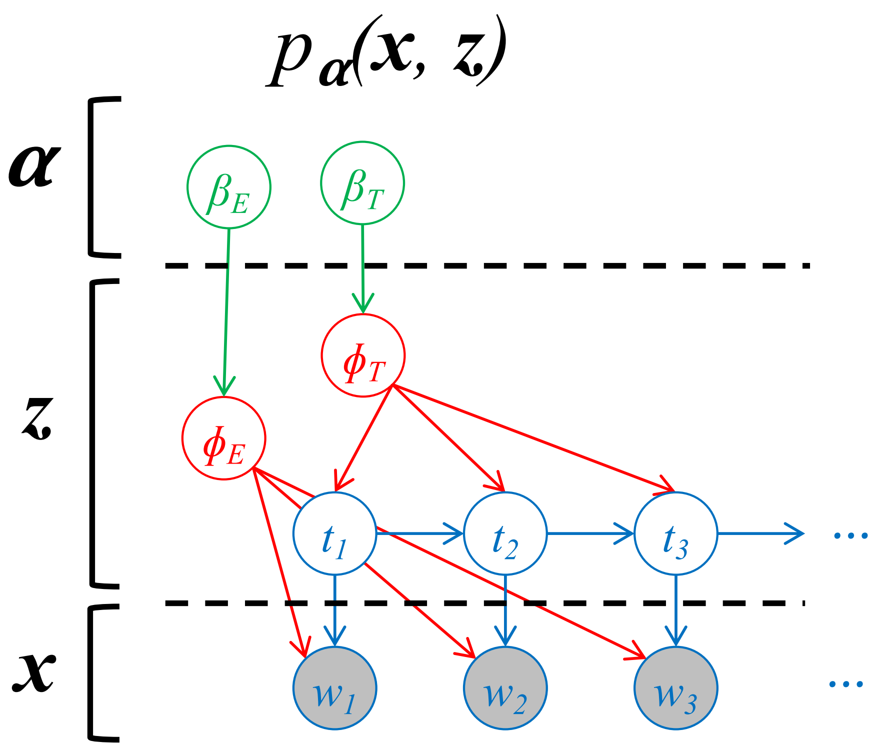
Whiteboard

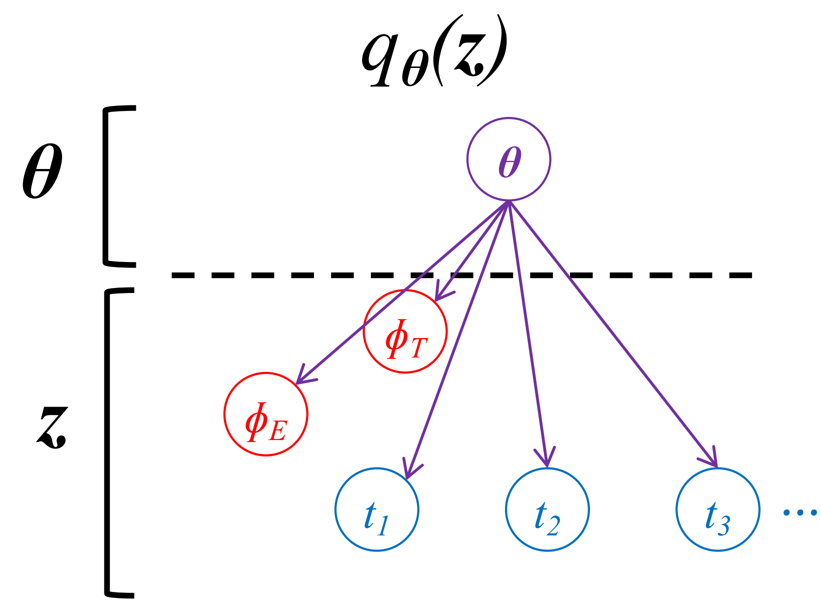
- Example: Unsupervised POS Tagging
- Variational Bayes
- Variational EM

UNDERSTANDING THE VARIATIONAL FRAMEWORK

Bayesian Unsupervised POS Tagging

- Given: sentences only (concatenated together into one long string)
- Goal: infer the POS tags for unlabeled sentences
- Model: Bayesian HMM





Variational Inference Framework

	β	ϕ_E, ϕ_T	$t_{1:T}$	optimization problem
1)	given	given	max	
2)	given	given	sum	
3)	given	max	sum	
4)	given	sum	sum	
5)	max	sum	sum	

Question:
What is the corresponding optimization problem?

A. Bayesian inference
B. empirical Bayes
C. MAP inference
D. marginal inference
E. MAP estimation

Table from Jason Eisner https://www.cs.jhu.edu/~jason/tutorials/variational.html

Variational Inference Framework

	β	ϕ_E, ϕ_T	$t_{1:T}$	optimization problem	name of (approximate) method
1)	given	given	max	$t = \text{argmax}_t\ p(t \mid \phi, \beta)$ MAP inference / decoding	max-product B.P. (greedy search)
2)	given	given	sum	$p(w \mid \phi, \beta) = \sum_t p(w, t \mid \phi, \beta)$ marginal inference	sum-product B.P. (variational inference)
3)	given	max	sum	$\phi = \text{argmax}_\phi\ p(w \mid \phi)\, p(\phi \mid \beta)$ MAP estimation	(variational) MAP-EM
4)	given	sum	sum	estimate $p(\phi \mid w, \beta) = \ldots$ and $p(t \mid w, \beta) = \ldots$ Bayesian inference	variational Bayes
5)	max	sum	sum	$\beta = \text{argmax}_\beta\ p(\beta \mid w)$ empirical Bayes	variational EM

Table from Jason Eisner https://www.cs.jhu.edu/~jason/tutorials/variational.html

VARIATIONAL EM RESULTS

Unsupervised POS Tagging

Bayesian Inference for HMMs

- **Task:** unsupervised POS tagging
- **Data:** 1 million words (i.e. unlabeled sentences) of WSJ text
- **Dictionary:** defines legal part-of-speech (POS) tags for each word type
- **Models:**
  - EM: standard HMM
  - VB: uncollapsed variational Bayesian HMM
  - Algo 1 (CVB): collapsed variational Bayesian HMM (strong indep. assumption)
  - Algo 2 (CVB): collapsed variational Bayesian HMM (weaker indep. assumption)
  - CGS: collapsed Gibbs Sampler for Bayesian HMM

Algo 1 mean field update:

$$q(z_t = k) \propto \frac{\mathbb{E}_{q(\mathbf{z}^{\neg t})}[C^{\neg t}_{k,w}] + \beta}{\mathbb{E}_{q(\mathbf{z}^{\neg t})}[C^{\neg t}_{k,\cdot}] + W\beta} \cdot \frac{\mathbb{E}_{q(\mathbf{z}^{\neg t})}[C^{\neg t}_{z_{t-1},k}] + \alpha}{\mathbb{E}_{q(\mathbf{z}^{\neg t})}[C^{\neg t}_{z_{t-1},\cdot}] + K\alpha} \cdot \frac{\mathbb{E}_{q(\mathbf{z}^{\neg t})}[C^{\neg t}_{k,z_{t+1}}] + \alpha + \mathbb{E}_{q(\mathbf{z}^{\neg t})}[\delta(z_{t-1} = k = z_{t+1})]}{\mathbb{E}_{q(\mathbf{z}^{\neg t})}[C^{\neg t}_{k,\cdot}] + K\alpha + \mathbb{E}_{q(\mathbf{z}^{\neg t})}[\delta(z_{t-1} = k)]}$$

CGS full conditional:

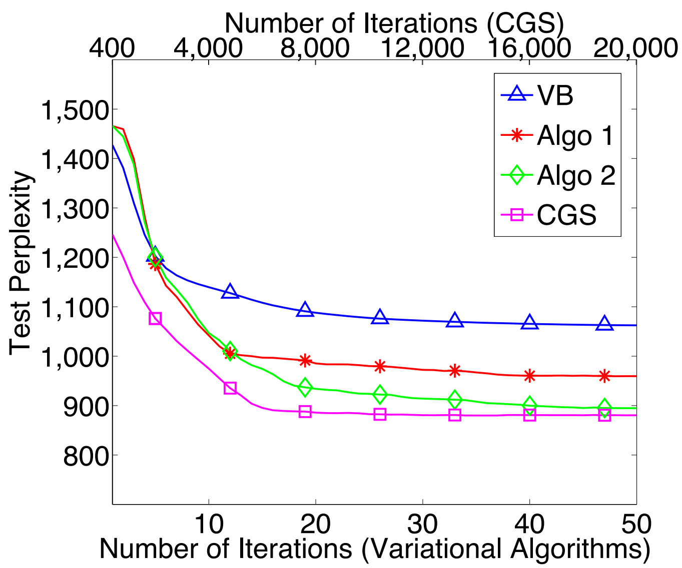
$$p(z_t = k | \mathbf{x}, \mathbf{z}^{\neg t}, \alpha, \beta) \propto \frac{C^{\neg t}_{k,w} + \beta}{C^{\neg t}_{k,\cdot} + W\beta} \cdot \frac{C^{\neg t}_{z_{t-1},k} + \alpha}{C^{\neg t}_{z_{t-1},\cdot} + K\alpha} \cdot \frac{C^{\neg t}_{k,z_{t+1}} + \alpha + \delta(z_{t-1} = k = z_{t+1})}{C^{\neg t}_{k,\cdot} + K\alpha + \delta(z_{t-1} = k)}$$

Figure from Wang & Blunsom (2013)

Unsupervised POS Tagging

Bayesian Inference for HMMs

- **Task:** unsupervised POS tagging
- **Data:** 1 million words (i.e. unlabeled sentences) of WSJ text
- **Dictionary:** defines legal part-of-speech (POS) tags for each word type
- **Models:**
  - EM: standard HMM
  - VB: uncollapsed variational Bayesian HMM
  - Algo 1 (CVB): collapsed variational Bayesian HMM (strong indep. assumption)
  - Algo 2 (CVB): collapsed variational Bayesian HMM (weaker indep. assumption)
  - CGS: collapsed Gibbs Sampler for Bayesian HMM



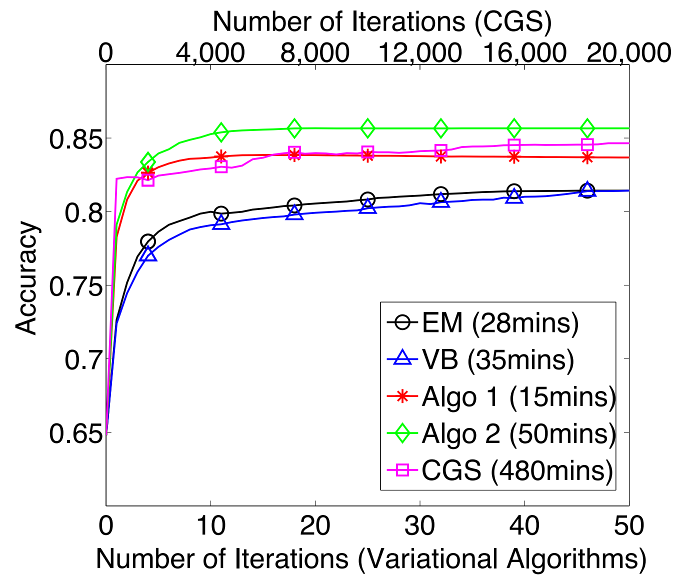


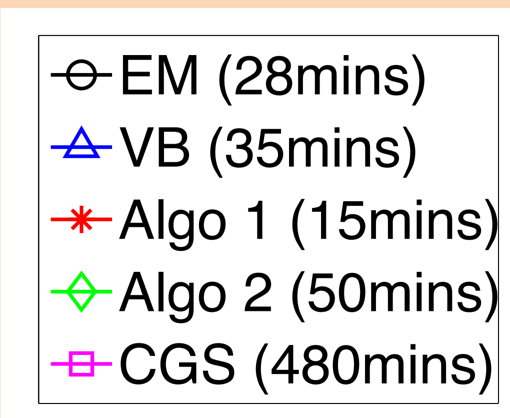
Figure from Wang & Blunsom (2013)

Unsupervised POS Tagging

Bayesian Inference for HMMs

- **Task:** unsupervised POS tagging
- **Data:** 1 million words (i.e. unlabeled sentences) of WSJ text
- **Dictionary:** defines legal part-of-speech (POS) tags for each word type
- **Models:**
  - EM: standard HMM
  - VB: uncollapsed variational Bayesian HMM
  - Algo 1 (CVB): collapsed variational Bayesian HMM (strong indep. assumption)
  - Algo 2 (CVB): collapsed variational Bayesian HMM (weaker indep. assumption)
  - CGS: collapsed Gibbs Sampler for Bayesian HMM

Speed:



- EM is slow b/c of log-space computations
- VB is slow b/c of digamma computations
- Algo 1 (CVB) is the fastest!
- Algo 2 (CVB) is slow b/c it computes dynamic parameters
- CGS: an order of magnitude slower than any deterministic algorithm

Figure from Wang & Blunsom (2013)

Stochastic Variational Bayesian HMM

- **Task:** Human Chromatin Segmentation
- **Goal:** unsupervised segmentation of the genome
- **Data:** from ENCODE, "250 million observations consisting of twelve assays carried out in the chronic myeloid leukemia cell line K562"
- **Metric:** "the false discovery rate (FDR) of predicting active promoter elements in the sequence"
- **Models:**
  - DBN HMM: dynamic Bayesian HMM trained with standard EM
  - SVIHMM: stochastic variational inference for a Bayesian HMM
- **Main Takeaway:**
  - the two models perform at similar levels of FDR
  - SVIHMM takes **one hour**
  - DBNHMM takes **days**

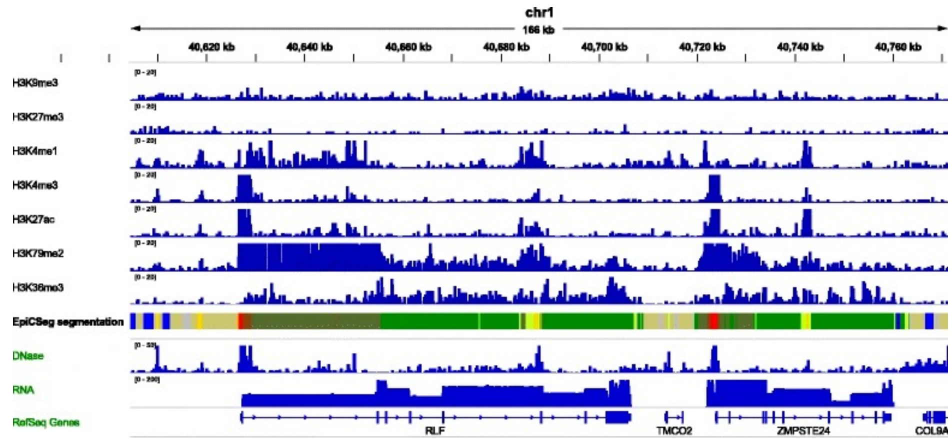


Figure from Foti et al. (2014)

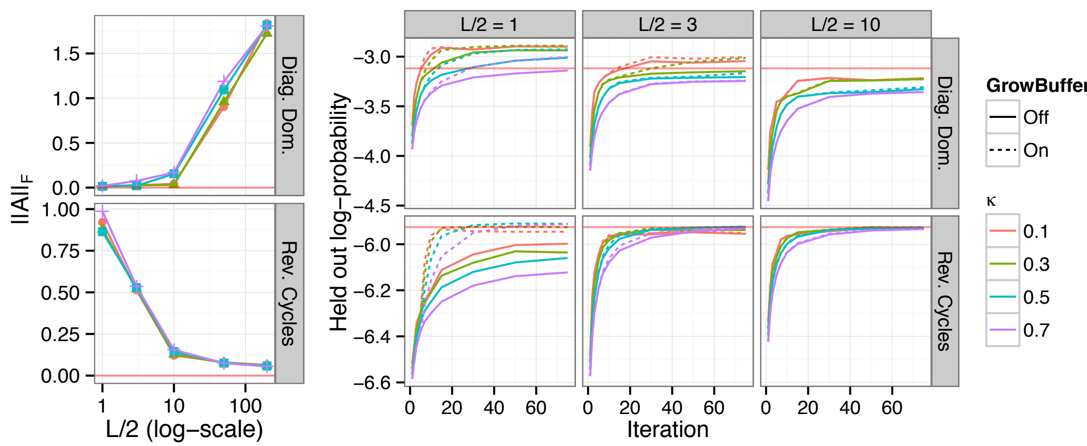


Figure from Mammana & Chung (2015)

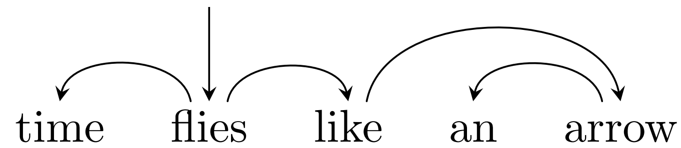
Grammar Induction

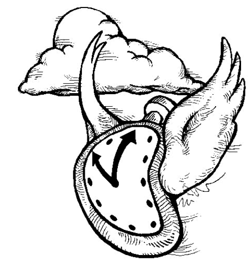
Question: Can maximizing (unsupervised) marginal likelihood produce useful results?

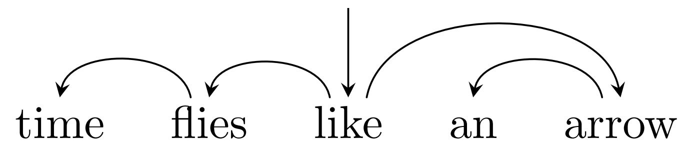
Answer: Let's look at an example…

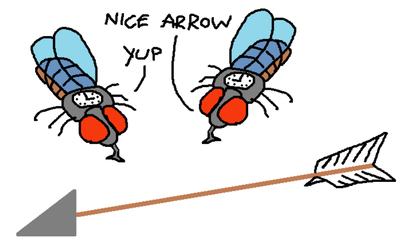
- **Babies** learn the syntax of their **native language** (e.g. English) just by **hearing** many sentences
- Can a **computer** similarly learn syntax of a **human language** just by looking at lots of example sentences?
  - This is the problem of Grammar Induction!
  - It's an unsupervised learning problem
  - We try to recover the **syntactic structure** for each sentence without any supervision

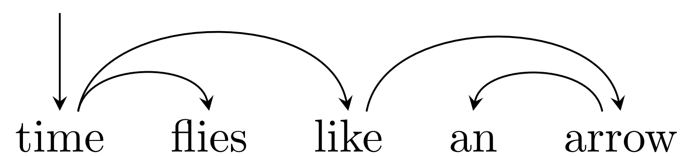
Grammar Induction



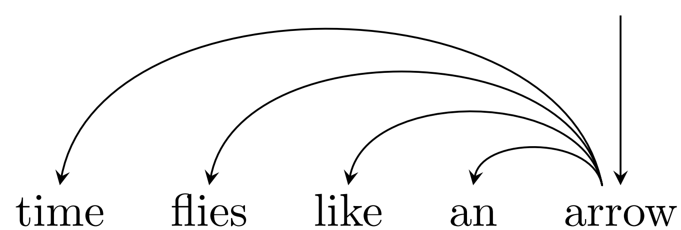








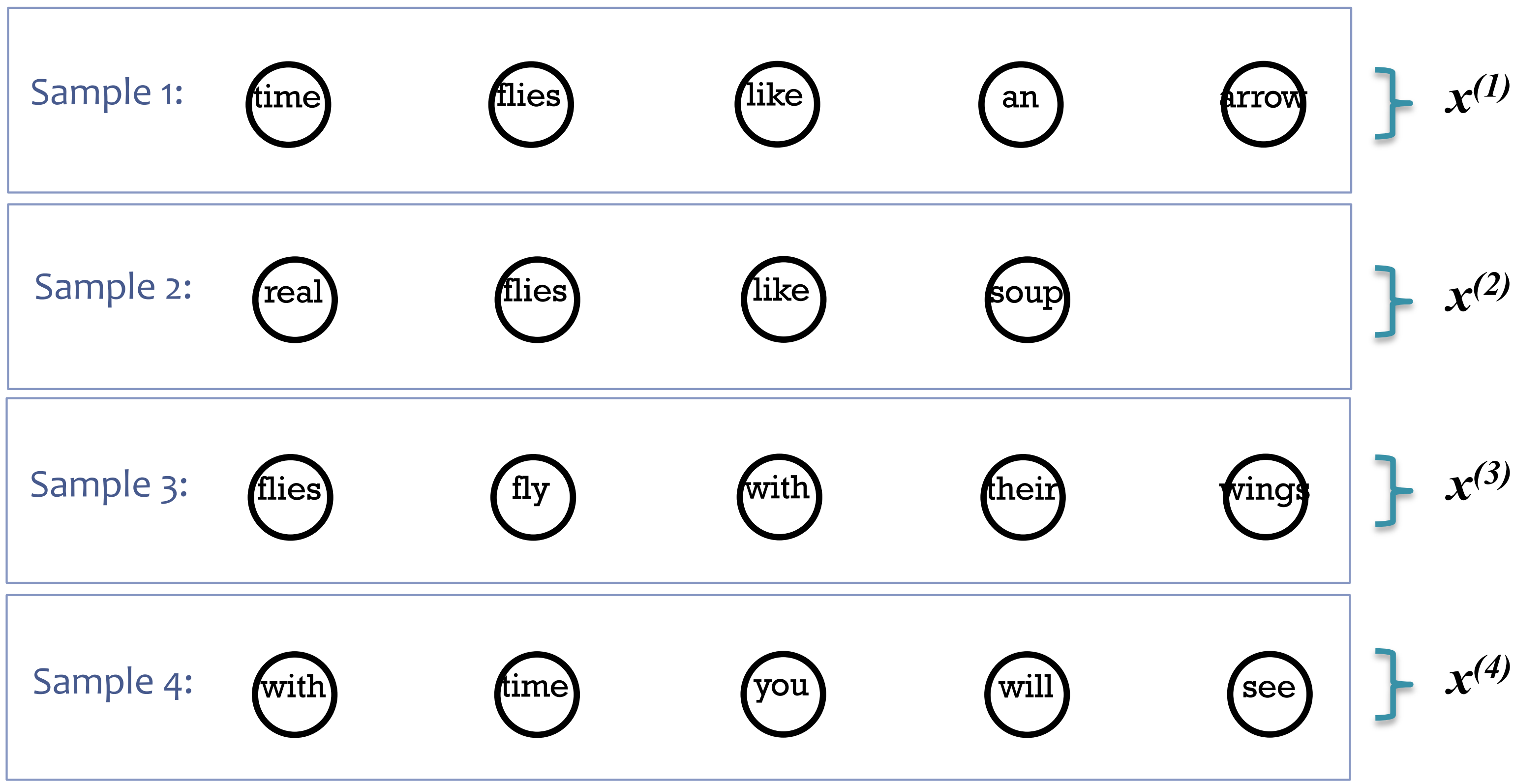
• • •



No semantic interpretation

Grammar Induction

Training Data: Sentences only, without parses



Test Data: Sentences **with** parses, so we can evaluate accuracy

Grammar Induction

Q: Does likelihood correlate with accuracy on a task we care about?

A: Yes, but there is still a wide range of accuracies for a particular likelihood value

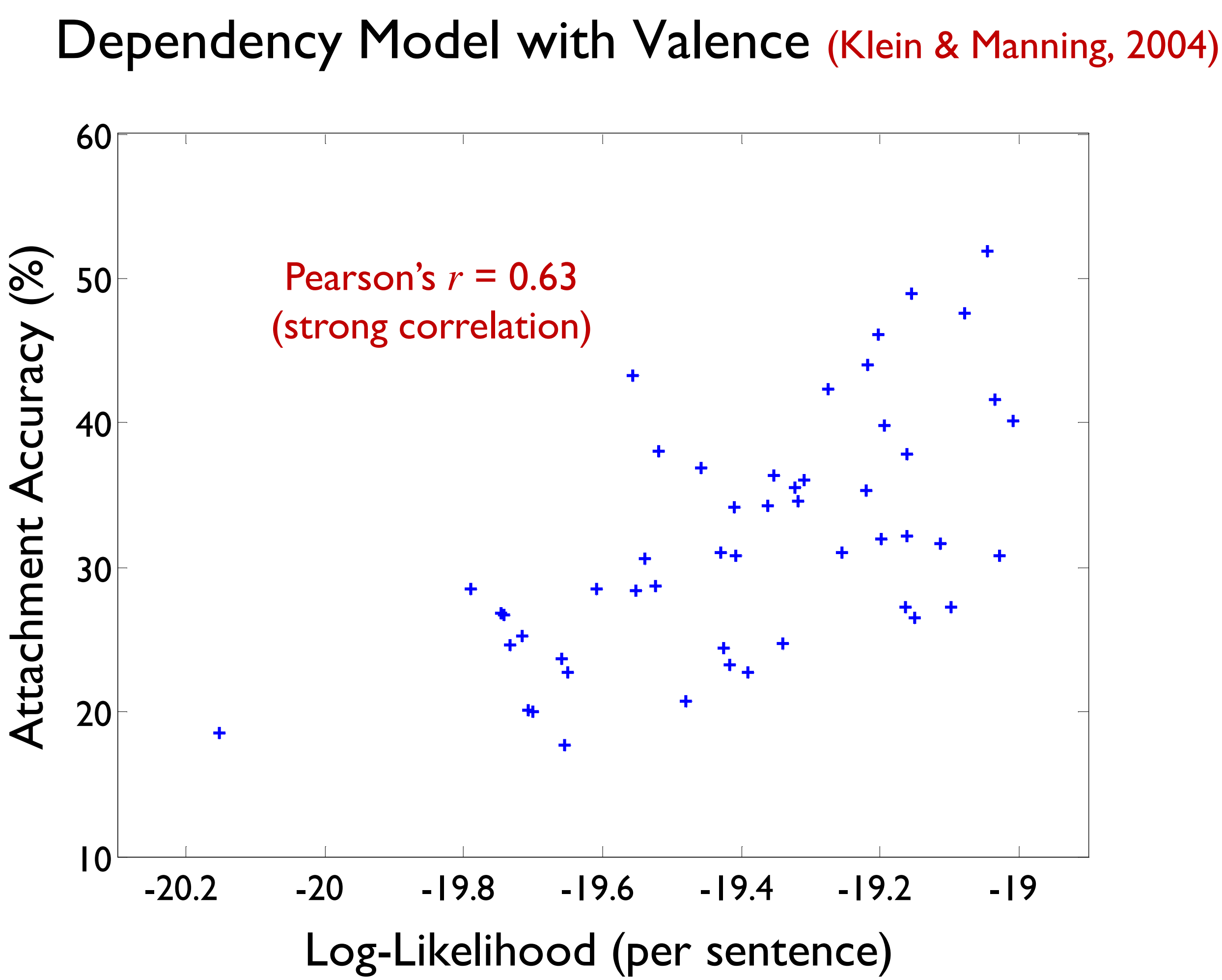
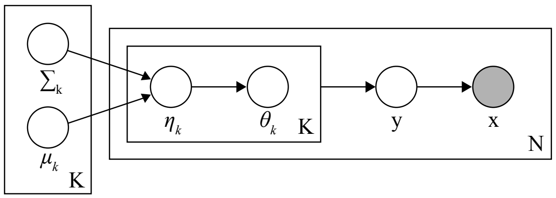


Figure from Gimpel & Smith (NAACL 2012) - slides

Grammar Induction

Graphical Model for Logistic Normal Probabilistic Grammar



y = syntactic parse
x = observed sentence

Settings:

EM Maximum likelihood estimate of $\boldsymbol{\theta}$ using the EM algorithm to optimize $p(\mathbf{x} \mid \boldsymbol{\theta})$ [14].

EM-MAP Maximum *a posteriori* estimate of $\boldsymbol{\theta}$ using the EM algorithm and a fixed symmetric Dirichlet prior with $\alpha > 1$ to optimize $p(\mathbf{x}, \boldsymbol{\theta} \mid \alpha)$. Tune α to maximize the likelihood of an unannotated development dataset, using grid search over $[1.1, 30]$.

VB-Dirichlet Use variational Bayes inference to estimate the posterior distribution $p(\boldsymbol{\theta} \mid \mathbf{x}, \alpha)$, which is a Dirichlet. Tune the symmetric Dirichlet prior's parameter α to maximize the likelihood of an unannotated development dataset, using grid search over $[0.0001, 30]$. Use the mean of the posterior Dirichlet as a point estimate for $\boldsymbol{\theta}$.

VB-EM-Dirichlet Use variational Bayes EM to optimize $p(\mathbf{x} \mid \boldsymbol{\alpha})$ with respect to $\boldsymbol{\alpha}$. Use the mean of the learned Dirichlet as a point estimate for $\boldsymbol{\theta}$ (similar to [5]).

VB-EM-Log-Normal Use variational Bayes EM to optimize $p(\mathbf{x} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma})$ with respect to $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$. Use the (exponentiated) mean of this Gaussian as a point estimate for $\boldsymbol{\theta}$.

Results:

	attachment accuracy (%)					
	Viterbi decoding			MBR decoding		
	$\lvert\mathbf{x}\rvert \le 10$	$\lvert\mathbf{x}\rvert \le 20$	all	$\lvert\mathbf{x}\rvert \le 10$	$\lvert\mathbf{x}\rvert \le 20$	all
Attach-Right	38.4	33.4	31.7	38.4	33.4	31.7
EM	45.8	39.1	34.2	46.1	39.9	35.9
EM-MAP, $\boldsymbol{\alpha} = 1.1$	45.9	39.5	34.9	46.2	40.6	36.7
VB-Dirichlet, $\boldsymbol{\alpha} = 0.25$	46.9	40.0	35.7	47.1	41.1	37.6
VB-EM-Dirichlet	45.9	39.4	34.9	46.1	40.6	36.9
VB-EM-Log-Normal, $\boldsymbol{\Sigma}_k^{(0)} = \mathbf{I}$	56.6	43.3	37.4	59.1	**45.9**	39.9
VB-EM-Log-Normal, families	**59.3**	**45.1**	**39.0**	**59.4**	**45.9**	**40.5**

Table 1: Attachment accuracy of different learning methods on unseen test data from the Penn Treebank of varying levels of difficulty imposed through a length filter. Attach-Right attaches each word to the word on its right and the last word to \$. EM and EM-MAP with a Dirichlet prior ($\alpha > 1$) are reproductions of earlier results [14, 18].

HIDDEN STATE CRFS

Case Study: Object Recognition

Data consists of images $\boldsymbol{x}$ and labels y.

pigeon

rhinoceros

leopard

llama

Case Study: Object Recognition

Data consists of images x and labels y.

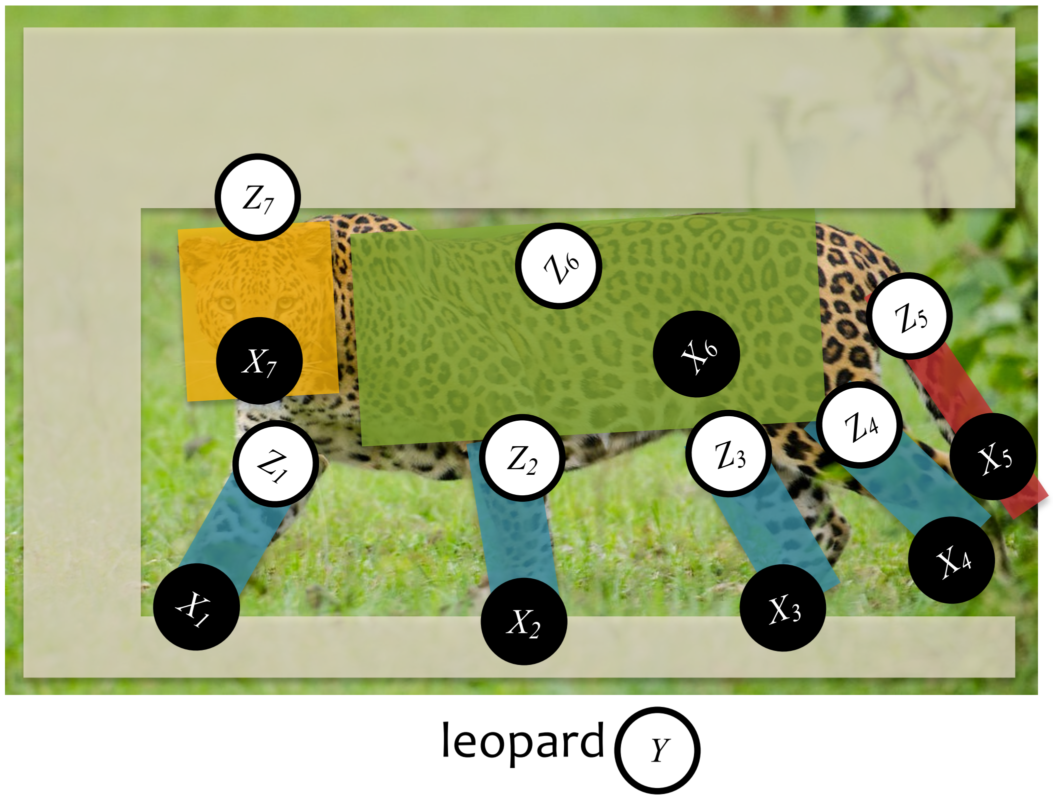
- Preprocess data into "patches"
- Posit a latent labeling z describing the object's parts (e.g. head, leg, tail, torso, grass)
- Define graphical model with these latent variables in mind
- z is not observed at train or test time

leopard

Case Study: Object Recognition

Data consists of images x and labels y.

- Preprocess data into "patches"
- Posit a latent labeling z describing the object's parts (e.g. head, leg, tail, torso, grass)
- Define graphical model with these latent variables in mind
- z is not observed at train or test time

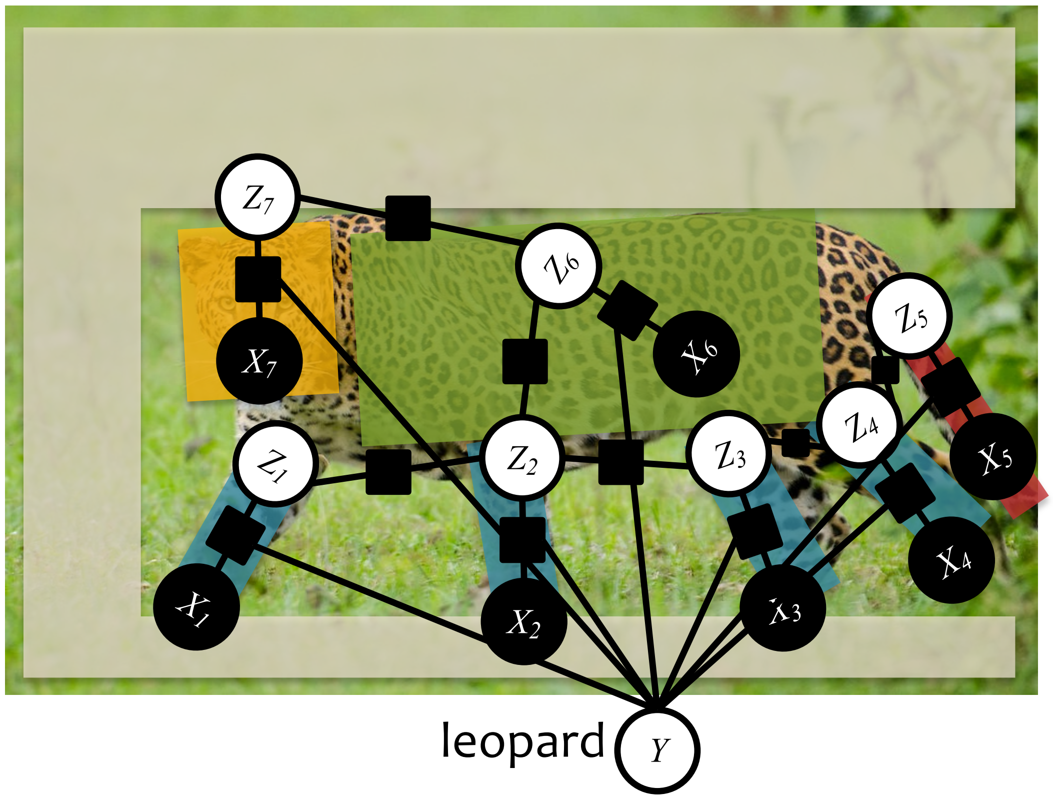


Case Study: Object Recognition

Data consists of images x and labels y.

- Preprocess data into "patches"
- Posit a latent labeling z describing the object's parts (e.g. head, leg, tail, torso, grass)

- Define graphical model with these latent variables in mind
- z is not observed at train or test time

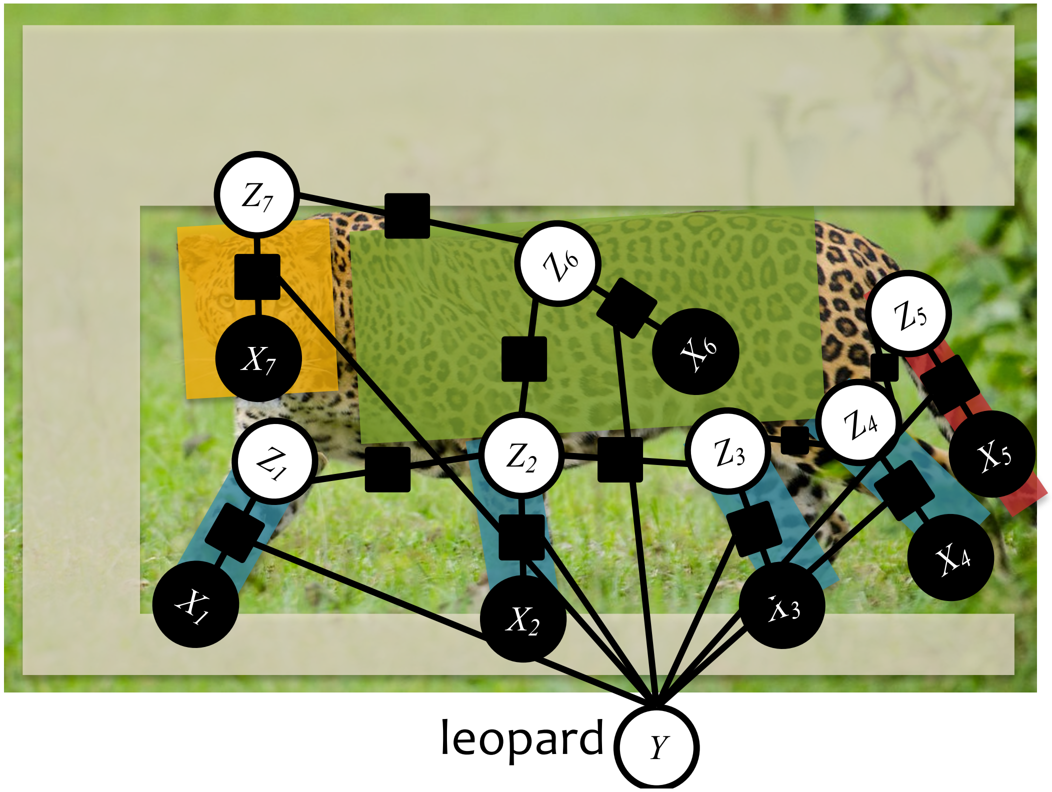


Hidden-state CRFs

Data: $\mathcal{D} = \{\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)}\}_{n=1}^{N}$

Joint model: $p_{\boldsymbol{\theta}}(\boldsymbol{y}, \boldsymbol{z} \mid \boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x}, \boldsymbol{\theta})} \prod_{\alpha} \psi_{\alpha}(\boldsymbol{y}_{\alpha}, \boldsymbol{z}_{\alpha}, \boldsymbol{x})$

Marginalized model: $p_{\boldsymbol{\theta}}(\boldsymbol{y} \mid \boldsymbol{x}) = \sum_{\boldsymbol{z}} p_{\boldsymbol{\theta}}(\boldsymbol{y}, \boldsymbol{z} \mid \boldsymbol{x})$



Hidden-state CRFs

Data: $\mathcal{D} = \{\boldsymbol{x}^{(n)}, \boldsymbol{y}^{(n)}\}_{n=1}^{N}$

Joint model: $p_{\boldsymbol{\theta}}(\boldsymbol{y}, \boldsymbol{z} \mid \boldsymbol{x}) = \frac{1}{Z(\boldsymbol{x}, \boldsymbol{\theta})} \prod_{\alpha} \psi_{\alpha}(\boldsymbol{y}_{\alpha}, \boldsymbol{z}_{\alpha}, \boldsymbol{x})$

Marginalized model: $p_{\boldsymbol{\theta}}(\boldsymbol{y} \mid \boldsymbol{x}) = \sum_{\boldsymbol{z}} p_{\boldsymbol{\theta}}(\boldsymbol{y}, \boldsymbol{z} \mid \boldsymbol{x})$

We can train using gradient based methods:

(the values $\boldsymbol{x}$ are omitted below for clarity)

$$\frac{d\ell(\boldsymbol{\theta}|\mathcal{D})}{d\boldsymbol{\theta}} = \sum_{n=1}^{N} \left(\mathbb{E}_{\boldsymbol{z} \sim p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{y}^{(n)})}[f_j(\boldsymbol{y}^{(n)}, \boldsymbol{z})] - \mathbb{E}_{\boldsymbol{y},\boldsymbol{z} \sim p_{\boldsymbol{\theta}}(\cdot,\cdot)}[f_j(\boldsymbol{y}, \boldsymbol{z})] \right)$$

$$= \sum_{n=1}^{N} \sum_{\alpha} \left(\sum_{\boldsymbol{z}_{\boldsymbol{\alpha}}} \underbrace{p_{\boldsymbol{\theta}}(\boldsymbol{z}_{\boldsymbol{\alpha}} \mid \boldsymbol{y}^{(n)})}_{\text{Inference on clamped factor graph}} f_{\alpha,j}(\boldsymbol{y}_{\alpha}^{(n)}, \boldsymbol{z}_{\alpha}) - \sum_{\boldsymbol{y}_{\boldsymbol{\alpha}}, \boldsymbol{z}_{\boldsymbol{\alpha}}} \underbrace{p_{\boldsymbol{\theta}}(\boldsymbol{y}_{\boldsymbol{\alpha}}, \boldsymbol{z}_{\boldsymbol{\alpha}})}_{\text{Inference on full factor graph}} f_{\alpha,j}(\boldsymbol{y}_{\boldsymbol{\alpha}}, \boldsymbol{z}_{\boldsymbol{\alpha}}) \right)$$

Inference on **clamped** factor graph

Inference on **full** factor graph