Machine Learning

CSE 6363 (Fall 2016)

Lecture 20 Conditional Random Field

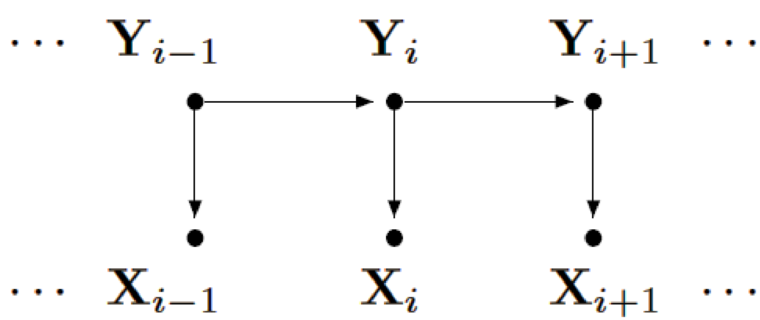
Heng Huang, Ph.D.

Department of Computer Science and Engineering

Generative Models

- Hidden Markov models (HMMs) and stochastic grammars
  - Assign a joint probability to paired observation and label sequences
  - The parameters typically trained to maximize the joint likelihood of train examples

Standard tool is the hidden Markov Model (HMM).



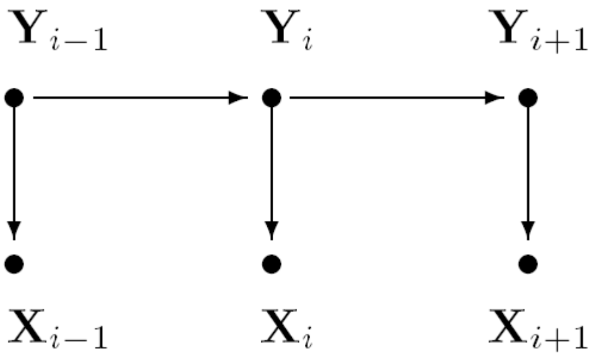
$$P(\mathbf{X}, \mathbf{Y}) \;=\; \prod_i P(\mathbf{X}_i \,|\, \mathbf{Y}_i)\, P(\mathbf{Y}_i \,|\, \mathbf{Y}_{i-1})$$

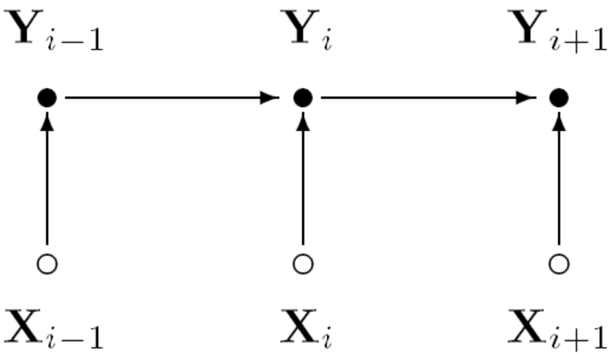
Conditional Models

- Conditional probability *P(label sequence* **y** | *observation sequence* **x***)* rather than joint probability *P*(**y, x**)
  - Specify the probability of possible label sequences given an observation sequence
- Allow arbitrary, non-independent features on the observation sequence X
- The probability of a transition between labels may depend on past and future observations
  - Relax strong independence assumptions in generative models

Implications of the model

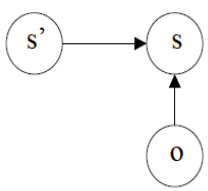
- Does this do what we want?
- Q: does Y[i-1] depend on X[i+1] ?
  - "a nodes is conditionally independent of its non-descendents given its parents"

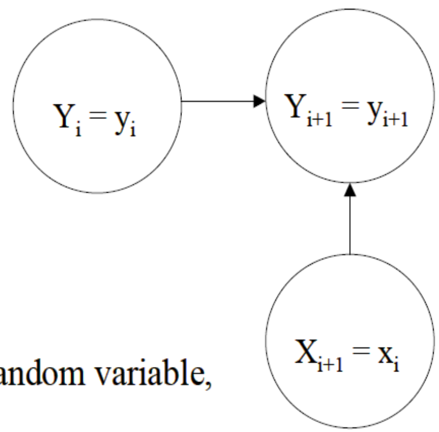




Discriminative Models
Maximum Entropy Markov Models (MEMMs)

- Exponential model
- Given training set X with label sequence Y:
  - Train a model θ that maximizes P(Y | X, θ)
  - For a new data sequence **x**, the predicted label **y** maximizes P(**y** | **x**, θ)
  - Notice the per-state normalization





$$\boxed{L(\Theta y) = \Sigma \log p_{\Theta y}(yi \mid xi)}$$

A = a, where A is a random variable, and a is an outcome

$$P(y' \mid y, x) = \frac{1}{Z(y, x)} \exp\left(\sum_k \underbrace{\lambda_k}_{\text{weight}} \underbrace{f_k(x, y, y')}_{\text{feature}} \right)$$

MEMM-Supervise Learning

In supervised learning we try to learn a function $h : \mathcal{X} \rightarrow \mathcal{Y}$ where $x \in \mathcal{X}$ are inputs and $y \in \mathcal{Y}$ are outputs.

- Binary classification: $\mathcal{Y} = \{-1, +1\}$
- Multiclass classification: $\mathcal{Y} = \{1, \ldots, K\}$ (finite set of labels)
- Regression: $\mathcal{Y} = \mathbb{R}$

The prediction is based on the *feature function* $\Phi : \mathcal{X} \rightarrow \mathcal{F}$ where usually $\mathcal{F} = \mathbb{R}^D$ (D-dimensional vector space)

MEMM-Linear Regression

- Training data: observations paired with outcomes ($n \in \mathbb{R}$)
- Observations have features (predictors, typically also real numbers)
- The model is a regression line $y = ax + b$ which best fits the observations
  - a is the slope
  - b is the intercept
  - This model has two parameters (or weigths)
  - One feature $= x$
  - Example:
    - ★ $x =$ number of vague adjectives in property descriptions
    - ★ $y =$ amount house sold over asking price

MEMM-Linear Regression

- More generally $y = w_0 + \sum_{i=0}^{N} w_i f_i$, where
  - y = outcome
  - w_0 = intercept
  - $f_1..f_N$ = features vector and $w_1..w_N$ weight vector
- We ignore w_0 by adding a special f_0 feature, then the equation is equivalent to dot product: $y = \mathbf{w} \cdot \mathbf{f}$

MEMM-Logistic Regression

- In logistic regression we use the linear model to do classification, i.e. assign probabilities to class labels
- For binary classification, predict $p(y = true|x)$. But predictions of linear regression model are $\in \mathbb{R}$, whereas $p(y = true|x) \in [0, 1]$
- Instead predict logit function of the probability:

$$ln\left(\frac{p(y = true|x)}{1 - p(y = true|x)}\right) = \mathbf{w} \cdot \mathbf{f} \qquad (1)$$

$$\frac{p(y = true|x)}{1 - p(y = true|x)} = e^{\mathbf{w} \cdot \mathbf{f}} \qquad (2)$$

- Solving for $p(y = true|x)$ we obtain:

$$p(y = true|x) = \frac{e^{\mathbf{w} \cdot \mathbf{f}}}{1 + e^{\mathbf{w} \cdot \mathbf{f}}} \qquad (3)$$

$$= \frac{exp\left(\sum_{i=0}^{N} w_i f_i\right)}{1 + exp\left(\sum_{i=0}^{N} w_i f_i\right)}$$

MEMM-Logistic Regression/Classification

- Example x belongs to class *true* if:

$$\frac{p(y = true|x)}{1 - p(y = true|x)} > 1 \tag{5}$$

$$e^{\mathbf{w} \cdot \mathbf{f}} > 1 \tag{6}$$

$$\mathbf{w} \cdot \mathbf{f} > 0 \tag{7}$$

$$\sum_{i=0}^{N} w_i f_i > 0 \tag{8}$$

- The equation $\sum_{i=0}^{N} w_i f_i = 0$ defines the hyperplane in N-dimensional space, with points above this hyperplane belonging to class *true*

MEMM-Logistic Regression/Learning

- Conditional likelihood estimation: choose the weights which make the probability of the observed values y be the highest, given the observations x
- For the training set with M examples:

$$\hat{\mathbf{w}} = \underset{w}{\operatorname{argmax}} \prod_{i=0}^{M} P(y^{(i)}|x^{(i)})$$

- A problem in convex optimization (not covered here)
  - L-BFGS (Limited-memory Broyden-Fletcher-Goldfarb-Shanno method)
  - gradient ascent
  - conjugate gradient
  - iterative scaling algorithms

MEMM-Maximum Entropy Model

- Logistic regression with more than two classes = multinomial logistic regression
- Also known as Maximum Entropy (MaxEnt)
- The MaxEnt equation generalizes (4) above:

$$p(c|x) = \frac{exp\left(\sum_{i=0}^{N} w_{ci} f_i\right)}{\sum_{c' \in C} exp\left(\sum_{i=0}^{N} w_{c'i} f_i\right)} \tag{9}$$

- The denominator is the normalization factor usually called Z used to make the score into a proper probability distribution

$$p(c|x) = \frac{1}{Z} exp \sum_{i=0}^{N} w_{ci} f_i$$

HMMs and MEMMs

- HMM POS tagging model:

$$\hat{T} = \underset{T}{\operatorname{argmax}} P(T|W) \tag{10}$$

$$= \underset{T}{\operatorname{argmax}} P(W|T)P(T) \tag{11}$$

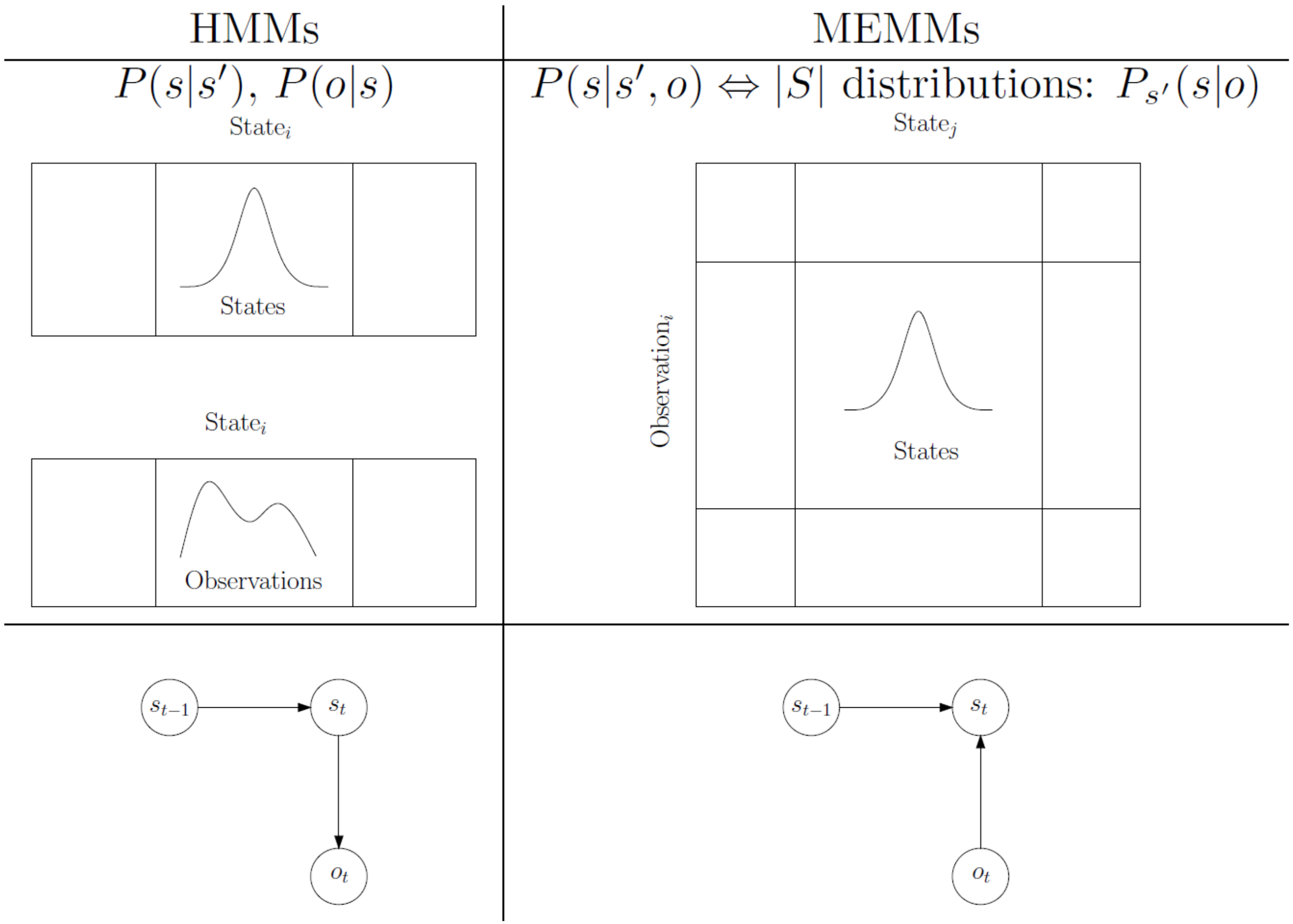
$$= \underset{T}{\operatorname{argmax}} \prod_i P(word_i|tag_i) \prod_i P(tag_i|tag_{i-1}) \tag{12}$$

- MEMM POS tagging model:

$$\hat{T} = \underset{T}{\operatorname{argmax}} P(T|W) \tag{13}$$

$$= \underset{T}{\operatorname{argmax}} \prod_i P(tag_i|word_i, tag_{i-1}) \tag{14}$$

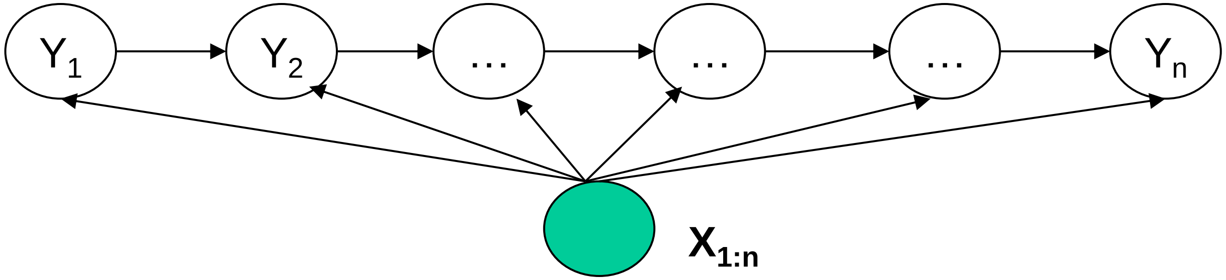
Maximum Entropy Markov Models (MEMMs)



HMM vs MEMM

HMMs	MEMMs
$\alpha_t(s)$ the probability of producing $o_1, \ldots, o_t$ and being in s at time t.	$\alpha_t(s)$ the probability of being in s at time t given $o_1, \ldots, o_t$.
$\alpha_{t+1}(s) = \sum_{s' \in S} \alpha_t(s') P(s\|s') P(o_{t+1}\|s)$	$\alpha_{t+1}(s) = \sum_{s' \in S} \alpha_t(s') P_{s'}(s\|o_{t+1})$
$\delta_t(s)$ the probability of the best path for producing $o_1, \ldots, o_t$ and being in s at time t.	$\delta_t(s)$ the probability of the best path that reaches s at time t given $o_1, \ldots, o_t$.
$\delta_{t+1}(s) = \max_{s' \in S} \delta_t(s') P(s\|s') P(o_{t+1}\|s)$	$\delta_{t+1}(s) = \max_{s' \in S} \delta_t(s') P_{s'}(s\|o_{t+1})$

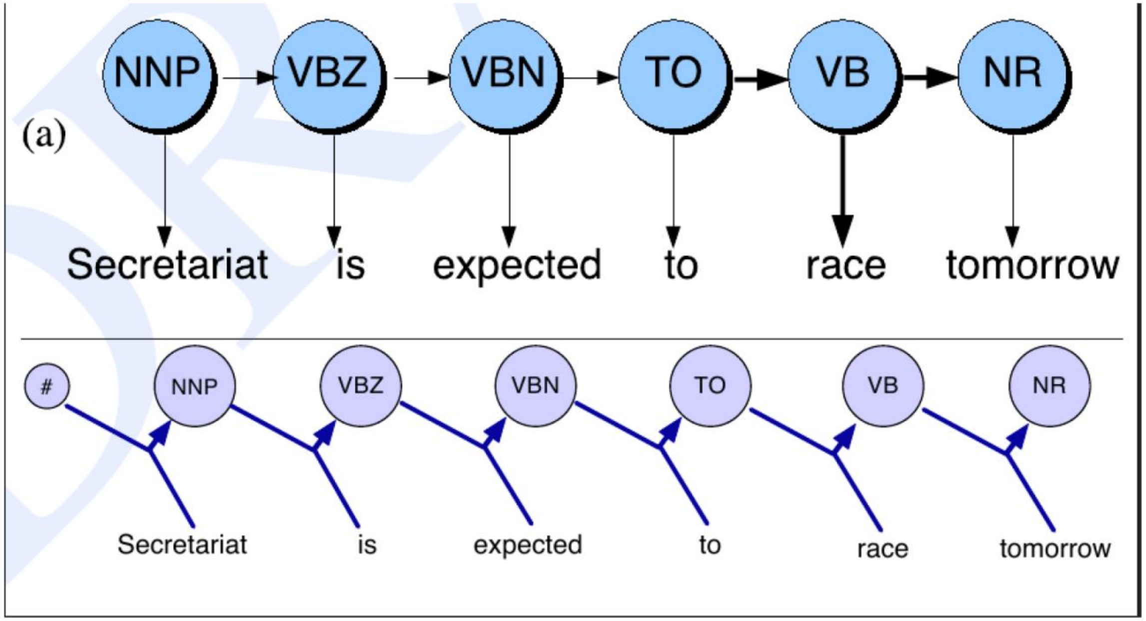
Maximum Entropy Markov Model (MEMM)



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

- 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**)

Viterbi in MEMMs



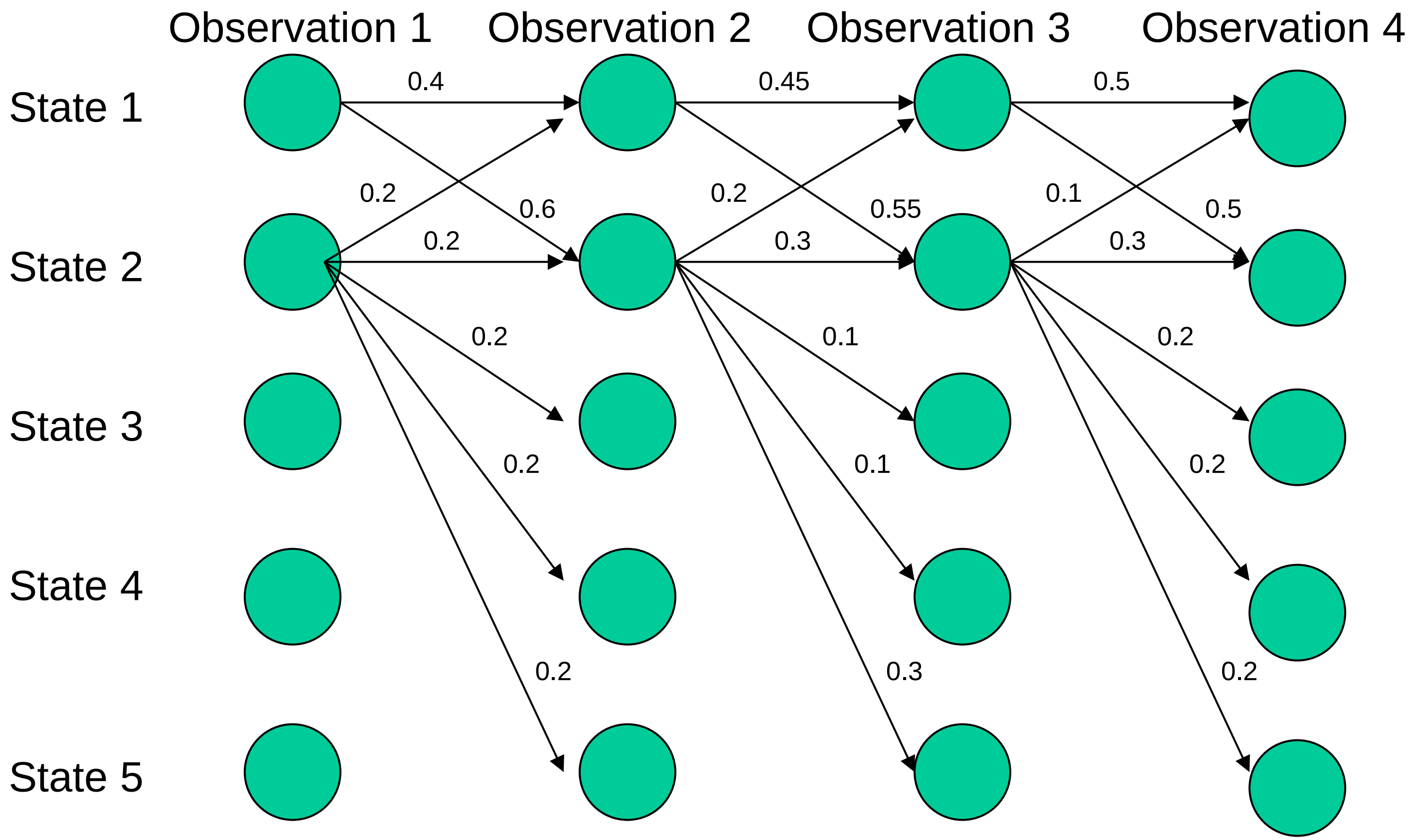
Decoding works almost the same as in HMM

Except entries in the DP table are values of $P(t_i | t_{i-1}, word_i)$

Recursive step: Viterbi value of time t for state j:

$$v_t(j) = \max_{i=i}^{N} v_{t-1}(i) P(s_j | s_i, o_t) \quad 1 \leq j \leq N, 1 < t \leq T$$

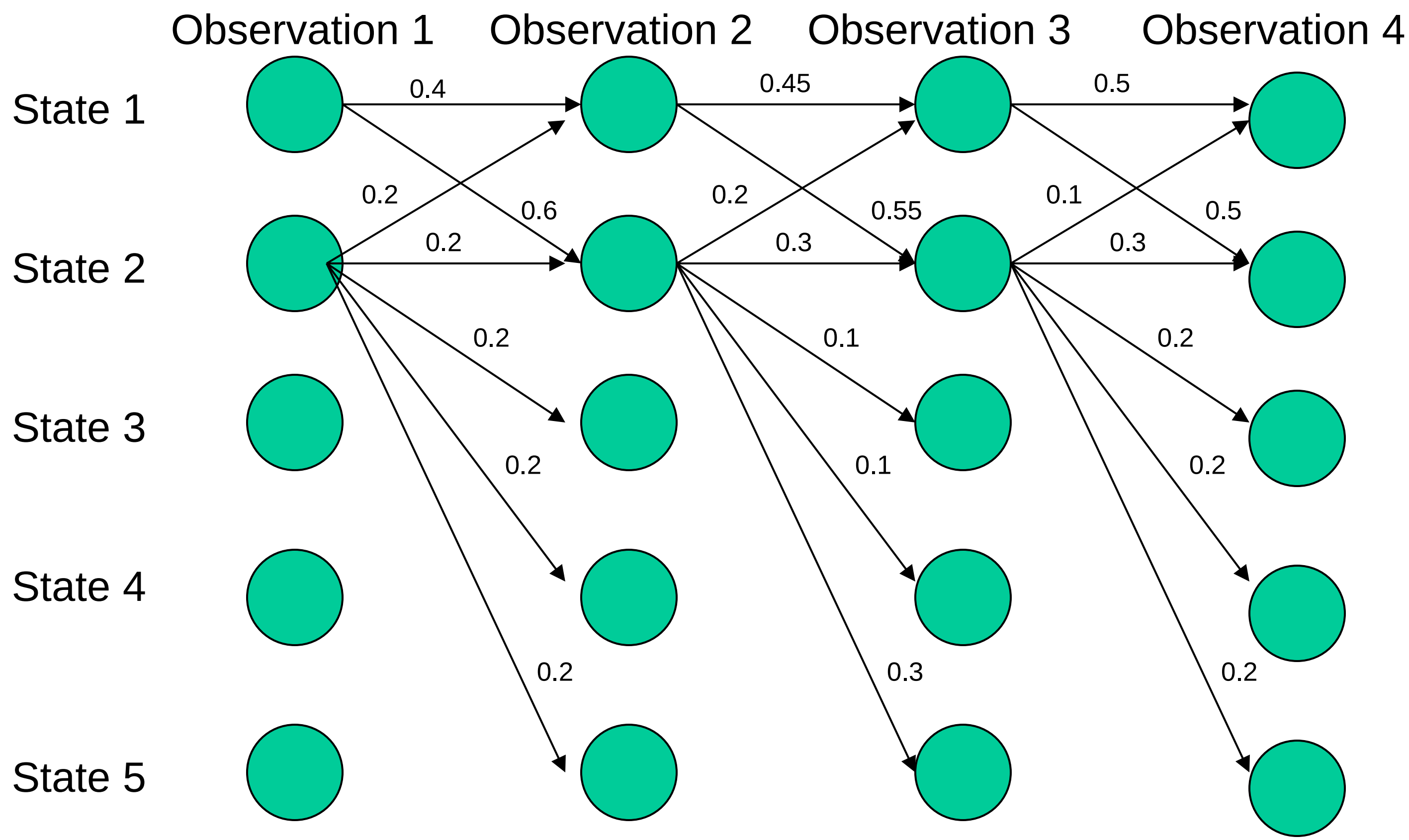
MEMM: Label Bias Problem



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

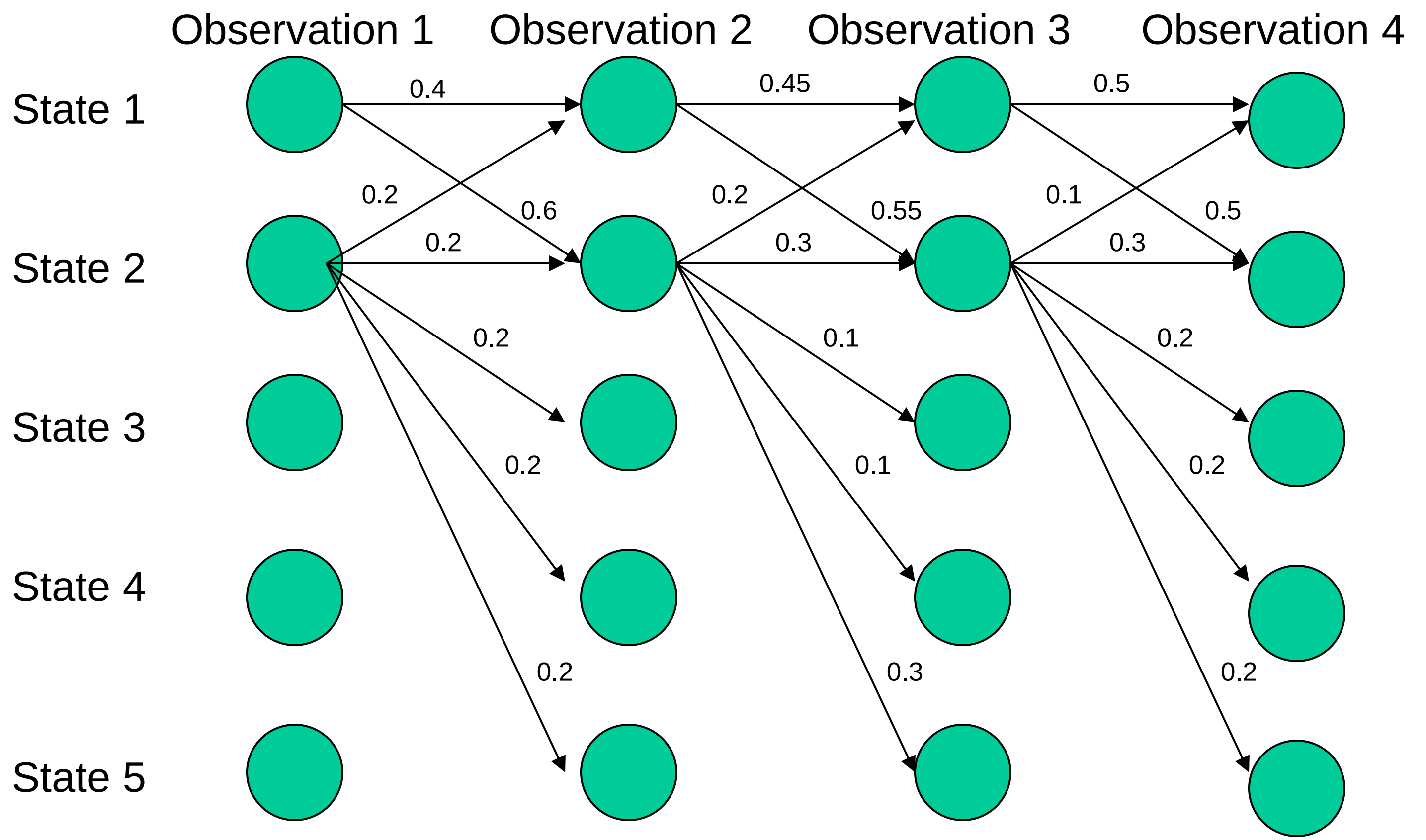
MEMM: Label Bias Problem



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

- 0.4 x 0.45 x 0.5 = 0.09

MEMM: Label Bias Problem

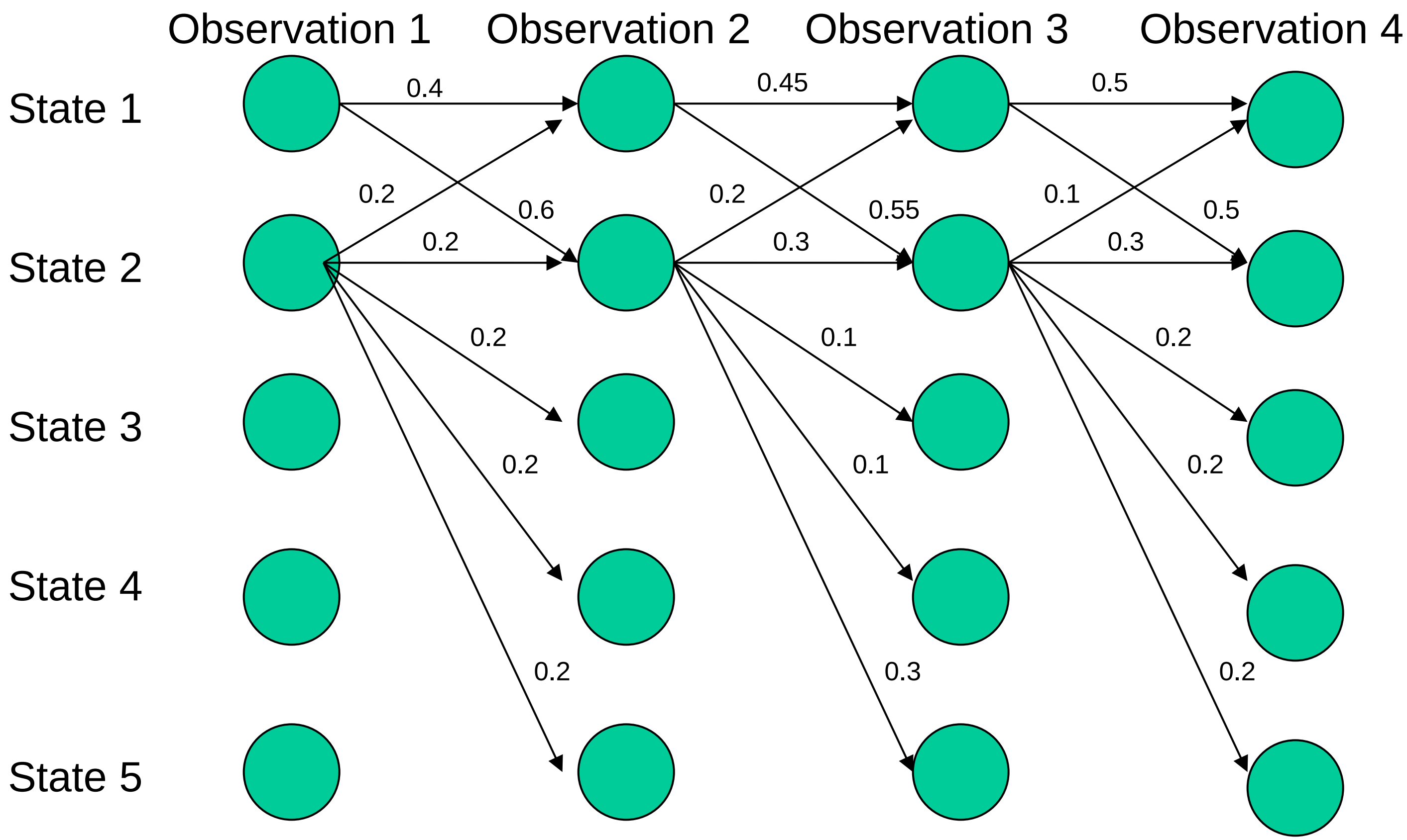


Probability of path 2->2->2->2 :

- 0.2 X 0.3 X 0.3 = 0.018

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

MEMM: Label Bias Problem

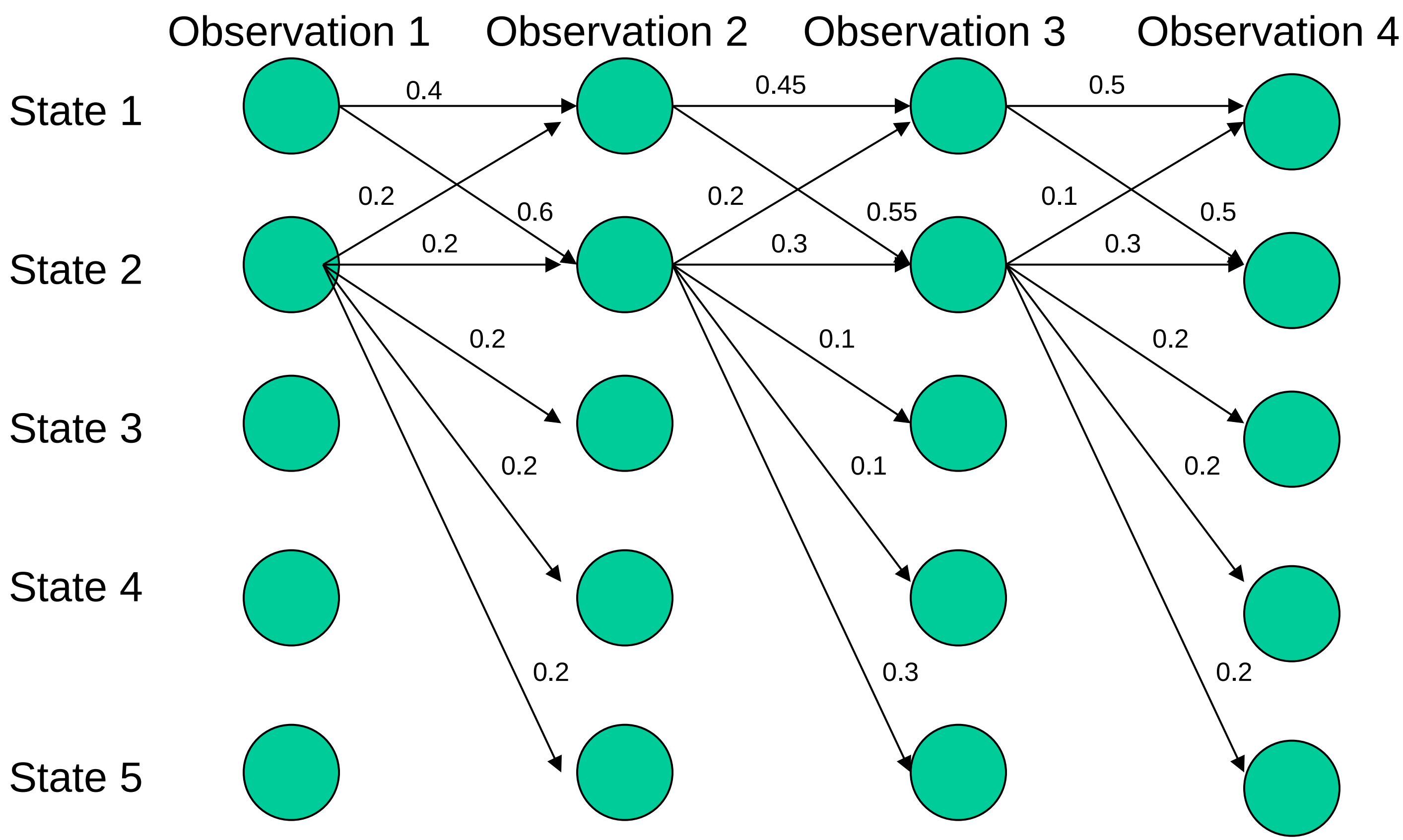


Probability of path 1->2->1->2:

- 0.6 X 0.2 X 0.5 = 0.06

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

MEMM: Label Bias Problem

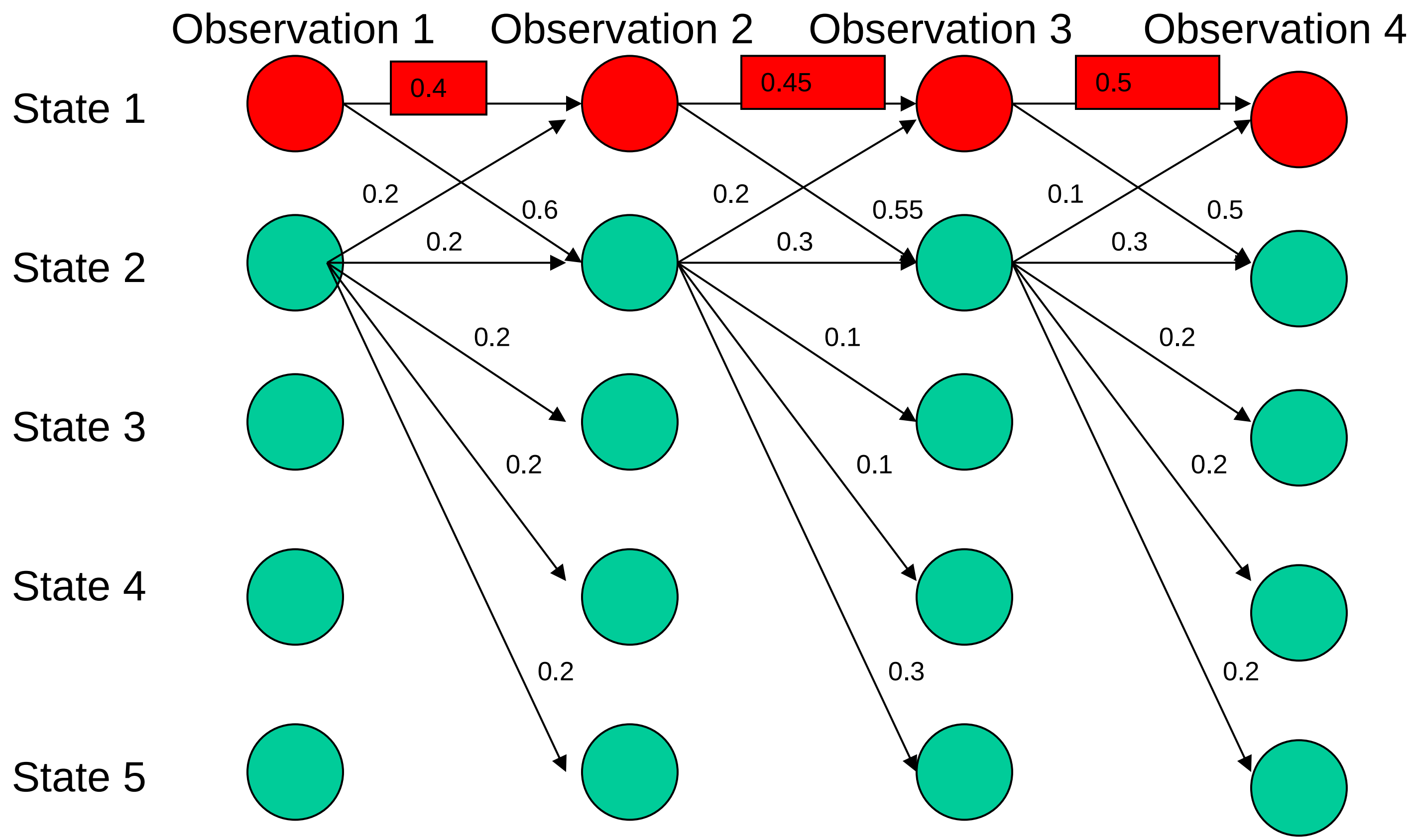


Probability of path 1->1->2->2:

- 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

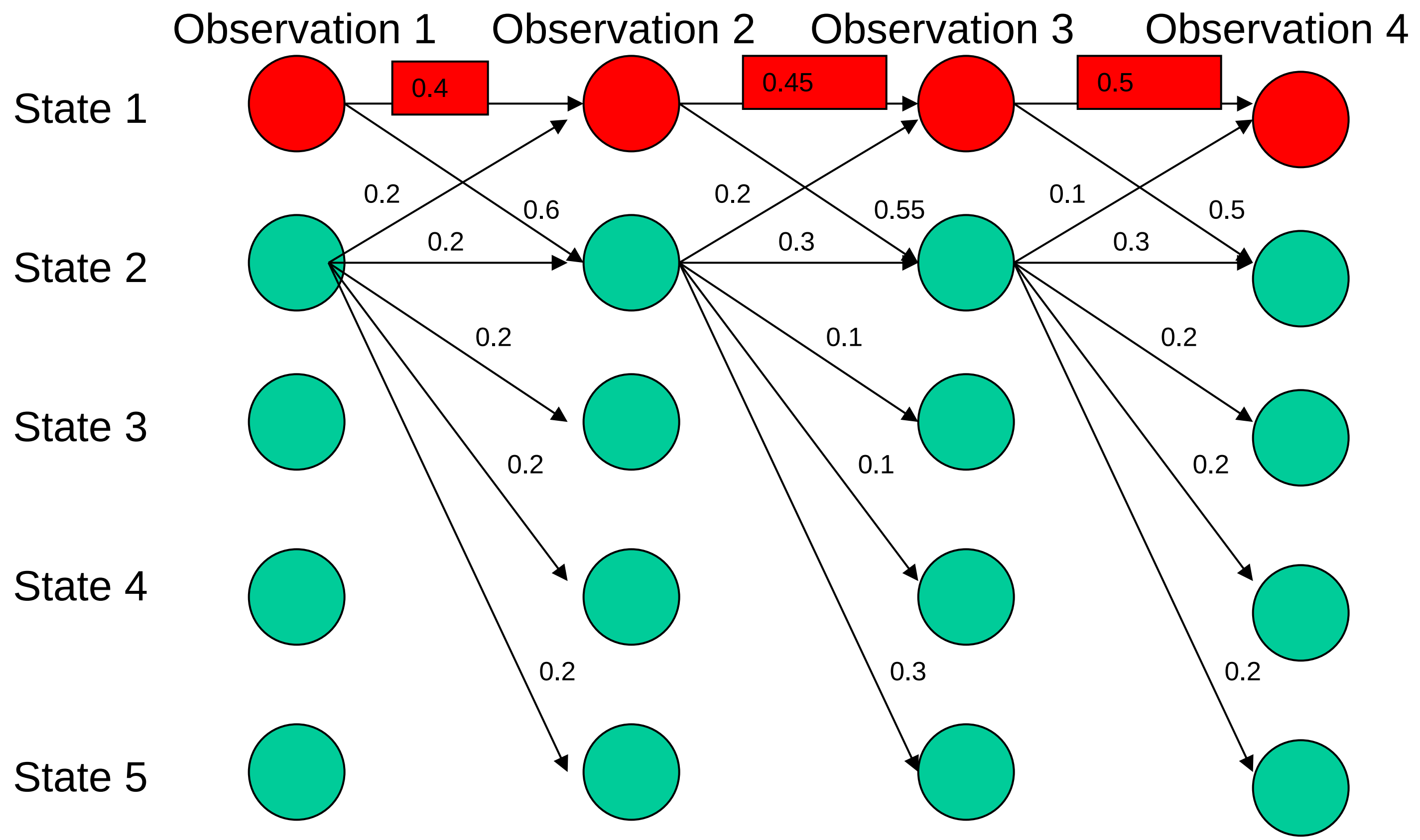
MEMM: Label Bias Problem



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?**

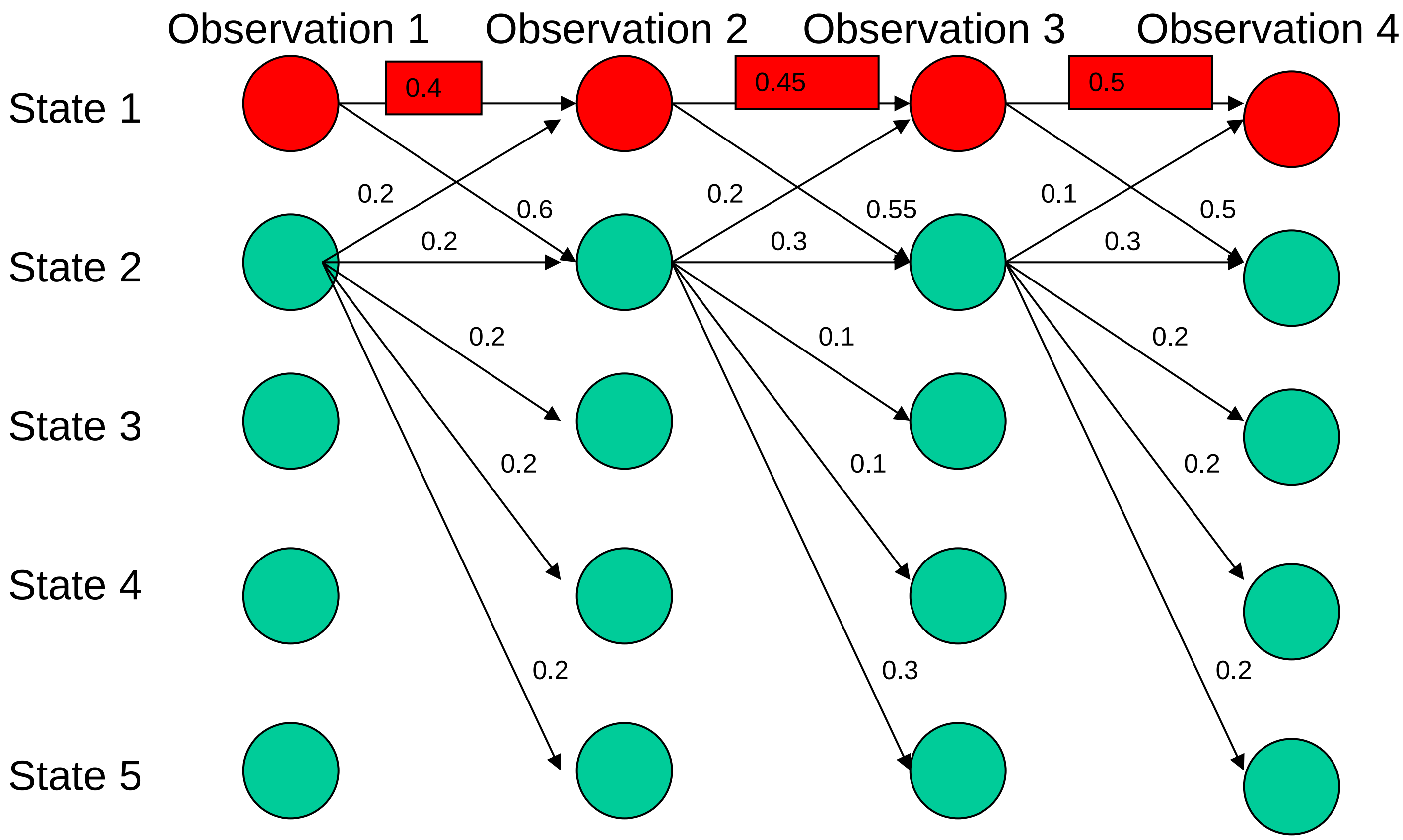
MEMM: Label Bias Problem



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

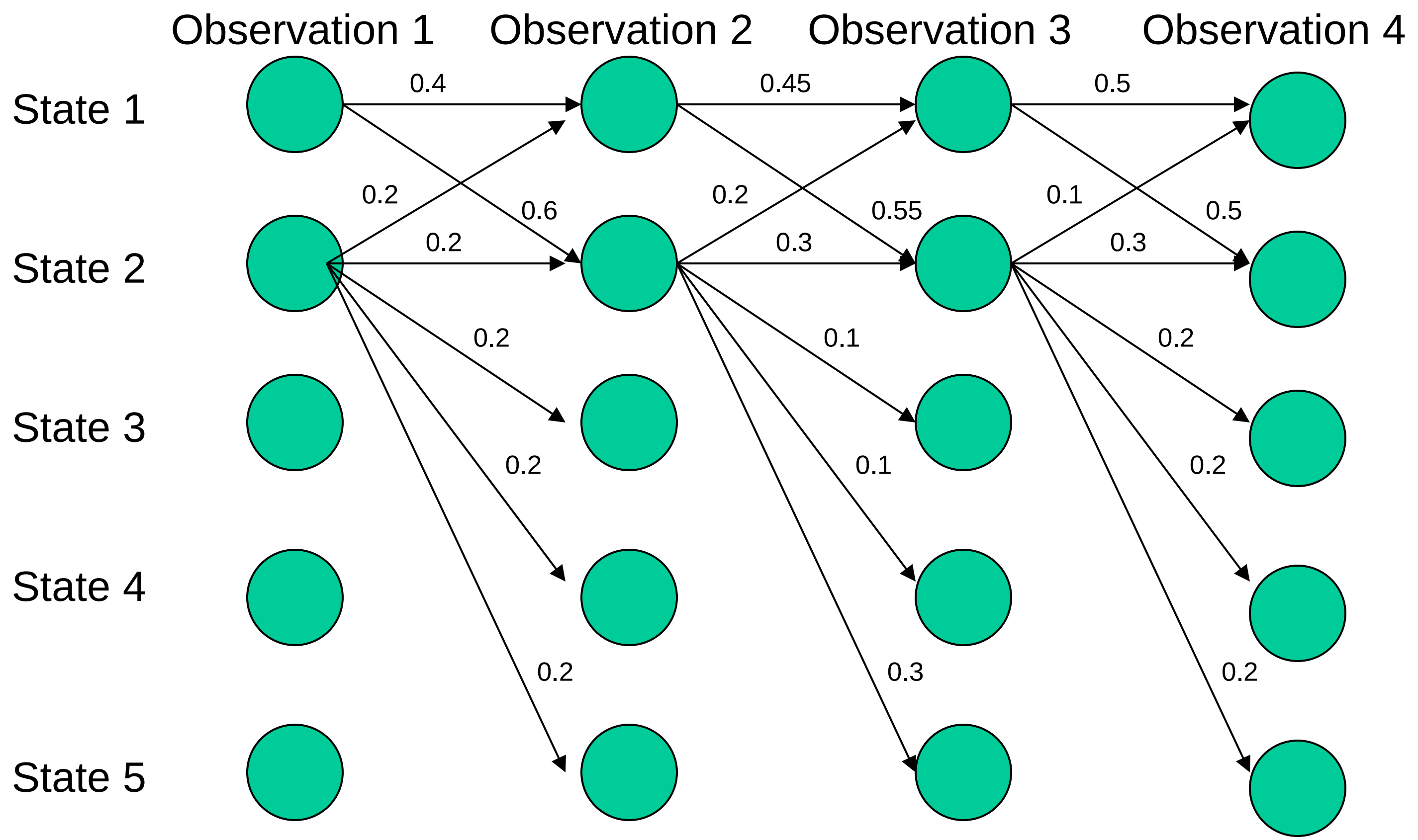
MEMM: Label Bias Problem



Label bias problem in MEMM:

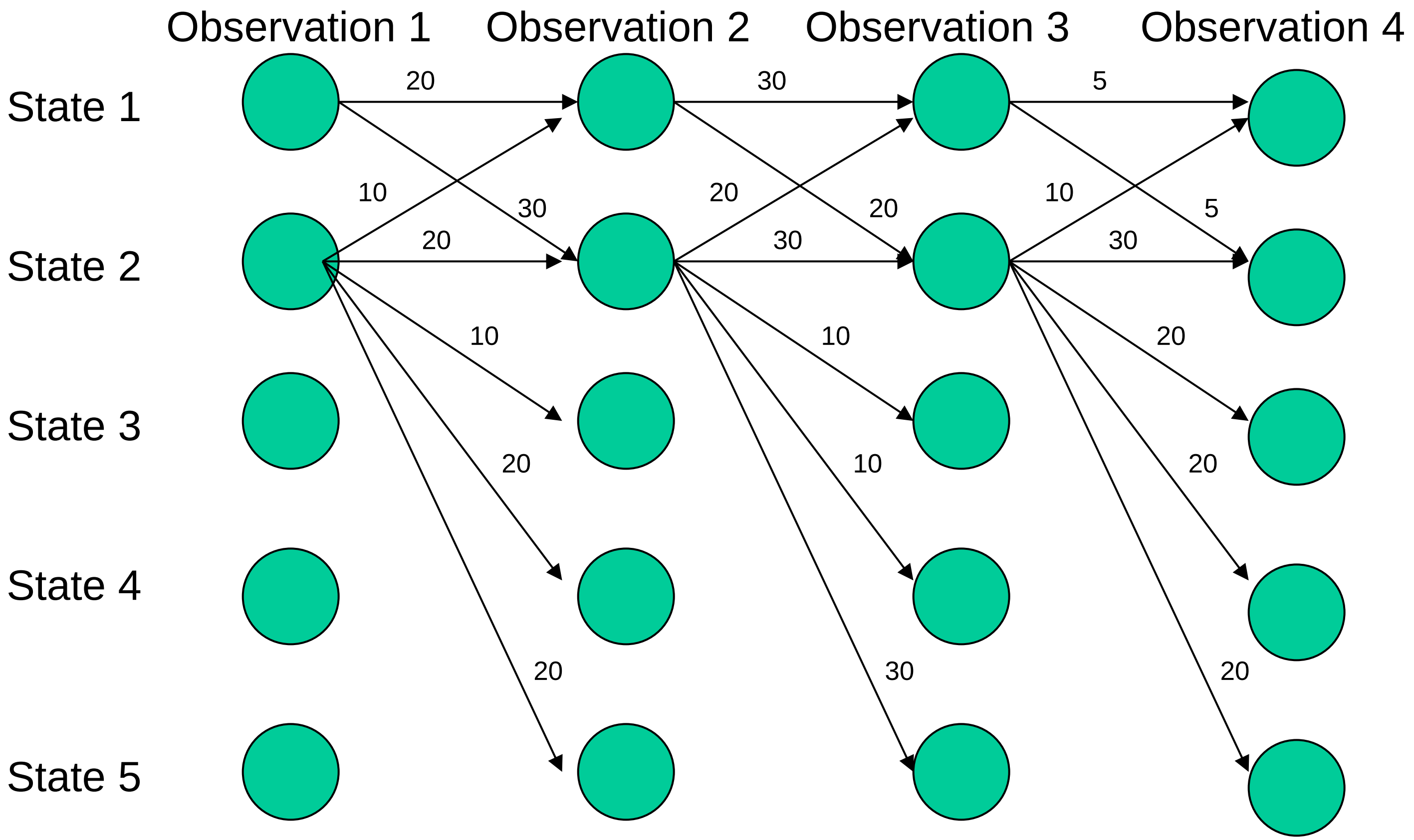
- Preference of states with lower number of transitions over others

Solution: Do not normalize probabilities locally



From local probabilities ….

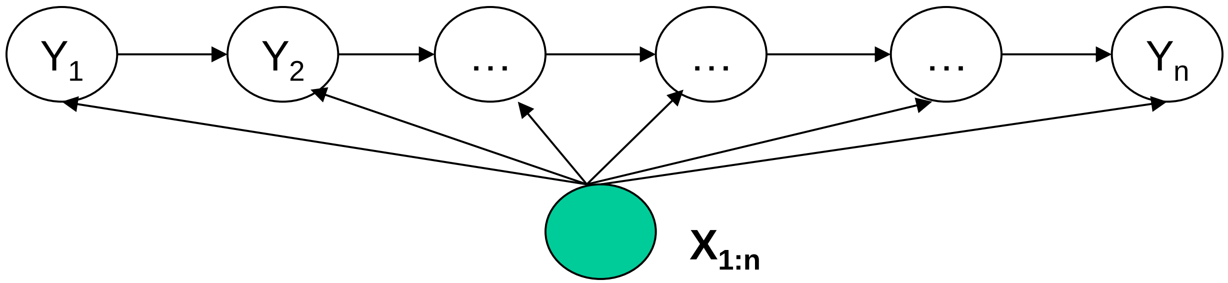
Solution: Do not normalize probabilities locally



From local probabilities to local potentials

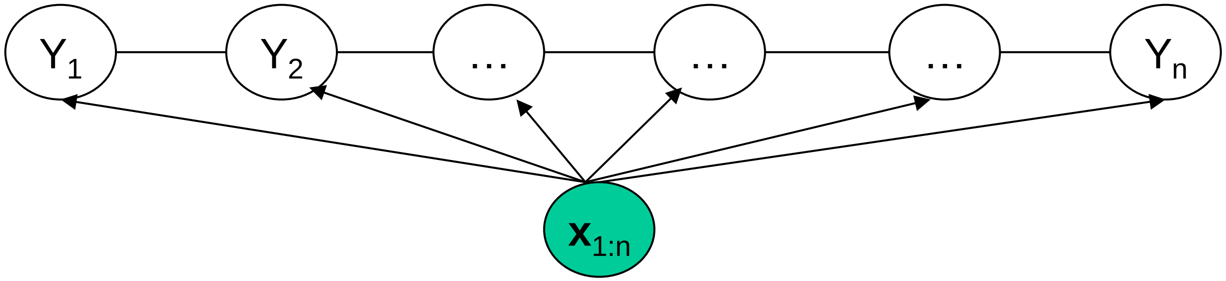
- States with lower transitions do not have an unfair advantage!

From MEMM ….



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

From MEMM to CRF

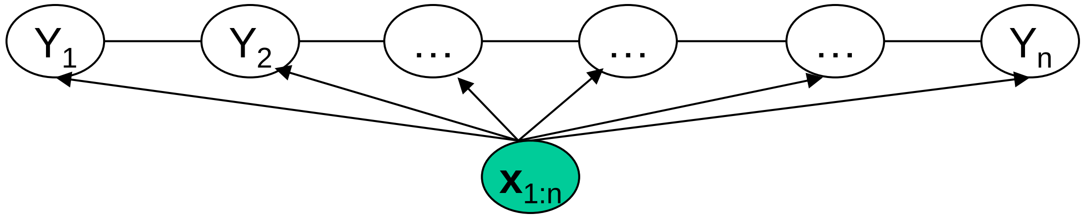


$$P(\mathbf{y}_{1:n}|\mathbf{x}_{1:n}) = \frac{1}{Z(\mathbf{x}_{1:n})}\prod_{i=1}^{n}\phi(y_i, y_{i-1}, \mathbf{x}_{1:n}) = \frac{1}{Z(\mathbf{x}_{1:n})}\prod_{i=1}^{n}\exp(\mathbf{w}^T\mathbf{f}(y_i, y_{i-1}, \mathbf{x}_{1:n}))$$

- CRF is a partially directed model
  - Discriminative model like MEMM
  - Usage of global normalizer Z(**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:



$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})}\exp(\sum_{i=1}^{n}(\sum_{k}\lambda_k f_k(y_i, y_{i-1}, \mathbf{x}) + \sum_{l}\mu_l g_l(y_i, \mathbf{x})))$$

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

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

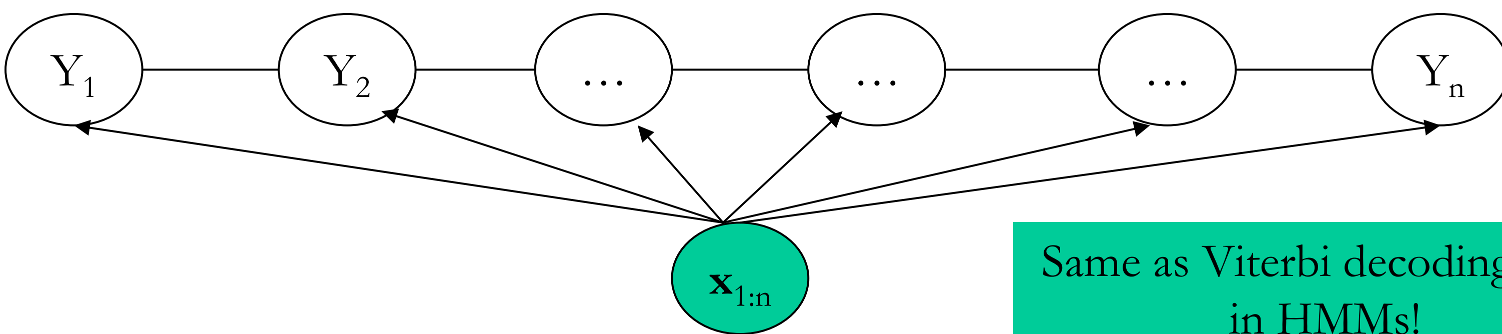
CRFs: Inference

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

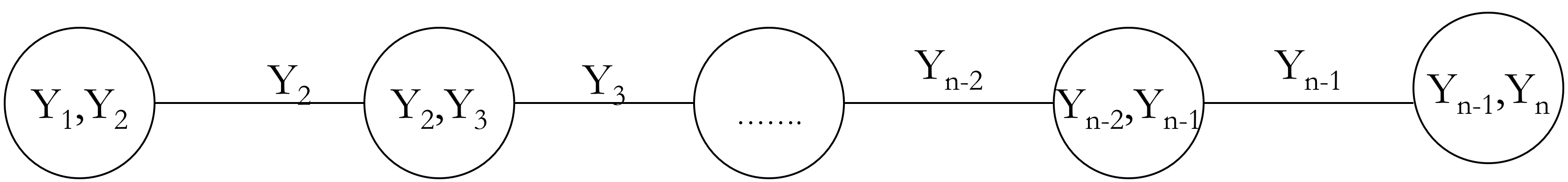
$$\mathbf{y}^* \quad = \quad \arg\max_{\mathbf{y}} \exp\left(\sum_{i=1}^{n}(\lambda^T \mathbf{f}(y_i, y_{i-1}, \mathbf{x}) + \mu^T \mathbf{g}(y_i, \mathbf{x}))\right)$$

  - Can ignore Z($\mathbf{x}$) because it is not a function of $\mathbf{y}$

- Run the max-product algorithm on the junction-tree of CRF:



Same as Viterbi decoding used in HMMs!

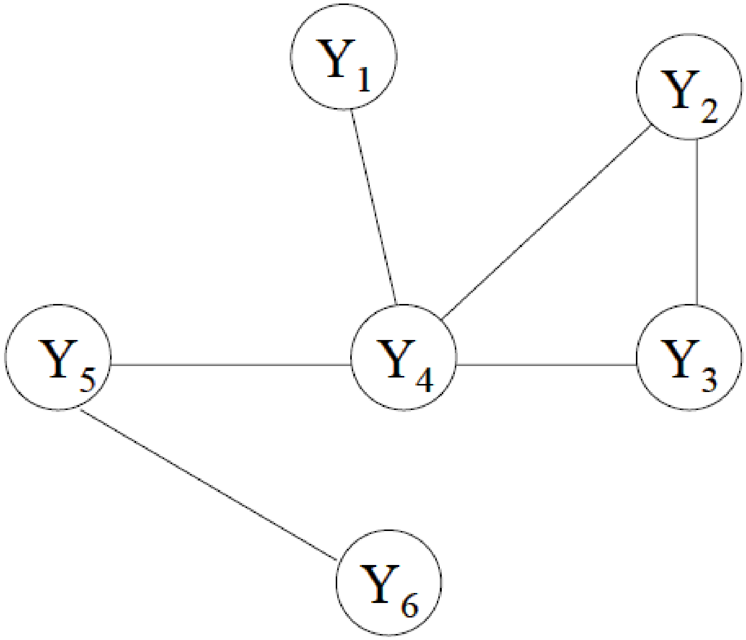


Recall: Random Field

Let G = (Y, E) be a graph where each vertex Y_v is a random variable

Suppose $P(Y_v \mid \text{all other } Y) = P(Y_v \mid \text{neighbors}(Y_v))$ then Y is a random field

Example:



- $P(Y_5 \mid \text{all other } Y) = P(Y_5 \mid Y_4, Y_6)$

Conditional Random Fields (CRFs)

- CRFs have all the advantages of MEMMs without label bias problem
  - MEMM uses per-state exponential model for the conditional probabilities of next states given the current state
  - CRF has a single exponential model for the joint probability of the entire sequence of labels given the observation sequence
- Undirected acyclic graph
- Allow some transitions “vote” more strongly than others depending on the corresponding observations

Definition of CRFs

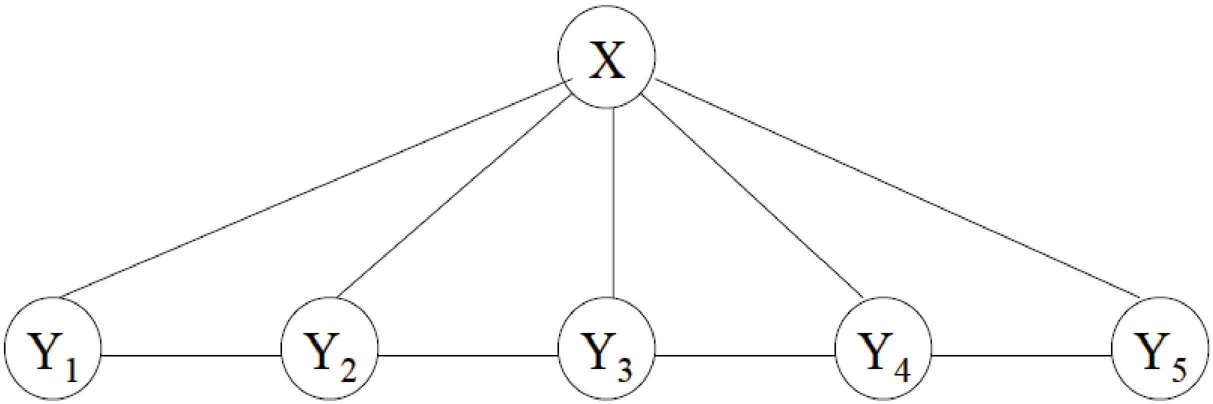
X is a random variable over data sequences to be labeled

Y is a random variable over corresponding label sequences

Definition. *Let $G = (V, E)$ be a graph such that $\mathbf{Y} = (\mathbf{Y}_v)_{v \in V}$, so that $\mathbf{Y}$ is indexed by the vertices of G. Then $(\mathbf{X}, \mathbf{Y})$ is a conditional random field in case, when conditioned on $\mathbf{X}$, the random variables $\mathbf{Y}_v$ obey the Markov property with respect to the graph: $p(\mathbf{Y}_v \,|\, \mathbf{X}, \mathbf{Y}_w, w \neq v) = p(\mathbf{Y}_v \,|\, \mathbf{X}, \mathbf{Y}_w, w \sim v)$, where $w \sim v$ means that w and v are neighbors in G.*

Example of CRFs

Suppose $P(Y_v \mid X, \text{all other } Y) = P(Y_v \mid X, \text{neighbors}(Y_v))$ then X with Y is a **conditional** random field



- $P(Y_3 \mid X, \text{all other } Y) = P(Y_3 \mid X, Y_2, Y_4)$
- Think of X as observations and Y as labels

Graphical comparison among HMMs, MEMMs and CRFs

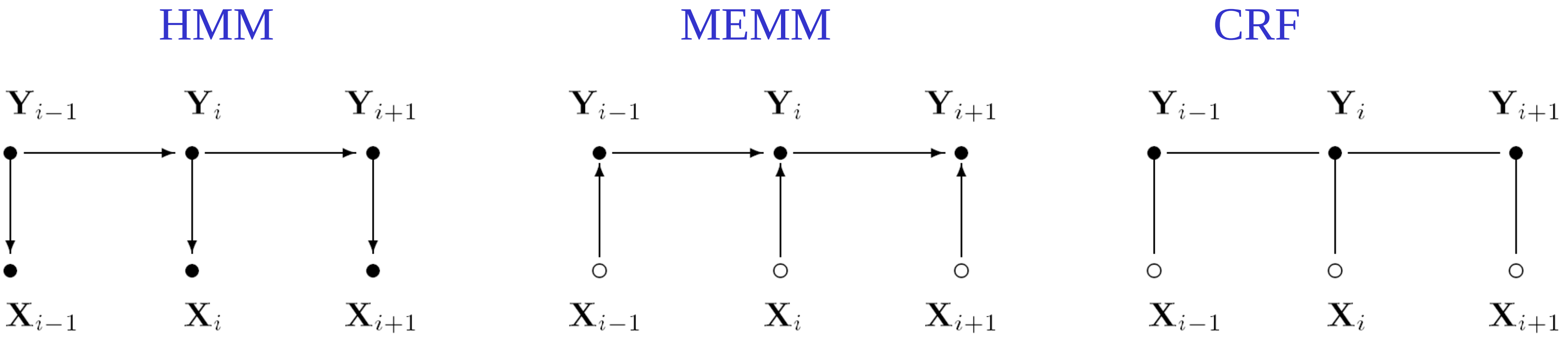
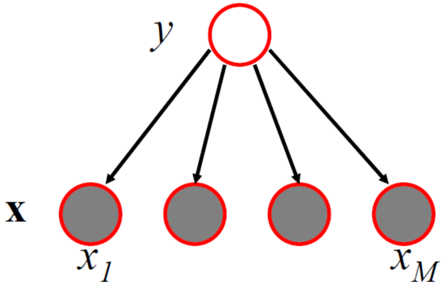


Figure 2. Graphical structures of simple HMMs (left), MEMMs (center), and the chain-structured case of CRFs (right) for sequences. An open circle indicates that the variable is not generated by the model.

Functional Models

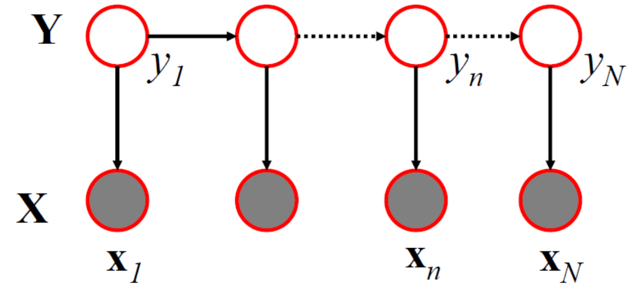
GENERATIVE

Naïve Bayes Classifier



$$p(y,\mathbf{x}) = p(y)\prod_{m=1}^{M} p(x_m \mid y)$$

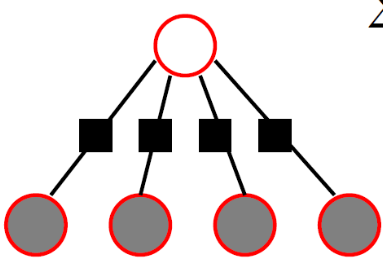
Hidden Markov Model



$$p(\mathbf{Y},\mathbf{X}) = \prod_{n=1}^{N} p(y_n \mid y_{n-1})\, p(\mathbf{x}_n \mid y_n)$$

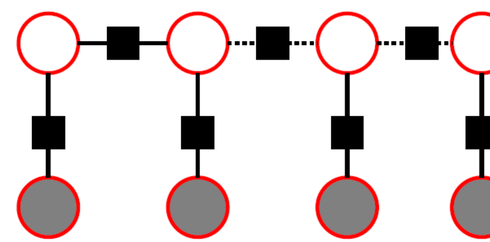
DISCRIMINATIVE

$$p(y \mid \mathbf{x}) = \frac{\exp\left\{\sum_{m=1}^{M} \lambda_m f_m(y,\mathbf{x})\right\}}{\sum_{y'} \exp\left\{\sum_{m=1}^{M} \lambda_m f_m(y',\mathbf{x})\right\}}$$



Logistic Regression

$$p(\mathbf{Y} \mid \mathbf{X}) = \frac{\exp\left\{\sum_{m=1}^{M} \lambda_m f_m(y_n, y_{n-1}, \mathbf{x}_n)\right\}}{\sum_{\mathbf{y}'} \exp\left\{\sum_{m=1}^{M} \lambda_m f_m(y_n', y_{n-1}', \mathbf{x}_n)\right\}}$$



Conditional Random Field

Conditional Distribution

If the graph $G = (V, E)$ of Y is a tree, the conditional distribution over the label sequence Y = y, given X = x, by fundamental theorem of random fields is:

$$p_\theta(\mathrm{y} \mid \mathrm{x}) \propto \exp\left(\sum_{e \in E, k} \lambda_k f_k(e, \mathrm{y}|_e, \mathrm{x}) + \sum_{v \in V, k} \mu_k g_k(v, \mathrm{y}|_v, \mathrm{x}) \right)$$

x is a data sequence

y is a label sequence

v is a vertex from vertex set V = set of label random variables

e is an edge from edge set E over V

f_k and g_k are given and fixed. g_k is a Boolean vertex feature; f_k is a Boolean edge feature

k is the number of features

$\theta = (\lambda_1, \lambda_2, \cdots, \lambda_n; \mu_1, \mu_2, \cdots, \mu_n)$; λ_k and μ_k are parameters to be estimated

$\mathrm{y}|_e$ is the set of components of y defined by edge *e*

$\mathrm{y}|_v$ is the set of components of y defined by vertex *v*

Training of CRFs (From Prof. Dietterich)

- First, we take the log of the equation

$$\log p_\theta(y \mid x) = \sum_{e \in E,k} \lambda_k f_k(e, \mathrm{y}|_e, \mathrm{x}) + \sum_{v \in V,k} \mu_k g_k(v, \mathrm{y}|_v, \mathrm{x}) - \log Z(\mathrm{x})$$

- Then, take the derivative of the above equation

$$\frac{\partial \log p_\theta(y \mid x)}{\partial \theta} = \frac{\partial}{\partial \theta}\left(\sum_{e \in E,k} \lambda_k f_k(e, \mathrm{y}|_e, \mathrm{x}) + \sum_{v \in V,k} \mu_k g_k(v, \mathrm{y}|_v, \mathrm{x}) - \log Z(\mathrm{x}) \right)$$

- For training, the first 2 items are easy to get.
- For example, for each λ_k, f_k is a sequence of Boolean numbers, such as 00101110100111.

$\lambda_k f_k(e, \mathrm{y}|_e, \mathrm{x})$ is just the total number of 1's in the sequence.

- The hardest thing is how to calculate $Z(\mathrm{x})$

Another Definition

CRF is a Markov Random Fields.

By the Hammersley-Clifford theorem, the probability of a label can be expressed as a Gibbs distribution, so that

$$p(y \mid x, \lambda, \mu) = \frac{1}{Z} \exp(\sum_j \lambda_j F_j(y, x))$$

$$F_j(y, x) = \sum_{i=1}^{n} f_j(y_{|c}, x, i)$$

clique

What is clique?

By only taking consideration of the one node and two nodes cliques, we have

$$p(y \mid x, \lambda, \mu) = \frac{1}{Z} \exp(\sum_j \lambda_j t_j(y_{|e}, x, i) + \sum_k \mu_k s_k(y_{|s}, x, i))$$

Definition (cont.)

Moreover, let us consider the problem in a first-order chain model, we have

$$p(y \mid x, \lambda, \mu) = \frac{1}{Z} \exp\Big(\sum_j \lambda_j t_j (y_{i-1}, y_i, x, i) + \sum_k \mu_k s_k (y_i, x, i)\Big)$$

For simplifying description, let $f_j(y, x)$ denote $t_j(y_{i-1}, y_i, x, i)$ and $s_k(y_i, x, i)$

$$p(y \mid x, \lambda, \mu) = \frac{1}{Z} \exp\Big(\sum_j \lambda_j F_j (y, x)\Big)$$

$$F_j (y, x) = \sum_{i=1}^{n} f_j (y_{|c}, x, i)$$