Generative and Discriminative Approaches to Graphical Models
CMSC 35900 Topics in AI

Lecture 2

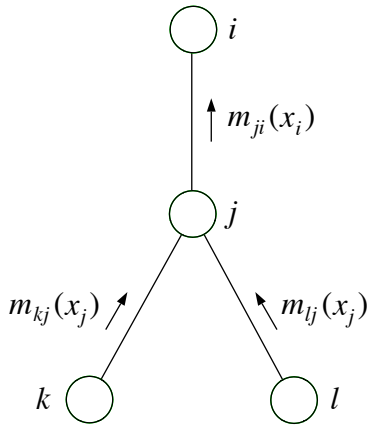
Yasemin Altun

January 26, 2007

Review of Inference on Graphical Models

- Elimination algorithm finds single marginal probability $p(f)$ by
  - Choosing an elimination ordering I such that f is the last node
  - Keep track of active potentials
  - Eliminate a node i by removing all active potentials referencing i, take product and sum over x_i, put this *message* on the active list.
- Consider trees with multiple queries. This is a special case of Junction Tree Algorithm, Sum-Product algorithm.
- Choose a root node arbitrarily
- A natural elimination order is depth-first traversal of the tree.

Inference on Trees for Single Query $p(r)$



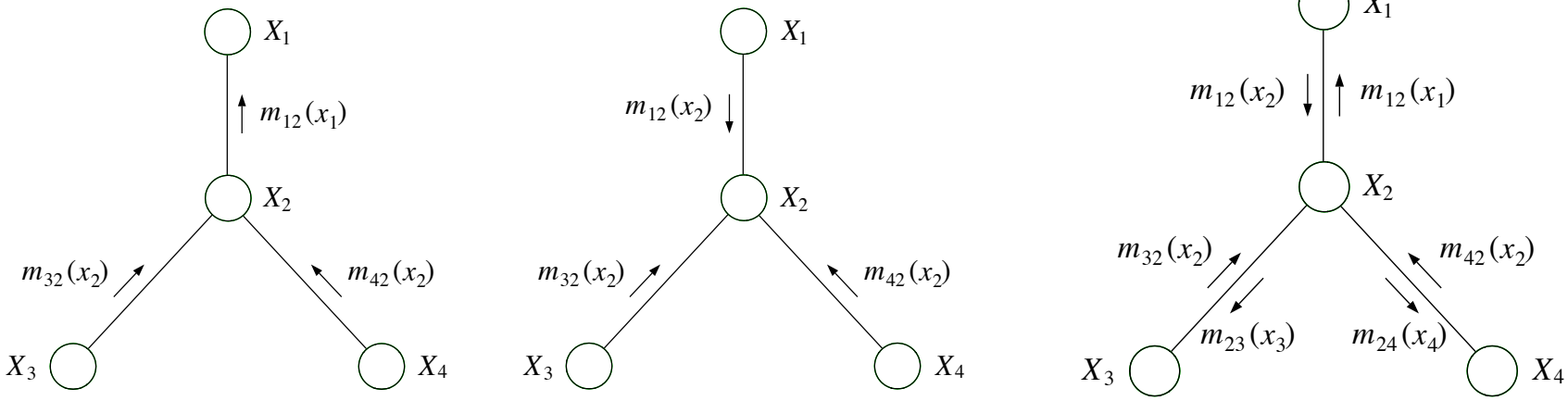
- Elimination of j node.
- If j is an evidence node $\phi^E(x_j) = 1[x_j = \bar{x}_j]$ else $\phi^E(x_j) = 1$

$$m_{ji}(x_i) = \sum_{x_j} \left(\phi_E(x_j)\phi(x_i, x_j) \prod_{k \in c(j)} m_{kj}(x_j) \right)$$

- The marginal of root x_r is proportional to its messages

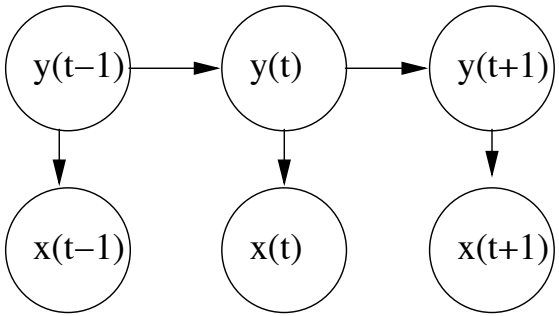
$$p(x_r|\bar{x}_e) = \phi^E(x_r) \prod_{k \in c(r)} m_{kr}(x_r)$$

Inference on Trees for Multiple Query



- Fig a) Choosing 1 as root, 2 has all the messages from nodes below it
- Fig b) To compute $p(x_2)$, $m_{12}(x_2)$ is needed
- Fig c) One pass from leaves to root and one pass backward gives the messages required to compute all node marginals

Inference on HMMs



a) HMM

- Hidden Markov Models are directed sequence models.
  - Let $\mathbf{x} = (x_1, \ldots, x_l)$ and $\mathbf{y} = (y_1, \ldots, y_l)$ be sequences.
  - $x_i \in \mathcal{O}$, where $\mathcal{O}$ is a set of possible observations.
  - $y_i \in \mathcal{Y}$, where $\mathcal{Y}$ is a set of possible labels.
  - $\pi_\sigma = p(y_1 = \sigma; \theta)$ initial transition.
  - $T_{\sigma,\sigma'} = p(y_{i+1} = \sigma | y_i = \sigma'; \theta)$ transition probabilities for $\sigma, \sigma' \in \mathcal{Y}$
  - $O_{\sigma,u} = p(x_i = u | y_i = \sigma; \theta)$ observation probabilities for $\sigma \in \mathcal{Y}, u \in \mathcal{O}$

$$P(\mathbf{x}, \mathbf{y}; \theta) = \pi_{y_1} \prod_{i=1}^{N} O_{y_i,x_i} \prod_{i=1}^{N-1} T_{y_{i+1},y_i}$$

Forward-Backward Algorithm for HMMs

- Set y_l as the root, $\mathbf{x}$ are evidence.
- Forward pass $\alpha_i(\sigma) = m_{(i-1)i}(x_i = \sigma)$

$$p(\bar{x}_{1:i}, y_i = \sigma) = \alpha_i(\sigma) = \left(\sum_{\sigma' \in \mathcal{Y}} \alpha_{i-1}(\sigma') T_{\sigma,\sigma'} \right) O_{\sigma,\bar{x}_i}$$

- Backward pass

$$p(\bar{x}_{i+1:l} | y_i = \sigma) = \beta_i(\sigma) = \sum_{\sigma' \in \mathcal{Y}} T_{\sigma',\sigma} O_{\sigma',\bar{x}_{i+1}} \beta_{i+1}(\sigma')$$

- Combining messages

$$\begin{aligned} P(y_i = \sigma | x_{1:l}) &= \frac{p(\bar{x}_{1:l} | y_i) p(y_i)}{p(\bar{x}_{i:l})} \\ &\propto p(\bar{x}_{1:i} | y_i) p(\bar{x}_{i+1:l} | y_i) p(y_i) = \alpha_i(\sigma) \beta_i(\sigma) \end{aligned}$$

Generalization to Junction Tree Algorithm

- Data structure:
  - Generate a tree whose nodes are the maximal cliques. Then, we get $p(x_C|x_{V\setminus C})$ using the sum-product. We can marginalize over this to get individual marginal probabilities.
  - Marginalization over different cliques of node i should yield to same $p(x_i|x_{V\setminus i})$.
  - There are multiple clique tree. We need a data structure that preserves global consistency via local consistency.
  - A *junction tree* is a clique tree for each clique pair C_i, C_j, sharing some variables C_s, C_s is included in all the nodes of the clique tree on the path from C_i and C_j.
  - There exists a junction tree of a graph iff the graph is triangulated.
- Elimination ordering
  - *Message-Passing Protocol:* A clique C_i can send a message to neighbor C_j if it has received messages from all its other neighbors.

Junction Tree Algorithm

- Moralize if the graph is directed
- Triangulate the graph G (using variable elimination algorithm)
- Construct a junction tree J from the triangulated graph via maximal weight spanning tree.
- Initialize the potential of cliques C of J as the product of potentials from G all assigned factors from the model.
- Propagate local messages on the junction tree

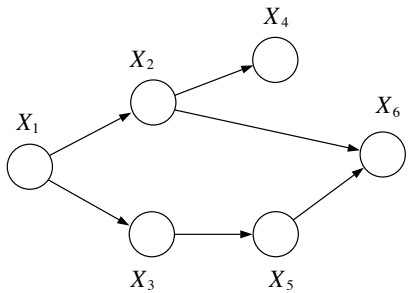
Finding maximum probability configurations

- Same formulation can be used exactly for finding MAP
- Multiplication distributes over max as well as sum

$$\max(ab, ac) = a \max(b, c)$$

- We can do MAP computation using exactly the same algorithm replacing sums with max

$$\begin{aligned}\max_{\mathbf{x}} p(\mathbf{x}) &= \max_{\mathbf{x}_1} \max_{\mathbf{x}_2} \max_{\mathbf{x}_3} \max_{\mathbf{x}_4} \max_{\mathbf{x}_5} p(\mathbf{x}_1)p(\mathbf{x}_2|\mathbf{x}_1)p(\mathbf{x}_3|\mathbf{x}_1)p(\mathbf{x}_4|\mathbf{x}_2)p(\mathbf{x}_5|\mathbf{x}_3)p(\mathbf{x}_6|\mathbf{x}_2, \mathbf{x}_5) \\ &= \max_{\mathbf{x}_1} p(\mathbf{x}_1) \max_{\mathbf{x}_2} p(\mathbf{x}_2|\mathbf{x}_1) \max_{\mathbf{x}_3} p(\mathbf{x}_3|\mathbf{x}_1) \max_{\mathbf{x}_4} p(\mathbf{x}_4|\mathbf{x}_2) \max_{\mathbf{x}_5} p(\mathbf{x}_5|\mathbf{x}_3)p(\mathbf{x}_6|\mathbf{x}_2, \mathbf{x}_5)\end{aligned}$$



Learning

- Parameter Learning: Given the graph structure, how to get the conditional probability distributions $P_\theta(X_i|X_{\Pi_i})$?
  - Parameters θ are unknown constants
  - Given fully observed training data D
  - Find their estimates maximizing the (penalized) log-likelihood

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \log P(D|\theta)(-\lambda R(\theta))$$

  - Parameter estimation with fully observed data
  - Parameter estimation with latent variables, Expectation Maximization (EM)
- Structure Learning: Given fully observed training data D, how to get the graph G and its parameters θ?
  - After studying the discriminative framework

Parameter Estimation with Complete Observations

- Random variable $\mathbf{X} = (X_1, \ldots, X_n)$. The data $D = (\mathbf{x}^1, \ldots, \mathbf{x}^m)$ is a set of m IID observations.
- Maximum likelihood estimate (MLE) can be found by maximizing the log probability of the data. In Bayes nets,

$$\ell(\theta; D) = \log p(D|\theta) = \log \prod_{j=1}^{m} p(\mathbf{x}^j|\theta) = \sum_{j=1}^{m} \sum_{i=1}^{n} \log p(x_i^j | x_{\pi_i}^j, \theta_i)$$

- Consider X_i be discrete rv with K possible values. Then, $p(x_i|X_{\pi_i}, \theta_i) = \theta_i(x_i, x_{\pi_i})$ is a multinomial distribution st $\sum_{x_i} \theta_i(x_i, x_{\pi_i}) = 1$.

$$\ell(\theta; D) = \sum_i \sum_{x_{i,\pi_i}} N(x_{i,\pi_i}) \log \theta_i(x_i, x_{\pi_i})$$

- $N(x_{i,\pi_i}) = \sum_{j=1}^{m} 1[x_{i,\pi_i}^j = x_{i,\pi_i}]$ observed count of x_{i,π_i} assignment in data

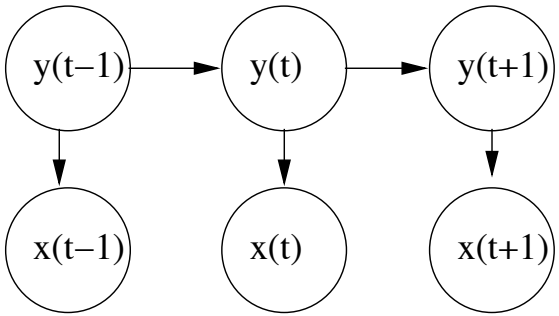
Parameter Estimation with Complete Observations

$$\ell(\theta; D) = \sum_i \sum_{x_{i,\pi_i}} N(x_{i,\pi_i}) \log \theta_i(x_i, x_{\pi_i})$$

- Estimation of θ_i is independent of θ_k for $k \neq i$.
- Estimating of θ_i, ignore data associated with nodes other than i and π_i and maximize $N(x_{i,\pi_i}) \log \theta_i(x_i, x_{\pi_i})$ wrt θ_i.
- Add a Lagrangian term for normalization and solve for θ_i
- ML estimate is the relative frequency.

$$\hat{\theta}_i(x_i, x_{\pi_i}) = N(x_i, x_{\pi_i}) / N(\pi_i)$$

Parameter Estimation in HMMs



a) HMM

- Hidden Markov Models are directed sequence models.
  - Let $\mathbf{x} = (x_1, \ldots, x_l)$ and $\mathbf{y} = (y_1, \ldots, y_l)$ be sequences.
  - $x_i \in \mathcal{O}$, where $\mathcal{O}$ is a set of possible observations.
  - $y_i \in \mathcal{Y}$, where $\mathcal{Y}$ is a set of possible labels.
  - $\pi_\sigma = p(y_1 = \sigma; \theta)$ initial transition.
  - $T_{\sigma,\sigma'} = p(y_{i+1} = \sigma | y_i = \sigma'; \theta)$ transition probabilities for $\sigma, \sigma' \in \mathcal{Y}$
  - $O_{\sigma,u} = p(x_i = u | y_i = \sigma; \theta)$ observation probabilities for $\sigma \in \mathcal{Y}, u \in \mathcal{O}$

$$P(\mathbf{x}, \mathbf{y}; \theta) = \pi_{y_1} \prod_{i=1}^{N} O_{y_i,x_i} \prod_{i=1}^{N-1} T_{y_{i+1},y_i}$$

Parameter Estimation in HMMs

$$P(\mathbf{x}, \mathbf{y}; \theta) = \pi_{y_1} \prod_{i=1}^{N} O_{y_i, x_i} \prod_{i=1}^{N-1} T_{y_{i+1}, y_i}$$

- MLE estimate
  - $\pi^*_\sigma = \frac{\sum_{j=1}^{m} 1[y_1^j = \sigma]}{m}$
  - $T^*_{\sigma,\sigma'} = \frac{\sum_{j=1}^{m} \sum_{i=1}^{l_j - 1} 1[y_{i+1}^j = \sigma \wedge y_i^j = \sigma']}{\sum_{j=1}^{m} \sum_{i=1}^{l_j - 1} y_i^j = \sigma'}$
  - $O^*_{\sigma,u} = \frac{\sum_{j=1}^{m} \sum_{i=1}^{l_j} 1[y_i^j = \sigma \wedge x_i^j = u]}{\sum_{j=1}^{m} \sum_{i=1}^{l_j} y_i^j = \sigma}$

MLE with Complete Observations for MRF

$$p(\mathbf{x}|\theta) = \frac{\prod_C \phi(x_C)}{\sum_{\mathbf{x}' \in \mathcal{Y}^n} \prod_c \phi(x'_c)}$$

- In MRFs, Z couples the parameters. There is no closed formed solution of θ.
- For potential functions $\phi_C(x_C) = \exp(\langle f(x_C), \theta_C \rangle)$

$$\begin{aligned}
\ell(\theta; D) &= \sum_{j=1}^{m} \log p(\mathbf{x}^j|\theta) \\
&= \sum_j \left(\sum_C \langle f(x_C), \theta_C \rangle - \log Z_\theta \right) \\
&= \sum_C N(x_C) \langle f(x_C), \theta_C \rangle - m \log Z_\theta
\end{aligned}$$

- Take the derivative of ℓ wrt θ, equate to 0 and solve for θ.
- The derivative of $\log Z$ wrt θ_C is the expectation

Parameter Estimation with Latent Variables

- When there are latent variables, we need to marginalize over the latent variables, which leads to coupling between parameters.
- Let X be observed variables, Z latent variables.

$$\ell(\theta; D) = \log \sum_z p(x, z|\theta)$$

- Expectation Maximization (EM) Algorithm is a general approach to maximum likelihood parameter estimation (MLE) with latent(hidden) variables.
- EM is a coordinate-descent algorithm to minimize Kullback-Leibler (KL) divergence

EM Algorithm

- Since Z is not observed, the log-likelihood of data is a marginal over Z which does not decompose.

$$\ell(\theta; x) := \log p(x|\theta) = \log \sum_z p(x, z|\theta)$$

- Replace this with *expected log-likelihood* wrt the *averaging distribution* $q(z|x)$. This is linear in $l_c(\theta; x, z) := \log p(x, z|\theta)$.

$$\langle \ell_c(\theta; x, z) \rangle_q := \sum_z q(z|x, \theta) \log p(x, z|\theta)$$

- This optimization is a lower bound on $l(\theta; x)$. Using Jensen's inequality

$$\begin{aligned} \ell(\theta; x) &= \log \sum_z p(x, z|\theta) = \log \sum_z q(z|x) \frac{p(x, z|\theta)}{q(z|x)} \\ &\leq \sum_z q(z|x) \log \frac{p(x, z|\theta)}{q(z|x)} = \mathcal{L}(q, \theta) \end{aligned}$$

EM Algorithm

- Until convergence
  - Maximize wrt q (E step):

$$q^{t+1} = \underset{q}{\operatorname{argmax}}\, \mathcal{L}(q, \theta^t)$$

  - Maximize wrt θ (M step):

$$\theta^{t+1} = \underset{\theta}{\operatorname{argmax}}\, \mathcal{L}(q^{t+1}, \theta)$$

EM Algorithm

- Until convergence
  - Maximize wrt q (E step): $q^{t+1} = \text{argmax}_q \mathcal{L}(q, \theta^t)$
  - Maximize wrt θ (M step): $\theta^{t+1} = \text{argmax}_\theta \mathcal{L}(q^{t+1}, \theta)$

$$\begin{aligned} \underset{\theta}{\text{argmax}}\, \mathcal{L}(q, \theta^t) &= \underset{\theta}{\text{argmax}} \sum_z q(z|x) \log p(x, z|\theta) \\ &- \sum_z q(z|x) \log q(z|x) \\ &= \underset{\theta}{\text{argmax}} \langle \ell_c(\theta; x, z) \rangle_q \end{aligned}$$

EM Algorithm

- Until convergence
  - Maximize wrt q (E step): $q^{t+1} = \text{argmax}_q \mathcal{L}(q, \theta^t)$

$$\begin{aligned} \ell(\theta; x) - \mathcal{L}(q, \theta) &= \sum_z q(z|x) \log p(x|\theta) - \sum_z q(z|x) \log \frac{p(x, z|\theta)}{q(z|x)} \\ &= D(q(z|x)||p(z|x, \theta)) \end{aligned}$$

    - Minimizing $\ell(\theta; x) - \mathcal{L}(q, \theta)$ is equivalent to maximizing $\mathcal{L}(q, \theta)$.
    - KL divergence is minimizes when $q(z|x) = p(z|x, \theta^t)$
    - Intuitively, given $p(x, z|\theta)$, $p(z|x, \theta^t)$ is the best guess for latent variables conditioned on x.
  - Maximize wrt θ (M step): $\theta^{t+1} = \text{argmax}_\theta \mathcal{L}(q^{t+1}, \theta)$
- M step increases a lower bound on likelihood
- E step closes the gap to yield $\ell(\theta^t; x) = \mathcal{L}(q^{t+1}, \theta^t)$

EM for HMMs

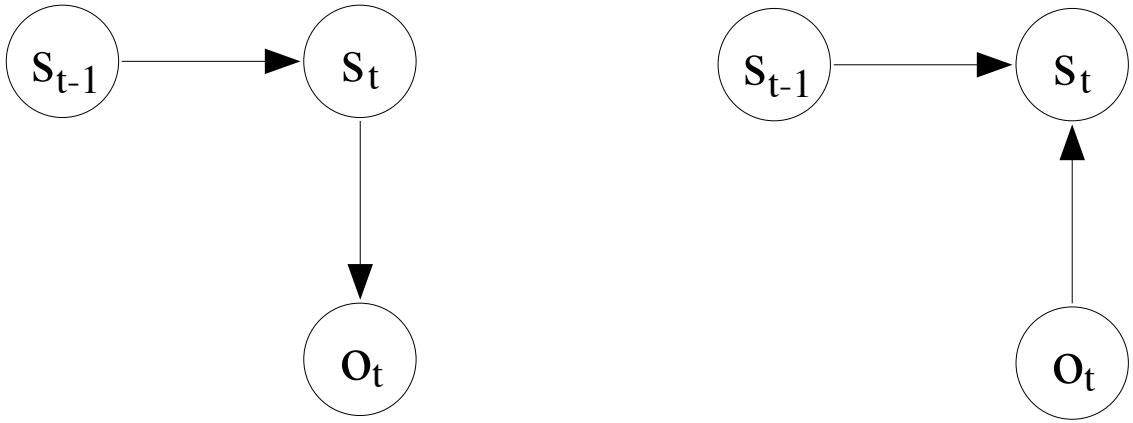
- E step: Get expected counts using Forward-Backward Algorithm.

$$q^t = p(z|x, \theta^{t-1})$$

- M step: Find MLE estimate wrt the expected counts (relative frequency).

$$\theta^t = \underset{\theta}{\text{argmax}} \langle \ell_c(\theta; x, z) \rangle_{q^t}$$

Discriminative Training of HMMs



- Maximum Entropy Markov Models (MEMM) [McCFrePer00] (Generalization to Bayes Nets is trivial.)

  a) HMM: $P(\mathbf{x}, \mathbf{y}; \theta) = \pi_{y_1} \prod_{i=1}^{N} O_{y_i, x_i} \prod_{i=1}^{N-1} T_{y_{i+1}, y_i}$

  b) MEMM: $P(\mathbf{y}|\mathbf{x}; \theta) = \pi_{y_1, x_1} \prod_{i=1}^{N-1} T_{y_{i+1}, y_i, x_{i+1}}$

Inference and Parameter Estimation in MEMMs

- Forward-Backward algorithm

$$\alpha_i(\sigma) = \sum_{\sigma' \in \mathcal{Y}} \alpha_{i-1}(\sigma') T_{\sigma,\sigma',\bar{x}_i}$$

$$\beta_i(\sigma) = \sum_{\sigma' \in \mathcal{Y}} T_{\sigma',\sigma,\bar{x}_{i+1}} \beta_{i+1}(\sigma')$$

- Representation for $\sigma, \bar{\sigma} \in \mathcal{Y}, u \in \mathcal{O}$

$$T_{\sigma,\sigma',\bar{u}} = \frac{1}{Z(\bar{u},\sigma')} \exp\left(\left\langle \theta, f(\bar{u},\sigma,\sigma)' \right\rangle\right)$$

$$Z(\bar{u},\sigma') = \sum_{\sigma} \exp\left(\left\langle \theta, f(\bar{u},\sigma,\sigma') \right\rangle\right)$$

- Coupling between θ due to Z results in no-closed-form solution. Use some convex optimization technique.
- Note the difference between undirected models where Z is a sum over all possible values of all nodes ($\mathcal{Y}^n$). Here, it is a sum over $\mathcal{Y}$.

Generative vs Discriminative Approaches

Generative framework: HMMs model $p(\mathbf{x}, \mathbf{y})$

- Advantages:
  - Efficient learning algorithm. Relative frequency.
  - Can handle missing observation (simply latent variable)
  - Naturally incorporates prior knowledge
- Disadvantages:
  - Harder problem $p(\mathbf{y}|\mathbf{x})p(\mathbf{y})$ vs $p(\mathbf{y}|\mathbf{x})$
  - Questionable independence assumption $x_i \perp x_j | y_i, \forall j \neq i$
  - Limited representation. Overlapping features are problematic.
    - Assuming independence violates the model.

$$p(x_i|y_i) = \prod_k p(f_k(x_i)|y_i)$$

$f_1(u) =$ Is u the word "Brown"?, $f_2(u) =$ Is u capitalized?

    - Conditioning increase the number of parameters.

$$p(x_i|y_i) = \prod_k p(f_k(x_i)|y_i, f_1(x_i), \ldots, f_{k-1}(x_i))$$

    - Increased complexity to capture dependency between y_i and x_{i-1}, $p(x_i|y_i, y_{i-1})$.

Generative vs Discriminative Approaches

Discriminative learning: MEMM model $p(\mathbf{y}|\mathbf{x})$

- Advantages:
  - Solves more direct problem.
  - Richer representation via kernels or feature selection. Conditioning on $\mathbf{x}$, overlapping features are trivial.
  - No independence assumption on observations.
  - Lower asymptotic error
- Disadvantages:
  - Expensive learning. There is no closed form solution as in HMMs. Therefore, we need Forward-Backward algorithm
  - Missing values in observation is problematic due to conditioning.
  - Requires more data ($O(d)$) to reach its asymptotic error, whereas generative models require $O(\log d)$, for d number of parameters [NgJor01].