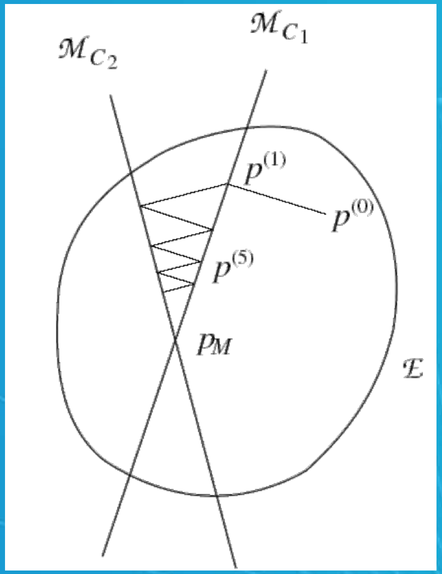
Probabilistic Graphical Models

Maximum likelihood learning of undirected GM

Eric Xing

Lecture 7, February 6, 2019

Reading: see class homepage



Recall: Learning Graphical Models

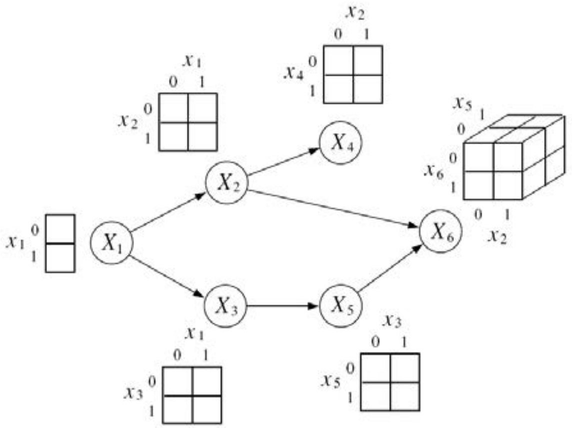
- Scenarios:
  - Completely observed GMs
    - directed
    - undirected
  - Partially or unobserved GMs
    - directed
    - undirected (an open research topic)
- Estimation principles:
  - Maximal likelihood estimation (MLE)
  - Bayesian estimation
  - Maximal conditional likelihood
  - Maximal "Margin"
  - Maximum entropy

- We use learning as a name for the process of estimating the parameters, and in some cases, the topology of the network, from data.

Recap: MLE for BNs

- Assuming the parameters for each CPD are globally independent, and all nodes are fully observed, then the log-likelihood function decomposes into a sum of local terms, one per node:

$$\ell(\theta; D) = \log p(D \mid \theta) = \log \prod_n \left(\prod_i p(x_{n,i} \mid \mathbf{x}_{\pi_i}, \theta_i) \right) = \sum_i \left(\sum_n \log p(x_{n,i} \mid \mathbf{x}_{\pi_i}, \theta_i) \right)$$



$$\theta_{ijk}^{ML} = \frac{n_{ijk}}{\sum_{i,j',k} n_{ij'k}}$$

MLE for undirected graphical models

- For directed graphical models, the log-likelihood decomposes into a sum of terms, one per family (node plus parents).
- For undirected graphical models, the log-likelihood does not decompose, because the normalization constant Z is a function of all the parameters

$$P(x_1, \ldots, x_n) = \frac{1}{Z} \prod_{c \in C} \psi_c(\mathbf{x}_c) \qquad Z = \sum_{x_1, \ldots, x_n} \prod_{c \in C} \psi_c(\mathbf{x}_c)$$

- In general, we will need to do inference (i.e., marginalization) to learn parameters for undirected models, even in the fully observed case.

Log Likelihood for UGMs with tabular clique potentials

- Sufficient statistics: for a UGM (V,E), the number of times that a configuration $\mathbf{x}$ (i.e., $\mathbf{X}_V=\mathbf{x}$) is observed in a dataset $D=\{\mathbf{x}_1,\ldots,\mathbf{x}_N\}$ can be represented as follows:

$$m(\mathbf{x}) \stackrel{\text{def}}{=} \sum_n \delta(\mathbf{x}, \mathbf{x}_n) \quad \text{(total count)}, \qquad \text{and} \qquad m(\mathbf{x}_c) \stackrel{\text{def}}{=} \sum_{\mathbf{x}_{V \setminus c}} m(\mathbf{x}) \quad \text{(clique count)}$$

- In terms of the counts, the log likelihood is given by:

$$p(D|\theta) = \prod_n \prod_{\mathbf{x}} p(\mathbf{x} \mid \theta)^{\delta(\mathbf{x},\mathbf{x}_n)}$$

$$\log p(D|\theta) = \sum_n \sum_{\mathbf{x}} \delta(\mathbf{x}, \mathbf{x}_n) \log p(\mathbf{x} \mid \theta) = \sum_{\mathbf{x}} \sum_n \delta(\mathbf{x}, \mathbf{x}_n) \log p(\mathbf{x} \mid \theta)$$

$$\ell = \sum_{\mathbf{x}} m(\mathbf{x}) \log \left(\frac{1}{Z} \prod_c \psi_c(\mathbf{x}_c) \right)$$

$$= \sum_c \sum_{\mathbf{x}_c} m(\mathbf{x}_c) \log \psi_c(\mathbf{x}_c) - N \log Z$$

- There is a nasty log Z in the likelihood

Log Likelihood for UGMs with tabular clique potentials

- Sufficient statistics: for a UGM (V,E), the number of times that a configuration $\mathbf{x}$ (i.e., $\mathbf{X}_V=\mathbf{x}$) is observed in a dataset $D=\{\mathbf{x}_1,\ldots,\mathbf{x}_N\}$ can be represented as follows:

$$m(\mathbf{x}) \stackrel{\text{def}}{=} \sum_n \delta(\mathbf{x}, \mathbf{x}_n) \quad \text{(total count)}, \qquad \text{and} \qquad m(\mathbf{x}_c) \stackrel{\text{def}}{=} \sum_{\mathbf{x}_{V \setminus c}} m(\mathbf{x}) \quad \text{(clique count)}$$

- In terms of the counts, the log likelihood is given by:

$$\log p(D|\theta) = \sum_c \sum_{\mathbf{x}_c} m(\mathbf{x}_c) \log \psi_c(\mathbf{x}_c) - N \log Z$$

- There is a nasty log Z in the likelihood

Derivative of log Likelihood

- Log-likelihood:

$$\ell = \sum_c \sum_{\mathbf{x}_c} m(\mathbf{x}_c) \log \psi_c(\mathbf{x}_c) - N \log Z$$

- First term:

$$\frac{\partial \ell_1}{\partial \psi_c(\mathbf{x}_c)} = \frac{m(\mathbf{x}_c)}{\psi_c(\mathbf{x}_c)}$$

- Second term:

$$\frac{\partial \log Z}{\partial \psi_c(\mathbf{x}_c)} = \frac{1}{Z} \frac{\partial}{\partial \psi_c(\mathbf{x}_c)} \left(\sum_{\tilde{\mathbf{x}}} \prod_d \psi_d(\tilde{\mathbf{x}}_d) \right)$$

$$= \frac{1}{Z} \sum_{\tilde{\mathbf{x}}} \delta(\tilde{\mathbf{x}}_c, \mathbf{x}_c) \frac{\partial}{\partial \psi_c(\mathbf{x}_c)} \left(\prod_d \psi_d(\tilde{\mathbf{x}}_d) \right)$$

Set the value of variables to $\mathbf{x}$

$$= \sum_{\tilde{\mathbf{x}}} \delta(\tilde{\mathbf{x}}_c, \mathbf{x}_c) \frac{1}{\psi_c(\tilde{\mathbf{x}}_c)} \frac{1}{Z} \prod_d \psi_d(\tilde{\mathbf{x}}_d)$$

$$= \frac{1}{\psi_c(\mathbf{x}_c)} \sum_{\tilde{\mathbf{x}}} \delta(\tilde{\mathbf{x}}_c, \mathbf{x}_c) p(\tilde{\mathbf{x}}) = \frac{p(\mathbf{x}_c)}{\psi_c(\mathbf{x}_c)}$$

Conditions on Clique Marginals

- Derivative of log-likelihood

$$\frac{\partial \ell}{\partial \psi_c(\mathbf{x}_c)} = \frac{m(\mathbf{x}_c)}{\psi_c(\mathbf{x}_c)} - N\frac{p(\mathbf{x}_c)}{\psi_c(\mathbf{x}_c)}$$

- Hence, for the maximum likelihood parameters, we know that:

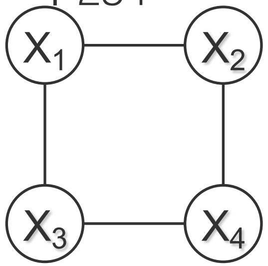
$$p^*_{MLE}(\mathbf{x}_c) = \frac{m(\mathbf{x}_c)}{N} \overset{\text{def}}{=} \tilde{p}(\mathbf{x}_c)$$

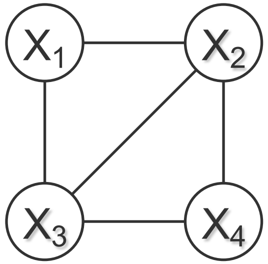
- In other words, at the maximum likelihood setting of the parameters, for each clique, the model marginals must be equal to the observed marginals (empirical counts).

- This doesn't tell us how to get the ML parameters, it just gives us a condition that must be satisfied when we have them.

MLE for undirected graphical models

- Is the graph decomposable (triangulated)?
- Are all the clique potentials defined on maximal cliques (not sub-cliques)? e.g., ψ_{123}, ψ_{234} not ψ_{12}, ψ_{23}, …





- Are the clique potentials full tables (or Gaussians), or parameterized more compactly, e.g. $\psi_c(\mathbf{x}_c) = \exp\left(\sum_c \theta_k f_k(\mathbf{x}_c)\right)$

Decomposable?	Max clique?	Tabular?	Method
√	√	√	Direct
-	-	√	IPF
-	-	-	Gradient
-	-	-	GIS

Two Workhorse Algorithms

- Iterative Proportional Fitting (IPF)
  - For MRFs with tabular potentials

- Generalized Iterative Scaling (GIS)
  - For MRFs with features based potentials

Iterative Proportional Fitting (IPF)

- From the derivative of the likelihood:

$$\frac{\partial \ell}{\partial \psi_c(\mathbf{x}_c)} = \frac{m(\mathbf{x}_c)}{\psi_c(\mathbf{x}_c)} - N\frac{p(\mathbf{x}_c)}{\psi_c(\mathbf{x}_c)}$$

- we can derive another relationship:

$$\frac{\tilde{p}(\mathbf{x}_c)}{\psi_c(\mathbf{x}_c)} = \frac{p(\mathbf{x}_c)}{\psi_c(\mathbf{x}_c)}$$

  in which ψ_c appears implicitly in the model marginal $p(\mathbf{x}_c)$.

- This is therefore a fixed-point equation for ψ_c.
  - Solving ψ_c in closed-form is hard, because it appears on both sides of this implicit nonlinear equation.
- The idea of IPF is to hold ψ_c fixed on the right hand side (both in the numerator and denominator) and solve for it on the left hand side. We cycle through all cliques, then iterate:

$$\psi_c^{(t+1)}(\mathbf{x}_c) = \psi_c^{(t)}(\mathbf{x}_c)\frac{\tilde{p}(\mathbf{x}_c)}{p^{(t)}(\mathbf{x}_c)}$$

Need to do inference here

Properties of IPF Updates

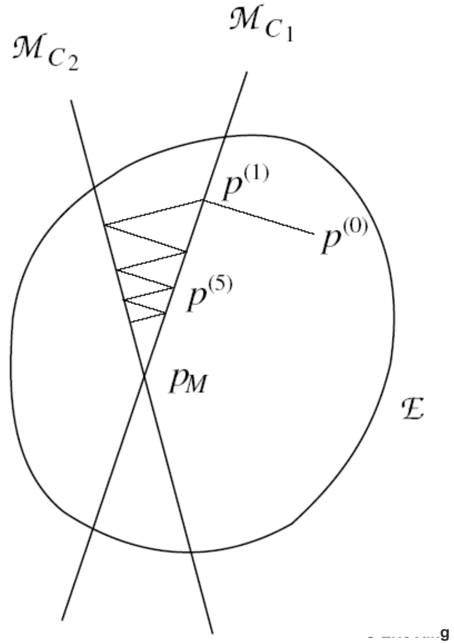
- IPF iterates a set of fixed-point equations:

$$\psi_c^{(t+1)}(\mathbf{x}_c) = \psi_c^{(t)}(\mathbf{x}_c)\frac{\tilde{p}(\mathbf{x}_c)}{p^{(t)}(\mathbf{x}_c)}$$

- However, we can prove it is also a coordinate ascent algorithm (coordinates = parameters of clique potentials).

- Hence at each step, it will increase the log-likelihood, and it will converge to a global maximum.

- I-projection: finding a distribution with the correct marginals that has the maximal entropy



KL Divergence View

- IPF can be seen as coordinate ascent in the likelihood using the way of expressing likelihoods using KL divergences.
- We can show that maximizing the log likelihood is equivalent to minimizing the KL divergence (cross entropy) from the observed distribution to the model distribution:

$$\max \ell \Leftrightarrow \min KL\left(\tilde{p}(x) \| p(x \mid \theta)\right) = \sum_{x} \tilde{p}(x) \log \frac{\tilde{p}(x)}{p(x \mid \theta)}$$

- Using a property of KL divergence based on the conditional chain rule: $p(x) = p(x_a)p(x_b/x_a)$:

$$\begin{aligned} KL\left(q(x_a, x_b) \| p(x_a, x_b)\right) &= \sum_{x_a, x_b} q(x_a) q(x_b \mid x_a) \log \frac{q(x_a) q(x_b \mid x_a)}{p(x_a) p(x_b \mid x_a)} \\ &= \sum_{x_a, x_b} q(x_a) q(x_b \mid x_a) \log \frac{q(x_a)}{p(x_a)} + \sum_{x_a, x_b} q(x_a) q(x_b \mid x_a) \log \frac{q(x_b \mid x_a)}{p(x_b \mid x_a)} \\ &= KL\left(q(x_a) \| p(x_a)\right) + \sum_{x_a} q(x_a) KL\left(q(x_b \mid x_a) \| p(x_b \mid x_a)\right) \end{aligned}$$

IPF minimizes KL divergence

- Putting things together, we have

$$KL\left(\tilde{p}(\mathbf{x}) \| p(\mathbf{x}|\theta)\right) = KL\left(\tilde{p}(\mathbf{x}_c) \| p(\mathbf{x}_c|\theta)\right) + \sum_{x_a} \tilde{p}(\mathbf{x}_c) KL\left(\tilde{p}(\mathbf{x}_{-c}|\mathbf{x}_c) \| p(\mathbf{x}_{-c}|\mathbf{x}_c)\right)$$

  It can be shown that changing the clique potential ψ_C has no effect on the conditional distribution, so the second term in unaffected.

- To minimize the first term, we set the marginal to the observed marginal, just as in IPF.
  - Note that this is only good when the model is decomposable !

- We can interpret IPF updates as retaining the "old" conditional probabilities $p^{(t)}(\mathbf{x}_{-c}|\mathbf{x}_c)$ while replacing the "old" marginal probability $p^{(t)}(\mathbf{x}_c)$ with the observed marginal $\tilde{p}(\mathbf{x}_c)$

Two Workhorse Algorithms

- Iterative Proportional Fitting (IPF)
  - For MRFs with tabular potentials

- Generalized Iterative Scaling (GIS)
  - For MRFs with features based potentials

Feature-based Clique Potentials

- So far we have discussed the most general form of an undirected graphical model in which cliques are parameterized by general "tabular" potential functions $\psi_c(\mathbf{x}_c)$.
- But for large cliques these general potentials are exponentially costly for inference and have exponential numbers of parameters that we must learn from limited data.
- One solution: change the graphical model to make cliques smaller. But this changes the dependencies, and may force us to make more independence assumptions than we would like.
- Another solution: keep the same graphical model, but use a less general parameterization of the clique potentials.
- This is the idea behind feature-based models.

Features

- Consider a clique $\mathbf{x}_c$ of random variables in a UGM, e.g. three consecutive characters $c_1c_2c_3$ in a string of English text.
- How would we build a model of $p(c_1c_2c_3)$?
  - If we use a single clique function over $c_1c_2c_3$, the full joint clique potential would be huge: 26^3-1 parameters.
  - However, we often know that some particular joint settings of the variables in a clique are quite likely or quite unlikely. e.g. ing, ate, ion, ?ed, qu?, jkx, zzz,...
- A "feature" is a function which is vacuous over all joint settings except a few particular ones on which it is high or low.
  - For example, we might have $f_{\text{ing}}(c_1c_2c_3)$ which is 1 if the string is 'ing' and 0 otherwise, and similar features for '?ed', etc.
- We can also define features when the inputs are continuous. Then the idea of a cell on which it is active disappears, but we might still have a compact parameterization of the feature.

Features as Micropotentials

- By exponentiating them, each feature function can be made into a "micropotential". We can **multiply** these micropotentials together to get a clique potential.
- Example: a clique potential $\psi(c_1 c_2 c_3)$ could be expressed as:

$$\psi_c(c_1, c_2, c_3) = e^{\theta_{\text{ing}} f_{\text{ing}}} \times e^{\theta_{\text{?ed}} f_{\text{?ed}}} \times \ldots$$

$$= \exp\left\{ \sum_{k=1}^{K} \theta_k f_k(c_1, c_2, c_3) \right\}$$

- This is still a potential over 26^3 possible settings, but only uses K parameters if there are K features.
  - By having one indicator function per combination of $\mathbf{x}_c$, we recover the standard tabular potential.

Combining Features

- Each feature has a weight θ_k which represents the numerical strength of the feature and whether it increases or decreases the probability of the clique.
- The marginal over the clique is a generalized exponential family distribution, actually, a GLIM:

$$p(c_1, c_2, c_3) \propto \exp\left\{ \begin{array}{l} \theta_{\text{ing}} f_{\text{ing}}(c_1, c_2, c_3) + \theta_{\text{?ed}} f_{\text{?ed}}(c_1, c_2, c_3) + \\ \theta_{\text{qu?}} f_{\text{qu?}}(c_1, c_2, c_3) + \theta_{\text{zzz}} f_{\text{zzz}}(c_1, c_2, c_3) + \cdots \end{array} \right\}$$

- In general, the features may be overlapping, unconstrained indicators or any function of any subset of the clique variables:

$$\psi_c(\mathbf{x}_c) \stackrel{\text{def}}{=} \exp\left\{ \sum_{i \in I_c} \theta_k f_k(\mathbf{x}_{c_i}) \right\}$$

- How can we combine feature into a probability model?

Feature Based Model

- We can multiply these clique potentials as usual:

$$p(\mathbf{x}) = \frac{1}{Z(\theta)} \prod_c \psi_c(\mathbf{x}_c) = \frac{1}{Z(\theta)} \exp\left\{ \sum_c \sum_{i \in I_c} \theta_k f_k(\mathbf{x}_{c_i}) \right\}$$

- However, in general we can forget about associating features with cliques and just use a simplified form:

$$p(\mathbf{x}) = \frac{1}{Z(\theta)} \exp\left\{ \sum_i \theta_i f_i(\mathbf{x}_{c_i}) \right\}$$

- This is just our friend the exponential family model, with the features as sufficient statistics!

- Learning: recall that in IPF, we have $\psi_c^{(t+1)}(\mathbf{x}_c) = \psi_c^{(t)}(\mathbf{x}_c) \frac{\tilde{p}(\mathbf{x}_c)}{p^{(t)}(\mathbf{x}_c)}$
  - Not obvious how to use this rule to update the weights and features individually !!!

MLE of Feature Based UGMs

- Scaled likelihood function

$$\tilde{\ell}(\theta; D) = \ell(\theta; D)/N = \frac{1}{N}\sum_n \log p(x_n \mid \theta)$$

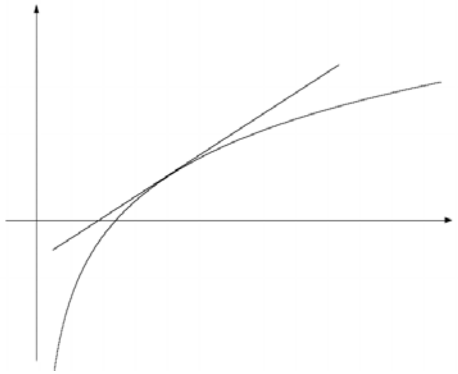
$$= \sum_x \tilde{p}(x) \log p(x \mid \theta)$$

$$= \sum_x \tilde{p}(x) \sum_i \theta_i f_i(x) - \log Z(\theta)$$

- Instead of optimizing this objective directly, we attack its lower bound
  - The logarithm has a linear upper bound …

$$\log Z(\theta) \le \mu Z(\theta) - \log \mu - 1$$

  - This bound holds for all μ, in particular, for $\mu = Z^{-1}(\theta^{(t)})$
- Thus we have



$$\tilde{\ell}(\theta; D) \ge \sum_x \tilde{p}(x) \sum_i \theta_i f_i(x) - \frac{Z(\theta)}{Z(\theta^{(t)})} - \log Z(\theta^{(t)}) + 1$$

Generalized Iterative Scaling (GIS)

- Lower bound of scaled loglikelihood

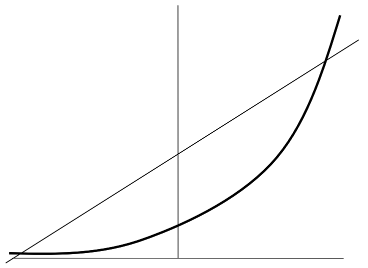
$$\tilde{\ell}(\theta;D) \ge \sum_x \tilde{p}(x)\sum_i \theta_i f_i(x) - \frac{Z(\theta)}{Z(\theta^{(t)})} - \log Z(\theta^{(t)}) + 1$$

- Define

$$\Delta\theta_i^{(t)} \overset{\text{def}}{=} \theta_i - \theta_i^{(t)}$$

$$\begin{aligned}
\tilde{\ell}(\theta;D) &\ge \sum_x \tilde{p}(x)\sum_i \theta_i f_i(x) - \frac{1}{Z(\theta^{(t)})}\sum_x \exp\left\{\sum_i \theta_i f_i(x)\right\} - \log Z(\theta^{(t)}) + 1 \\
&= \sum_i \theta_i \sum_x \tilde{p}(x) f_i(x) - \frac{1}{Z(\theta^{(t)})}\sum_x \exp\left\{\sum_i \theta_i^{(t)} f_i(x)\right\} \exp\left\{\sum_i \Delta\theta_i^{(t)} f_i(x)\right\} - \log Z(\theta^{(t)}) + 1 \\
&= \sum_i \theta_i \sum_x \tilde{p}(x) f_i(x) - \sum_x p(x \mid \theta^{(t)}) \exp\left\{\sum_i \Delta\theta_i^{(t)} f_i(x)\right\} - \log Z(\theta^{(t)}) + 1
\end{aligned}$$

- Relax again
  - Assume $f_i(x) \ge 0, \quad \sum_i f_i(x) = 1$
  - Convexity of exponential: $\exp\left(\sum_i \pi_i x_i\right) \le \sum_i \pi_i \exp(x_i)$



- We have:

$$\tilde{\ell}(\theta;D) \ge \sum_i \theta_i \sum_x \tilde{p}(x) f_i(x) - \sum_x p(x \mid \theta^{(t)}) \sum_i f_i(x) \exp\left(\Delta\theta_i^{(t)}\right) - \log Z(\theta^{(t)}) + 1 \overset{\text{def}}{=} \Lambda(\theta)$$

GIS

- Lower bound of scaled loglikelihood

$$\tilde{\ell}(\theta;D) \geq \sum_i \theta_i \sum_x \tilde{p}(x) f_i(x) - \sum_x p(x \mid \theta^{(t)}) \sum_i f_i(x) \exp\left(\Delta\theta_i^{(t)}\right) - \log Z(\theta^{(t)}) + 1 \overset{\text{def}}{=} \Lambda(\theta)$$

- Take derivative:

$$\frac{\partial \Lambda}{\partial \theta_i} = \sum_x \tilde{p}(x) f_i(x) - \exp\left(\Delta\theta_i^{(t)}\right) \sum_x p(x \mid \theta^{(t)}) f_i(x)$$

- Set to zero

$$e^{\Delta\theta_i^{(t)}} = \frac{\sum_x \tilde{p}(x) f_i(x)}{\sum_x p(x \mid \theta^{(t)}) f_i(x)} = \frac{\sum_x \tilde{p}(x) f_i(x)}{\sum_x p^{(t)}(x) f_i(x)} Z(\theta^{(t)})$$

  - where $p^{(t)}(x)$ is the unnormalized version of $p(x|\theta^{(t)})$

- Update

$$\theta_i^{(t+1)} = \theta_i^{(t)} + \Delta\theta_i^{(t)} \Rightarrow p^{(t+1)}(x) = p^{(t)}(x) \prod_i e^{\Delta\theta_i^{(t)} f_i(x)}$$

$$p^{(t+1)}(x) = \frac{p^{(t)}(x)}{Z(\theta^{(t)})} \prod_i \left(\frac{\sum_x \tilde{p}(x) f_i(x)}{\sum_x p^{(t)}(x) f_i(x)} Z(\theta^{(t)}) \right)^{f_i(x)}$$

$$\Rightarrow \quad = \frac{p^{(t)}(x)}{Z(\theta^{(t)})} \prod_i \left(\frac{\sum_x \tilde{p}(x) f_i(x)}{\sum_x p^{(t)}(x) f_i(x)} \right)^{f_i(x)} \left(Z(\theta^{(t)}) \right)^{\sum_i f_i(x)}$$

$$= p^{(t)}(x) \prod_i \left(\frac{\sum_x \tilde{p}(x) f_i(x)}{\sum_x p^{(t)}(x) f_i(x)} \right)^{f_i(x)}$$

Recall IPF:

$$\psi_c^{(t+1)}(\mathbf{x}_c) = \psi_c^{(t)}(\mathbf{x}_c) \frac{\tilde{p}(\mathbf{x}_c)}{p^{(t)}(\mathbf{x}_c)}$$

Summary

- IPF is a general algorithm for finding MLE of UGMs.
  - a fixed-point equation for ψ_c over single cliques, coordinate ascent
  - I-projection in the clique marginal space
  - Requires the potential to be fully parameterized
  - The clique described by the potentials do not have to be max-clique
  - For fully decomposable model, reduces to a single step iteration
- GIS
  - Iterative scaling on general UGM with feature-based potentials
  - IPF is a special case of GIS which the clique potential is built on features defined as an indicator function of clique configurations.

GIS:

$$p^{(t+1)}(x) = p^{(t)}(x)\prod_i \left(\frac{\sum_x \tilde{p}(x) f_i(x)}{\sum_x p^{(t)}(x) f_i(x)}\right)^{f_i(x)}$$

$$\theta_i^{(t+1)} = \theta_i^{(t)} + \log\left(\frac{\sum_x \tilde{p}(x) f_i(x)}{\sum_x p^{(t)}(x) f_i(x)}\right)$$

IPF:

$$\psi_c^{(t+1)}(\mathbf{x}_c) = \psi_c^{(t)}(\mathbf{x}_c)\frac{\tilde{p}(\mathbf{x}_c)}{p^{(t)}(\mathbf{x}_c)}$$

Where does the exponential form come from?

- Review: Maximum Likelihood for exponential family

$$\ell(\theta; D) = \sum_x m(x) \log p(x \mid \theta)$$
$$= \sum_x m(x) \left(\sum_i \theta_i f_i(x) - \log Z(\theta) \right)$$
$$= \sum_x m(x) \sum_i \theta_i f_i(x) - N \log Z(\theta)$$

$$\frac{\partial}{\partial \theta_i} \ell(\theta; D) = \sum_x m(x) f_i(x) - N \frac{\partial}{\partial \theta_i} \log Z(\theta)$$
$$= \sum_x m(x) f_i(x) - N \sum_x p(x \mid \theta) f_i(x)$$

$$\Rightarrow \quad \sum_x p(x \mid \theta) f_i(x) = \sum_x \frac{m(x)}{N} f_i(x) = \sum_x \tilde{p}(x \mid \theta) f_i(x)$$

- i.e., At ML estimate, the expectations of the sufficient statistics under the model must match empirical feature average.

Maximum Entropy

- We can approach the modeling problem from an entirely different point of view. Begin with some fixed feature expectations:

$$\sum_x p(x) f_i(x) = \alpha_i$$

- Assuming expectations are consistent, there may exist many distributions which satisfy them. Which one should we select?
    - The most uncertain or flexible one, i.e., the one with maximum entropy.
- This yields a new optimization problem:

$$\begin{aligned} \max_p \ & \mathrm{H}\big(p(x)\big) = -\sum_x p(x) \log p(x) \\ \text{s.t.} \ & \sum_x p(x) f_i(x) = \alpha_i \\ & \sum_x p(x) = 1 \end{aligned}$$

This is a **variational** definition of a distribution!

Solution to the MaxEnt Problem

- To solve the MaxEnt problem, we use Lagrange multipliers:

$$L = -\sum_x p(x)\log p(x) - \sum_i \theta_i\left(\sum_x p(x) f_i(x) - \alpha_i\right) - \mu\left(\sum_x p(x) - 1\right)$$

$$\frac{\partial L}{\partial p(x)} = 1 + \log p(x) - \sum_i \theta_i f_i(x) - \mu$$

$$p^*(x) = e^{\mu-1}\exp\left\{\sum_i \theta_i f_i(x)\right\}$$

$$Z(\theta) = e^{\mu-1} = \sum_x \exp\left\{\sum_i \theta_i f_i(x)\right\} \qquad (\text{since } \textstyle\sum_x p^*(x) = 1)$$

$$p(x|\theta) = \frac{1}{Z(\theta)}\exp\left\{\sum_i \theta_i f_i(x)\right\}$$

- So feature constraints + MaxEnt $\Rightarrow$ exponential family.
- Problem is strictly convex w.r.t. p, so solution is unique.

A more general MaxEnt problem

$$\min_p \quad \mathrm{KL}\big(p(x) \,\|\, h(x)\big)$$

$$\stackrel{\text{def}}{=} \sum_x p(x) \log \frac{p(x)}{h(x)} = -\mathrm{H}(p) - \sum_x p(x) \log h(x)$$

$$\text{s.t.} \quad \sum_x p(x) f_i(x) = \alpha_i$$

$$\sum_x p(x) = 1$$

$$\Rightarrow \qquad p(x|\theta) = \frac{1}{Z(\theta)} h(x) \exp\left\{ \sum_i \theta_i f_i(x) \right\}$$

Constraints from Data

- Where do the constraints α_i come from?
- Just as before, measure the empirical counts on the training data:

$$\alpha_i = \sum_x \frac{m(\mathbf{x})}{N} f_i(\mathbf{x}) = \sum_x \tilde{p}(\mathbf{x}) f_i(\mathbf{x})$$

- This also ensures consistency automatically.
- Known as the "method of moments". (c.f. law of large numbers)
- We have seen a case of convex duality:
  - In one case, we assume exponential family and show that ML implies model expectations must match empirical expectations.
  - In the other case, we assume model expectations must match empirical feature counts and show that MaxEnt implies exponential family distribution.
  - No duality gap $\Rightarrow$ yield the same value of the objective

Geometric interpretation

- All exponential family distribution:

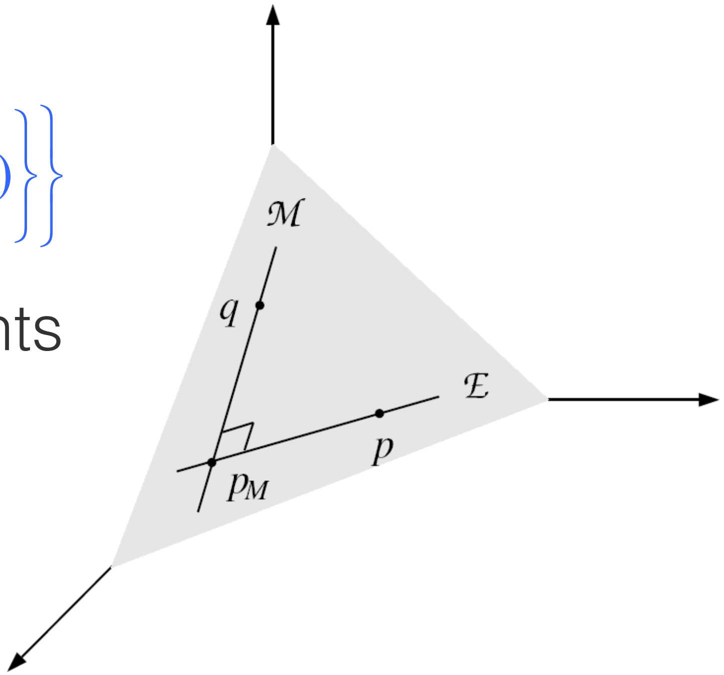
$$\mathcal{E} = \left\{ p(x) \, : \, p(x|\theta) = \frac{1}{Z(\theta)} h(x) \exp\left\{ \sum_i \theta_i f_i(x) \right\} \right\}$$

- All distributions satisfying moment constraints

$$\mathcal{M} = \left\{ p(x) \, : \, \sum_x p(x) f_i(x) = \sum_x \tilde{p}(x) f_i(x) \right\}$$

- Pythagorean theorem

$$\mathrm{KL}(q \,\|\, p) = \mathrm{KL}(q \,\|\, p_M) + \mathrm{KL}(p_M \,\|\, p)$$



MaxEnt:

$$\min_p \quad \mathrm{KL}(q \,\|\, h)$$
$$\text{s.t.} \qquad q \in \mathcal{M}$$
$$\mathrm{KL}(q \,\|\, h) = \mathrm{KL}(q \,\|\, p_M) + \cancel{\mathrm{KL}(p_M \,\|\, h)}$$

MaxLik:

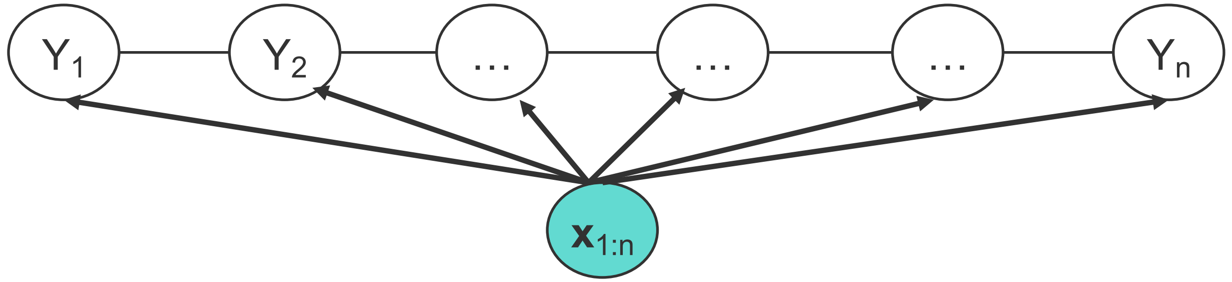
$$\min_p \quad \mathrm{KL}(\tilde{p} \,\|\, p)$$
$$\text{s.t.} \qquad q \in \mathcal{E}$$
$$\mathrm{KL}(\tilde{p} \,\|\, p) = \cancel{\mathrm{KL}(\tilde{p} \,\|\, p_M)} + \mathrm{KL}(p_M \,\|\, p)$$

Summary

- Exponential family distribution can be viewed as the solution to an variational expression --- the maximum entropy!
- The max-entropy principle to parameterization offers a dual perspective to the MLE.

Supplementary

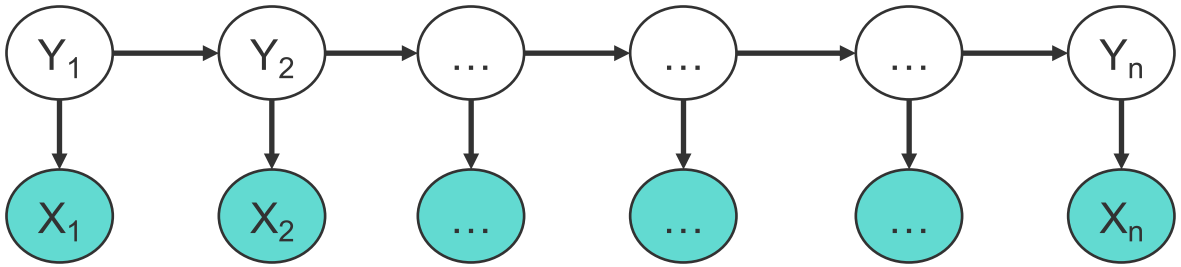
Case Study: Conditional Random Fields



$$P(\mathbf{y}_{1:n}|\mathbf{x}_{1:n}) = \frac{1}{Z(\mathbf{x}_{1:n})}\prod_{i=1}^{n}\phi(y_i, y_{i-1}, \mathbf{x}_{1:n}) = \frac{1}{Z(\mathbf{x}_{1:n}, \mathbf{w})}\prod_{i=1}^{n}\exp(\mathbf{w}^T\mathbf{f}(y_i, y_{i-1}, \mathbf{x}_{1:n}))$$

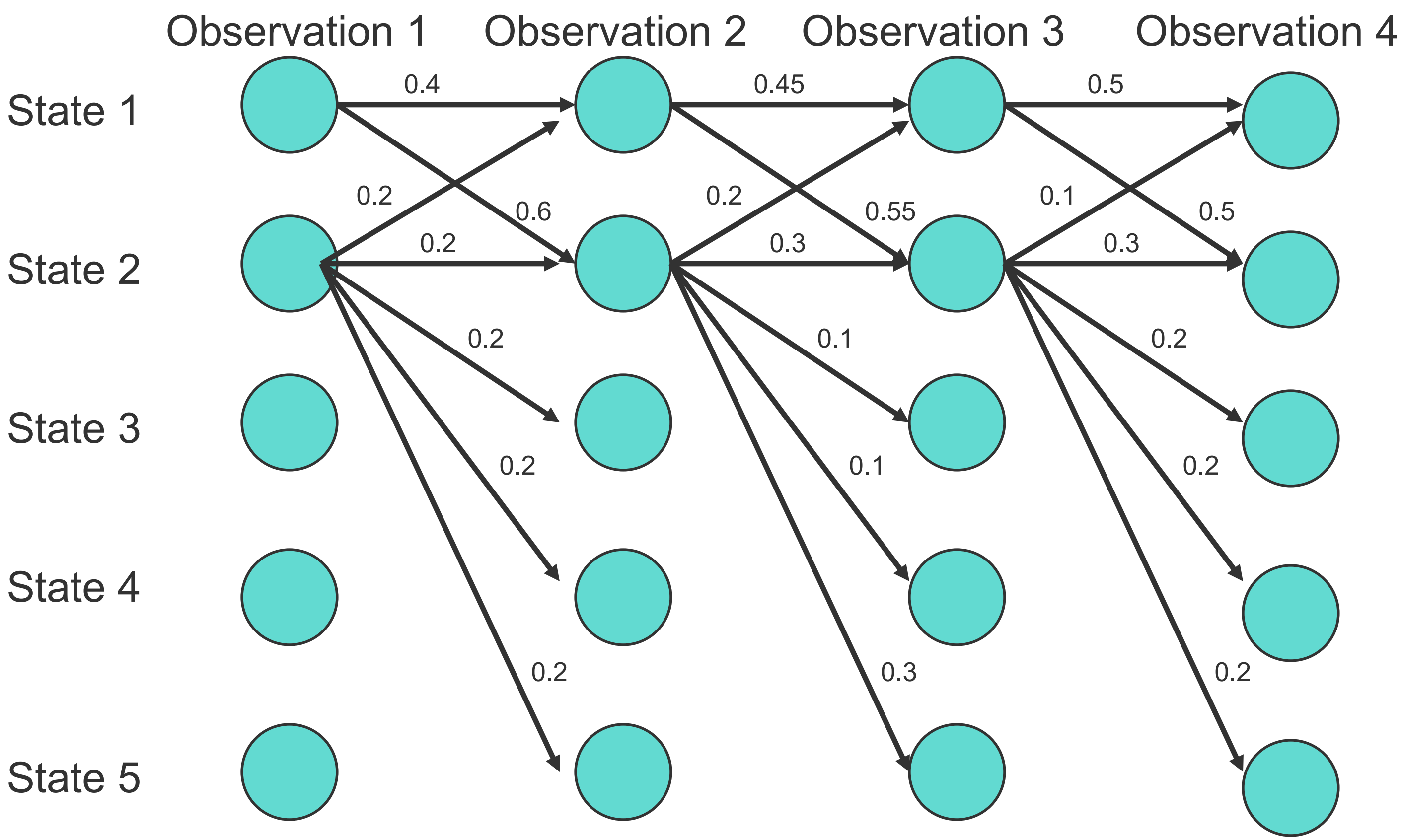
- Models dependence between each state and the full observation sequence explicitly
  - More expressive than HMMs
- Discriminative model
  - Completely ignores modeling P(X): saves modeling effort
  - Learning objective function consistent with predictive function: P(Y|X)

Shortcomings of Hidden Markov Model (1): locality of features



- HMM models capture dependences between each state and only its corresponding observation
  - NLP example: In a sentence segmentation task, each segmental state may depend not just on a single word (and the adjacent segmental stages), but also on the (non-local) features of the whole line such as line length, indentation, amount of white space, etc.
- Mismatch between learning objective function and prediction objective function
  - HMM learns a joint distribution of states and observations P(Y, X), but in a prediction task, we need the conditional probability P(Y|X)

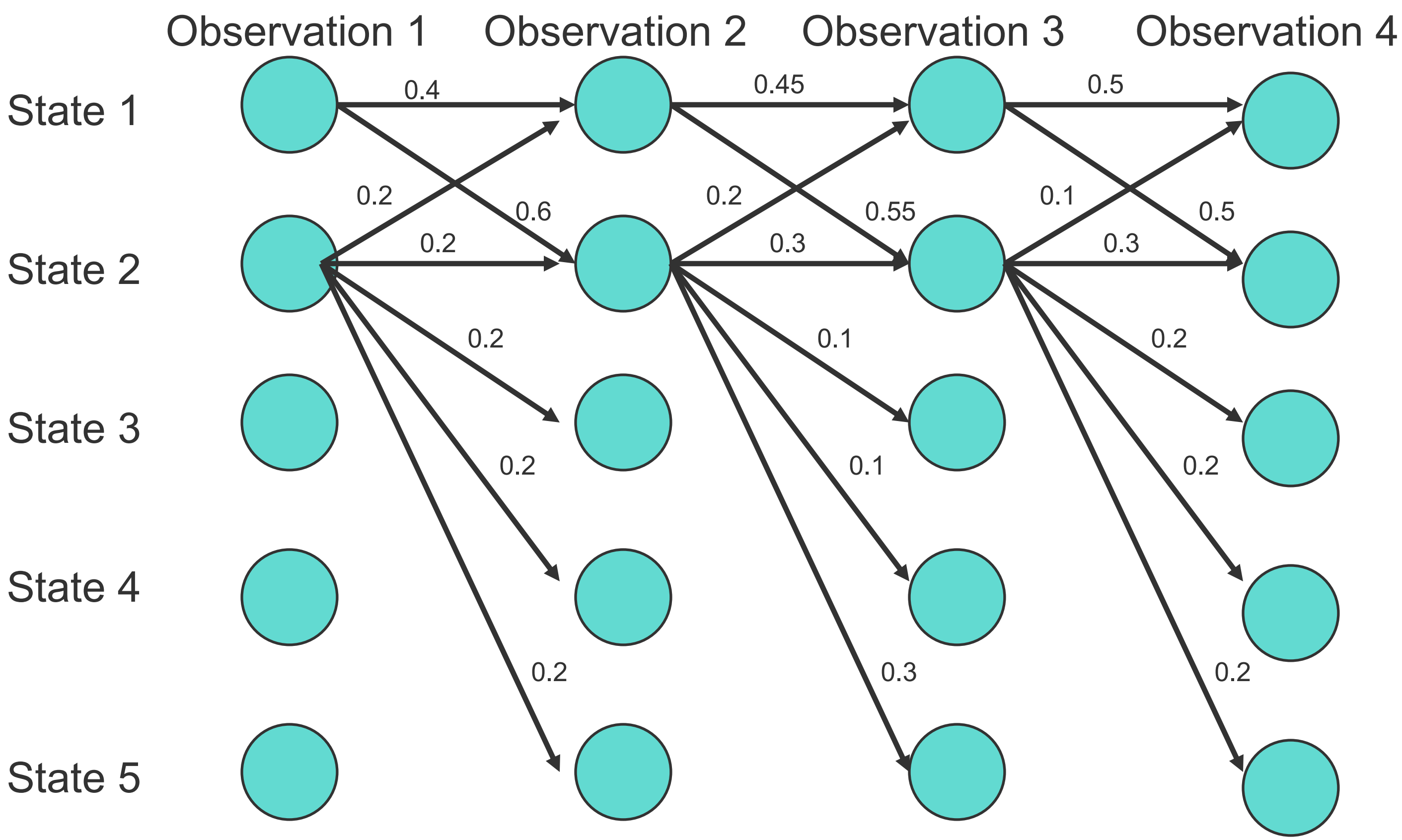
Then, shortcomings of HMM (2): the Label bias problem



What the local transition probabilities say:

- State 1 almost always prefers to go to state 2
- State 2 almost always prefer to stay in state 2

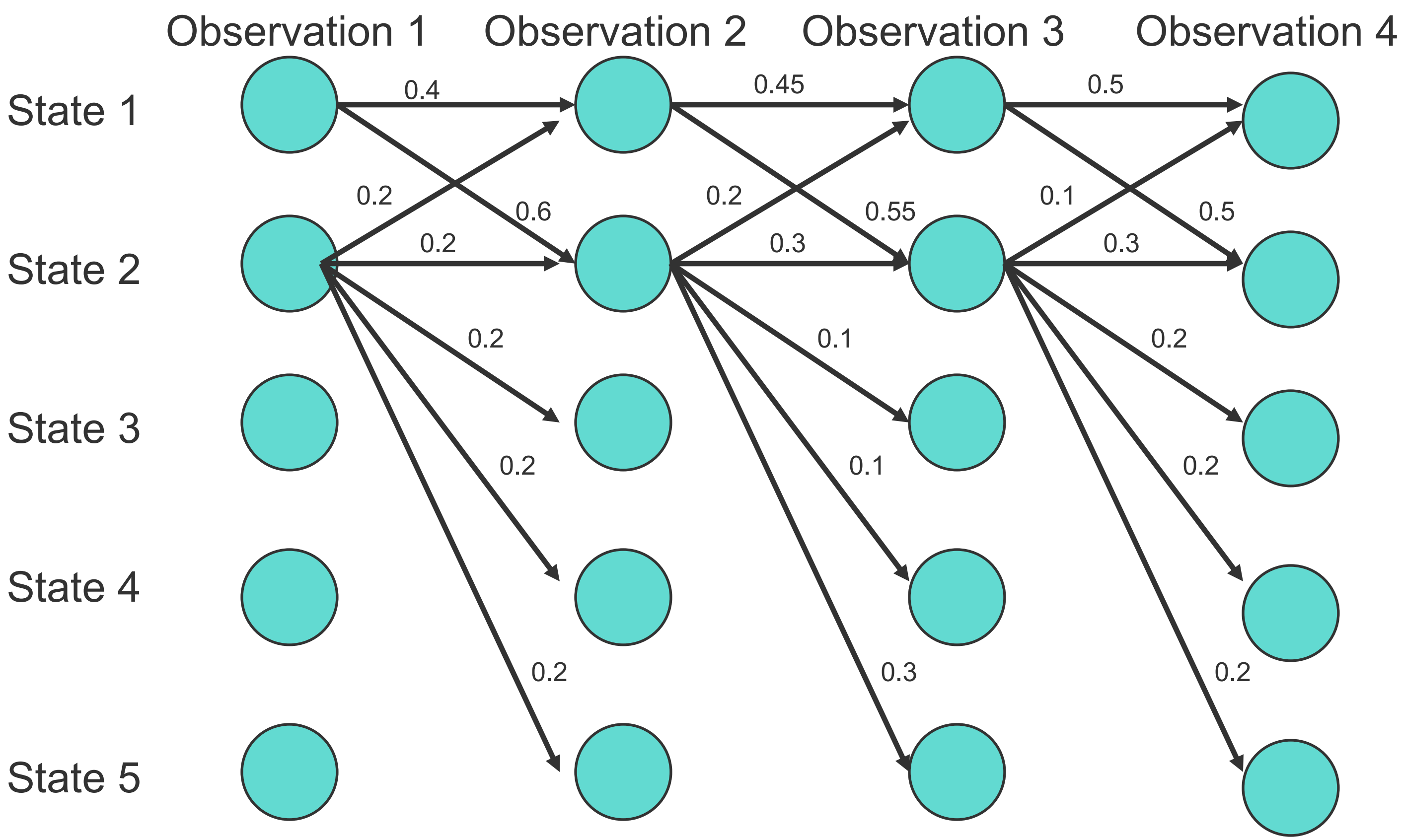
The Label bias problem



Probability of path 1-> 1-> 1-> 1:

- 0.4 x 0.45 x 0.5 = 0.09

The Label bias problem

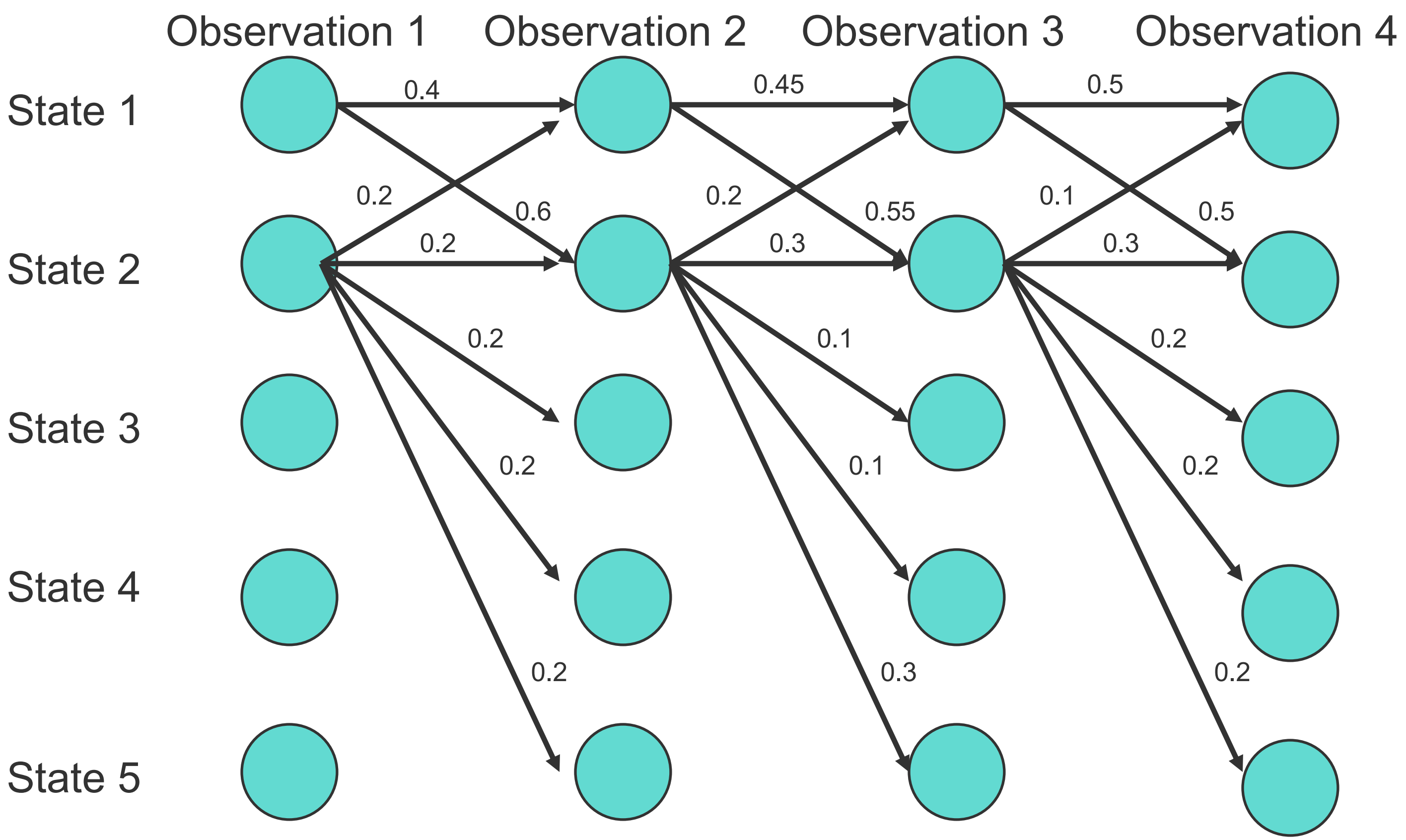


Probability of path 2->2->2->2 :

- 0.2 X 0.3 X 0.3 = 0.018

Other paths:
1-> 1-> 1-> 1: 0.09

The Label bias problem

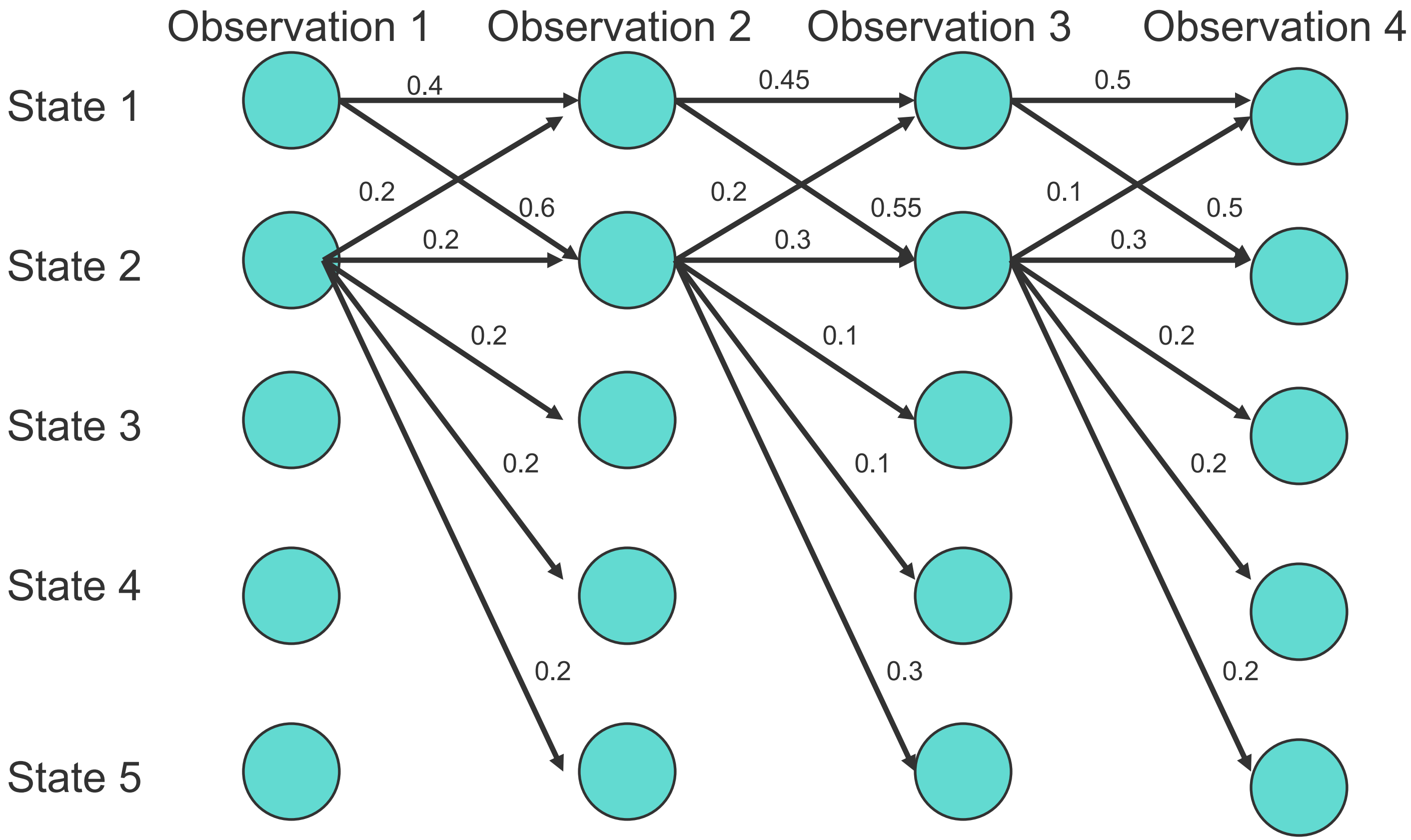


Probability of path 1->2->1->2:

- 0.6 X 0.2 X 0.5 = 0.06

Other paths:
1->1->1->1: 0.09
2->2->2->2: 0.018

The Label bias problem

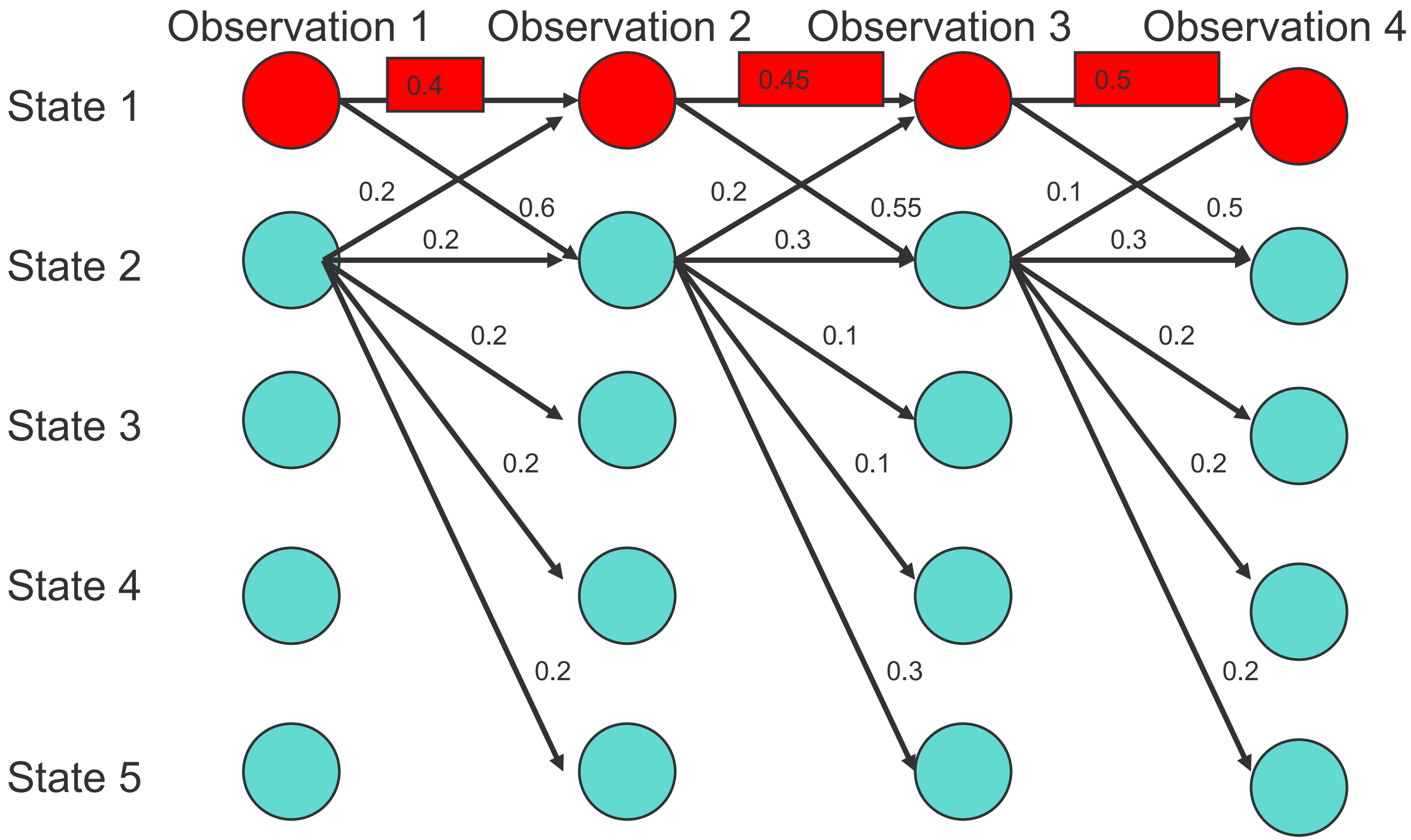


Probability of path 1->1->2->2:

- 0.4 X 0.55 X 0.3 = 0.066

Other paths:
1->1->1->1: 0.09
2->2->2->2: 0.018
1->2->1->2: 0.06

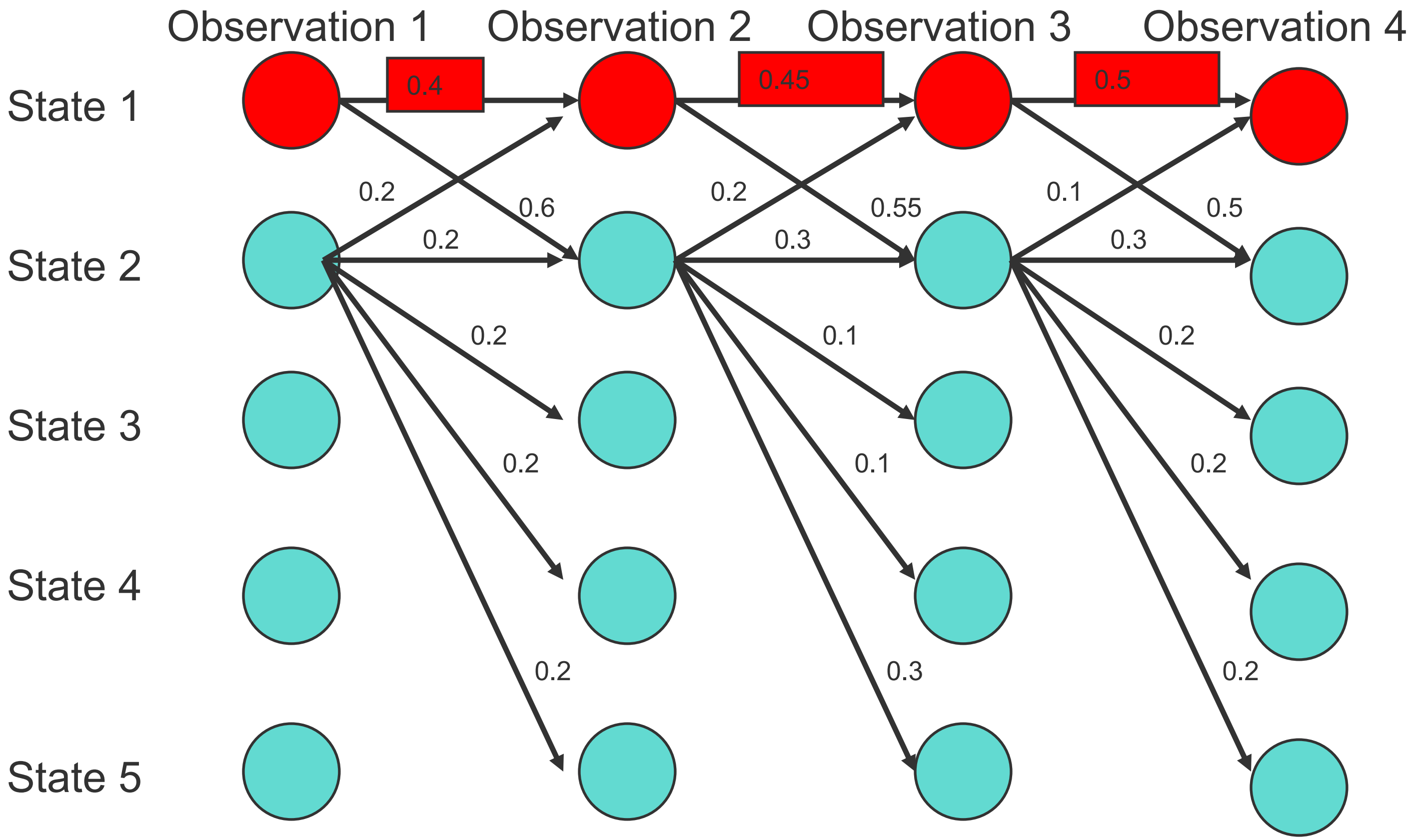
The Label bias problem



Most Likely Path: 1-> 1-> 1-> 1

- Although **locally** it seems state 1 wants to go to state 2 and state 2 wants to remain in state 2.
- **why?**

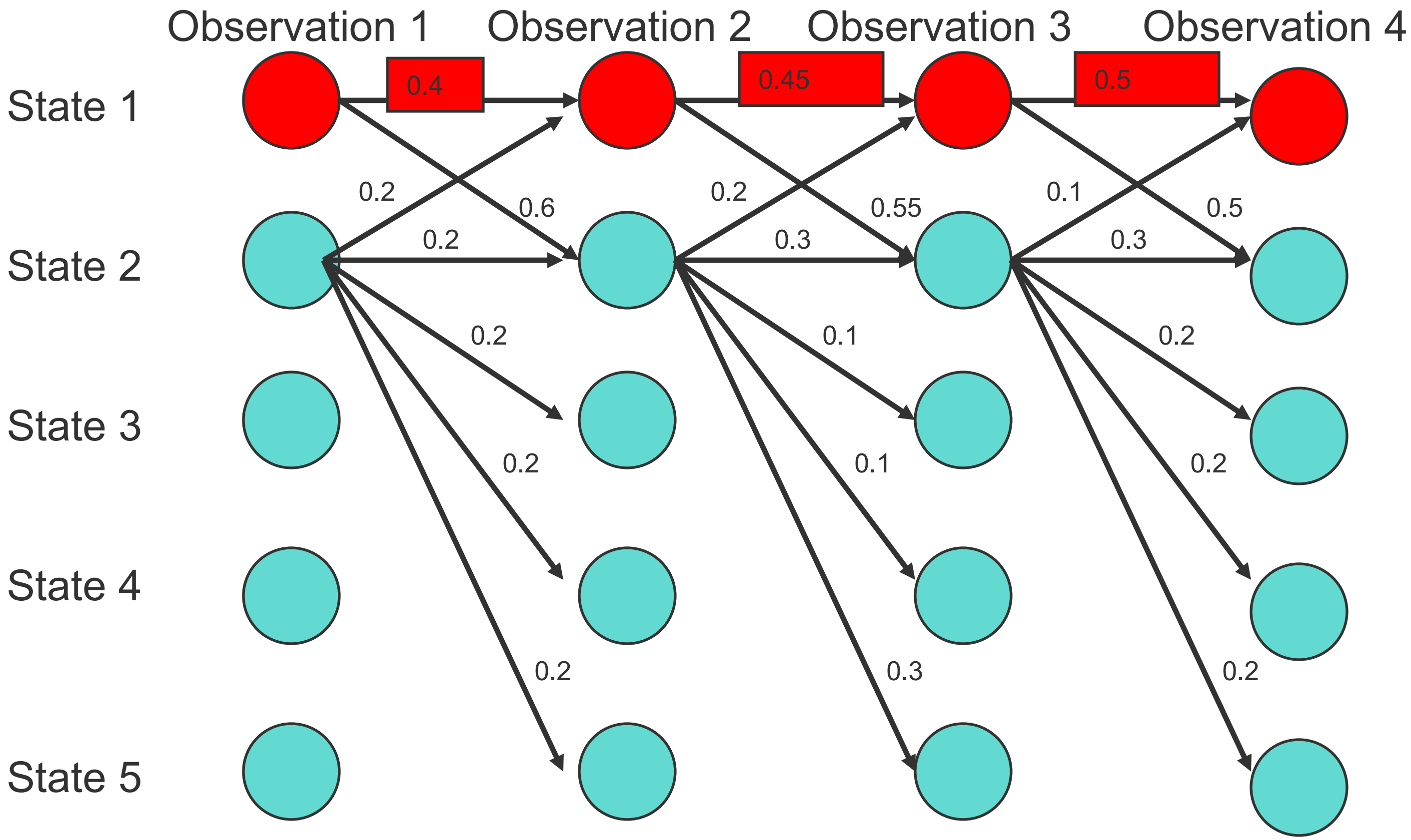
The Label bias problem



Most Likely Path: 1-> 1-> 1-> 1

- State 1 has only two transitions but state 2 has 5:
  - Average transition probability from state 2 is lower

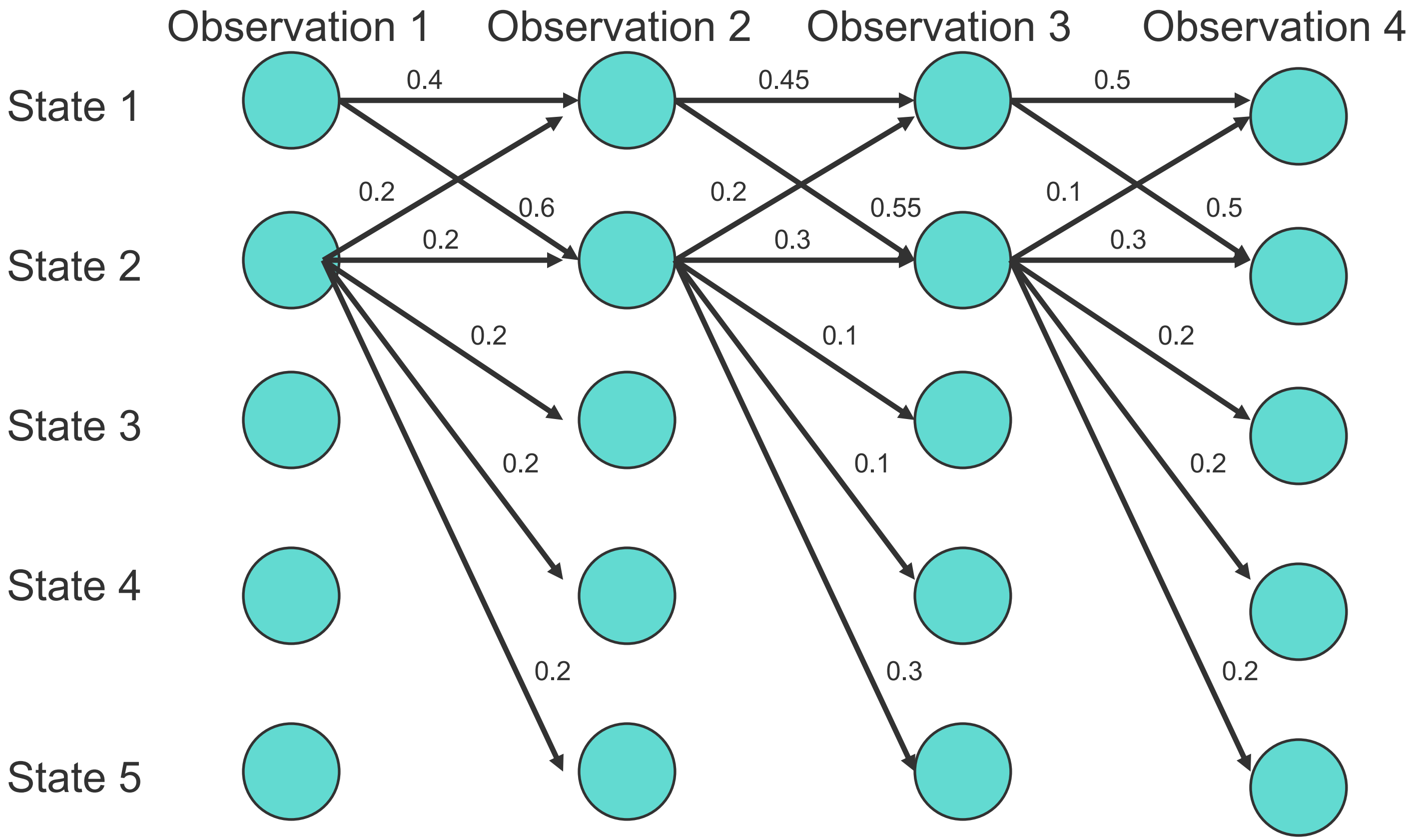
The Label bias problem



Label bias problem in MEMM:

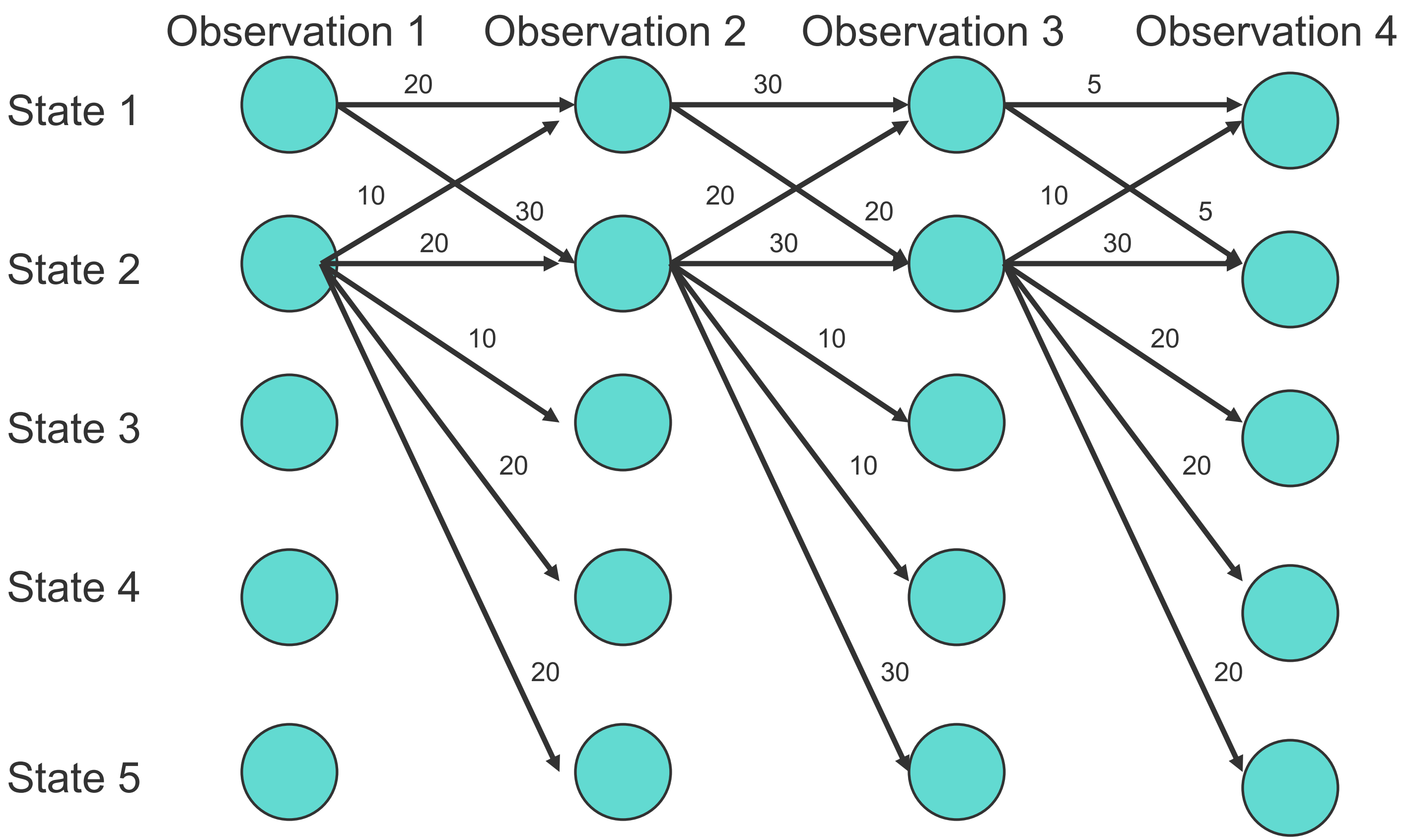
- Preference of states with lower number of transitions over others

Solution:
Do not normalize probabilities locally



From local probabilities ….

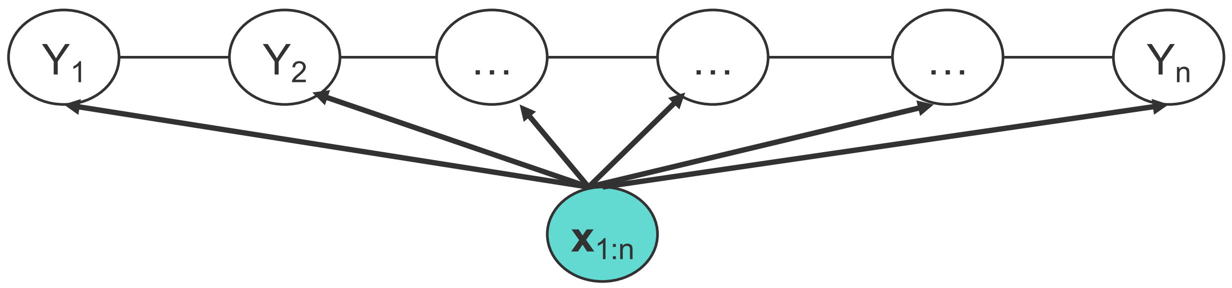
Solution:
Do not normalize probabilities locally



From local probabilities to local potentials

- States with lower transitions do not have an unfair advantage!

From HMM to CRF

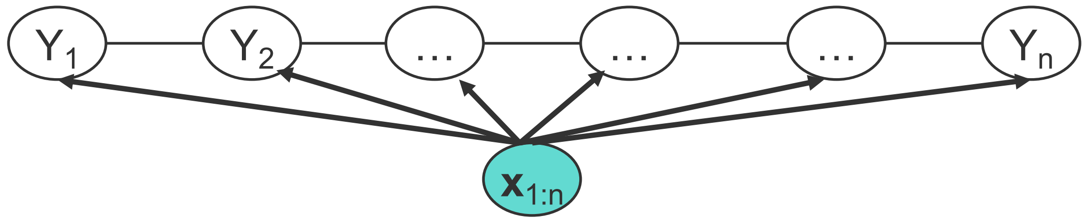


$$P(\mathbf{y}_{1:n}|\mathbf{x}_{1:n}) = \frac{1}{Z(\mathbf{x}_{1:n})}\prod_{i=1}^{n}\phi(y_i, y_{i-1}, \mathbf{x}_{1:n}) = \frac{1}{Z(\mathbf{x}_{1:n}, \mathbf{w})}\prod_{i=1}^{n}\exp(\mathbf{w}^T\mathbf{f}(y_i, y_{i-1}, \mathbf{x}_{1:n}))$$

- CRF is a partially directed model
  - Discriminative model like MEMM
  - Usage of global normalizer Z($\mathbf{x}$) overcomes the label bias problem of MEMM
  - Models the dependence between each state and the entire observation sequence (like MEMM)

Conditional Random Fields

- General parametric form:



$$
\begin{aligned}
P(\mathbf{y}|\mathbf{x}) &= \frac{1}{Z(\mathbf{x},\lambda,\mu)} \exp\Big(\sum_{i=1}^{n}\Big(\sum_{k}\lambda_k f_k(y_i, y_{i-1}, \mathbf{x}) + \sum_{l}\mu_l g_l(y_i, \mathbf{x})\Big)\Big) \\
&= \frac{1}{Z(\mathbf{x},\lambda,\mu)} \exp\Big(\sum_{i=1}^{n}\big(\lambda^T \mathbf{f}(y_i, y_{i-1}, \mathbf{x}) + \mu^T \mathbf{g}(y_i, \mathbf{x})\big)\Big)
\end{aligned}
$$

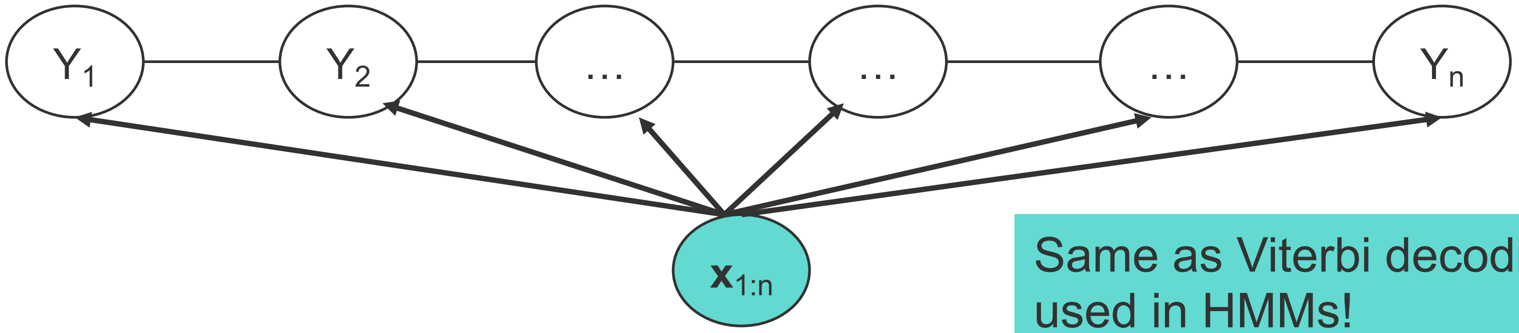
$$
\text{where } Z(\mathbf{x},\lambda,\mu) = \sum_{\mathbf{y}} \exp\Big(\sum_{i=1}^{n}\big(\lambda^T \mathbf{f}(y_i, y_{i-1}, \mathbf{x}) + \mu^T \mathbf{g}(y_i, \mathbf{x})\big)\Big)
$$

CRFs: Inference

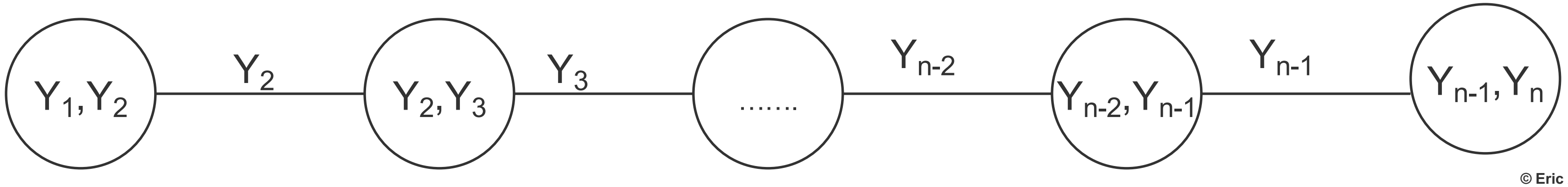
- Given CRF parameters λ and μ, find the $\mathbf{y}^*$ that maximizes P(**y**|**x**)

$$\mathbf{y}^* = \arg\max_{\mathbf{y}} \exp\left(\sum_{i=1}^{n}(\lambda^T \mathbf{f}(y_i, y_{i-1}, \mathbf{x}) + \mu^T \mathbf{g}(y_i, \mathbf{x}))\right)$$

  - Can ignore Z(**x**) because it is not a function of **y**
- Run the max-product algorithm on the junction-tree of CRF:



Same as Viterbi decoding used in HMMs!



CRF learning

- Given $\{(\mathbf{x}_d, \mathbf{y}_d)\}_{d=1}^N$, find λ^*, μ^* such that

$$
\begin{aligned}
\lambda*, \mu* &= \arg\max_{\lambda,\mu} L(\lambda,\mu) = \arg\max_{\lambda,\mu} \prod_{d=1}^{N} P(\mathbf{y}_d|\mathbf{x}_d,\lambda,\mu) \\
&= \arg\max_{\lambda,\mu} \prod_{d=1}^{N} \frac{1}{Z(\mathbf{x}_d,\lambda,\mu)} \exp\Big(\sum_{i=1}^{n} \big(\lambda^T \mathbf{f}(y_{d,i}, y_{d,i-1}, \mathbf{x}_d) + \mu^T \mathbf{g}(y_{d,i}, \mathbf{x}_d)\big)\Big) \\
&= \arg\max_{\lambda,\mu} \sum_{d=1}^{N} \Big(\sum_{i=1}^{n} \big(\lambda^T \mathbf{f}(y_{d,i}, y_{d,i-1}, \mathbf{x}_d) + \mu^T \mathbf{g}(y_{d,i}, \mathbf{x}_d)\big) - \log Z(\mathbf{x}_d,\lambda,\mu)\Big)
\end{aligned}
$$

- Computing the gradient w.r.t λ:

Gradient of the log-partition function in an exponential family is the expectation of the sufficient statistics.

$$
\nabla_\lambda L(\lambda,\mu) = \sum_{d=1}^{N} \Big(\sum_{i=1}^{n} \mathbf{f}(y_{d,i}, y_{d,i-1}, \mathbf{x}_d) - \sum_{\mathbf{y}} \big(P(\mathbf{y}|\mathbf{x_d}) \sum_{i=1}^{n} \mathbf{f}(y_{d,i}, y_{d,i-1}, \mathbf{x}_d)\big)\Big)
$$

CRF learning

$$\nabla_\lambda L(\lambda, \mu) = \sum_{d=1}^{N} \left(\sum_{i=1}^{n} \mathbf{f}(y_{d,i}, y_{d,i-1}, \mathbf{x}_d) - \boxed{\sum_{\mathbf{y}} \left(P(\mathbf{y}|\mathbf{x_d}) \sum_{i=1}^{n} \mathbf{f}(y_i, y_{i-1}, \mathbf{x}_d) \right) \Bigg)} \right.$$

- Computing the model expectations:
  - Requires exponentially large number of summations: Is it intractable?

$$\sum_{\mathbf{y}} \left(P(\mathbf{y}|\mathbf{x}_d) \sum_{i=1}^{n} \mathbf{f}(y_i, y_{i-1}, \mathbf{x}_d) \right) = \sum_{i=1}^{n} \left(\sum_{\mathbf{y}} \mathbf{f}(y_i, y_{i-1}, \mathbf{x}_d) P(\mathbf{y}|\mathbf{x}_d) \right)$$

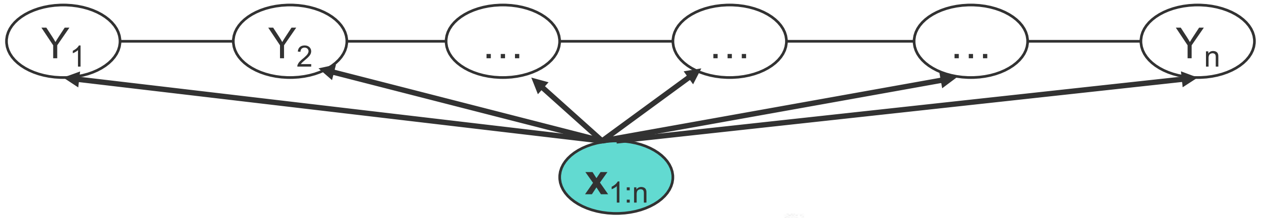
$$= \sum_{i=1}^{n} \sum_{y_i, y_{i-1}} \mathbf{f}(y_i, y_{i-1}, \mathbf{x}_d) P(y_i, y_{i-1}|\mathbf{x}_d)$$

Expectation of **f** over the corresponding marginal probability of neighboring nodes!!

- Tractable!
  - Can compute marginals using the sum-product algorithm on the chain

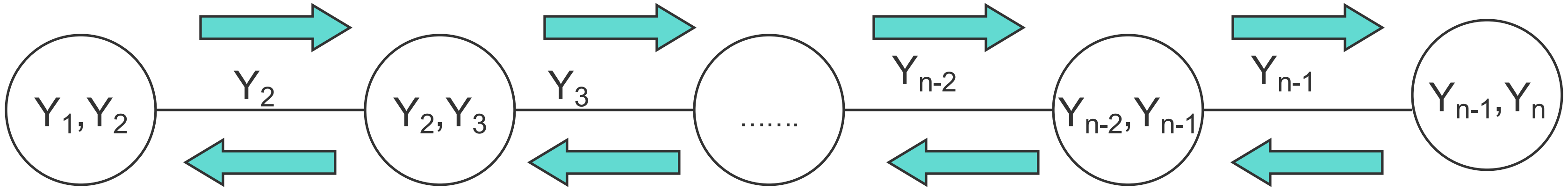
CRF learning

- Computing marginals using junction-tree calibration:



- Junction Tree Initialization:

$$\alpha^0(y_i, y_{i-1}) = \exp(\lambda^T \mathbf{f}(y_i, y_{i-1}, \mathbf{x}_d) + \mu^T \mathbf{g}(y_i, \mathbf{x}_d))$$



- After calibration:

Also called forward-backward algorithm

$$P(y_i, y_{i-1}|\mathbf{x}_d) \propto \alpha(y_i, y_{i-1})$$

$$\Rightarrow P(y_i, y_{i-1}|\mathbf{x}_d) = \frac{\alpha(y_i, y_{i-1})}{\sum_{y_i, y_{i-1}} \alpha(y_i, y_{i-1})} = \alpha'(y_i, y_{i-1})$$

CRF learning

- Computing feature expectations using calibrated potentials:

$$\sum_{y_i,y_{i-1}} \mathbf{f}(y_i, y_{i-1}, \mathbf{x}_d) P(y_i, y_{i-1}|\mathbf{x}_d) = \sum_{y_i,y_{i-1}} \mathbf{f}(y_i, y_{i-1}, \mathbf{x}_d) \alpha'(y_i, y_{i-1})$$

- Now we know how to compute $\nabla_\lambda L(\lambda,\mu)$:

$$\begin{aligned}
\nabla_\lambda L(\lambda, \mu) &= \sum_{d=1}^{N} \left(\sum_{i=1}^{n} \mathbf{f}(y_{d,i}, y_{d,i-1}, \mathbf{x}_d) - \sum_{\mathbf{y}} \left(P(\mathbf{y}|\mathbf{x_d}) \sum_{i=1}^{n} \mathbf{f}(y_i, y_{i-1}, \mathbf{x}_d) \right) \right) \\
&= \sum_{d=1}^{N} \left(\sum_{i=1}^{n} \left(\mathbf{f}(y_{d,i}, y_{d,i-1}, \mathbf{x}_d) - \sum_{y_i,y_{i-1}} \alpha'(y_i, y_{i-1}) \mathbf{f}(y_i, y_{i-1}, \mathbf{x}_d) \right) \right)
\end{aligned}$$

- Learning can now be done using gradient ascent:

$$\begin{aligned}
\lambda^{(t+1)} &= \lambda^{(t)} + \eta \nabla_\lambda L(\lambda^{(t)}, \mu^{(t)}) \\
\mu^{(t+1)} &= \mu^{(t)} + \eta \nabla_\mu L(\lambda^{(t)}, \mu^{(t)})
\end{aligned}$$

CRF learning

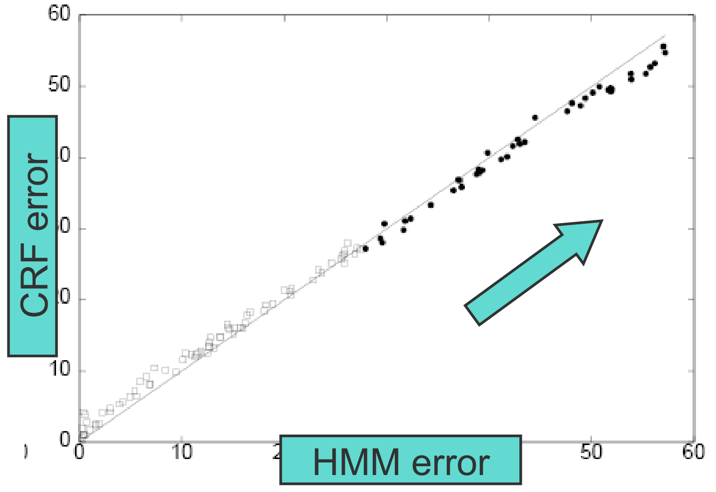
- In practice, we use a Gaussian Regularizer for the parameter vector to improve generalizability

$$\lambda*, \mu* \quad = \quad \arg\max_{\lambda,\mu} \sum_{d=1}^{N} \log P(\mathbf{y}_d | \mathbf{x}_d, \lambda, \mu) - \frac{1}{2\sigma^2}(\lambda^T \lambda + \mu^T \mu)$$

- In practice, gradient ascent has very slow convergence
  - Alternatives:
    - Conjugate Gradient method
    - Limited Memory Quasi-Newton Methods

CRFs: some empirical results

- Comparison of error rates on synthetic data



Data is increasingly higher order in the direction of arrow

CRFs achieve the lowest error rate for higher order data

CRFs: some empirical results

- Parts of Speech tagging

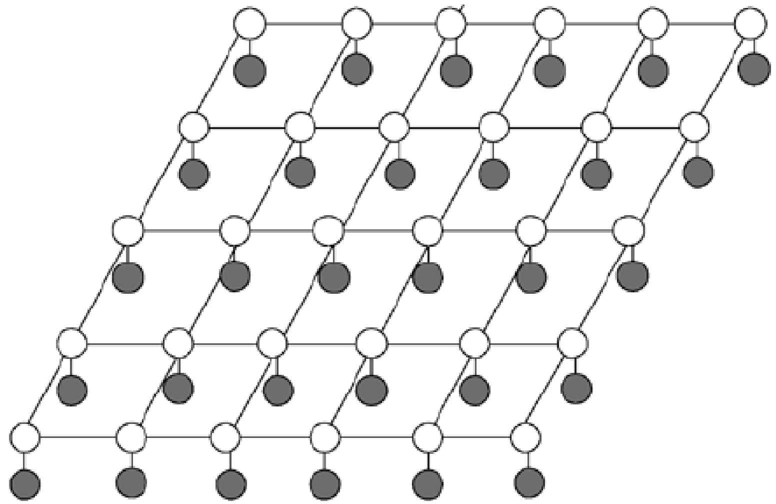
model	*error*	*oov error*
HMM	5.69%	45.99%
MEMM	6.37%	54.61%
CRF	5.55%	48.05%
MEMM^{+}	4.81%	26.99%
CRF^{+}	4.27%	23.76%

$^{+}$Using spelling features

  - Using same set of features: HMM >=< CRF > MEMM
  - Using additional overlapping features: CRF^{+} > MEMM^{+} >> HMM

Other CRFs

- So far we have discussed only 1-dimensional chain CRFs
  - Inference and learning: exact
- We could also have CRFs for arbitrary
  - E.g: Grid CRFs
  - Inference and learning no longer tractable
  - Approximate techniques used
    - MCMC Sampling
    - Variational Inference
    - Loopy Belief Propagation
  - We will discuss these techniques soon



Applications of CRF in Vision: Image Segmentation

Stereo Matching

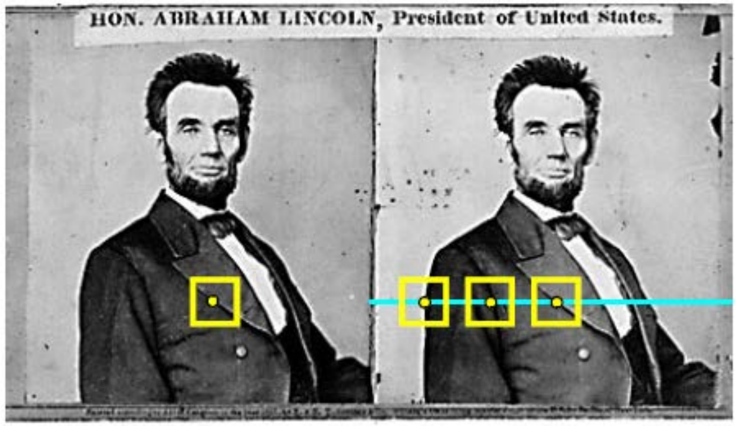


Image Restoration

Application: Image Segmentation

$\phi_i(y_i, x) \in \mathbb{R}^{\approx 1000}$: local image features, e.g. bag-of-words

$\rightarrow \quad \langle w_i, \phi_i(y_i, x)\rangle$: local classifier (like logistic-regression)

$\phi_{i,j}(y_i, y_j) = [\![y_i = y_j]\!] \in \mathbb{R}^1$: test for same label

$\rightarrow \quad \langle w_{ij}, \phi_{ij}(y_i, y_j)\rangle$: penalizer for label changes (if $w_{ij} > 0$)

combined: $\operatorname{argmax}_y p(y|x)$ is smoothed version of local cues

original

local classification

local + smoothness

Application: Handwriting Recognition

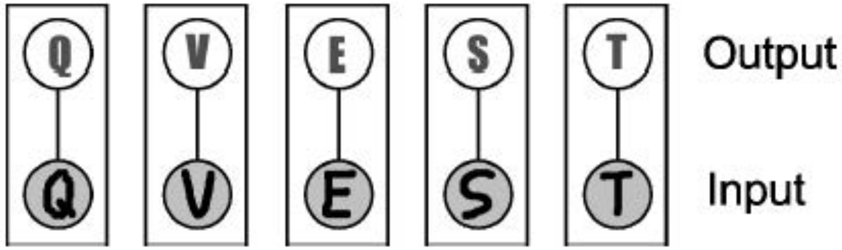
$\phi_i(y_i, x) \in \mathbb{R}^{\approx 1000}$: image representation (pixels, gradients)

$\rightarrow \quad \langle w_i, \phi_i(y_i, x) \rangle$: local classifier if x_i is letter y_i

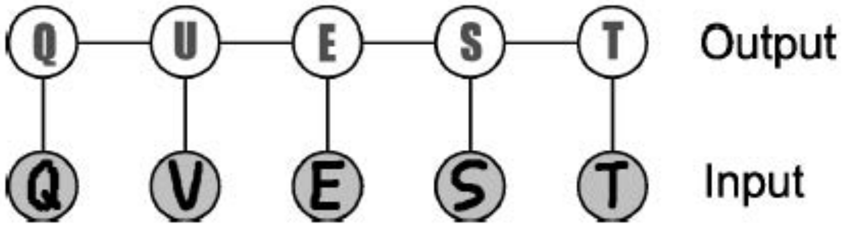
$\phi_{i,j}(y_i, y_j) = e_{y_i} \otimes e_{y_j} \in \mathbb{R}^{26 \cdot 26}$: letter/letter indicator

$\rightarrow \quad \langle w_{ij}, \phi_{ij}(y_i, y_j) \rangle$: encourage/suppress letter combinations

combined: $\mathrm{argmax}_y\, p(y|x)$ is "corrected" version of local cues



local classification



local + "correction"

Application: Pose Estimation

$\phi_i(y_i, x) \in \mathbb{R}^{\approx 1000}$: local image representation, e.g. HoG

$\rightarrow \quad \langle w_i, \phi_i(y_i, x) \rangle$: local confidence map

$\phi_{i,j}(y_i, y_j) = \mathit{good\_fit}(y_i, y_j) \in \mathbb{R}^1$: test for geometric fit

$\rightarrow \quad \langle w_{ij}, \phi_{ij}(y_i, y_j) \rangle$: penalizer for unrealistic poses

together: $\operatorname{argmax}_y p(y|x)$ is sanitized version of local cues

original   local classification   local + geometry

Feature Functions for CRF in Vision

$\phi_i(y_i, x)$: local representation, high-dimensional

$\rightarrow \quad \langle w_i, \phi_i(y_i, x) \rangle$: local classifier

$\phi_{i,j}(y_i, y_j)$: prior knowledge, low-dimensional

$\rightarrow \quad \langle w_{ij}, \phi_{ij}(y_i, y_j) \rangle$: penalize outliers

learning adjusts parameters:

- unary w_i: learn local classifiers and their importance
- binary w_{ij}: learn importance of smoothing/penalization

$\mathrm{argmax}_y\, p(y|x)$ is cleaned up version of local prediction

Case Study: Image Segmentation

- Image segmentation (FG/BG) by modeling of interactions btw RVs
  - Images are noisy.
  - Objects occupy continuous regions in an image.

[Nowozin,Lampert 2012]

Input image

Pixel-wise separate optimal labeling

Locally-consistent joint optimal labeling

$$Y^* = \arg\max_{y\in\{0,1\}^n} \Bigg[\underbrace{\sum_{i\in S} V_i(y_i, X)}_{\text{Unary Term}} + \underbrace{\sum_{i\in S}\sum_{j\in N_i} V_{i,j}(y_i, y_j)}_{\text{Pairwise Term}} \Bigg].$$

Y: labels
X: data (features)
S: pixels
N_i: neighbors of pixel i

Discriminative Random Fields

- A special type of CRF
  - The unary and pairwise potentials are designed using local discriminative classifiers.
  - Posterior

$$P(Y \mid X) = \frac{1}{Z}\exp\Big(\sum_{i\in S} \overbrace{A_i(y_i, X)}^{\text{Association}} + \sum_{i\in S}\sum_{j\in N_i} \overbrace{I_{ij}(y_i, y_j, X)}^{\text{Interaction}}\Big)$$

- Association Potential
  - Local discriminative model for site i: using logistic link with GLM.

$$A_i(y_i, X) = \log P(y_i \mid f_i(X)) \qquad P(y_i = 1 \mid f_i(X)) = \frac{1}{1+\exp(-(w^T f_i(X)))} = \sigma(w^T f_i(X))$$

- Interaction Potential
  - Measure of how likely site i and j have the same label given X

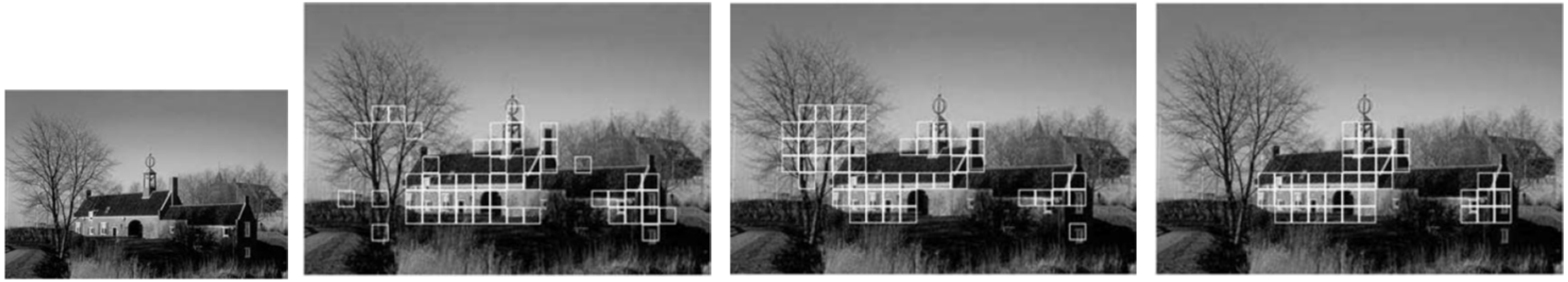
$$I_{ij}(y_i, y_j, X) = \underbrace{k y_i y_j}_{\text{(1) Data-independent smoothing term}} + \underbrace{(1-k)(2\sigma(y_i y_j \mu_{ij}(X)) - 1))}_{\text{(2) Data-dependent pairwise logistic function}}$$

S. Kumar and M. Hebert. Discriminative Random Fields. IJCV, 2006.

DRF Results

- Task: Detecting man-made structure in natural scenes.
  - Each image is divided in non-overlapping 16x16 tile blocks.

- An example



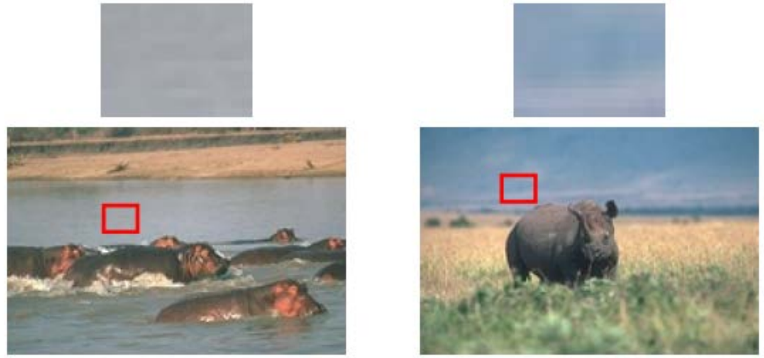
Input image Logistic MRF DRF

- Logistic: No smoothness in the labels
- MRF: Smoothed False positive. Lack of neighborhood interaction of the data

S. Kumar and M. Hebert. Discriminative Random Fields. IJCV, 2006.

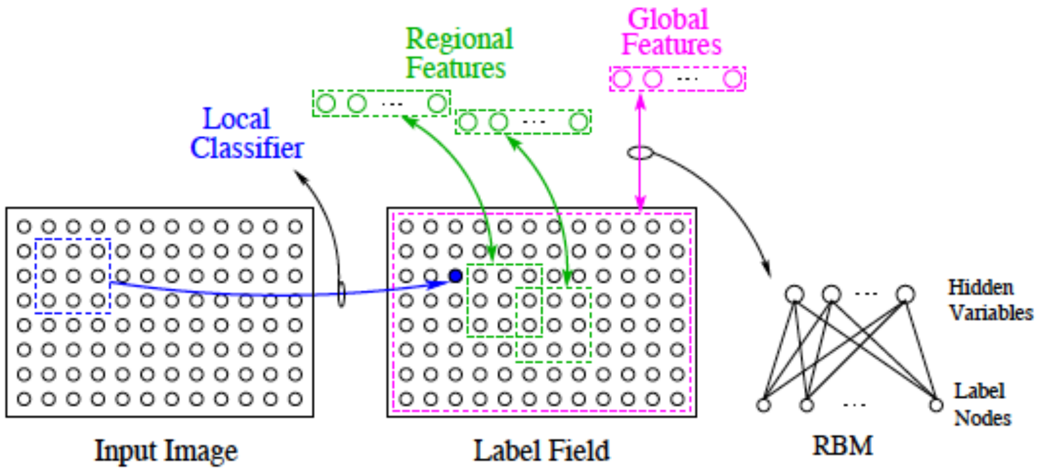
Multiscale Conditional Random Fields

- Considering features in different scales
  - Local Features (site)
  - Regional Label Features (small patch)
  - Global Label Features (big patch or the wh
- The conditional probability *P(L|X)* is formulated by



$$P(L \mid X) = \frac{1}{Z} \prod_{s} P_s(L \mid X)$$

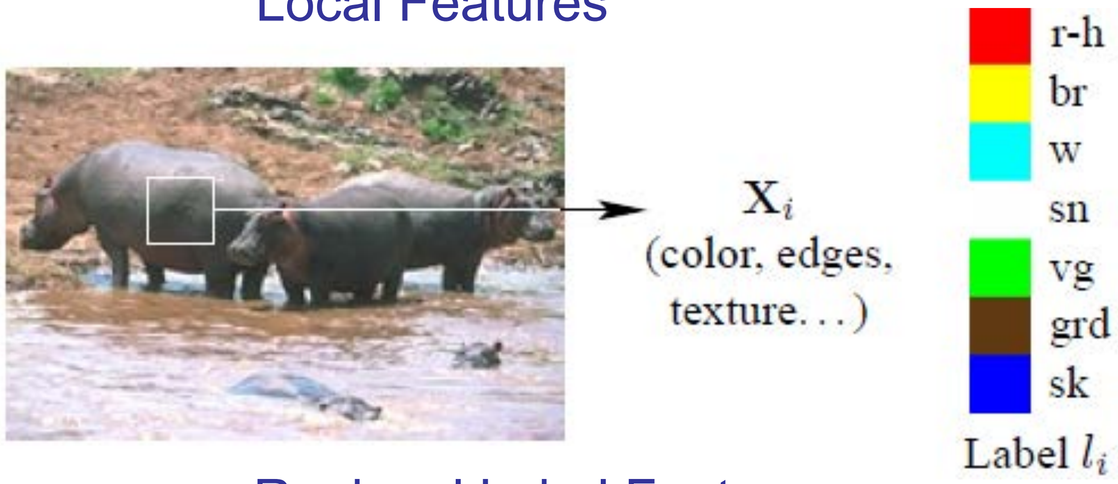
$$Z = \sum_{L} \prod_{s} P_s(L \mid X)$$



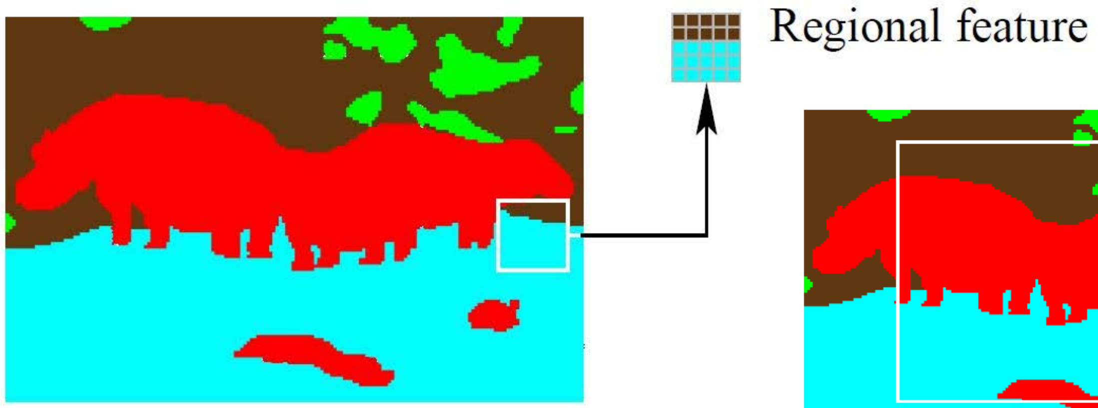
He, X. et. al.: Multiscale conditional random fields for image labeling. CVPR 2004

Multiscale Conditional Random Fields

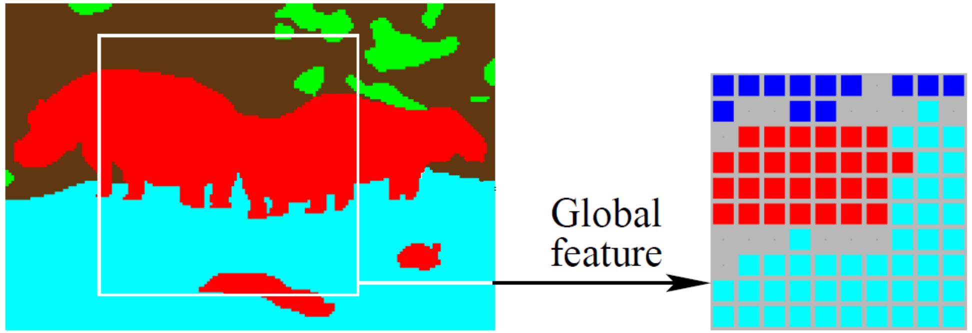
Local Features



Regional Label Features

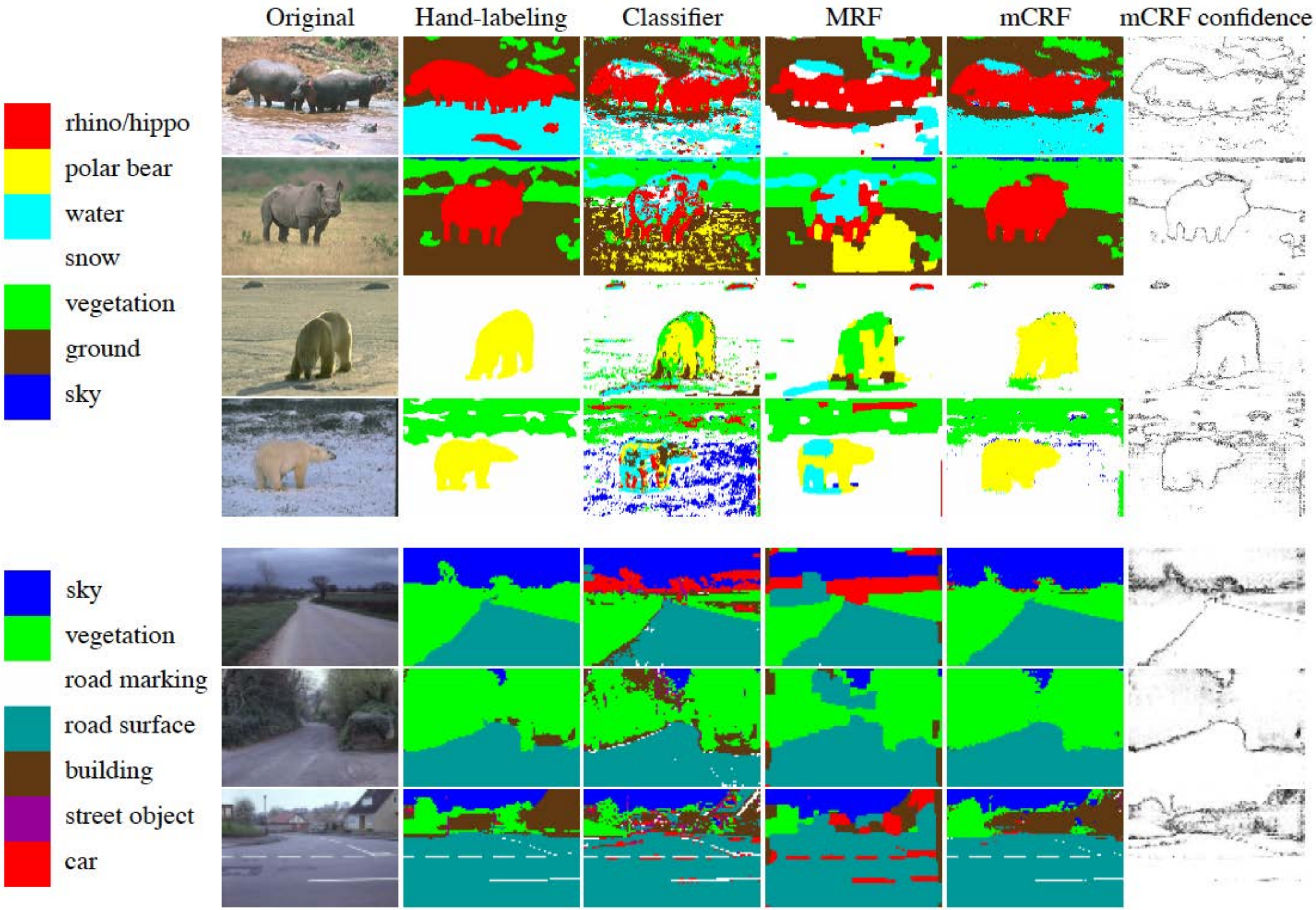


Global Label Features



He, X. et. al.: Multiscale conditional random fields for image labeling. CVPR 2004

mC

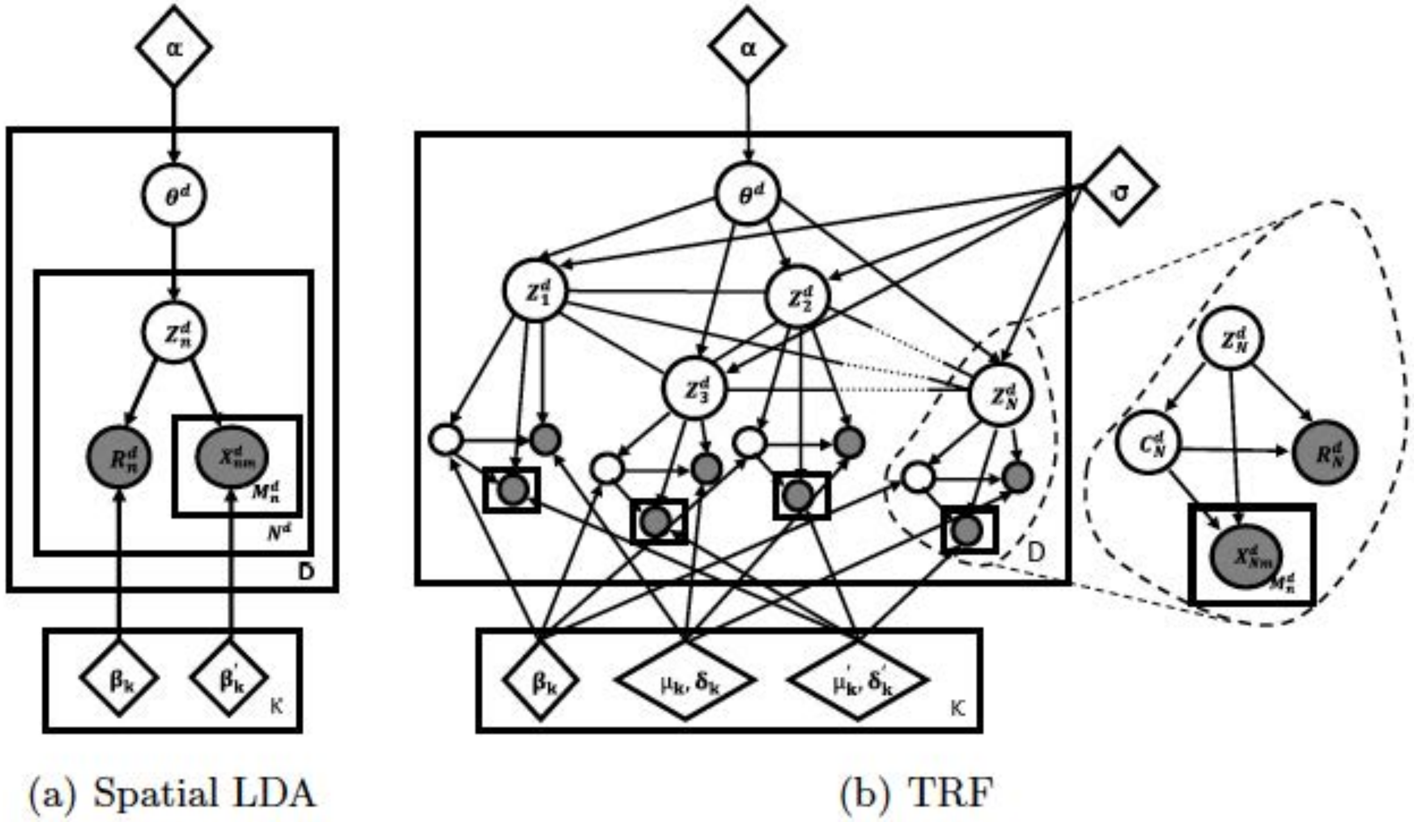


He, X. et. al.: Multiscale conditional random fields for image labeling. CVPR 2004

Topic Random Fields

- Spatial MRF over topic assignments

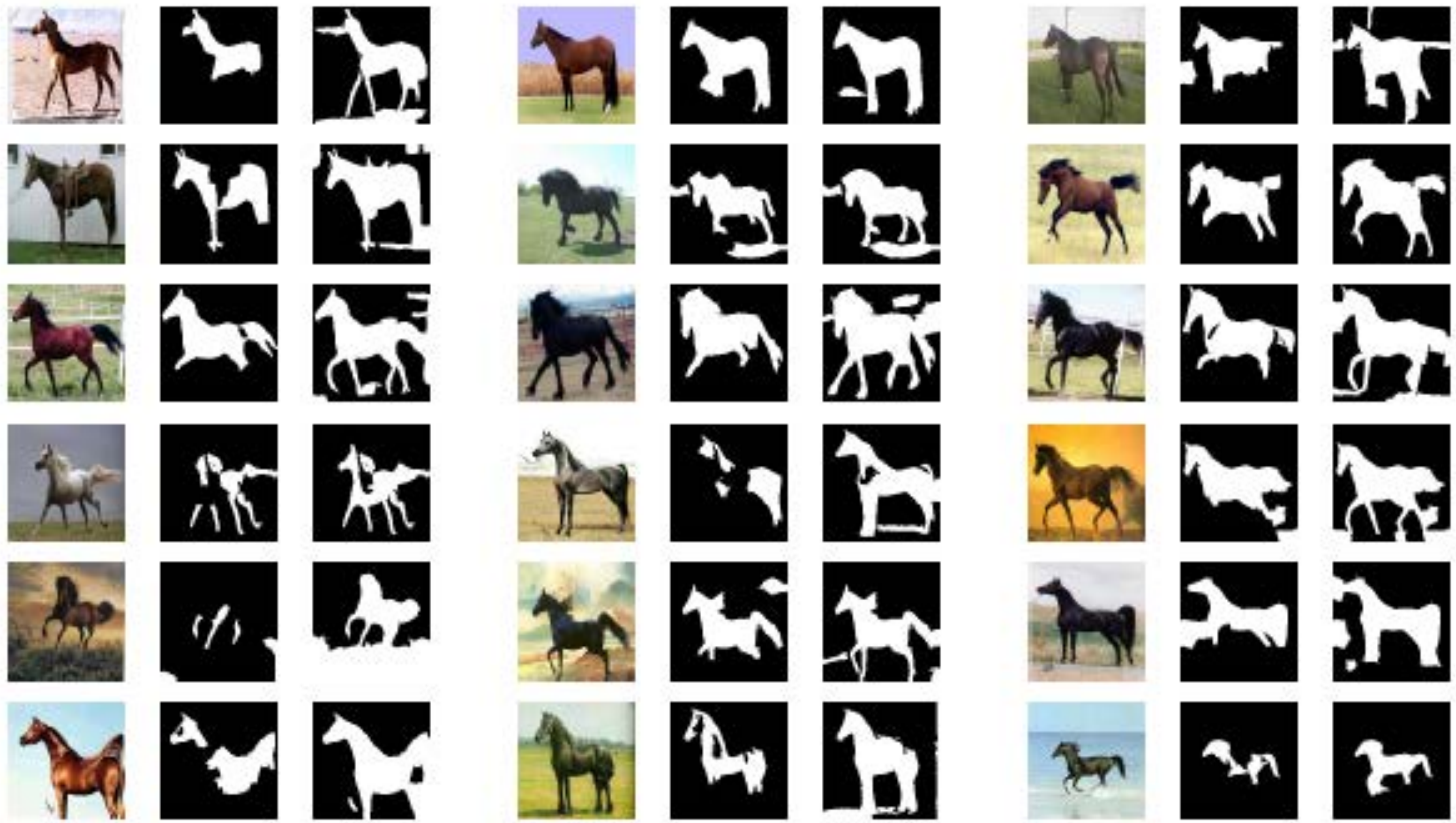
$$p(\mathbf{z}^d|\boldsymbol{\theta}^d,\sigma) = \frac{1}{A(\boldsymbol{\theta}^d,\sigma)} \exp\left[\sum_n \sum_k z^d_{nk} \log \theta^d_k + \sum_{n\sim m} \sigma I(z^d_n = z^d_m)\right]$$



Zhao, B. et. al.: Topic random fields for image segmentation. ECCV 2010

TRF Results

Spatial LDA vs. Topic Random Fields



Zhao, B. et. al.: Topic random fields for image segmentation. ECCV 2010

Summary

- Conditional Random Fields are partially directed discriminative models
- They overcome the label bias problem of MEMMs by using a global normalizer
- Inference for 1-D chain CRFs is exact
  - Same as Max-product or Viterbi decoding
- Learning also is exact
  - globally optimum parameters can be learned
  - Requires using sum-product or forward-backward algorithm
- CRFs involving arbitrary graph structure are intractable in general
  - E.g.: Grid CRFs
  - Inference and learning require approximation techniques
    - MCMC sampling
    - Variational methods
    - Loopy BP