

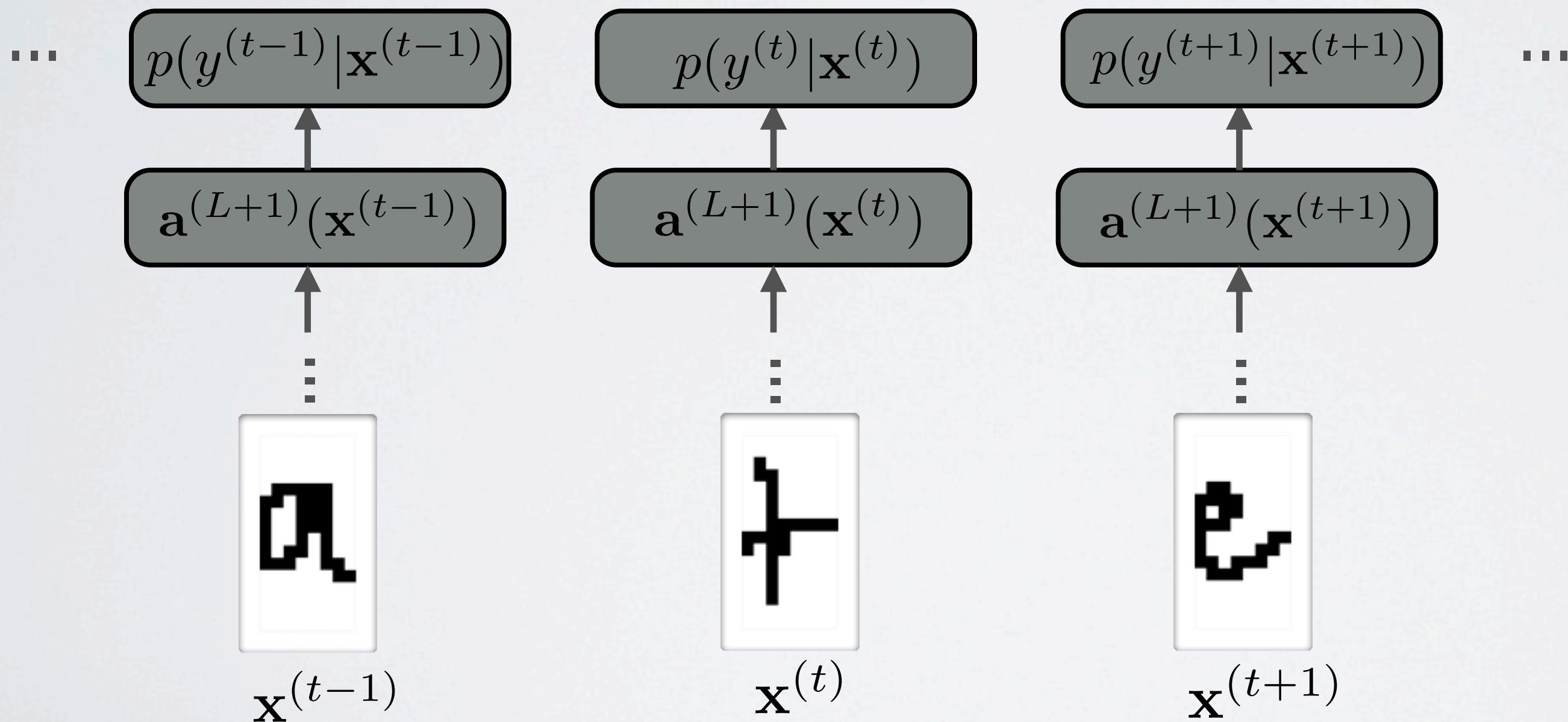
# Neural networks

Conditional random fields - motivation

# CONDITIONAL RANDOM FIELD

**Topics:** sequence classification

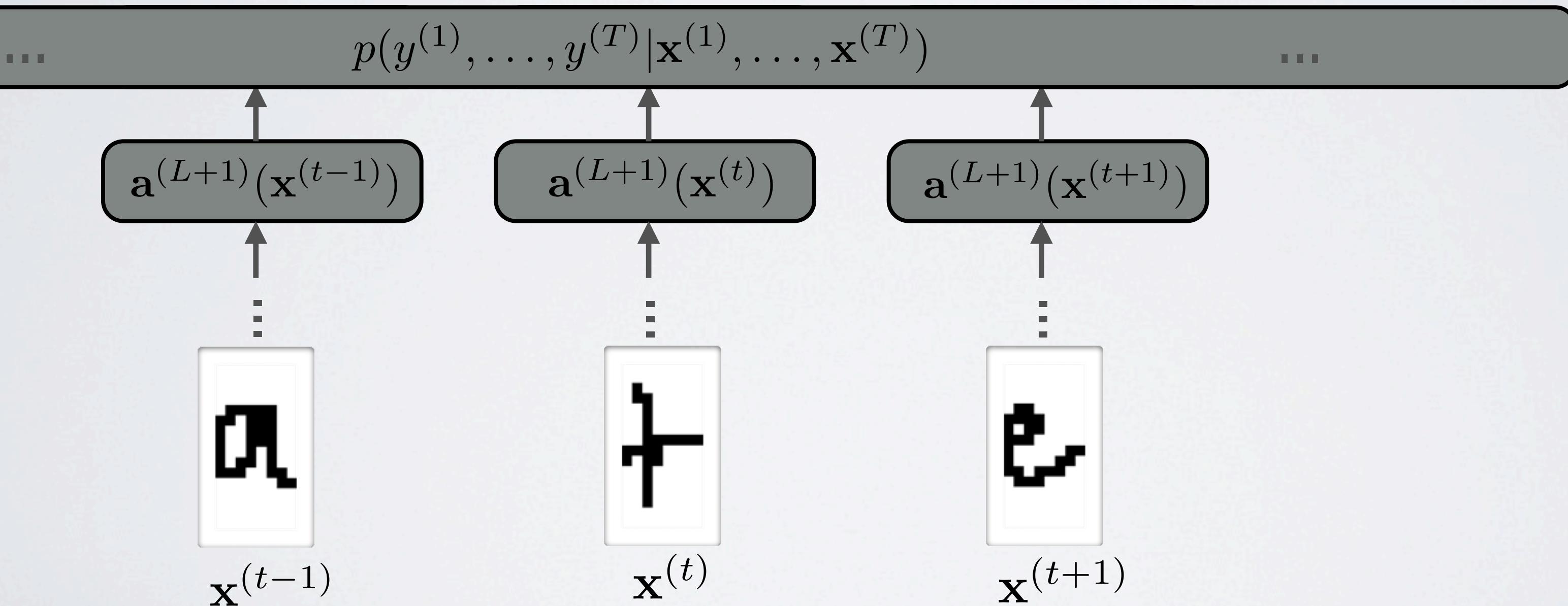
- What if examples organized in a sequence



# CONDITIONAL RANDOM FIELD

**Topics:** sequence classification

- What if examples organized in a sequence



# NOTATION

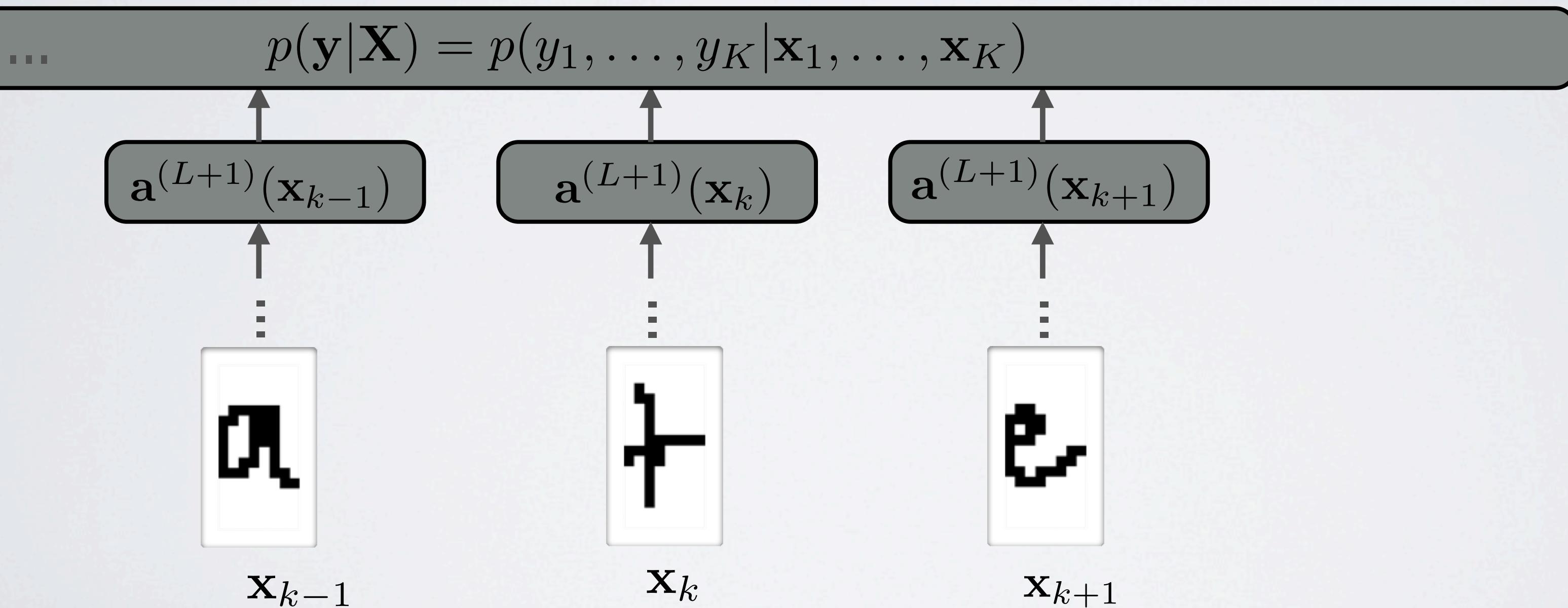
**Topics:** notation for inputs and targets

- Training set  $\{(\mathbf{X}^{(t)}, \mathbf{y}^{(t)})\}$  is a set of input and target sequences pairs:
  - inputs are  $\mathbf{X}^{(t)} = [\mathbf{x}_1^{(t)}, \dots, \mathbf{x}_{K_t}^{(t)}]$
  - targets are  $\mathbf{y}^{(t)} = [y_1^{(t)}, \dots, y_{K_t}^{(t)}]$
  - $K_t$  is the length of the  $t^{\text{th}}$  sequence

# CONDITIONAL RANDOM FIELD

**Topics:** sequence classification

- For a given example  $(\mathbf{X}, \mathbf{y})$ :



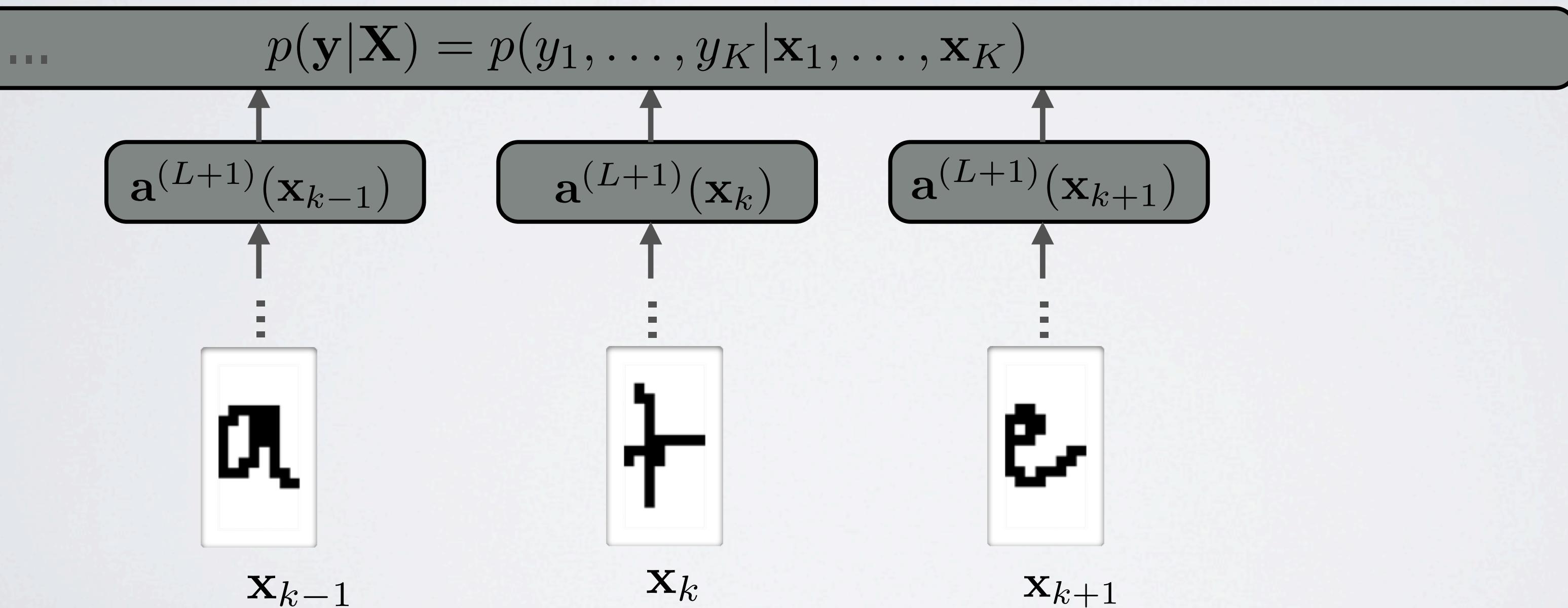
# Neural networks

Conditional random fields - linear chain CRF

# CONDITIONAL RANDOM FIELD

**Topics:** sequence classification

- For a given example  $(\mathbf{X}, \mathbf{y})$ :



# LINEAR CHAN CRF

**Topics:** lateral weights

- Regular classification:

$$\begin{aligned}
 p(\mathbf{y}|\mathbf{X}) &= \prod_k p(y_k|\mathbf{x}_k) = \prod_k \exp(a^{(L+1)}(\mathbf{x}_k)_{y_k}) / Z(\mathbf{x}_k) \\
 &= \exp\left(\sum_k a^{(L+1)}(\mathbf{x}_k)_{y_k}\right) / \left(\prod_k Z(\mathbf{x}_k)\right)
 \end{aligned}$$

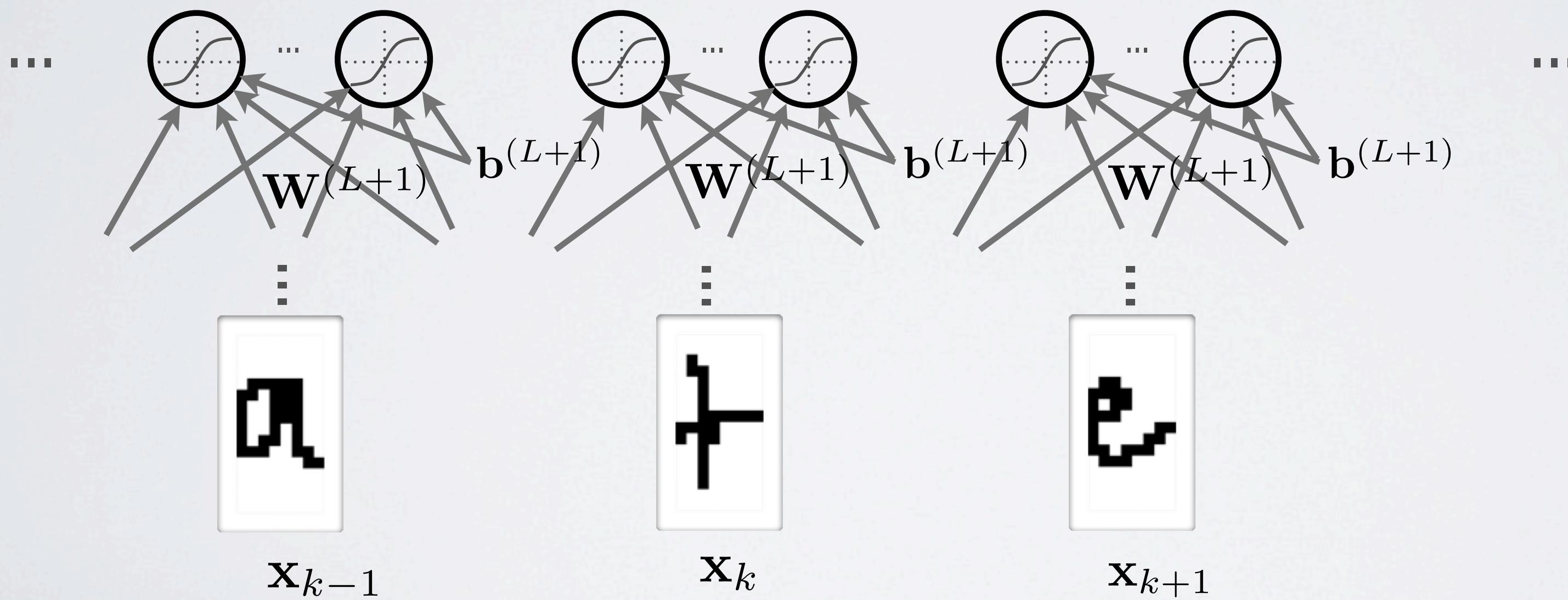
- Sequence classification with linear chain:

$$p(\mathbf{y}|\mathbf{X}) = \exp\left(\underbrace{\sum_{k=1}^K a^{(L+1)}(\mathbf{x}_k)_{y_k}}_{\text{is } y_k \text{ likely given input?}} + \underbrace{\sum_{k=1}^{K-1} V_{y_k, y_{k+1}}}_{\text{is } y_k \text{ followed by } y_{k+1} \text{ likely?}}\right) / \underbrace{Z(\mathbf{X})}_{\text{partition function}}$$

# LINEAR CHAN CRF

**Topics:** lateral weights

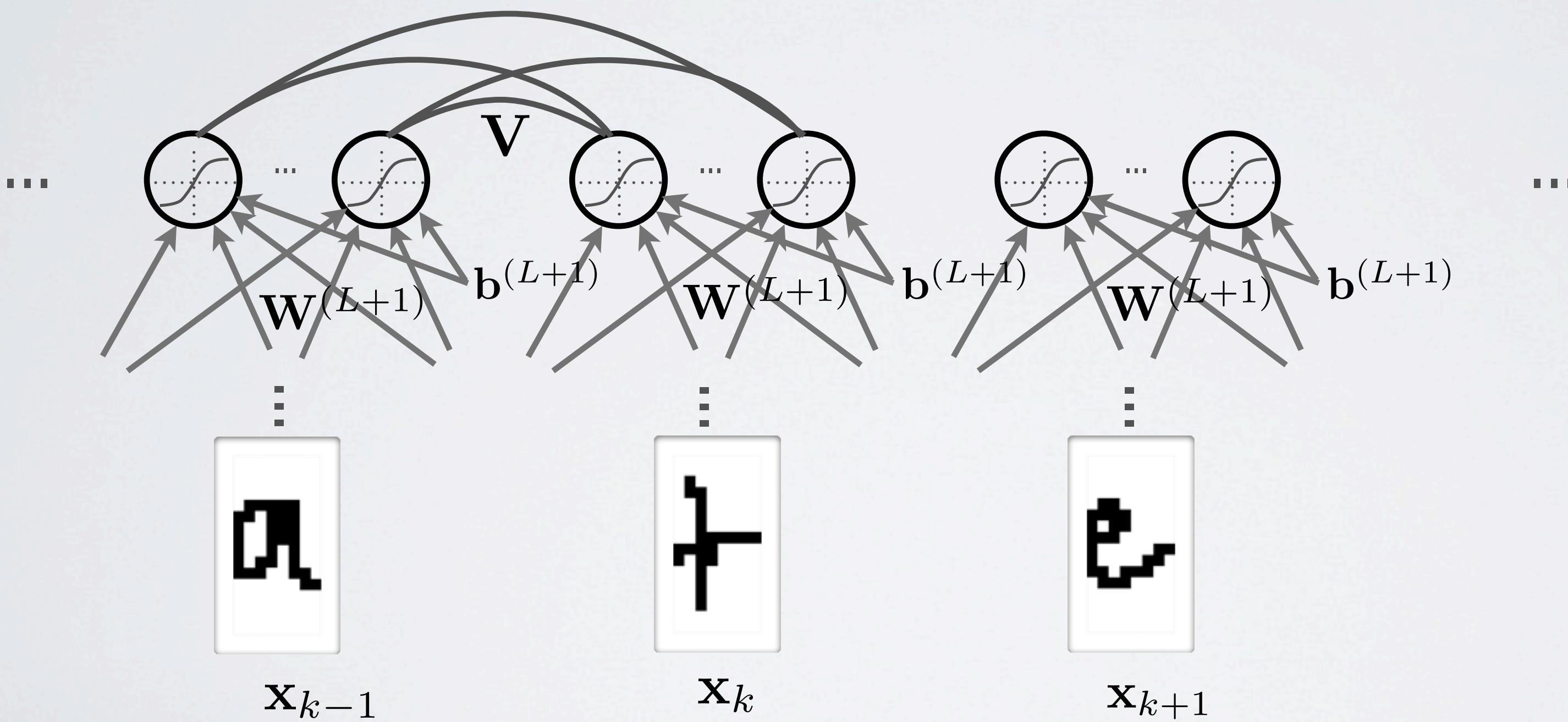
- Sequence classification with linear chain:



# LINEAR CHAN CRF

**Topics:** lateral weights

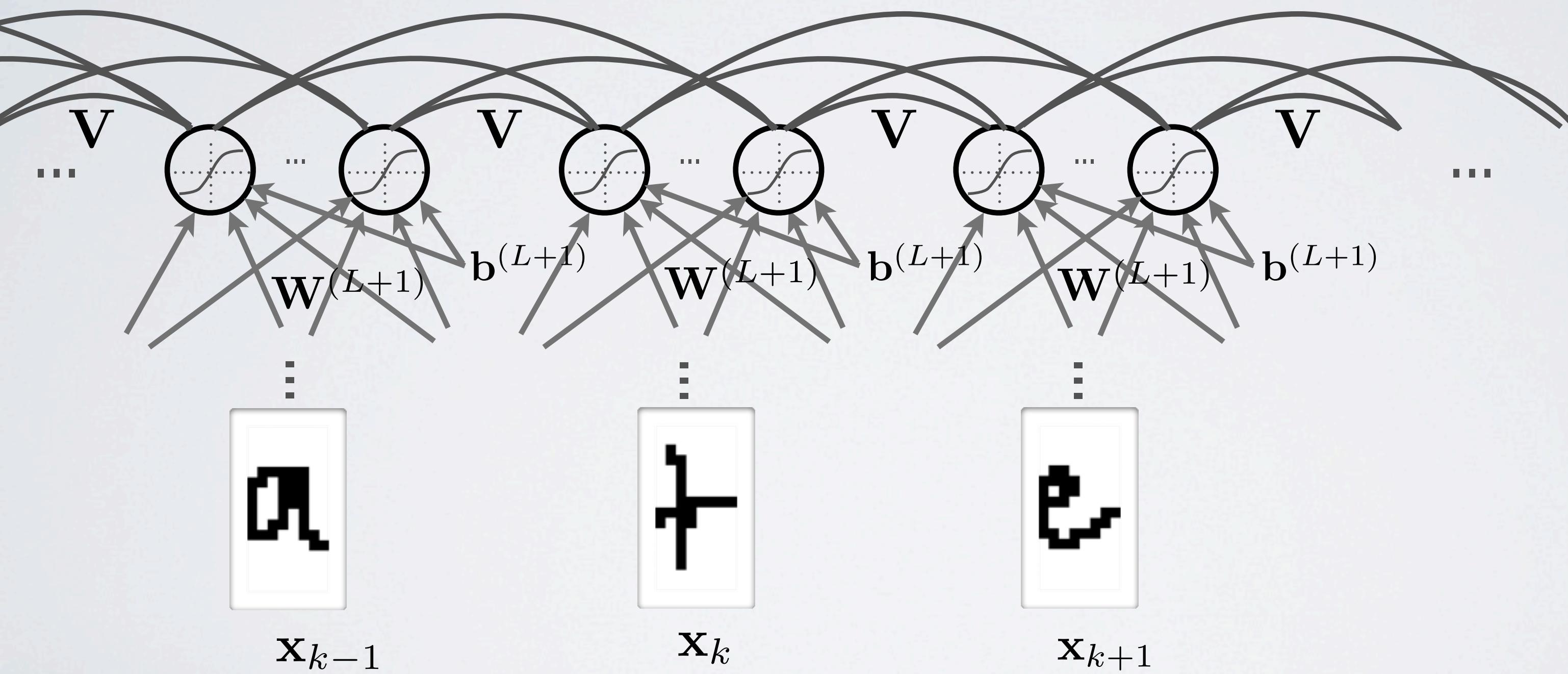
- Sequence classification with linear chain:



# LINEAR CHAN CRF

**Topics:** lateral weights

- Sequence classification with linear chain:



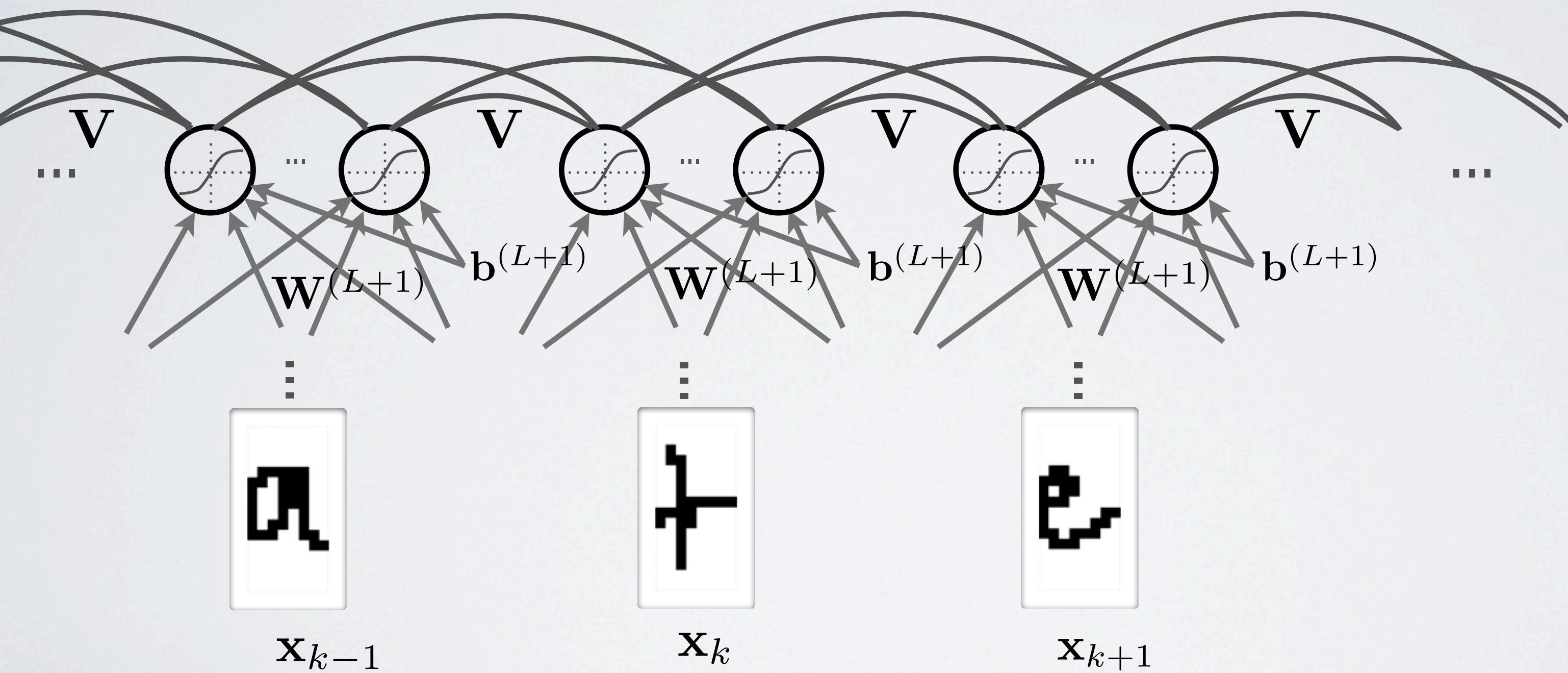
# Neural networks

Conditional random fields - context window

# LINEAR CHAN CRF

**Topics:** lateral weights

- Sequence classification with linear chain:



# LINEAR CHAN CRF

## Topics: context window

- Could incorporate a context window to the prediction at each position
  - ▶ e.g. context window of radius 1

$$p(\mathbf{y}|\mathbf{X}) = \exp \left( \sum_{k=1}^K a^{(L+1,0)}(\mathbf{x}_k)_{y_k} + \sum_{k=1}^{K-1} V_{y_k, y_{k+1}} + \right) / Z(\mathbf{X})$$

# LINEAR CHAN CRF

## Topics: context window

- Could incorporate a context window to the prediction at each position
  - ▶ e.g. context window of radius 1

$$\begin{aligned}
 p(\mathbf{y}|\mathbf{X}) = & \exp \left( \sum_{k=1}^K a^{(L+1,0)}(\mathbf{x}_k)_{y_k} + \sum_{k=1}^{K-1} V_{y_k, y_{k+1}} + \right. \\
 & \left. \sum_{k=2}^K a^{(L+1,-1)}(\mathbf{x}_{k-1})_{y_k} + \sum_{k=1}^{K-1} a^{(L+1,+1)}(\mathbf{x}_{k+1})_{y_k} \right) / Z(\mathbf{X})
 \end{aligned}$$

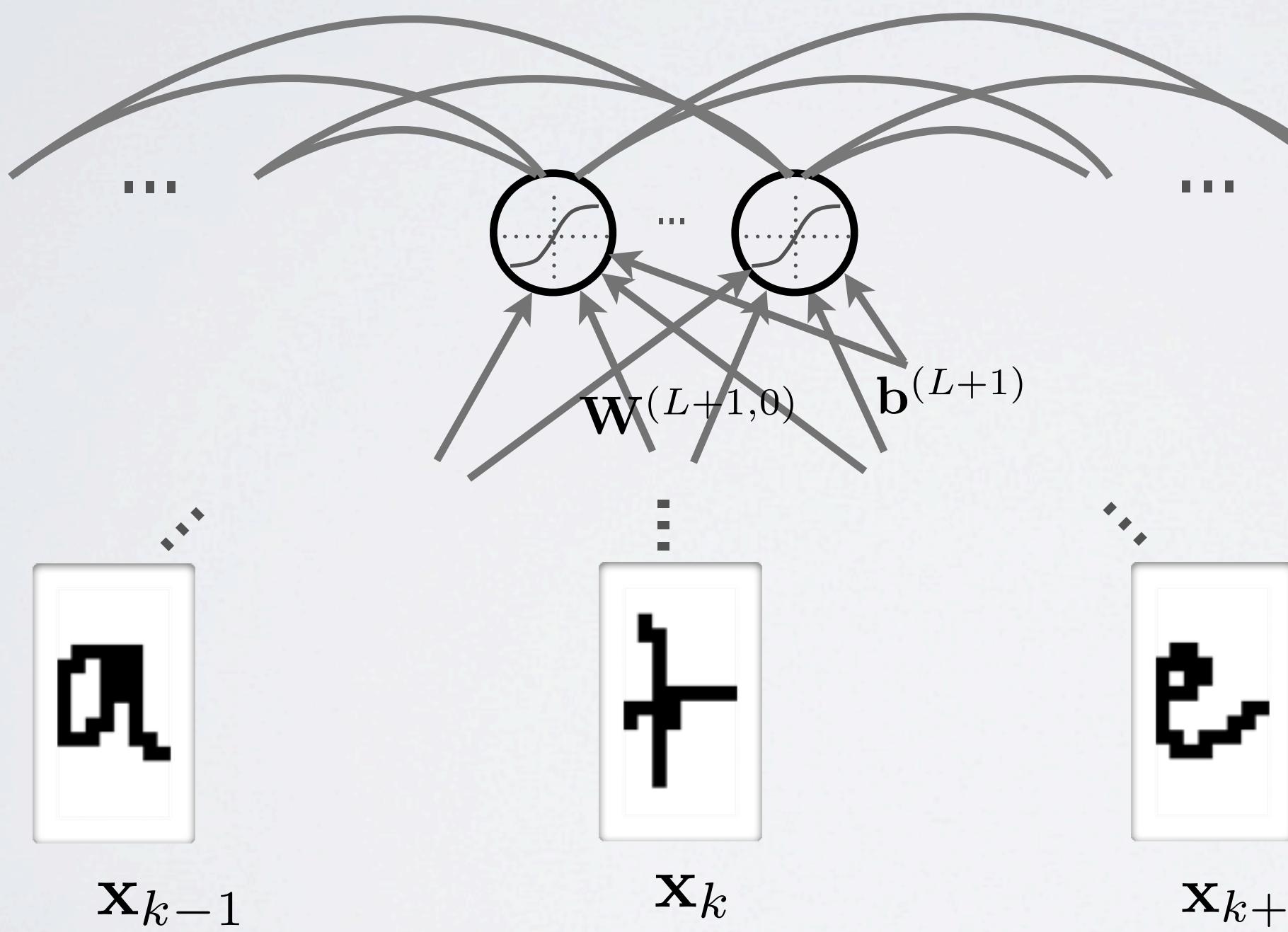
is  $y_k$  likely given input  
 on the left ?

is  $y_k$  likely given input  
 on the right ?

# LINEAR CHAN CRF

**Topics:** context window

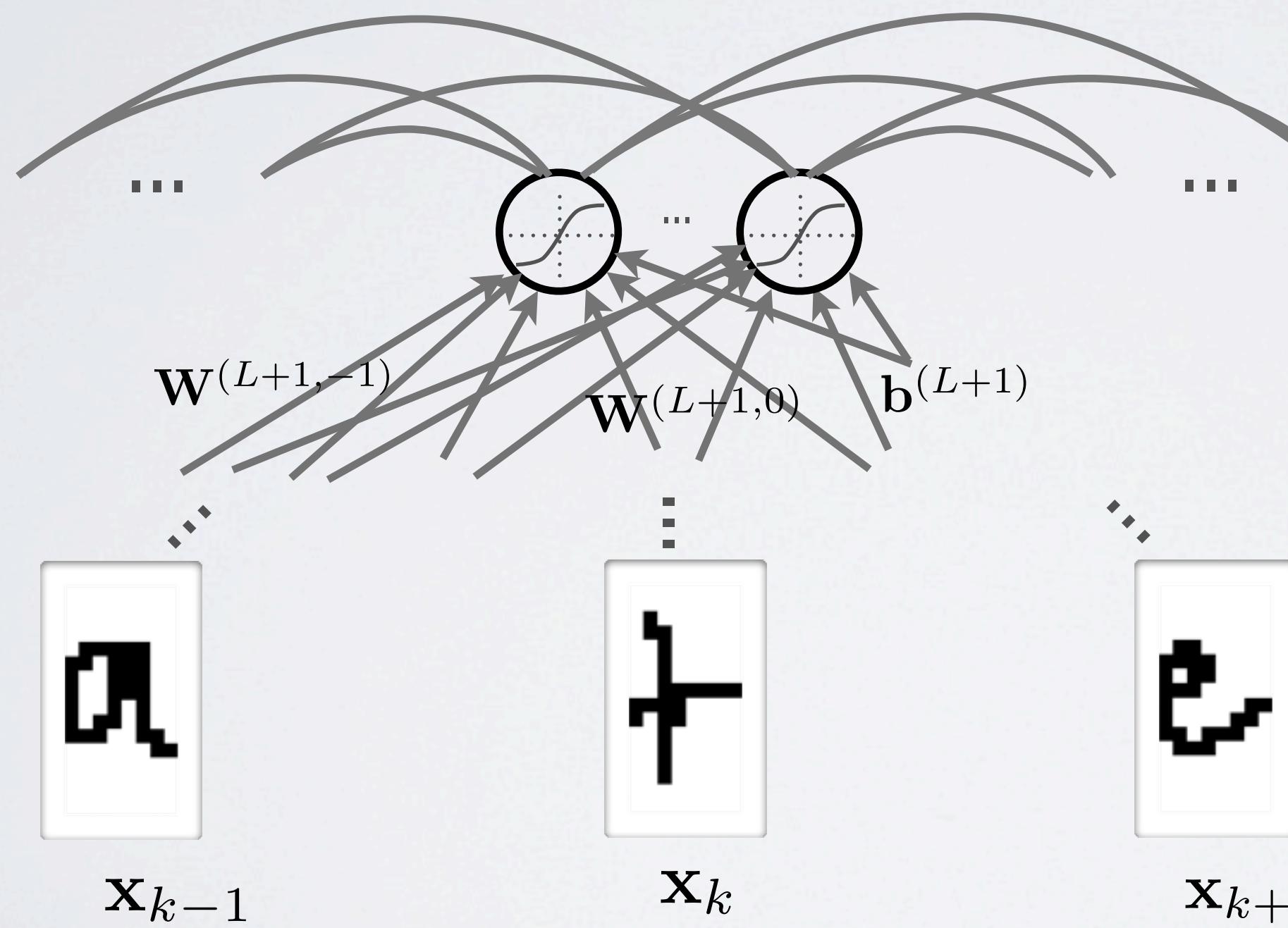
- Sequence classification with linear chain:



# LINEAR CHAN CRF

**Topics:** context window

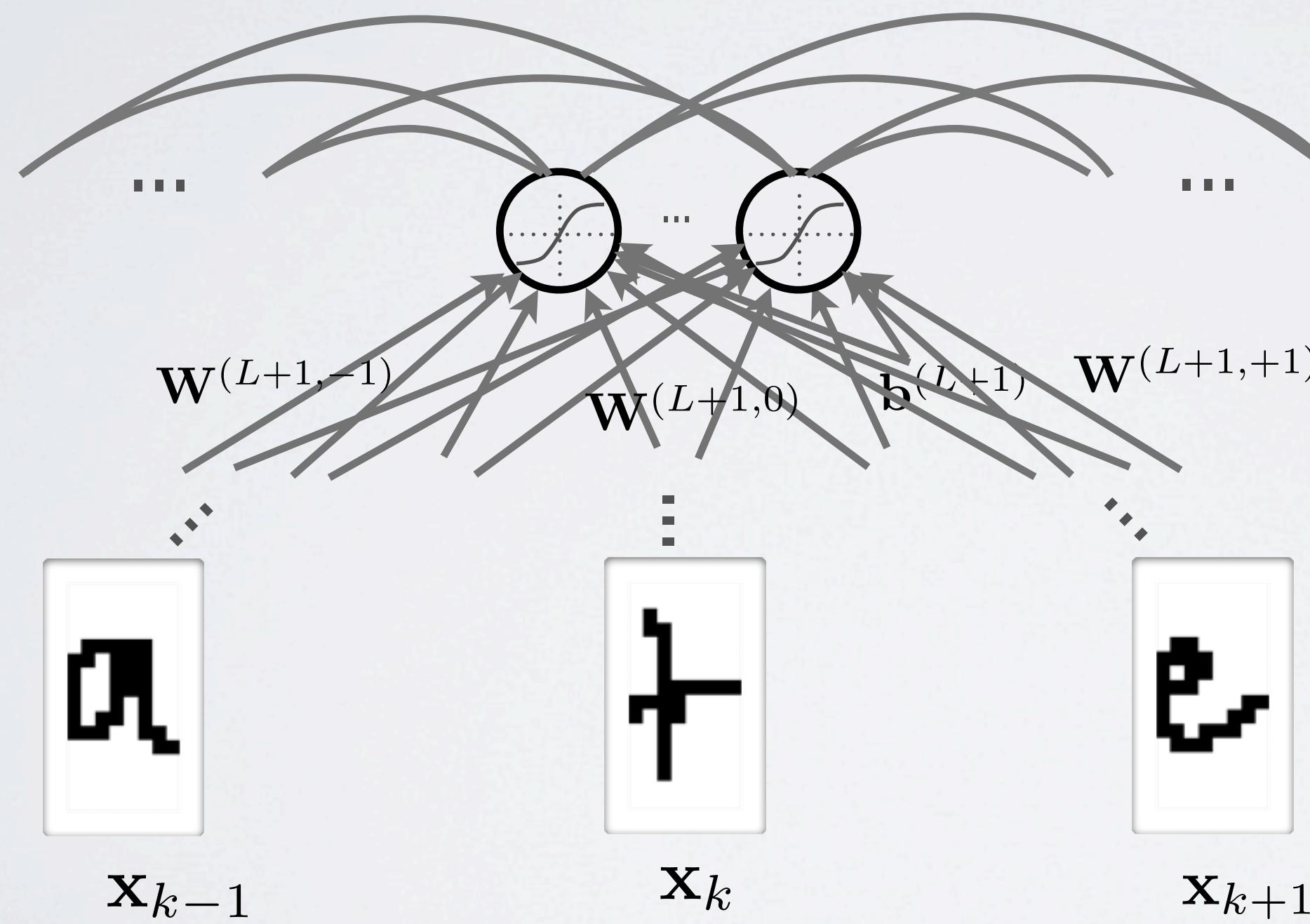
- Sequence classification with linear chain:



# LINEAR CHAN CRF

**Topics:** context window

- Sequence classification with linear chain:



# LINEAR CHAN CRF

**Topics:** context window

- Could instead feed the window to a single neural network
  - ▶ neural network can learn about the whole context jointly

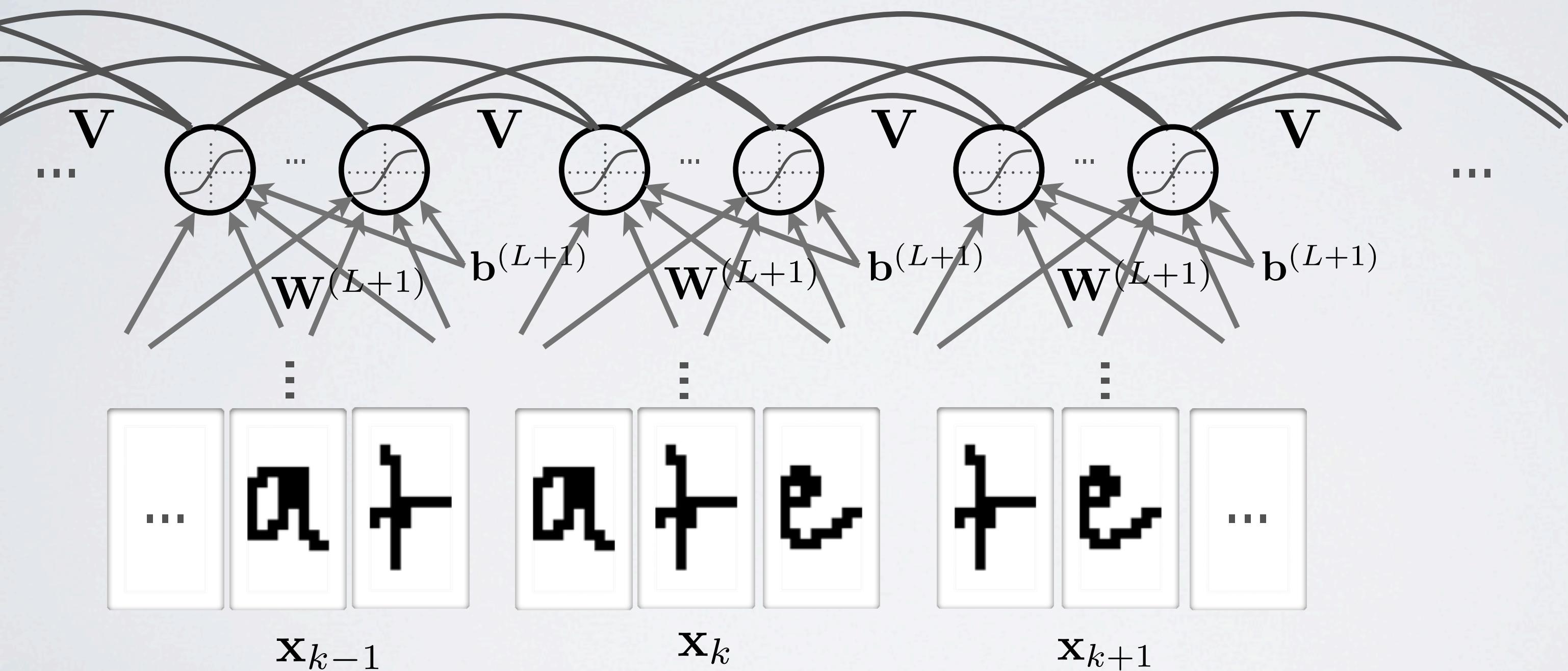
$$p(\mathbf{y}|\mathbf{X}) = \exp \left( \sum_{k=1}^K a^{(L+1)}(\mathbf{x}_{k-1}, \mathbf{x}_k, \mathbf{x}_{k+1}) y_k + \sum_{k=1}^{K-1} V_{y_k, y_{k+1}} \right) / Z(\mathbf{X})$$

where  $\mathbf{x}_0 = 0$  and  $\mathbf{x}_{K+1} = 0$  (or some chosen special vectors that indicate beginning/end of sequences)

# LINEAR CHAN CRF

**Topics:** context window

- Sequence classification with linear chain:



# LINEAR CHAN CRF

**Topics:** unary and pairwise log-factors

- For brevity, let's assume this notation:

- ▶ unary log-factors

$$a_u(y_k) = a^{(L+1,0)}(\mathbf{x}_k)_{y_k} + \mathbf{1}_{k>1} a^{(L+1,-1)}(\mathbf{x}_{k-1})_{y_k} + \mathbf{1}_{k<K} a^{(L+1,+1)}(\mathbf{x}_{k+1})_{y_k}$$

or

$$a_u(y_k) = a^{(L+1)}(\mathbf{x}_{k-1}, \mathbf{x}_k, \mathbf{x}_{k+1})_{y_k}$$

- ▶ pairwise log-factors

$$a_p(y_k, y_{k+1}) = \mathbf{1}_{1 \leq k < K} V_{y_k, y_{k+1}}$$

- Then we have:

$$p(\mathbf{y}|\mathbf{X}) = \exp \left( \sum_{k=1}^K a_u(y_k) + \sum_{k=1}^{K-1} a_p(y_k, y_{k+1}) \right) / Z(\mathbf{X})$$

# Neural networks

Conditional random fields - computing the partition function

# LINEAR CHAN CRF

**Topics:** unary and pairwise log-factors

- For brevity, let's assume this notation:

- ▶ unary log-factors

$$a_u(y_k) = a^{(L+1,0)}(\mathbf{x}_k)_{y_k} + \mathbf{1}_{k>1} a^{(L+1,-1)}(\mathbf{x}_{k-1})_{y_k} + \mathbf{1}_{k<K} a^{(L+1,+1)}(\mathbf{x}_{k+1})_{y_k}$$

or

$$a_u(y_k) = a^{(L+1)}(\mathbf{x}_{k-1}, \mathbf{x}_k, \mathbf{x}_{k+1})_{y_k}$$

- ▶ pairwise log-factors

$$a_p(y_k, y_{k+1}) = \mathbf{1}_{1 \leq k < K} V_{y_k, y_{k+1}}$$

- Then we have:

$$p(\mathbf{y}|\mathbf{X}) = \exp \left( \sum_{k=1}^K a_u(y_k) + \sum_{k=1}^{K-1} a_p(y_k, y_{k+1}) \right) / Z(\mathbf{X})$$

# INFERENCE

**Topics:** computing  $p(\mathbf{y}|\mathbf{X})$

- Then we have:

$$p(\mathbf{y}|\mathbf{X}) = \exp \left( \sum_{k=1}^K a_u(y_k) + \sum_{k=1}^{K-1} a_p(y_k, y_{k+1}) \right) / Z(\mathbf{X})$$

where

$$Z(\mathbf{X}) = \sum_{y'_1} \sum_{y'_2} \cdots \sum_{y'_K} \exp \left( \sum_{k=1}^K a_u(y'_k) + \sum_{k=1}^{K-1} a_p(y'_k, y'_{k+1}) \right)$$



hard to compute

# INFERENCE

**Topics:** computing  $p(\mathbf{y}|\mathbf{X})$

$$Z(\mathbf{X}) = \sum_{y'_K} \exp(a_u(y'_K))$$

$$\left( \sum_{y'_{K-1}} \exp(a_u(y'_{K-1}) + a_p(y'_{K-1}, y'_K)) \right)$$

...

$$\left( \sum_{y'_2} \exp(a_u(y'_2) + a_p(y'_2, y'_3)) \right) \left( \sum_{y'_1} \exp(a_u(y'_1) + a_p(y'_1, y'_2)) \right) \dots$$

# INFERENCE

**Topics:** computing  $p(\mathbf{y}|\mathbf{X})$

$$Z(\mathbf{X}) = \sum_{y'_K} \exp(a_u(y'_K))$$

$$\left( \sum_{y'_{K-1}} \exp(a_u(y'_{K-1}) + a_p(y'_{K-1}, y'_K)) \right)$$

...

$$\left( \sum_{y'_2} \exp(a_u(y'_2) + a_p(y'_2, y'_3)) \right)$$

$$\alpha_1(y'_2) \left( \sum_{y'_1} \exp(a_u(y'_1) + a_p(y'_1, y'_2)) \right) \dots \right)$$

# INFERENCE

**Topics:** computing  $p(\mathbf{y}|\mathbf{X})$

$$Z(\mathbf{X}) = \sum_{y'_K} \exp(a_u(y'_K))$$

$$\left( \sum_{y'_{K-1}} \exp(a_u(y'_{K-1}) + a_p(y'_{K-1}, y'_K)) \right)$$

...

$$\alpha_2(y'_3) \left( \sum_{y'_2} \exp(a_u(y'_2) + a_p(y'_2, y'_3)) \right)$$

$$\alpha_1(y'_2) \left( \sum_{y'_1} \exp(a_u(y'_1) + a_p(y'_1, y'_2)) \right) \dots \right)$$

# INFERENCE

**Topics:** computing  $p(\mathbf{y}|\mathbf{X})$

$$Z(\mathbf{X}) = \sum_{y'_K} \exp(a_u(y'_K))$$

$$\alpha_{K-1}(y'_K) \left( \sum_{y'_{K-1}} \exp(a_u(y'_{K-1}) + a_p(y'_{K-1}, y'_K)) \right)$$

...

$$\alpha_2(y'_3) \left( \sum_{y'_2} \exp(a_u(y'_2) + a_p(y'_2, y'_3)) \right)$$

$$\alpha_1(y'_2) \left( \sum_{y'_1} \exp(a_u(y'_1) + a_p(y'_1, y'_2)) \right) \dots \right)$$

# INFERENCE

**Topics:** computing  $p(\mathbf{y}|\mathbf{X})$

$$Z(\mathbf{X}) = \sum_{y'_K} \exp(a_u(y'_K))$$

$$\alpha_{K-1}(y'_K) \left( \sum_{y'_{K-1}} \exp(a_u(y'_{K-1}) + a_p(y'_{K-1}, y'_K)) \right)$$

...

$$\alpha_2(y'_3) \left( \sum_{y'_2} \exp(a_u(y'_2) + a_p(y'_2, y'_3)) \right)$$

$$\alpha_1(y'_2) \left( \sum_{y'_1} \exp(a_u(y'_1) + a_p(y'_1, y'_2)) \right) \dots \right)$$

# INFERENCE

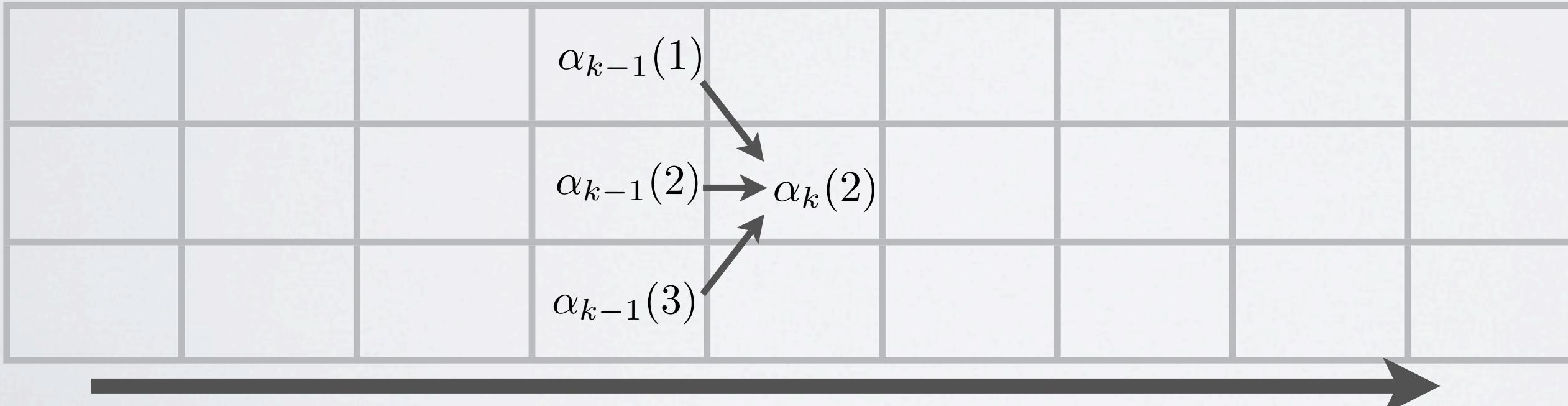
nb. of classes

**Topics:** computing  $p(\mathbf{y}|\mathbf{X})$

- Algorithm goes as follows:

- initialize, for all values of  $y'_2$  :  $\alpha_1(y'_2) \leftarrow \sum_{y'_1} \exp(a_u(y'_1) + a_p(y'_1, y'_2))$
- for  $k = 2$  to  $K-1$ , for all values of  $y'_{k+1}$  :
  - $\alpha_k(y'_{k+1}) \leftarrow \sum_{y'_k} \exp(a_u(y'_k) + a_p(y'_k, y'_{k+1})) \alpha_{k-1}(y'_k)$
- $Z(\mathbf{X}) \leftarrow \sum_{y'_K} \exp(a_u(y'_K)) \alpha_{K-1}(y'_K)$

Complexity in  $O(KC^2)$



# INFERENCE

**Topics:** computing  $p(\mathbf{y}|\mathbf{X})$

$$\begin{aligned}
 Z(\mathbf{X}) &= \sum_{y'_1} \exp(a_u(y'_1)) \\
 &\quad \left( \sum_{y'_2} \exp(a_u(y'_2) + a_p(y'_1, y'_2)) \right. \\
 &\quad \dots \\
 &\quad \left. \left( \sum_{y'_{K-1}} \exp(a_u(y'_{K-1}) + a_p(y'_{K-2}, y'_{K-1})) \right. \right. \\
 &\quad \left. \left. \left( \sum_{y'_K} \exp(a_u(y'_K) + a_p(y'_{K-1}, y'_K)) \right) \right) \dots \right)
 \end{aligned}$$

# INFERENCE

**Topics:** computing  $p(\mathbf{y}|\mathbf{X})$

$$\begin{aligned}
 Z(\mathbf{X}) &= \sum_{y'_1} \exp(a_u(y'_1)) \\
 &\quad \left( \sum_{y'_2} \exp(a_u(y'_2) + a_p(y'_1, y'_2)) \right. \\
 &\quad \dots \\
 &\quad \left. \left( \sum_{y'_{K-1}} \exp(a_u(y'_{K-1}) + a_p(y'_{K-2}, y'_{K-1})) \right. \right. \\
 &\quad \beta_K(y'_{K-1}) \left. \left( \sum_{y'_K} \exp(a_u(y'_K) + a_p(y'_{K-1}, y'_K)) \right) \right) \dots
 \end{aligned}$$

# INFERENCE

**Topics:** computing  $p(\mathbf{y}|\mathbf{X})$

$$\begin{aligned} Z(\mathbf{X}) &= \sum_{y'_1} \exp(a_u(y'_1)) \\ &\quad \left( \sum_{y'_2} \exp(a_u(y'_2) + a_p(y'_1, y'_2)) \right. \\ &\quad \quad \quad \dots \end{aligned}$$

$$\begin{aligned} &\beta_{K-1}(y'_{K-2}) \left( \sum_{y'_{K-1}} \exp(a_u(y'_{K-1}) + a_p(y'_{K-2}, y'_{K-1})) \right. \\ &\quad \left. \beta_K(y'_{K-1}) \left( \sum_{y'_K} \exp(a_u(y'_K) + a_p(y'_{K-1}, y'_K)) \right) \right) \dots \end{aligned}$$

# INFERENCE

**Topics:** computing  $p(\mathbf{y}|\mathbf{X})$

$$Z(\mathbf{X}) = \sum_{y'_1} \exp(a_u(y'_1))$$

$$\beta_2(y'_1) \left( \sum_{y'_2} \exp(a_u(y'_2) + a_p(y'_1, y'_2)) \right)$$

...

$$\beta_{K-1}(y'_{K-2}) \left( \sum_{y'_{K-1}} \exp(a_u(y'_{K-1}) + a_p(y'_{K-2}, y'_{K-1})) \right)$$

$$\beta_K(y'_{K-1}) \left( \sum_{y'_K} \exp(a_u(y'_K) + a_p(y'_{K-1}, y'_K)) \right) \dots \right)$$

# INFERENCE

**Topics:** computing  $p(\mathbf{y}|\mathbf{X})$

$$Z(\mathbf{X}) = \sum_{y'_1} \exp(a_u(y'_1))$$

$$\beta_2(y'_1) \left( \sum_{y'_2} \exp(a_u(y'_2) + a_p(y'_1, y'_2)) \right)$$

...

$$\beta_{K-1}(y'_{K-2}) \left( \sum_{y'_{K-1}} \exp(a_u(y'_{K-1}) + a_p(y'_{K-2}, y'_{K-1})) \right)$$

$$\beta_K(y'_{K-1}) \left( \sum_{y'_K} \exp(a_u(y'_K) + a_p(y'_{K-1}, y'_K)) \right) \dots \right)$$

# INFERENCE

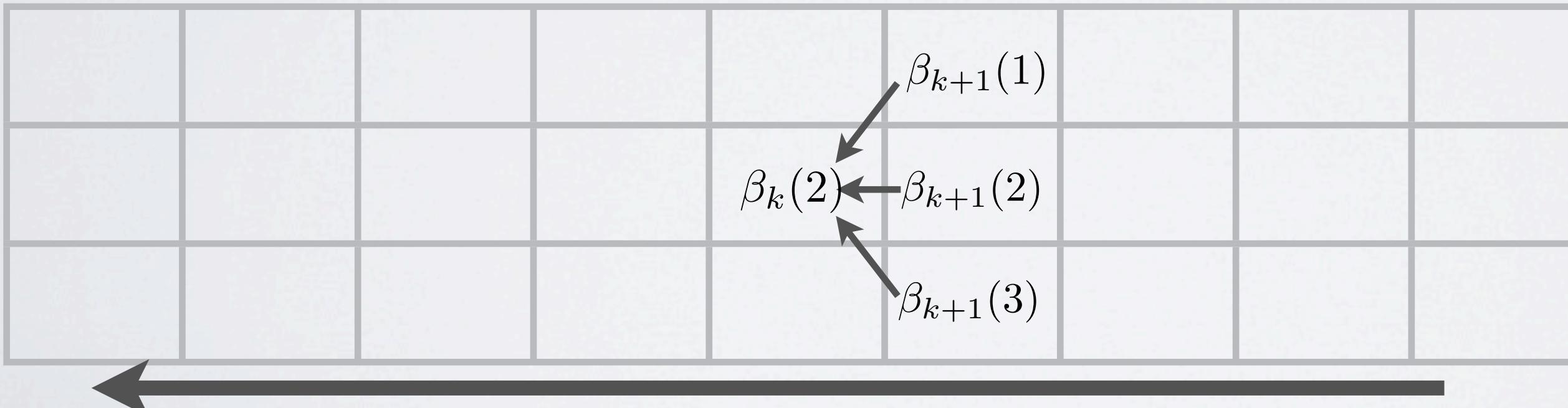
**Topics:** computing  $p(\mathbf{y}|\mathbf{X})$

- Algorithm goes as follows:

- initialize, for all values of  $y'_{K-1}$ :  $\beta_K(y'_{K-1}) \Leftarrow \sum_{y'_K} \exp(a_u(y'_K) + a_p(y'_{K-1}, y'_K))$
- for  $k = K-1$  to 2, for all values of  $y'_{k-1}$ :
  - $\beta_k(y'_{k-1}) \Leftarrow \sum_{y'_k} \exp(a_u(y'_k) + a_p(y'_{k-1}, y'_k)) \beta_{k+1}(y'_k)$
- $Z(\mathbf{X}) \Leftarrow \sum_{y'_1} \exp(a_u(y'_1)) \beta_2(y'_1)$

nb. of classes

Complexity in  $O(KC^2)$



# INFERENCE

**Topics:** forward/backward or belief propagation

- Computing both tables is often referred to as the forward/backward algorithm for CRFs
  - ▶  $\alpha$  is computed with a forward pass
  - ▶  $\beta$  is computed with a backward pass
- It has other names
  - ▶ belief propagation
  - ▶ sum-product
- $\alpha$  gives the summation from the left
- $\beta$  gives the summation from the right

# INFERENCE

**Topics:** stable implementation of belief propagation

- For a stable implementation, should work in log space

$$\log \alpha_k(y'_{k+1}) \Leftarrow \log \sum_{y'_k} \exp(a_u(y'_k) + a_p(y'_k, y'_{k+1}) + \log \alpha_{k-1}(y'_k))$$

$$\log \beta_k(y'_{k-1}) \Leftarrow \log \sum_{y'_k} \exp(a_u(y'_k) + a_p(y'_{k-1}, y'_k) + \log \beta_{k+1}(y'_k))$$

- Log-sum-exp operations are more stable if computed like this:

$$\log \sum_i \exp(z_i) = \underbrace{\max_i(z_i) + \log \sum_i \exp(z_i - \max_i(z_i))}_{\text{numerically stable}}$$

# Neural networks

Conditional random fields - computing marginals

# INFERENCE

**Topics:** computing  $p(\mathbf{y}|\mathbf{X})$

- Then we have:

$$p(\mathbf{y}|\mathbf{X}) = \exp \left( \sum_{k=1}^K a_u(y_k) + \sum_{k=1}^{K-1} a_p(y_k, y_{k+1}) \right) / Z(\mathbf{X})$$

where

$$Z(\mathbf{X}) = \sum_{y'_1} \sum_{y'_2} \cdots \sum_{y'_K} \exp \left( \sum_{k=1}^K a_u(y'_k) + \sum_{k=1}^{K-1} a_p(y'_k, y'_{k+1}) \right)$$



hard to compute

# INFERENCE

**Topics:** forward/backward or belief propagation

- Computing both tables is often referred to as the forward/backward algorithm for CRFs
  - ▶  $\alpha$  is computed with a forward pass
  - ▶  $\beta$  is computed with a backward pass
- It has other names
  - ▶ belief propagation
  - ▶ sum-product
- $\alpha$  gives the summation from the left
- $\beta$  gives the summation from the right

# INFERENCE

**Topics:** computing  $p(y_k|\mathbf{X})$ ,  $p(y_k, y_{k+1}|\mathbf{X})$

- The  $\alpha / \beta$  tables can be used to compute marginals

$$p(y_k|\mathbf{X}) = \frac{\exp(a_u(y_k) + \log \alpha_{k-1}(y_k) + \log \beta_{k+1}(y_k))}{\sum_{y'_k} \exp(a_u(y'_k) + \log \alpha_{k-1}(y'_k) + \log \beta_{k+1}(y'_k))}$$

$$p(y_k, y_{k+1}|\mathbf{X}) = \frac{\exp \left( \begin{array}{l} a_u(y_k) + a_p(y_k, y_{k+1}) + a_u(y_{k+1}) \\ + \log \alpha_{k-1}(y_k) + \log \beta_{k+2}(y_{k+1}) \end{array} \right)}{\sum_{y'_k} \sum_{y'_{k+1}} \exp \left( \begin{array}{l} a_u(y'_k) + a_p(y'_k, y'_{k+1}) + a_u(y'_{k+1}) \\ + \log \alpha_{k-1}(y'_k) + \log \beta_{k+2}(y'_{k+1}) \end{array} \right)}$$

# Neural networks

Conditional random fields - performing classification

# INFERENCE

**Topics:** computing  $p(\mathbf{y}|\mathbf{X})$

- Then we have:

$$p(\mathbf{y}|\mathbf{X}) = \exp \left( \sum_{k=1}^K a_u(y_k) + \sum_{k=1}^{K-1} a_p(y_k, y_{k+1}) \right) / Z(\mathbf{X})$$

where

$$Z(\mathbf{X}) = \sum_{y'_1} \sum_{y'_2} \cdots \sum_{y'_K} \exp \left( \sum_{k=1}^K a_u(y'_k) + \sum_{k=1}^{K-1} a_p(y'_k, y'_{k+1}) \right)$$



hard to compute

# INFERENCE

**Topics:** computing  $p(y_k|\mathbf{X})$ ,  $p(y_k, y_{k+1}|\mathbf{X})$

- The  $\alpha / \beta$  tables can be used to compute marginals

$$p(y_k|\mathbf{X}) = \frac{\exp(a_u(y_k) + \log \alpha_{k-1}(y_k) + \log \beta_{k+1}(y_k))}{\sum_{y'_k} \exp(a_u(y'_k) + \log \alpha_{k-1}(y'_k) + \log \beta_{k+1}(y'_k))}$$

$$p(y_k, y_{k+1}|\mathbf{X}) = \frac{\exp \left( \begin{array}{l} a_u(y_k) + a_p(y_k, y_{k+1}) + a_u(y_{k+1}) \\ + \log \alpha_{k-1}(y_k) + \log \beta_{k+2}(y_{k+1}) \end{array} \right)}{\sum_{y'_k} \sum_{y'_{k+1}} \exp \left( \begin{array}{l} a_u(y'_k) + a_p(y'_k, y'_{k+1}) + a_u(y'_{k+1}) \\ + \log \alpha_{k-1}(y'_k) + \log \beta_{k+2}(y'_{k+1}) \end{array} \right)}$$

# CLASSIFICATION

**Topics:** making a prediction (option 1)

- At each position  $k$ , pick label  $y_k$  with highest marginal probability  $p(y_k | \mathbf{X})$ 
  - ▶ this choice is the one that minimizes the sum of the classification errors over the whole sequence, assuming the CRF is the true distribution

$$\min_{\mathbf{y}^*} \mathbb{E} \left[ \sum_k \mathbf{1}_{y_k \neq y_k^*} \right]$$

$$\min_{\mathbf{y}^*} \mathrm{E}\left[\sum_k 1_{y_k \neq y_k^*}\right]$$

$$= \min_{\mathbf{y}^*} \sum_{y_1} \cdots \sum_{y_K} \left( \sum_k 1_{y_k \neq y_k^*} \right) p(\mathbf{y} | \mathbf{X})$$


---



---



---

$$\min_{\mathbf{y}^*} \mathbb{E} \left[ \sum_k 1_{y_k \neq y_k^*} \right]$$

$$= \min_{\mathbf{y}^*} \sum_{y_1} \cdots \sum_{y_K} \left( \sum_k 1_{y_k \neq y_k^*} \right) p(\mathbf{y} | \mathbf{X})$$

$$= \min_{y_1^*} \dots \min_{y_K^*} \sum_{y_1} \cdots \sum_{y_K} \left( \sum_k 1_{y_k \neq y_k^*} \right) p(\mathbf{y} | \mathbf{X})$$

$$\min_{\mathbf{y}^*} \mathbb{E} \left[ \sum_k 1_{y_k \neq y_k^*} \right]$$

$$= \min_{\mathbf{y}^*} \sum_{y_1} \cdots \sum_{y_K} \left( \sum_k 1_{y_k \neq y_k^*} \right) p(\mathbf{y} | \mathbf{X})$$

$$= \min_{y_1^*} \dots \min_{y_K^*} \sum_{y_1} \cdots \sum_{y_K} \left( \sum_k 1_{y_k \neq y_k^*} \right) p(\mathbf{y} | \mathbf{X})$$

$$= \min_{y_1^*} \dots \min_{y_K^*} \sum_k \sum_{y_1} \cdots \sum_{y_K} 1_{y_k \neq y_k^*} p(\mathbf{y} | \mathbf{X})$$

$$\begin{aligned}
& \min_{\mathbf{y}^*} \mathbb{E} \left[ \sum_k 1_{y_k \neq y_k^*} \right] \\
&= \min_{\mathbf{y}^*} \sum_{y_1} \cdots \sum_{y_K} \left( \sum_k 1_{y_k \neq y_k^*} \right) p(\mathbf{y} | \mathbf{X}) \\
&= \min_{y_1^*} \dots \min_{y_K^*} \sum_{y_1} \cdots \sum_{y_K} \left( \sum_k 1_{y_k \neq y_k^*} \right) p(\mathbf{y} | \mathbf{X}) \\
&= \min_{y_1^*} \dots \min_{y_K^*} \sum_k \sum_{y_1} \cdots \sum_{y_K} 1_{y_k \neq y_k^*} p(\mathbf{y} | \mathbf{X}) \\
&= \min_{y_1^*} \dots \min_{y_K^*} \sum_k \sum_{y_k} 1_{y_k \neq y_k^*} \sum_{y_1} \cdots \sum_{y_{k-1}} \sum_{y_{k+1}} \sum_{y_K} p(\mathbf{y} | \mathbf{X})
\end{aligned}$$

$$\begin{aligned}
& \min_{\mathbf{y}^*} \mathbb{E} \left[ \sum_k 1_{y_k \neq y_k^*} \right] \\
&= \min_{\mathbf{y}^*} \sum_{y_1} \cdots \sum_{y_K} \left( \sum_k 1_{y_k \neq y_k^*} \right) p(\mathbf{y} | \mathbf{X}) \\
&= \min_{y_1^*} \dots \min_{y_K^*} \sum_{y_1} \cdots \sum_{y_K} \left( \sum_k 1_{y_k \neq y_k^*} \right) p(\mathbf{y} | \mathbf{X}) \\
&= \min_{y_1^*} \dots \min_{y_K^*} \sum_k \sum_{y_1} \cdots \sum_{y_K} 1_{y_k \neq y_k^*} p(\mathbf{y} | \mathbf{X}) \\
&= \min_{y_1^*} \dots \min_{y_K^*} \sum_k \sum_{y_k} 1_{y_k \neq y_k^*} \sum_{y_1} \cdots \sum_{y_{k-1}} \sum_{y_{k+1}} \sum_{y_K} p(\mathbf{y} | \mathbf{X}) \\
&= \min_{y_1^*} \dots \min_{y_K^*} \sum_k \sum_{y_k} 1_{y_k \neq y_k^*} p(y_k | \mathbf{X})
\end{aligned}$$

$$\begin{aligned}
& \min_{\mathbf{y}^*} \mathbb{E} \left[ \sum_k 1_{y_k \neq y_k^*} \right] \\
&= \min_{\mathbf{y}^*} \sum_{y_1} \cdots \sum_{y_K} \left( \sum_k 1_{y_k \neq y_k^*} \right) p(\mathbf{y} | \mathbf{X}) \\
&= \min_{y_1^*} \dots \min_{y_K^*} \sum_{y_1} \cdots \sum_{y_K} \left( \sum_k 1_{y_k \neq y_k^*} \right) p(\mathbf{y} | \mathbf{X}) \\
&= \min_{y_1^*} \dots \min_{y_K^*} \sum_k \sum_{y_1} \cdots \sum_{y_K} 1_{y_k \neq y_k^*} p(\mathbf{y} | \mathbf{X}) \\
&= \min_{y_1^*} \dots \min_{y_K^*} \sum_k \sum_{y_k} 1_{y_k \neq y_k^*} \sum_{y_1} \cdots \sum_{y_{k-1}} \sum_{y_{k+1}} \sum_{y_K} p(\mathbf{y} | \mathbf{X}) \\
&= \min_{y_1^*} \dots \min_{y_K^*} \sum_k \sum_{y_k} 1_{y_k \neq y_k^*} p(y_k | \mathbf{X}) \\
&= \min_{y_1^*} \left( \sum_{y_1} 1_{y_1 \neq y_1^*} p(y_1 | \mathbf{X}) \right) + \cdots + \min_{y_K^*} \left( \sum_{y_K} 1_{y_K \neq y_K^*} p(y_K | \mathbf{X}) \right)
\end{aligned}$$

$$\begin{aligned}
& \min_{\mathbf{y}^*} \mathbb{E} \left[ \sum_k 1_{y_k \neq y_k^*} \right] \\
&= \min_{\mathbf{y}^*} \sum_{y_1} \cdots \sum_{y_K} \left( \sum_k 1_{y_k \neq y_k^*} \right) p(\mathbf{y} | \mathbf{X}) \\
&= \min_{y_1^*} \dots \min_{y_K^*} \sum_{y_1} \cdots \sum_{y_K} \left( \sum_k 1_{y_k \neq y_k^*} \right) p(\mathbf{y} | \mathbf{X}) \\
&= \min_{y_1^*} \dots \min_{y_K^*} \sum_k \sum_{y_1} \cdots \sum_{y_K} 1_{y_k \neq y_k^*} p(\mathbf{y} | \mathbf{X}) \\
&= \min_{y_1^*} \dots \min_{y_K^*} \sum_k \sum_{y_k} 1_{y_k \neq y_k^*} \sum_{y_1} \cdots \sum_{y_{k-1}} \sum_{y_{k+1}} \sum_{y_K} p(\mathbf{y} | \mathbf{X}) \\
&= \min_{y_1^*} \dots \min_{y_K^*} \sum_k \sum_{y_k} 1_{y_k \neq y_k^*} p(y_k | \mathbf{X}) \\
&= \min_{y_1^*} \left( \sum_{y_1} 1_{y_1 \neq y_1^*} p(y_1 | \mathbf{X}) \right) + \cdots + \min_{y_K^*} \left( \sum_{y_K} 1_{y_K \neq y_K^*} p(y_K | \mathbf{X}) \right) \\
&= \min_{y_1^*} (1 - p(y_1^* | \mathbf{X})) + \cdots + \min_{y_K^*} (1 - p(y_K^* | \mathbf{X}))
\end{aligned}$$

$$\begin{aligned}
& \min_{\mathbf{y}^*} \mathbb{E} \left[ \sum_k 1_{y_k \neq y_k^*} \right] \\
&= \min_{\mathbf{y}^*} \sum_{y_1} \cdots \sum_{y_K} \left( \sum_k 1_{y_k \neq y_k^*} \right) p(\mathbf{y} | \mathbf{X}) \\
&= \min_{y_1^*} \dots \min_{y_K^*} \sum_{y_1} \cdots \sum_{y_K} \left( \sum_k 1_{y_k \neq y_k^*} \right) p(\mathbf{y} | \mathbf{X}) \\
&= \min_{y_1^*} \dots \min_{y_K^*} \sum_k \sum_{y_1} \cdots \sum_{y_K} 1_{y_k \neq y_k^*} p(\mathbf{y} | \mathbf{X}) \\
&= \min_{y_1^*} \dots \min_{y_K^*} \sum_k \sum_{y_k} 1_{y_k \neq y_k^*} \sum_{y_1} \cdots \sum_{y_{k-1}} \sum_{y_{k+1}} \sum_{y_K} p(\mathbf{y} | \mathbf{X}) \\
&= \min_{y_1^*} \dots \min_{y_K^*} \sum_k \sum_{y_k} 1_{y_k \neq y_k^*} p(y_k | \mathbf{X}) \\
&= \min_{y_1^*} \left( \sum_{y_1} 1_{y_1 \neq y_1^*} p(y_1 | \mathbf{X}) \right) + \cdots + \min_{y_K^*} \left( \sum_{y_K} 1_{y_K \neq y_K^*} p(y_K | \mathbf{X}) \right) \\
&= \min_{y_1^*} (1 - p(y_1^* | \mathbf{X})) + \cdots + \min_{y_K^*} (1 - p(y_K^* | \mathbf{X})) \\
&= 1 - \max_{y_1^*} p(y_1^* | \mathbf{X}) + \cdots + 1 - \max_{y_K^*} p(y_K^* | \mathbf{X})
\end{aligned}$$

# CLASSIFICATION

**Topics:** making a prediction (option 2)

- Find most probable prediction:

$$\arg \max_{\mathbf{y}^*} p(\mathbf{y} | \mathbf{X})$$

- A Viterbi decoding algorithm can be used for that
  - ▶ forward pass: replace  $\log \sum_{y'_k} \exp(\cdot)$  by  $\max_{y'_k} (\cdot)$  and fill the table
  - ▶ backward pass: starting from the maximal value  $y_K^*$  at the last position, follow the backward trace of the max operations to decode the maximizing sequence

# CLASSIFICATION

**Topics:** making a prediction (option 2)

- Algorithm goes as follows:

Complexity in  $O(KC^2)$

- initialize, for all values of  $y'_2$  :

$$\begin{aligned} - \alpha_1^*(y'_2) &\leftarrow \max_{y'_1} a_u(y'_1) + a_p(y'_1, y'_2) \\ - \alpha_1^\leftarrow(y'_2) &\leftarrow \arg \max_{y'_1} a_u(y'_1) + a_p(y'_1, y'_2) \end{aligned}$$

- for  $k = 2$  to  $K-1$ , for all values of  $y'_{k+1}$  :

$$\begin{aligned} - \alpha_k^*(y'_{k+1}) &\leftarrow \max_{y'_k} a_u(y'_k) + a_p(y'_k, y'_{k+1}) + \alpha_{k-1}^*(y'_k) \\ - \alpha_k^\leftarrow(y'_{k+1}) &\leftarrow \arg \max_{y'_k} a_u(y'_k) + a_p(y'_k, y'_{k+1}) + \alpha_{k-1}^*(y'_k) \end{aligned}$$

- $\max_{\mathbf{y}^*} p(\mathbf{y}|\mathbf{X}) \leftarrow \max_{y'_K} a_u(y'_K) + \alpha_{K-1}^*(y'_K)$

- $y_K^* \leftarrow \arg \max_{y'_K} a_u(y'_K) + \alpha_{K-1}^*(y'_K)$
- for  $k = K-1$  to 1:  $y_k^* \leftarrow \alpha_k^\leftarrow(y_{k+1}^*)$

} backward decoding

# Neural networks

Conditional random fields - factors, sufficient statistics and linear CRF

# LINEAR CHAN CRF

**Topics:** factor, sufficient statistic

- CRFs are often written in the standard form:

$$p(\mathbf{y}|\mathbf{X}) = \frac{1}{Z(\mathbf{X})} \prod_f \underbrace{\Psi_f(\mathbf{y}, \mathbf{X})}_{\text{factors or compatibility functions}}$$

- A parametrization of the factor that is often used is:

$$\Psi_f(\mathbf{y}, \mathbf{X}) = \exp \left( \sum_s \theta_{f,s} \underbrace{t_{f,s}(\mathbf{y}, \mathbf{X})}_{\text{sufficient statistics}} \right)$$

# LINEAR CHAN CRF

**Topics:** factor, sufficient statistic

- With hidden units, the CRF factors could be:

$$\Psi_f(\mathbf{y}, \mathbf{X}) = \begin{cases} \phi_f(y_k, \mathbf{x}_{k-1}) = \exp(a^{(L+1, -1)}(\mathbf{x}_{k-1})_{y_k}) \\ \phi_f(y_k, \mathbf{x}_k) = \exp(a^{(L+1, 0)}(\mathbf{x}_k)_{y_k}) \\ \phi_f(y_k, \mathbf{x}_{k+1}) = \exp(a^{(L+1, +1)}(\mathbf{x}_{k+1})_{y_k}) \\ \phi_f(y_k, y_{k+1}) = \exp(V_{y_k, y_{k+1}}) \end{cases}$$

- There is no simple form for the sufficient statistics

# LINEAR CHAN CRF

**Topics:** factor, sufficient statistic

- With hidden units, the CRF factors could be:

unary factors

$$\Psi_f(\mathbf{y}, \mathbf{X}) = \begin{cases} \phi_f(y_k, \mathbf{x}_{k-1}) = \exp(a^{(L+1, -1)}(\mathbf{x}_{k-1})_{y_k}) \\ \phi_f(y_k, \mathbf{x}_k) = \exp(a^{(L+1, 0)}(\mathbf{x}_k)_{y_k}) \\ \phi_f(y_k, \mathbf{x}_{k+1}) = \exp(a^{(L+1, +1)}(\mathbf{x}_{k+1})_{y_k}) \\ \phi_f(y_k, y_{k+1}) = \exp(V_{y_k, y_{k+1}}) \end{cases}$$

- There is no simple form for the sufficient statistics

# LINEAR CHAN CRF

**Topics:** factor, sufficient statistic

- With hidden units, the CRF factors could be:

$$\Psi_f(\mathbf{y}, \mathbf{X}) = \begin{cases} \text{unary factors} \\ \phi_f(y_k, \mathbf{x}_{k-1}) = \exp(a^{(L+1, -1)}(\mathbf{x}_{k-1})_{y_k}) \\ \phi_f(y_k, \mathbf{x}_k) = \exp(a^{(L+1, 0)}(\mathbf{x}_k)_{y_k}) \\ \phi_f(y_k, \mathbf{x}_{k+1}) = \exp(a^{(L+1, +1)}(\mathbf{x}_{k+1})_{y_k}) \\ \phi_f(y_k, y_{k+1}) = \exp(V_{y_k, y_{k+1}}) \\ \text{pairwise factors} \end{cases}$$

- There is no simple form for the sufficient statistics

# LINEAR CHAN CRF

## Topics: linear CRF

- In a linear (linear chain) CRF (i.e. without hidden units), the CRF factors can be written as:

$$\Psi_f(\mathbf{y}, \mathbf{X}) = \begin{cases} \phi_f(y_k) = \exp(b_c^{(L+1)} 1_{y_k=c}) \\ \phi_f(y_k, x_{k-1,i}) = \exp(W_{c,i}^{(L+1,-1)} x_{k-1,i} 1_{y_k=c}) \\ \phi_f(y_k, x_{k,i}) = \exp(W_{c,i}^{(L+1,0)} x_{k,i} 1_{y_k=c}) \\ \phi_f(y_k, x_{k+1,i}) = \exp(W_{c,i}^{(L+1,+1)} x_{k+1,i} 1_{y_k=c}) \\ \phi_f(y_k, y_{k+1}) = \exp(V_{c,c'} 1_{y_k=c} 1_{y_{k+1}=c'}) \end{cases}$$

# LINEAR CHAN CRF

## Topics: linear CRF

- In a linear (linear chain) CRF (i.e. without hidden units), the CRF factors can be written as:

$$\Psi_f(\mathbf{y}, \mathbf{X}) = \begin{cases} \phi_f(y_k) = \exp(b_c^{(L+1)} \mathbf{1}_{y_k=c}) \\ \phi_f(y_k, x_{k-1,i}) = \exp(W_{c,i}^{(L+1,-1)} x_{k-1,i} \mathbf{1}_{y_k=c}) \\ \phi_f(y_k, x_{k,i}) = \exp(W_{c,i}^{(L+1,0)} x_{k,i} \mathbf{1}_{y_k=c}) \\ \phi_f(y_k, x_{k+1,i}) = \exp(W_{c,i}^{(L+1,+1)} x_{k+1,i} \mathbf{1}_{y_k=c}) \\ \phi_f(y_k, y_{k+1}) = \exp(V_{c,c'} \mathbf{1}_{y_k=c} \mathbf{1}_{y_{k+1}=c'}) \end{cases}$$

sufficient  
statistics

# LINEAR CHAN CRF

## Topics: linear CRF

- In a linear (linear chain) CRF (i.e. without hidden units), the CRF factors can be written as:

$$\Psi_f(\mathbf{y}, \mathbf{X}) = \begin{cases} \phi_f(y_k) = \exp \left( b_c^{(L+1)} \mathbf{1}_{y_k=c} \right) \\ \phi_f(y_k, x_{k-1,i}) = \exp \left( W_{c,i}^{(L+1,-1)} x_{k-1,i} \mathbf{1}_{y_k=c} \right) \\ \phi_f(y_k, x_{k,i}) = \exp \left( W_{c,i}^{(L+1,0)} x_{k,i} \mathbf{1}_{y_k=c} \right) \\ \phi_f(y_k, x_{k+1,i}) = \exp \left( W_{c,i}^{(L+1,+1)} x_{k+1,i} \mathbf{1}_{y_k=c} \right) \\ \phi_f(y_k, y_{k+1}) = \exp \left( V_{c,c'} \mathbf{1}_{y_k=c} \mathbf{1}_{y_{k+1}=c'} \right) \end{cases}$$

parameters      sufficient statistics  
 $\theta_{f,s}$                    $\theta_{f,s}$

# Neural networks

Conditional random fields - Markov network

# LINEAR CHAN CRF

**Topics:** factor, sufficient statistic

- With hidden units, the CRF factors could be:

$$\Psi_f(\mathbf{y}, \mathbf{X}) = \begin{cases} \phi_f(y_k, \mathbf{x}_{k-1}) = \exp(a^{(L+1, -1)}(\mathbf{x}_{k-1})_{y_k}) \\ \phi_f(y_k, \mathbf{x}_k) = \exp(a^{(L+1, 0)}(\mathbf{x}_k)_{y_k}) \\ \phi_f(y_k, \mathbf{x}_{k+1}) = \exp(a^{(L+1, +1)}(\mathbf{x}_{k+1})_{y_k}) \\ \phi_f(y_k, y_{k+1}) = \exp(V_{y_k, y_{k+1}}) \end{cases}$$

- There is no simple form for the sufficient statistics

# LINEAR CHAN CRF

**Topics:** factor, sufficient statistic

- With hidden units, the CRF factors could be:

unary factors

$$\Psi_f(\mathbf{y}, \mathbf{X}) = \begin{cases} \phi_f(y_k, \mathbf{x}_{k-1}) = \exp(a^{(L+1, -1)}(\mathbf{x}_{k-1})_{y_k}) \\ \phi_f(y_k, \mathbf{x}_k) = \exp(a^{(L+1, 0)}(\mathbf{x}_k)_{y_k}) \\ \phi_f(y_k, \mathbf{x}_{k+1}) = \exp(a^{(L+1, +1)}(\mathbf{x}_{k+1})_{y_k}) \\ \phi_f(y_k, y_{k+1}) = \exp(V_{y_k, y_{k+1}}) \end{cases}$$

- There is no simple form for the sufficient statistics

# LINEAR CHAN CRF

**Topics:** factor, sufficient statistic

- With hidden units, the CRF factors could be:

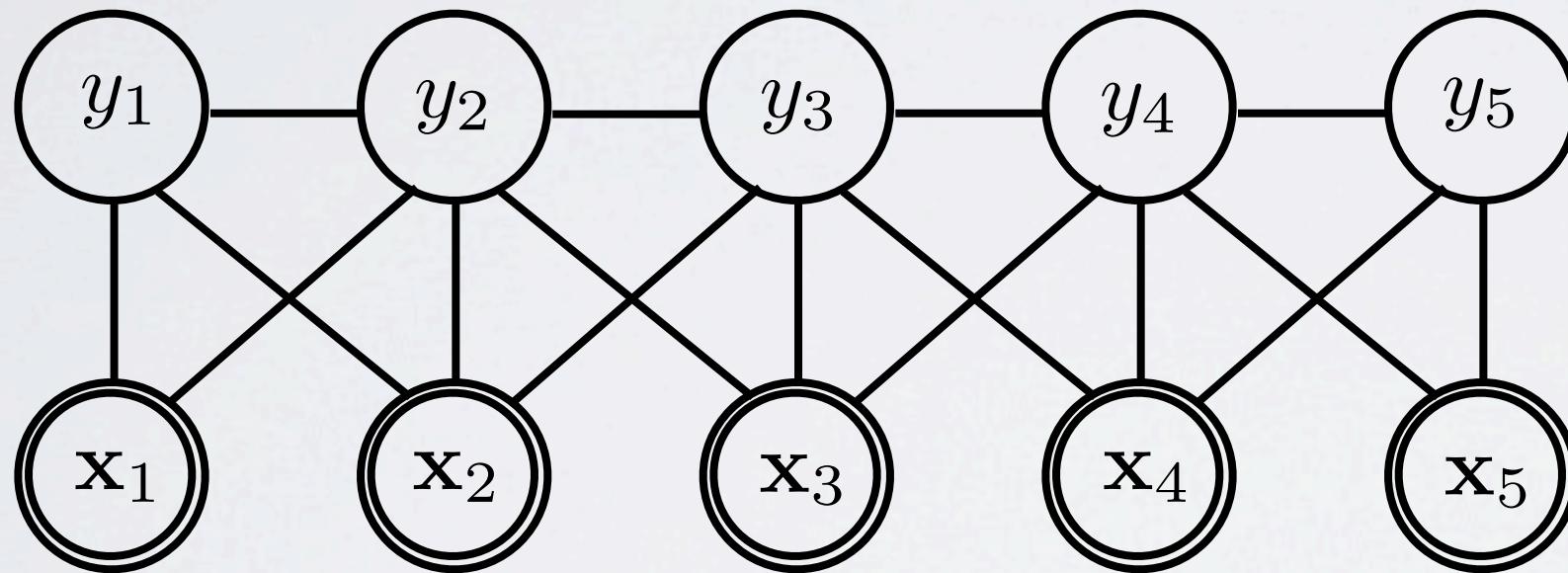
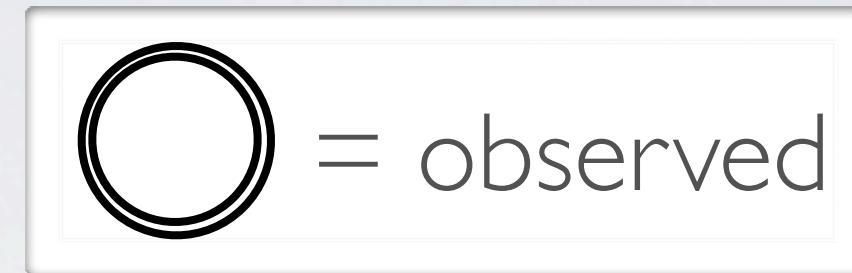
$$\Psi_f(\mathbf{y}, \mathbf{X}) = \begin{cases} \text{unary factors} \\ \phi_f(y_k, \mathbf{x}_{k-1}) = \exp(a^{(L+1, -1)}(\mathbf{x}_{k-1})_{y_k}) \\ \phi_f(y_k, \mathbf{x}_k) = \exp(a^{(L+1, 0)}(\mathbf{x}_k)_{y_k}) \\ \phi_f(y_k, \mathbf{x}_{k+1}) = \exp(a^{(L+1, +1)}(\mathbf{x}_{k+1})_{y_k}) \\ \phi_f(y_k, y_{k+1}) = \exp(V_{y_k, y_{k+1}}) \\ \text{pairwise factors} \end{cases}$$

- There is no simple form for the sufficient statistics

# MARKOV NETWORK VISUALIZATION

**Topics:** Markov network

- Illustration for  $K=5$

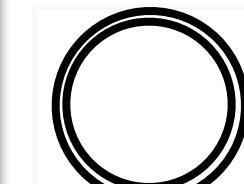


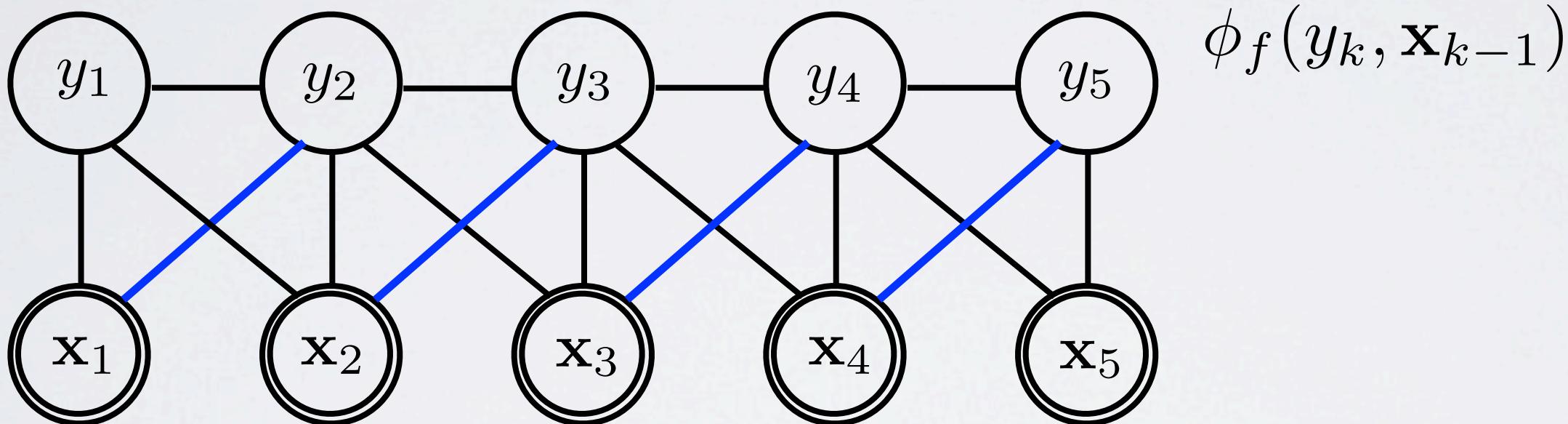
- Network with an edge between each node (random variable) that shares a factor

# MARKOV NETWORK VISUALIZATION

**Topics:** Markov network

- Illustration for  $K=5$

 = observed

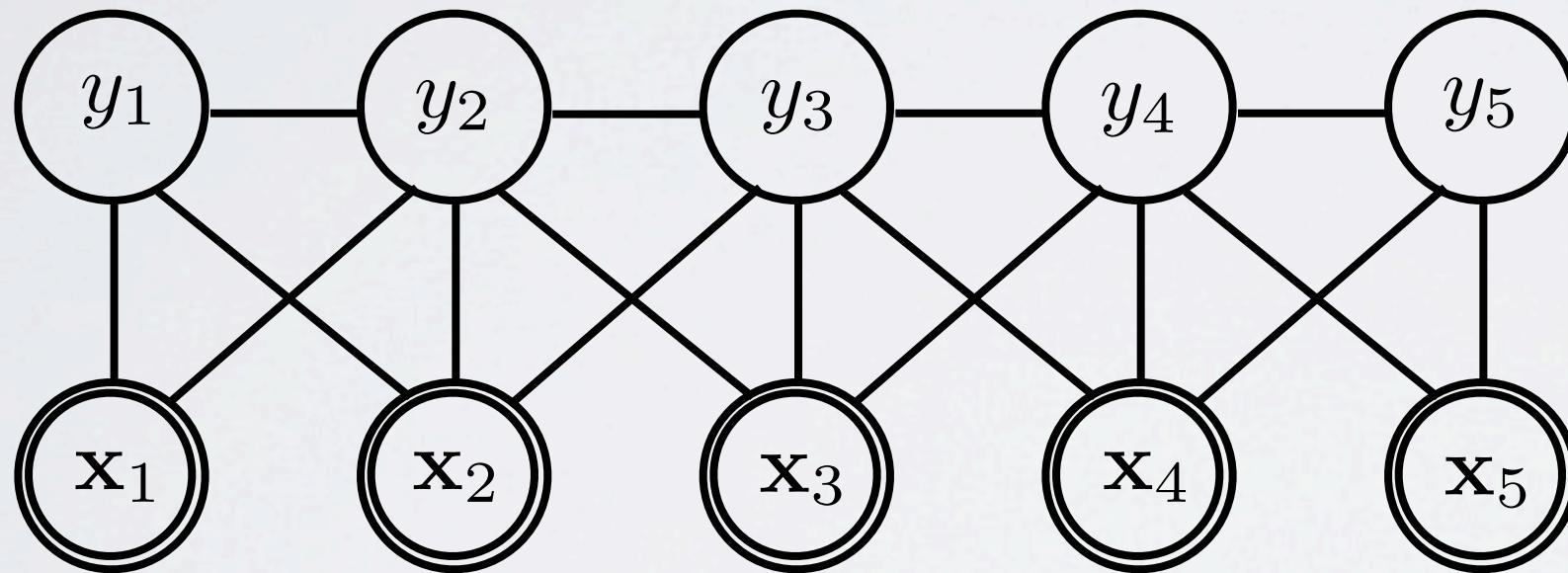
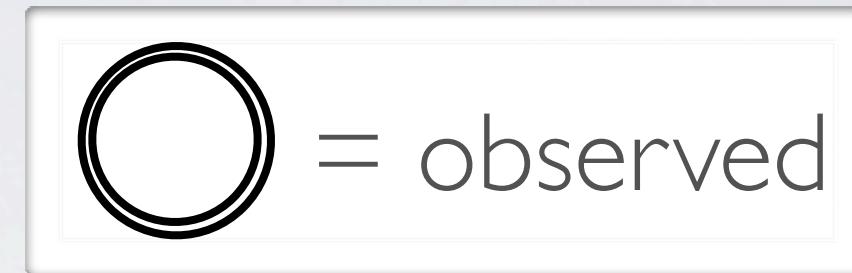


- Network with an edge between each node (random variable) that shares a factor

# MARKOV NETWORK VISUALIZATION

**Topics:** Markov network

- Illustration for  $K=5$

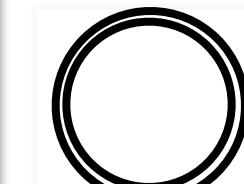


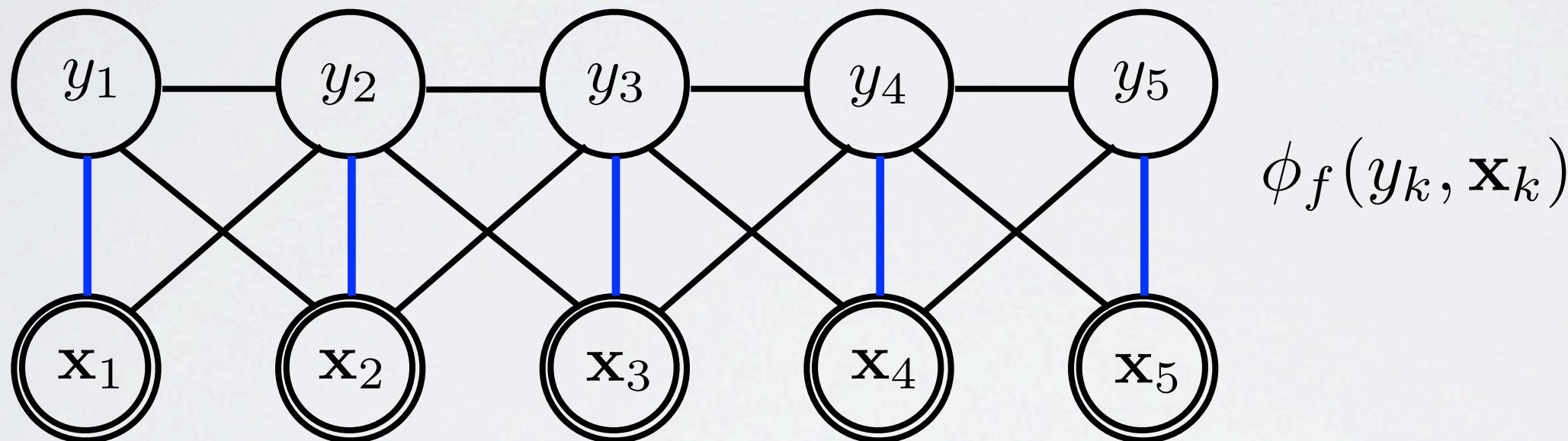
- Network with an edge between each node (random variable) that shares a factor

# MARKOV NETWORK VISUALIZATION

**Topics:** Markov network

- Illustration for  $K=5$

 = observed

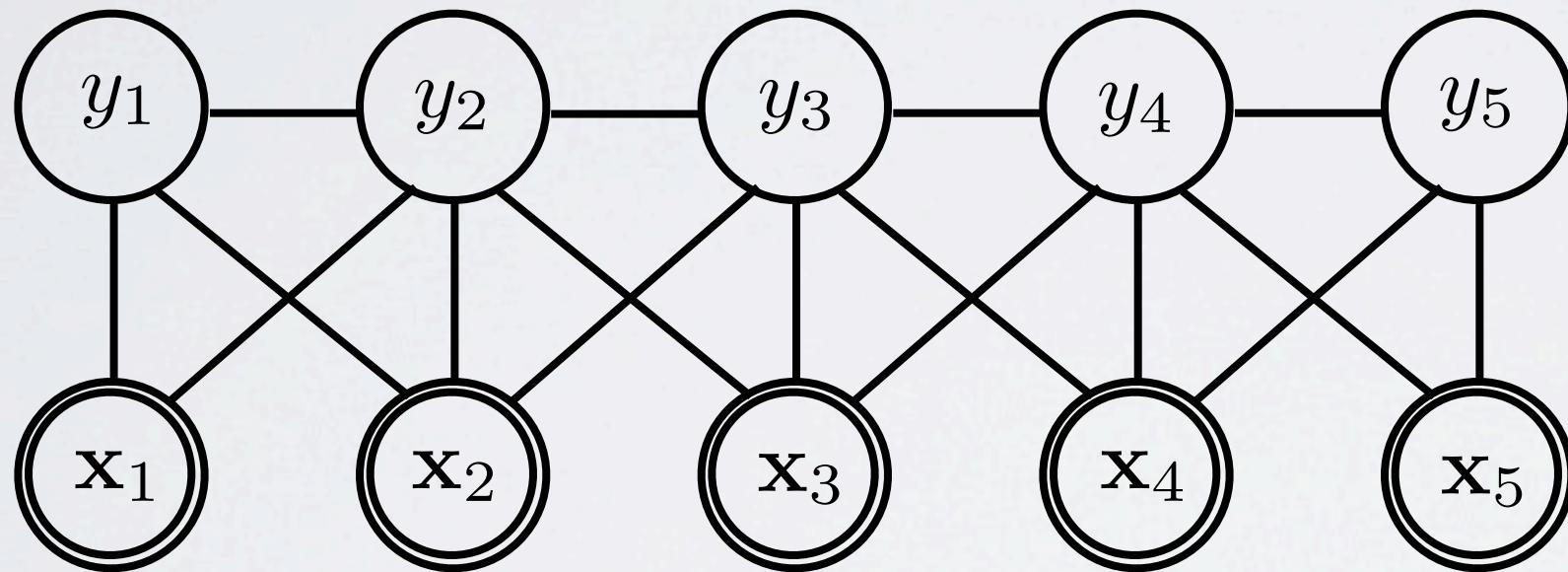
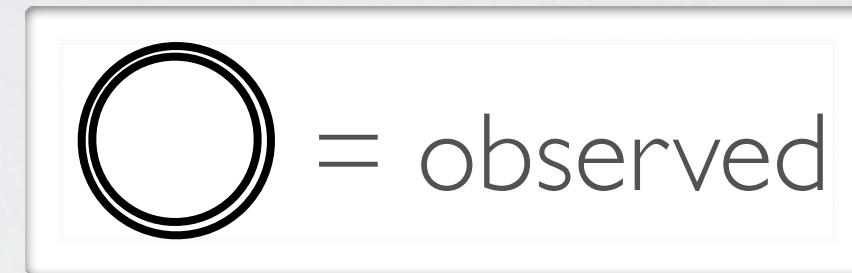


- Network with an edge between each node (random variable) that shares a factor

# MARKOV NETWORK VISUALIZATION

**Topics:** Markov network

- Illustration for  $K=5$

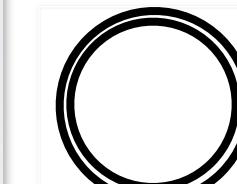


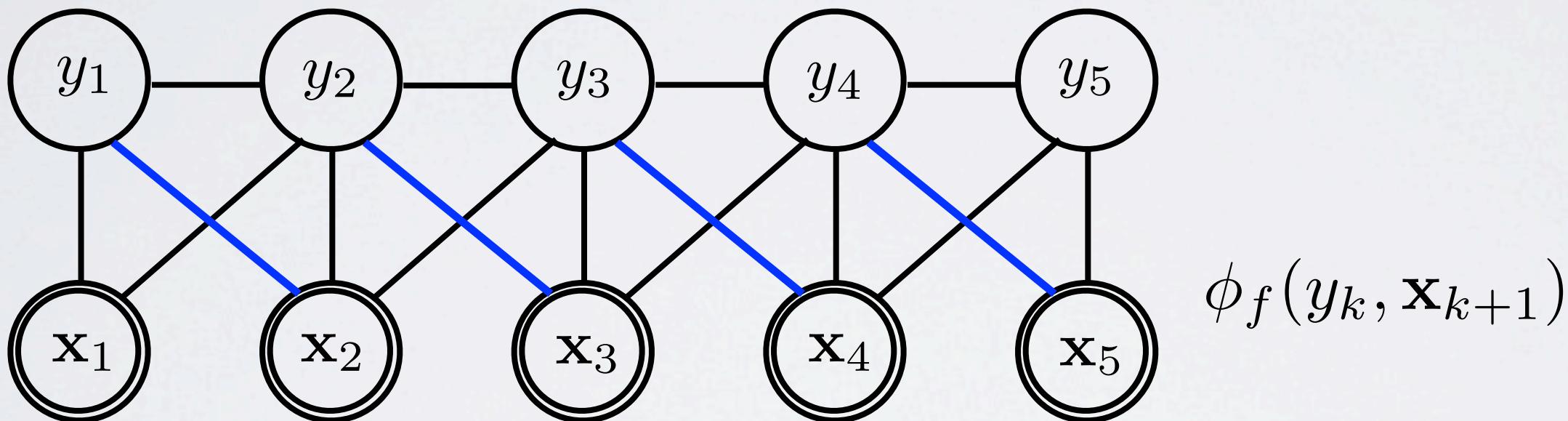
- Network with an edge between each node (random variable) that shares a factor

# MARKOV NETWORK VISUALIZATION

**Topics:** Markov network

- Illustration for  $K=5$

 = observed

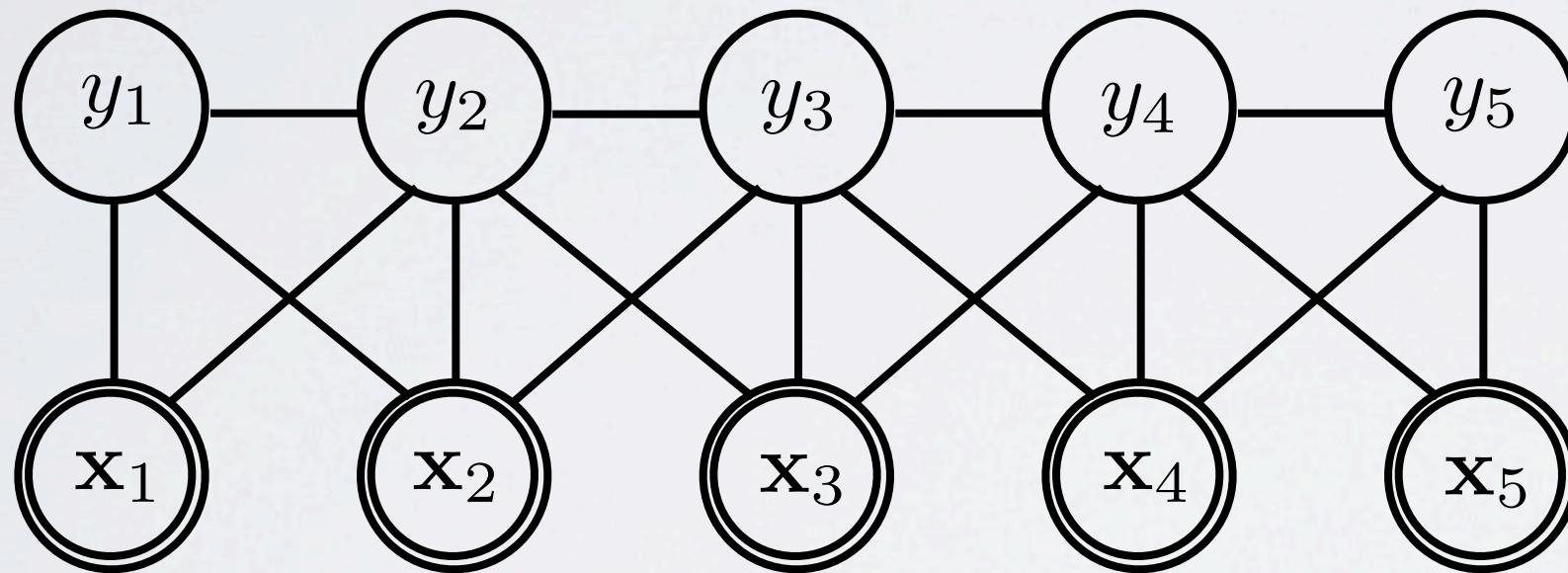
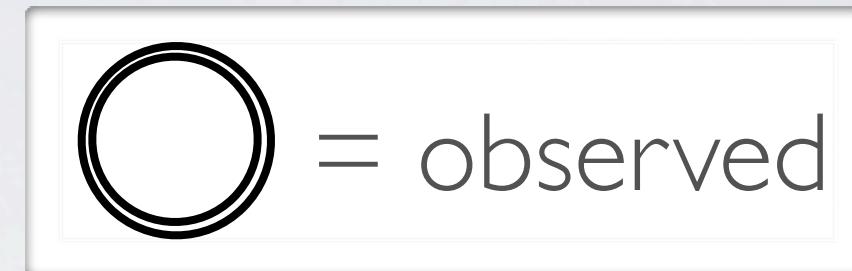


- Network with an edge between each node (random variable) that shares a factor

# MARKOV NETWORK VISUALIZATION

**Topics:** Markov network

- Illustration for  $K=5$

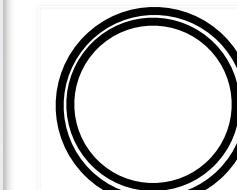


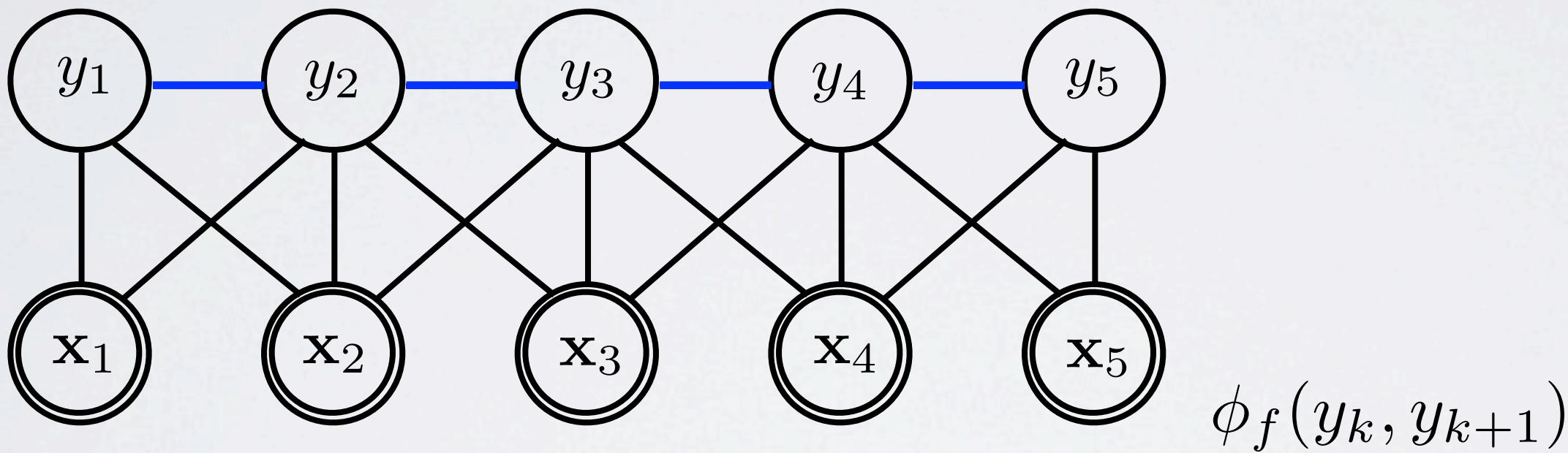
- Network with an edge between each node (random variable) that shares a factor

# MARKOV NETWORK VISUALIZATION

**Topics:** Markov network

- Illustration for  $K=5$

 = observed

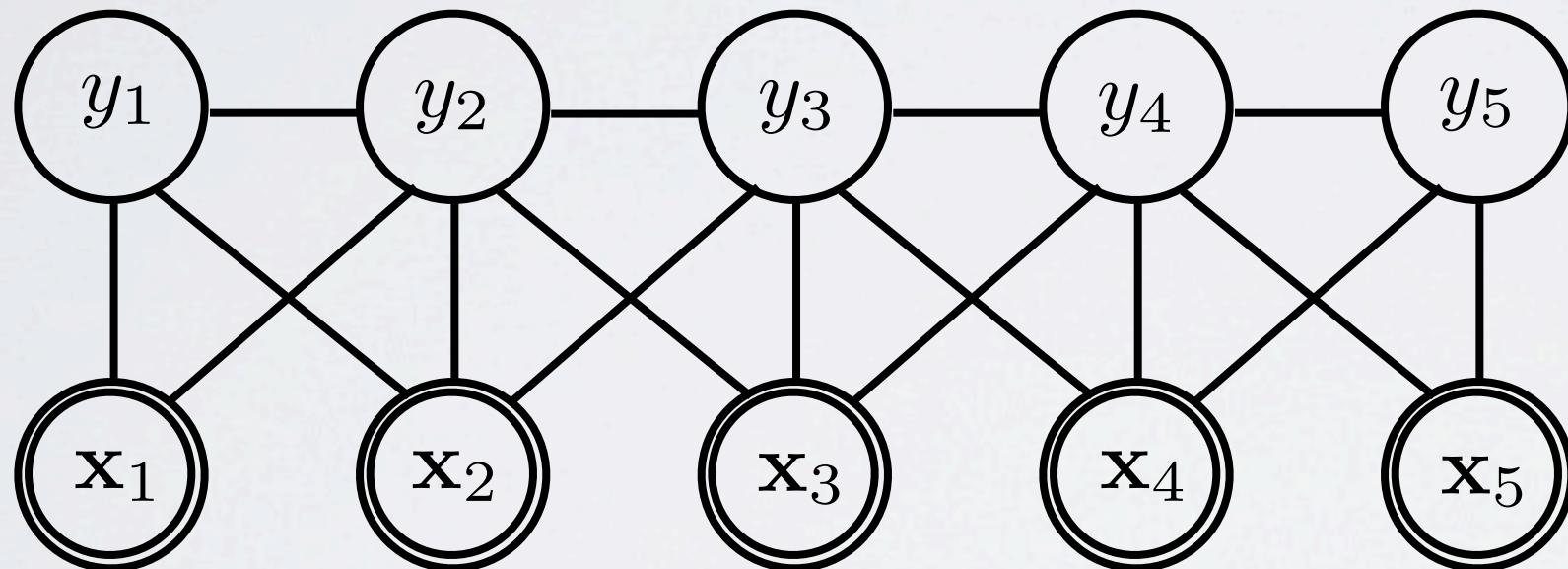
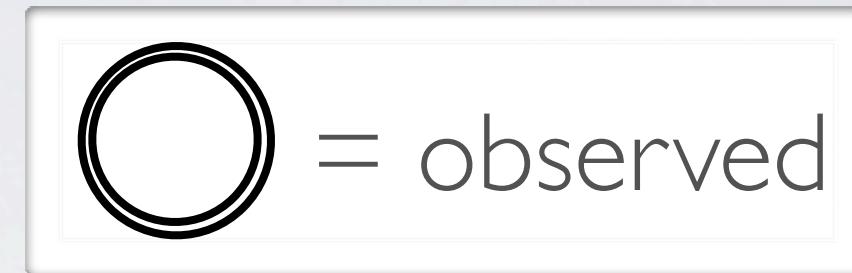


- Network with an edge between each node (random variable) that shares a factor

# MARKOV NETWORK VISUALIZATION

**Topics:** Markov network

- Illustration for  $K=5$

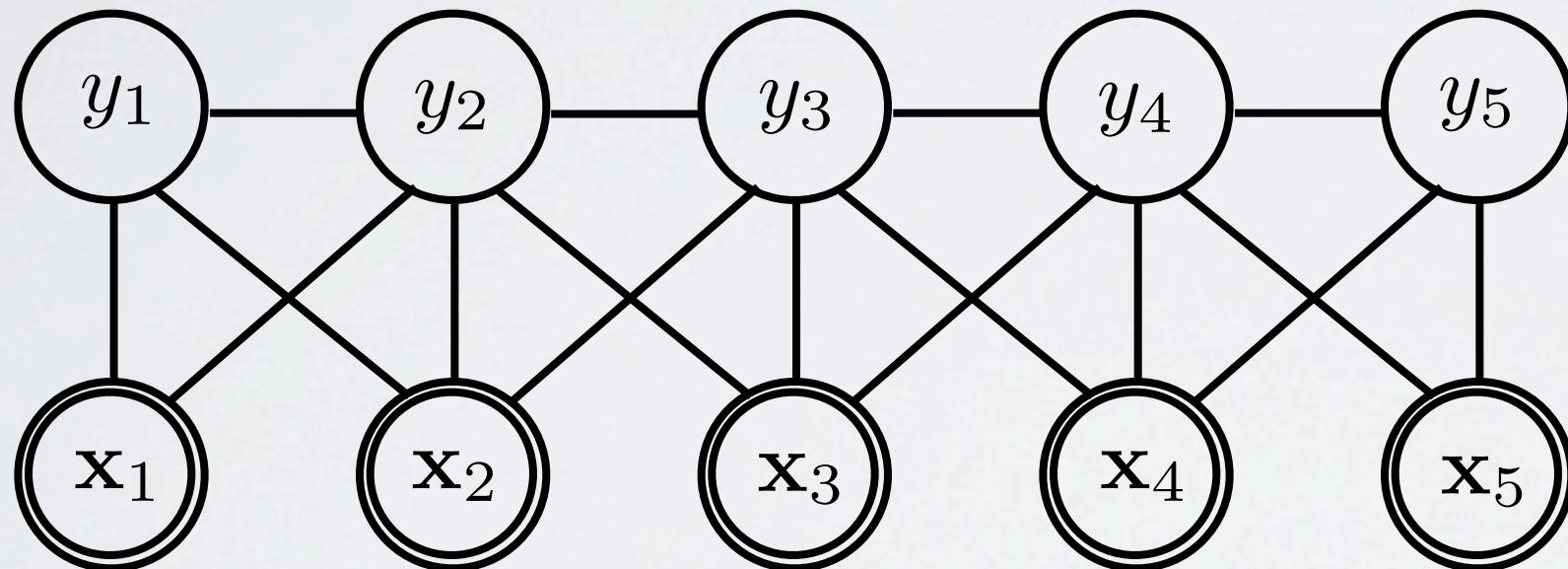
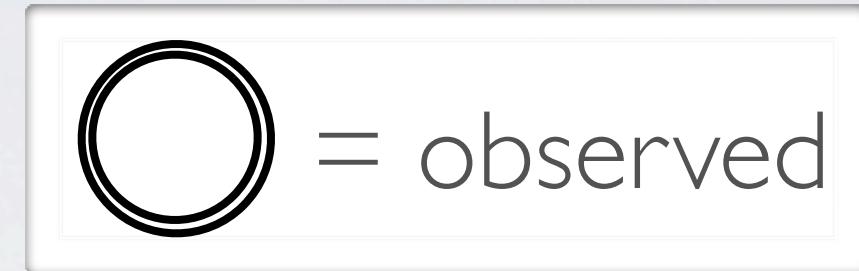


- Network with an edge between each node (random variable) that shares a factor

# MARKOV NETWORK VISUALIZATION

**Topics:** Markov network

- Illustration for  $K=5$

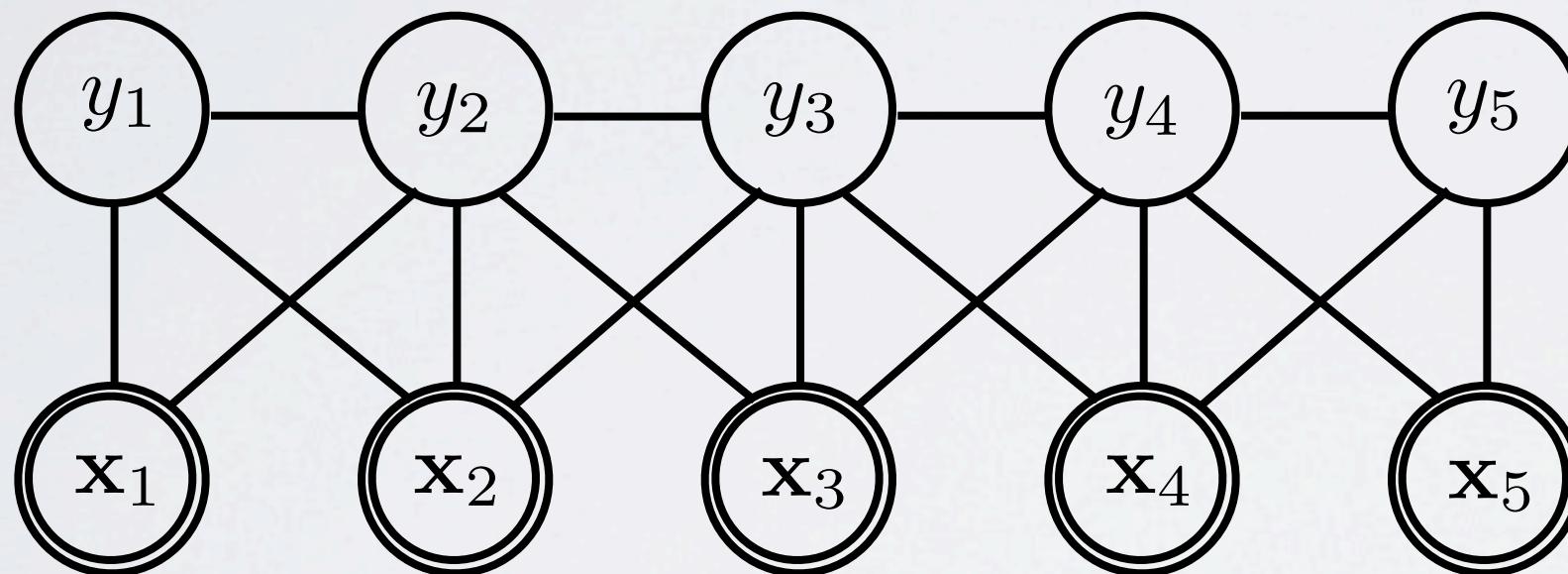
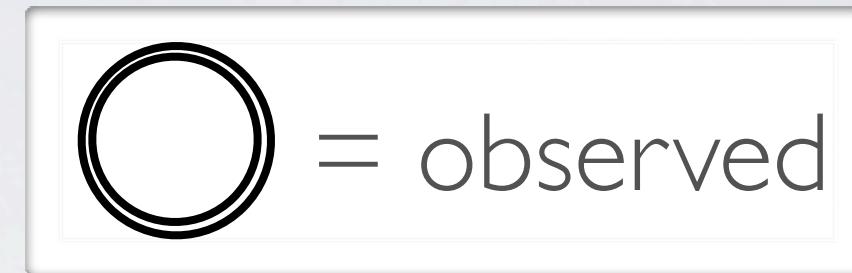


- Local Markov property
  - ▶ each node is independent of other nodes **given** its neighbors

# MARKOV NETWORK VISUALIZATION

**Topics:** Markov network

- Illustration for  $K=5$

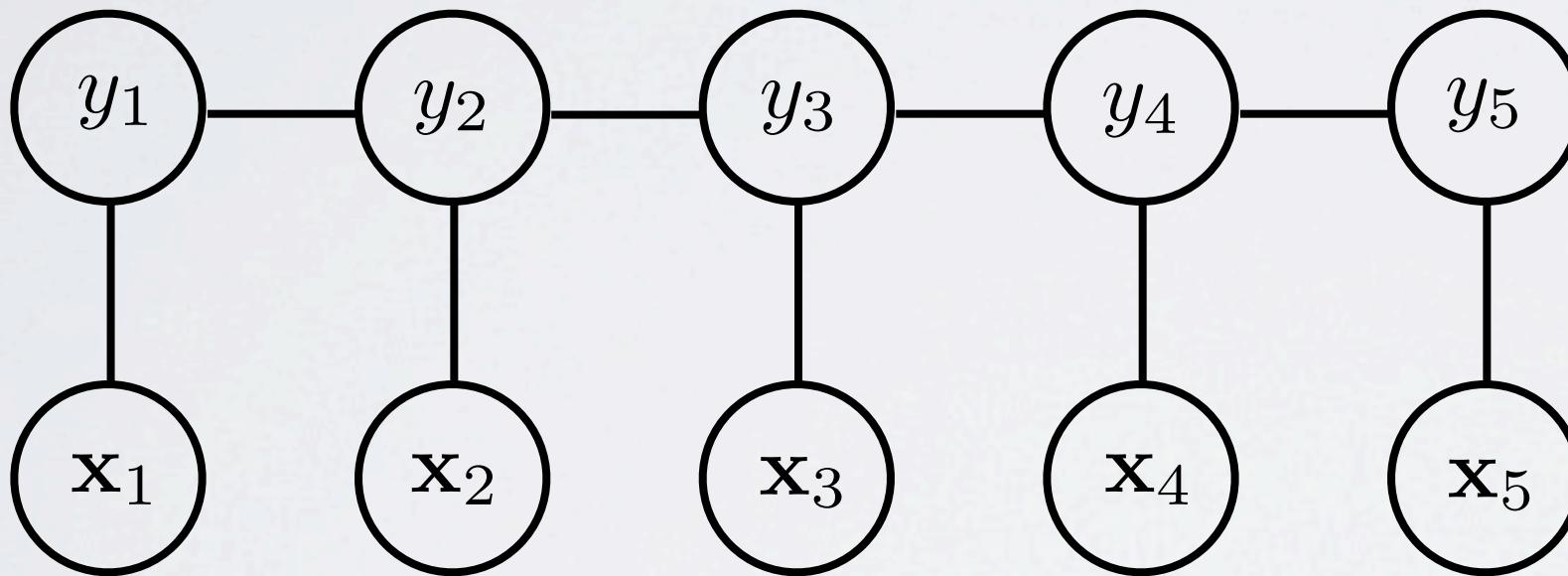


- General conditional independence
  - ▶ two nodes are conditionally independent if all paths between them contain at least one of the conditioning node

# DIRECTED VS. UNDIRECTED

**Topics:** undirected graphical model, directed graphical model

- CRF is an undirected graphical model

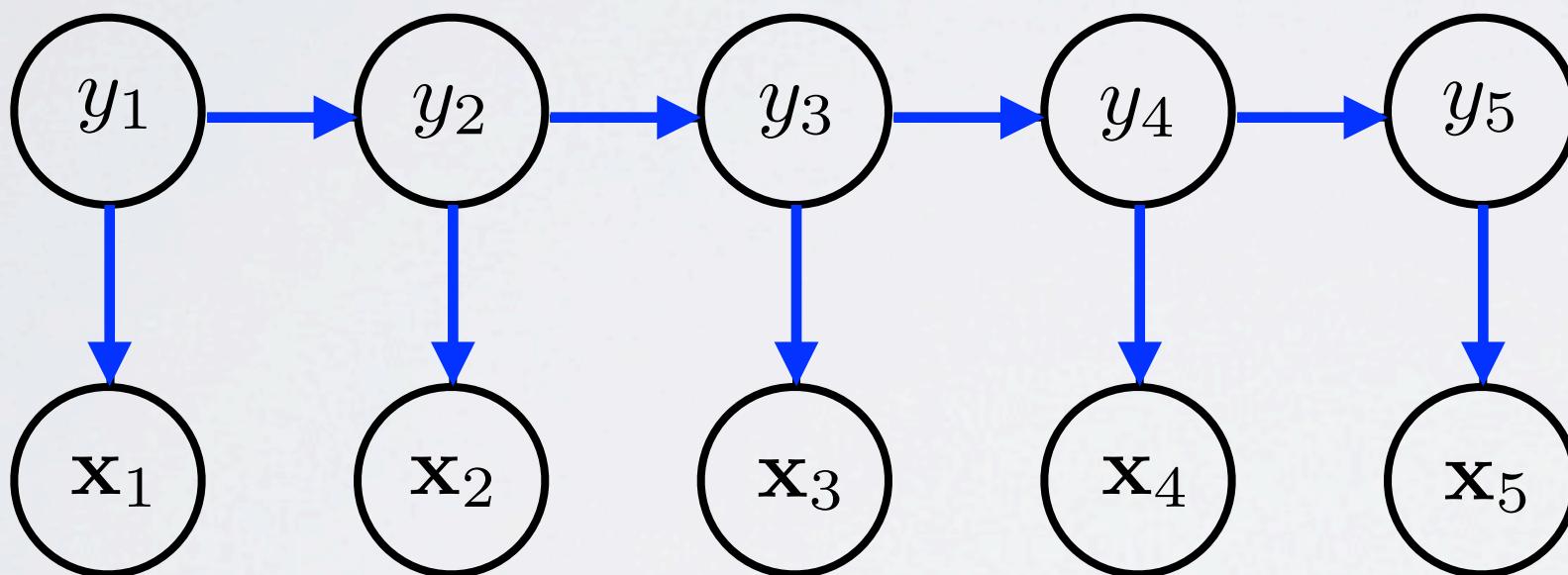


$$\left. \begin{array}{l} \phi_f(y_k, \mathbf{x}_k) \\ \phi_f(y_k, y_{k+1}) \end{array} \right\} \text{only need to be non-negative}$$

# DIRECTED VS. UNDIRECTED

**Topics:** undirected graphical model, directed graphical model

- HMM is a directed graphical model



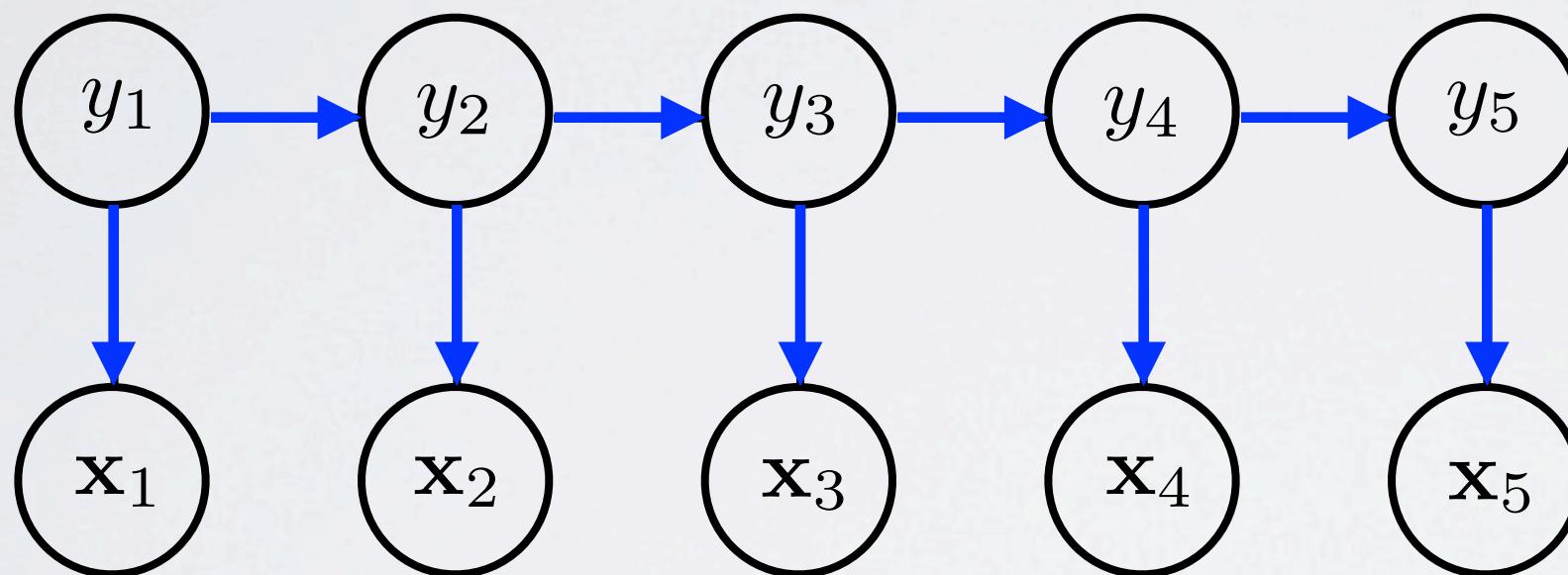
$$\phi_f(y_k, \mathbf{x}_k)$$

$$\phi_f(y_k, y_{k+1})$$

# DIRECTED VS. UNDIRECTED

**Topics:** undirected graphical model, directed graphical model

- HMM is a directed graphical model



$$\phi_f(y_k, \mathbf{x}_k) = p(\mathbf{x}_k | y_k)$$

$$\phi_f(y_k, y_{k+1}) = p(y_{k+1} | y_k)$$

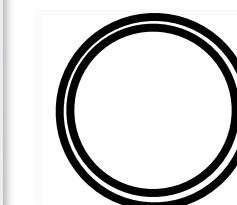
# Neural networks

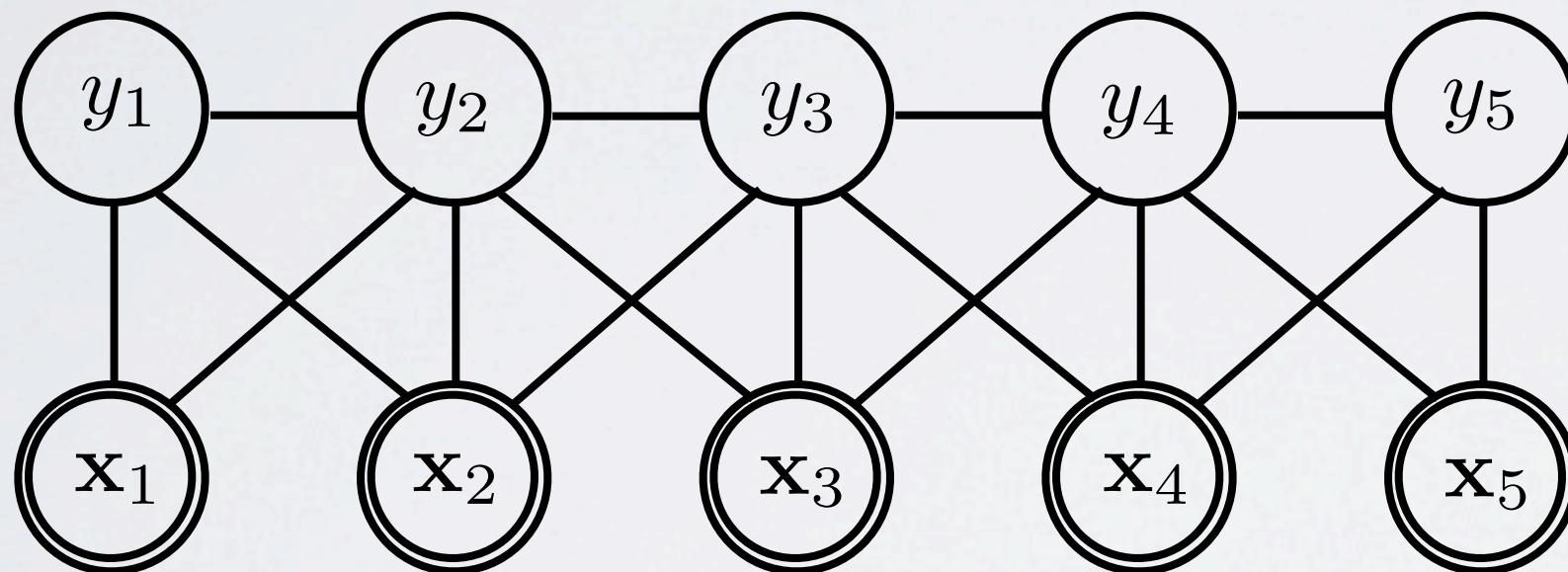
Conditional random fields - factor graph

# MARKOV NETWORK VISUALIZATION

**Topics:** Markov network

- Illustration for  $K=5$

 = observed

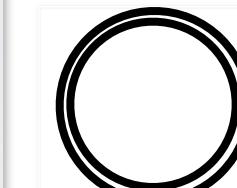


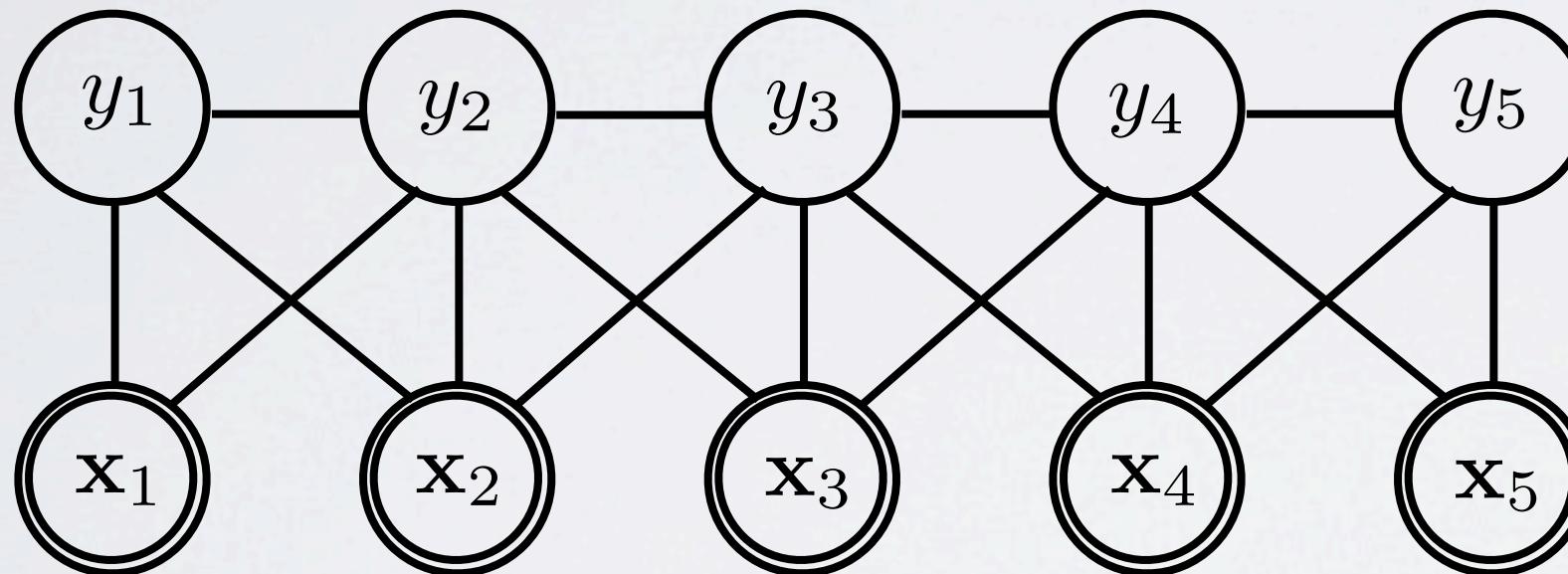
- Each edge is associated with one of more factors

# MARKOV NETWORK VISUALIZATION

**Topics:** Markov network

- Illustration for  $K=5$

 = observed



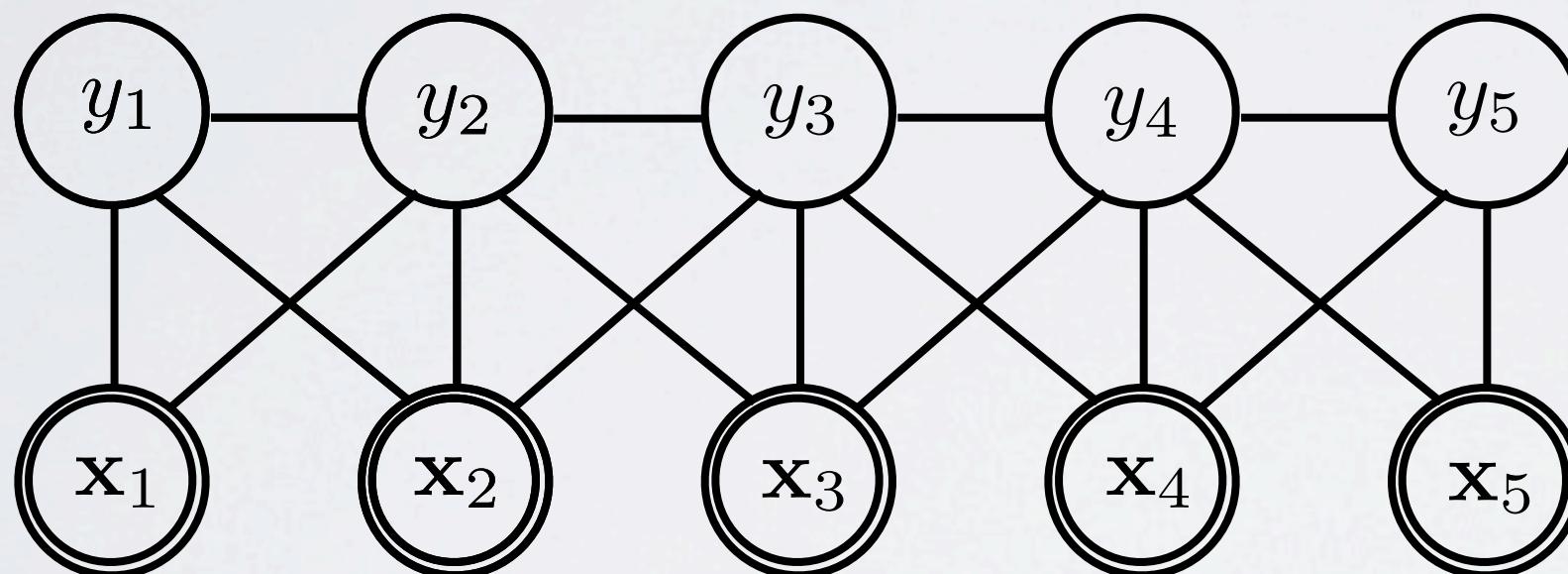
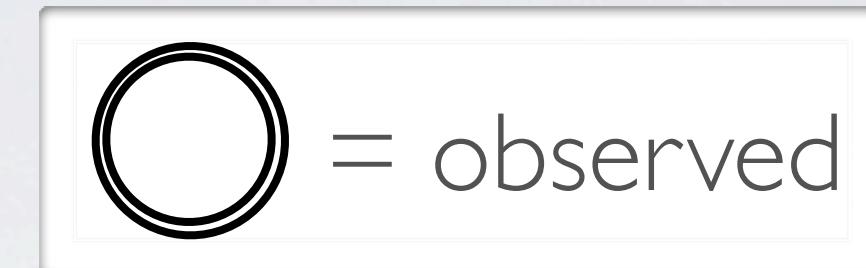
- Representation is ambiguous:
  - ▶ clique between  $y_1$ ,  $y_2$  and  $\mathbf{x}_2$ , but we didn't explicitly define a joint factor  $\phi_f(y_1, y_2, \mathbf{x}_2)$
  - ▶ in fact, we defined  $\phi_f(y_1, y_2)$ ,  $\phi_f(y_2, \mathbf{x}_2)$  and  $\phi_f(y_1, \mathbf{x}_2)$ , which is equivalent to having:  

$$\phi_f(y_1, y_2, \mathbf{x}_2) = \phi_f(y_1, y_2)\phi_f(y_2, \mathbf{x}_2)\phi_f(y_1, \mathbf{x}_2)$$

# FACTOR GRAPH VISUALIZATION

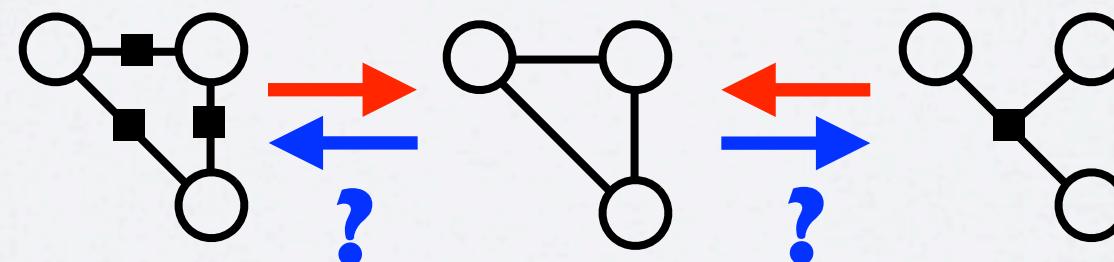
**Topics:** factor graph

- Factor graphs better represent factors



Factor graph 1      Markov network      Factor graph 2

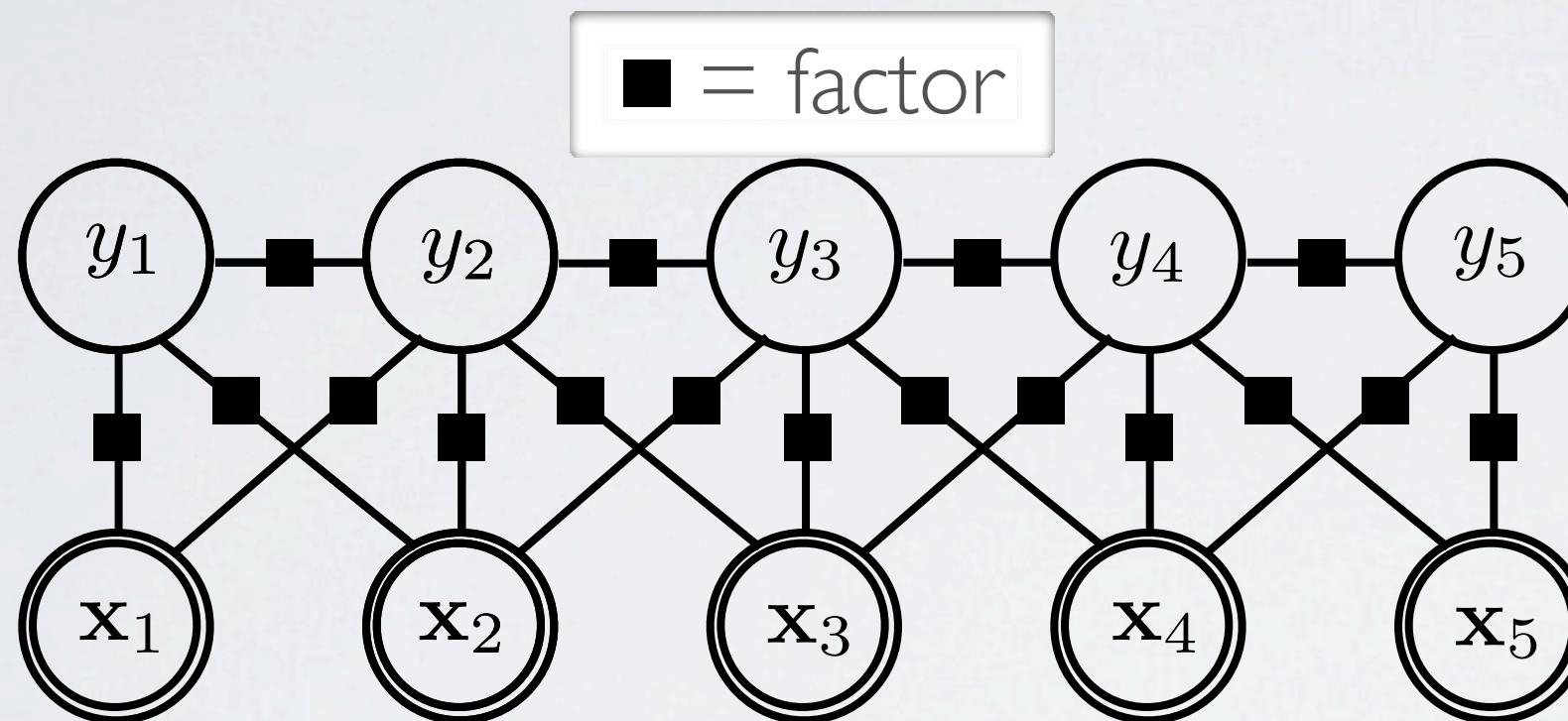
- This is less ambiguous:



# FACTOR GRAPH VISUALIZATION

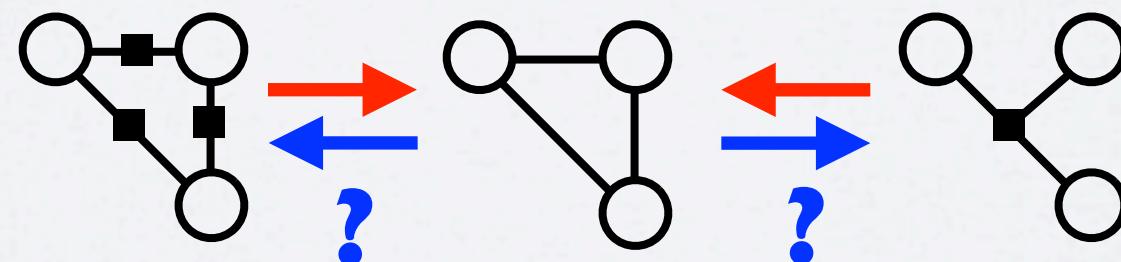
**Topics:** factor graph

- Factor graphs better represent factors



Factor graph 1      Markov network      Factor graph 2

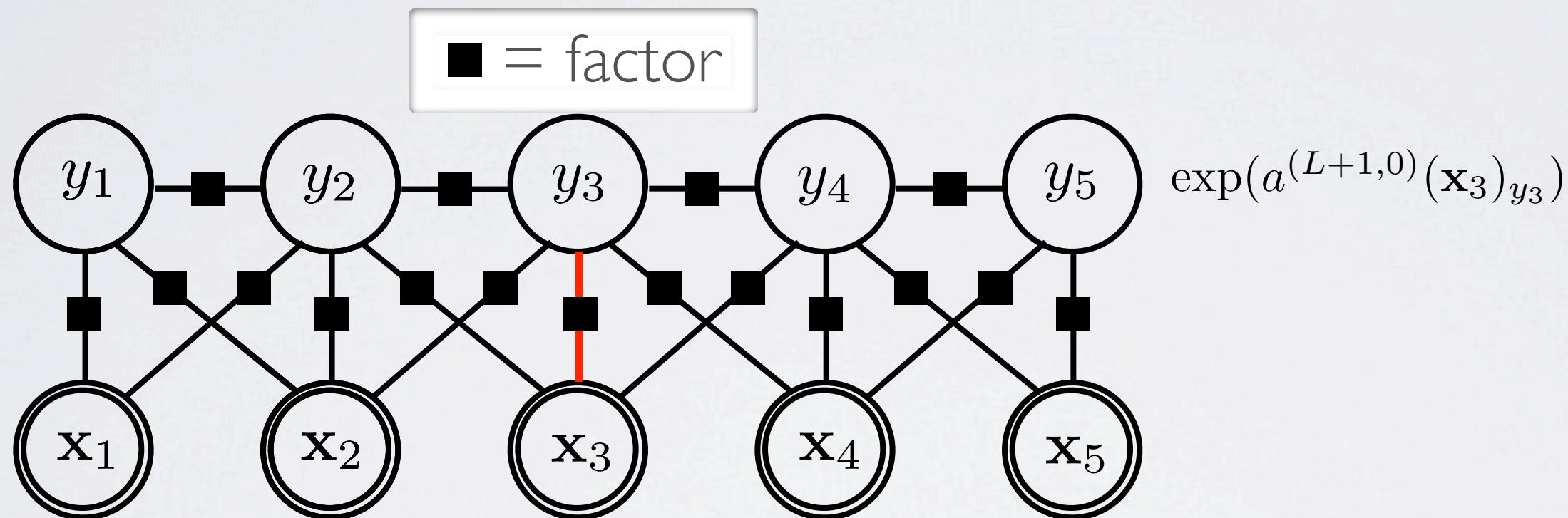
- This is less ambiguous:



# FACTOR GRAPH VISUALIZATION

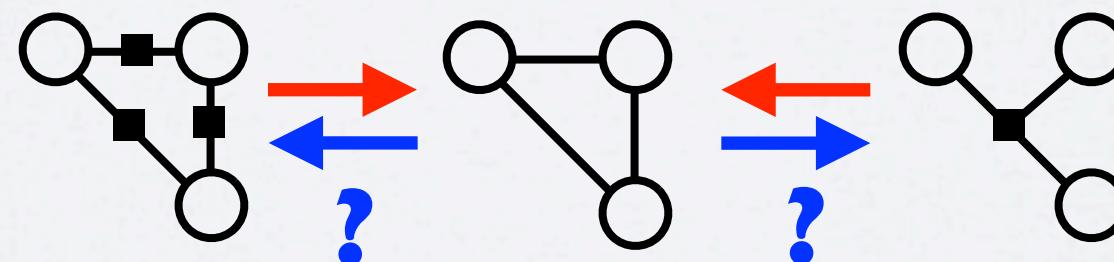
**Topics:** factor graph

- Factor graphs better represent factors



Factor graph I      Markov network      Factor graph 2

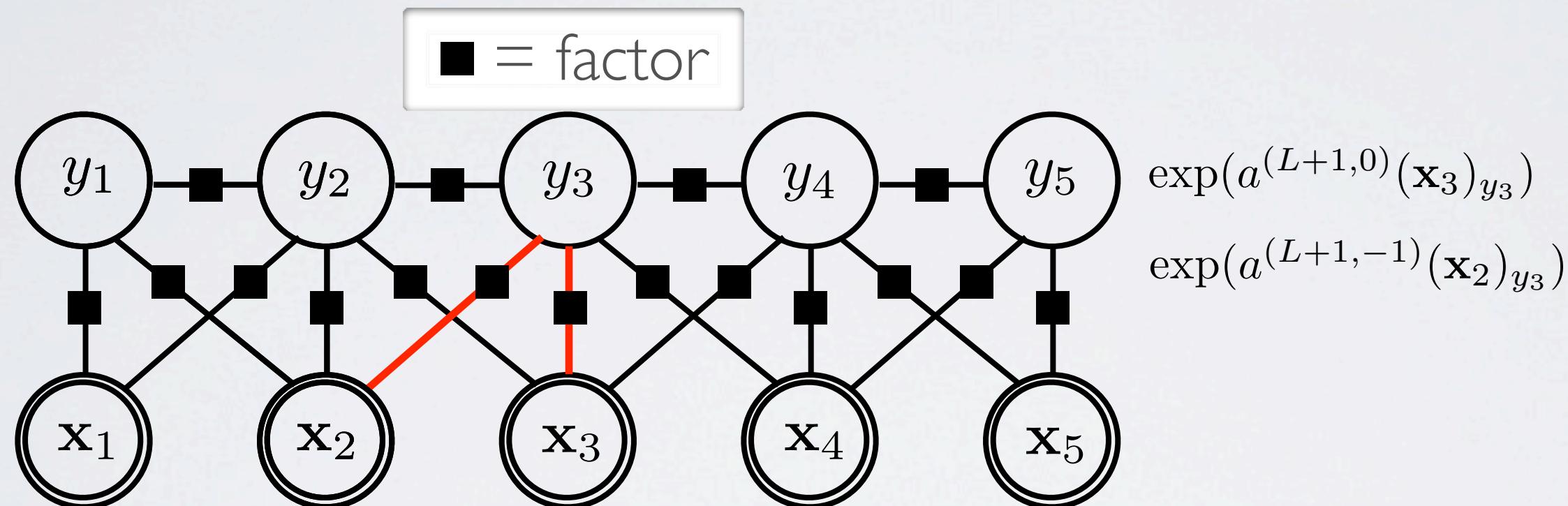
- This is less ambiguous:



# FACTOR GRAPH VISUALIZATION

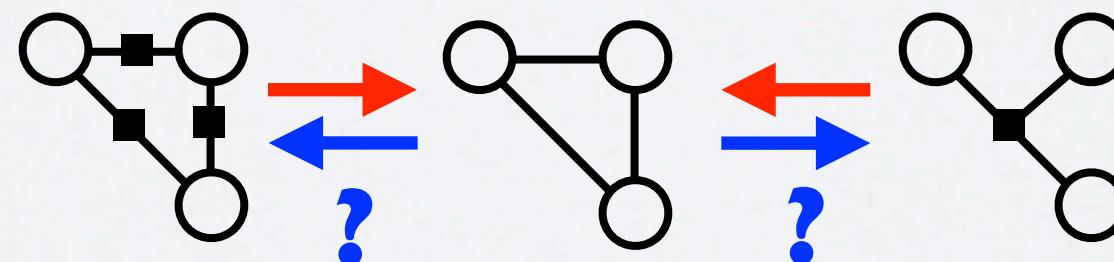
**Topics:** factor graph

- Factor graphs better represent factors



Factor graph 1      Markov network      Factor graph 2

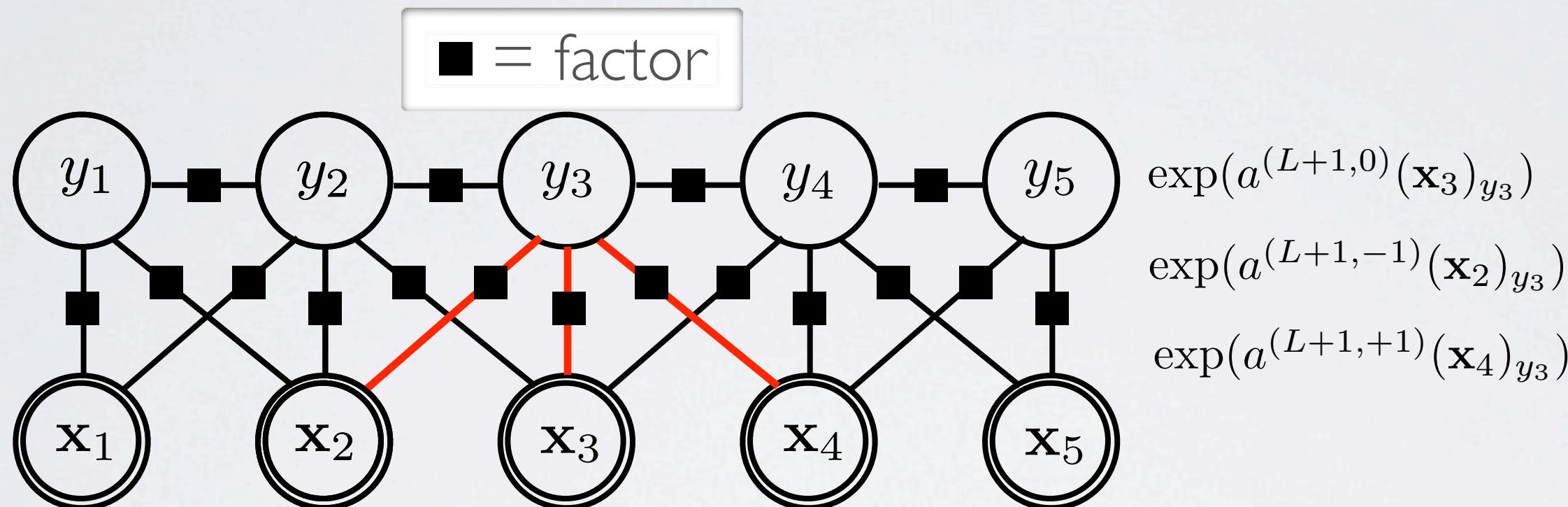
- This is less ambiguous:



# FACTOR GRAPH VISUALIZATION

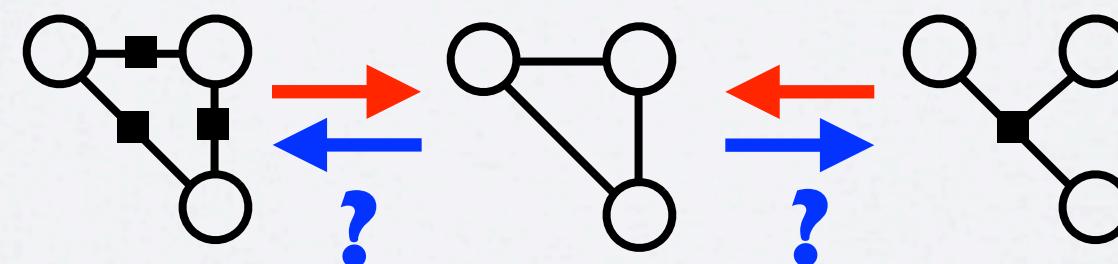
**Topics:** factor graph

- Factor graphs better represent factors



Factor graph I      Markov network      Factor graph 2

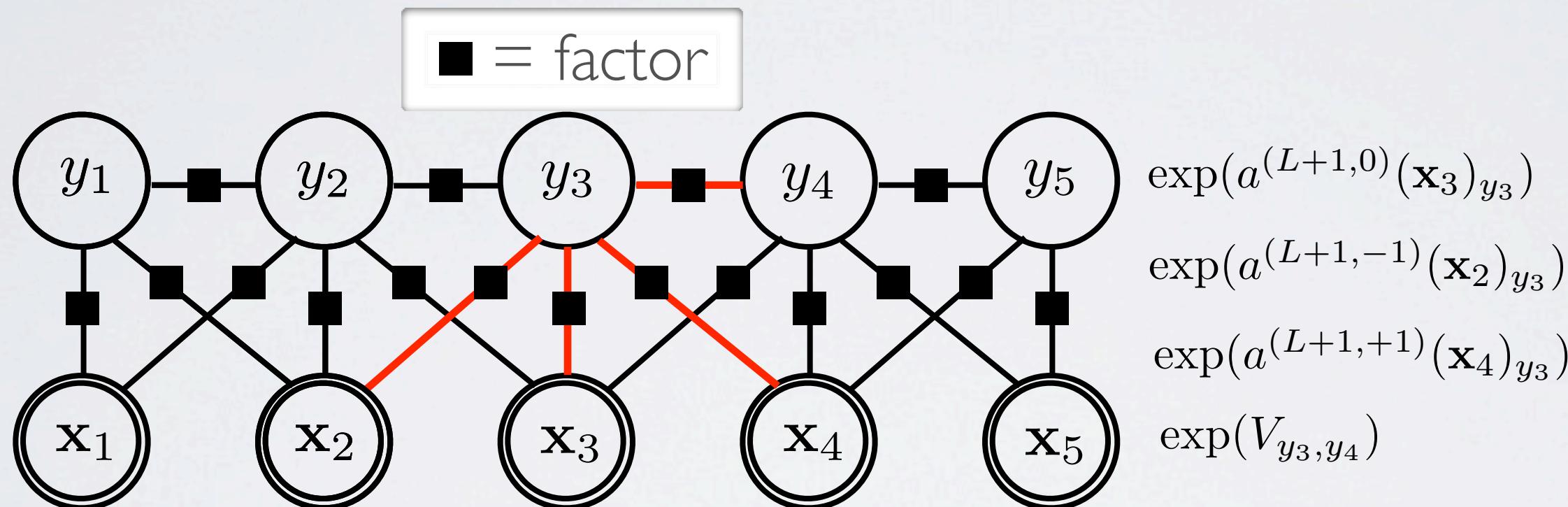
- This is less ambiguous:



# FACTOR GRAPH VISUALIZATION

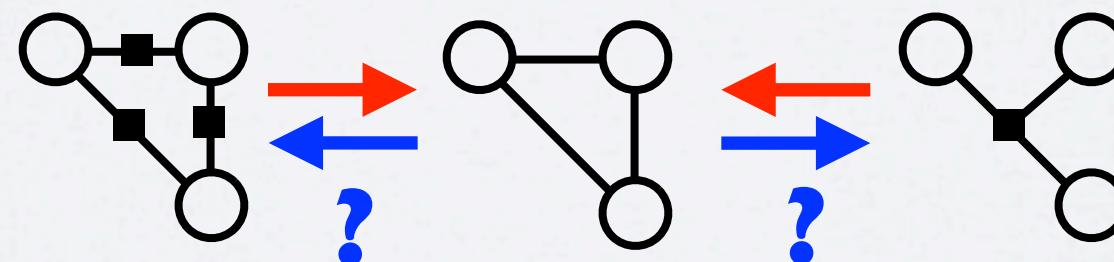
**Topics:** factor graph

- Factor graphs better represent factors



Factor graph 1      Markov network      Factor graph 2

- This is less ambiguous:



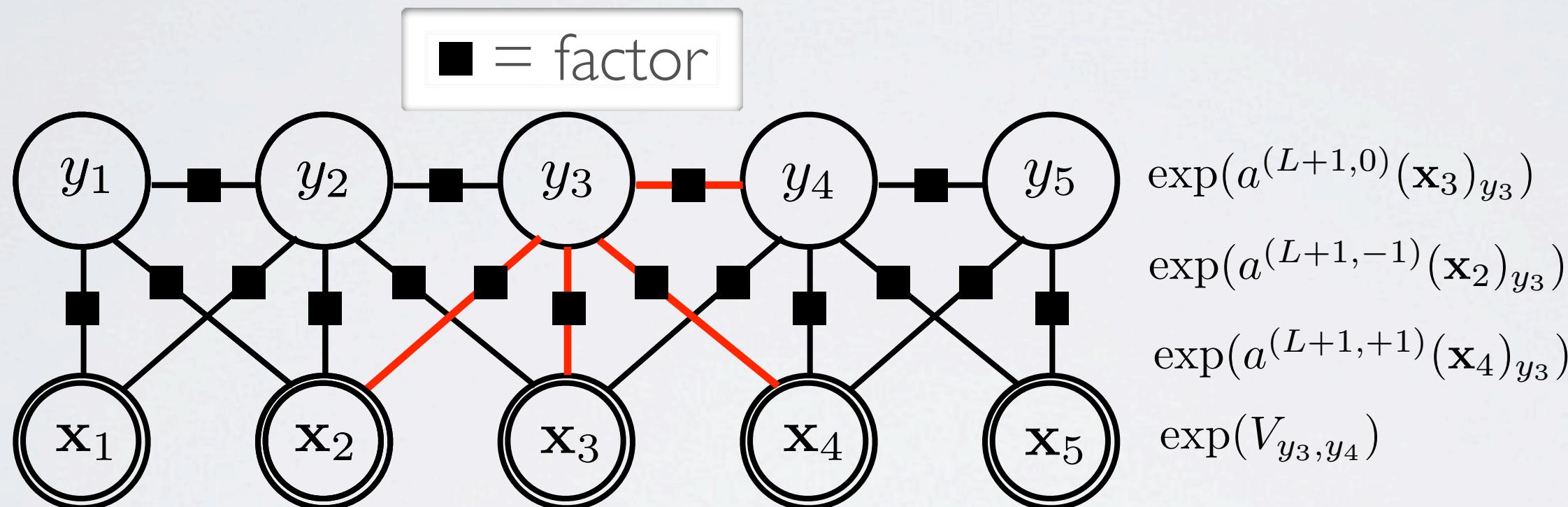
# Neural networks

Conditional random fields - belief propagation

# FACTOR GRAPH VISUALIZATION

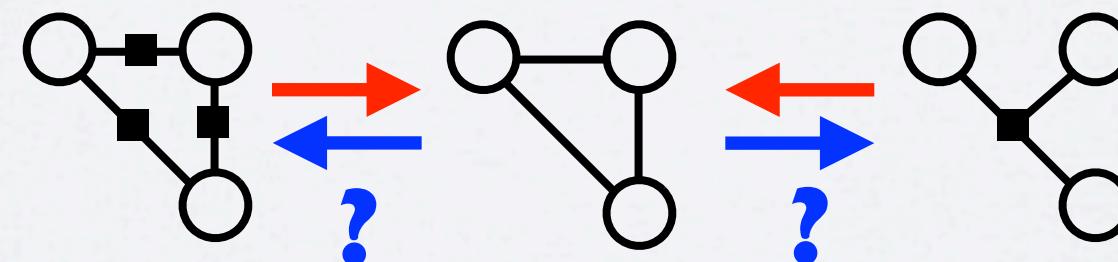
**Topics:** factor graph

- Factor graphs better represent factors



Factor graph 1      Markov network      Factor graph 2

- This is less ambiguous:

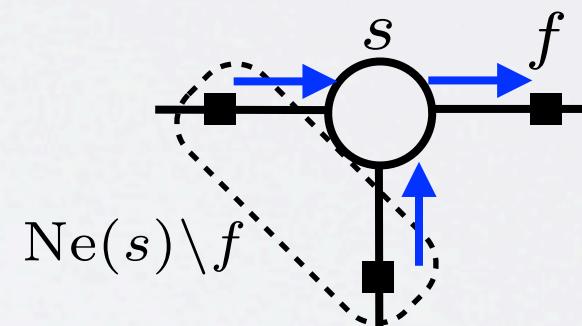


# BELIEF PROPAGATION

**Topics:** belief propagation, message passing

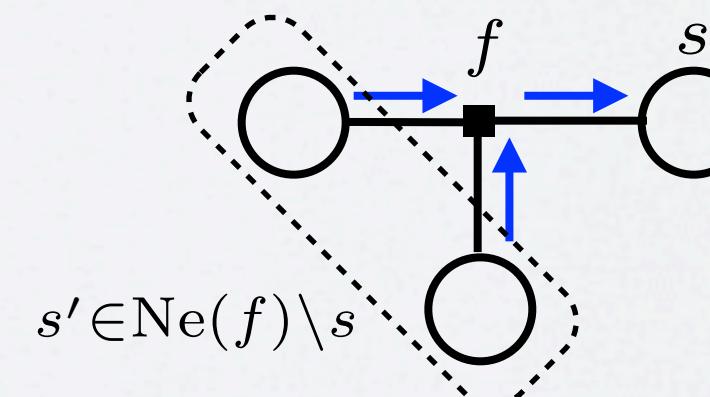
- Factor graphs better represent the computations needed to do inference
  - ▶ we can write the forward-backward algorithm seen before into a general message passing form
  - ▶ there are two types of message:
    - from a variable node ( $\circ$ ) to its neighbor factor nodes ( $\blacksquare$ ):

$$\mu_{s \rightarrow f}(i) = \prod_{f' \in \text{Ne}(s) \setminus f} \mu_{f' \rightarrow s}(i)$$



- from a factor node ( $\blacksquare$ ) to its neighbor variable nodes ( $\circ$ ):

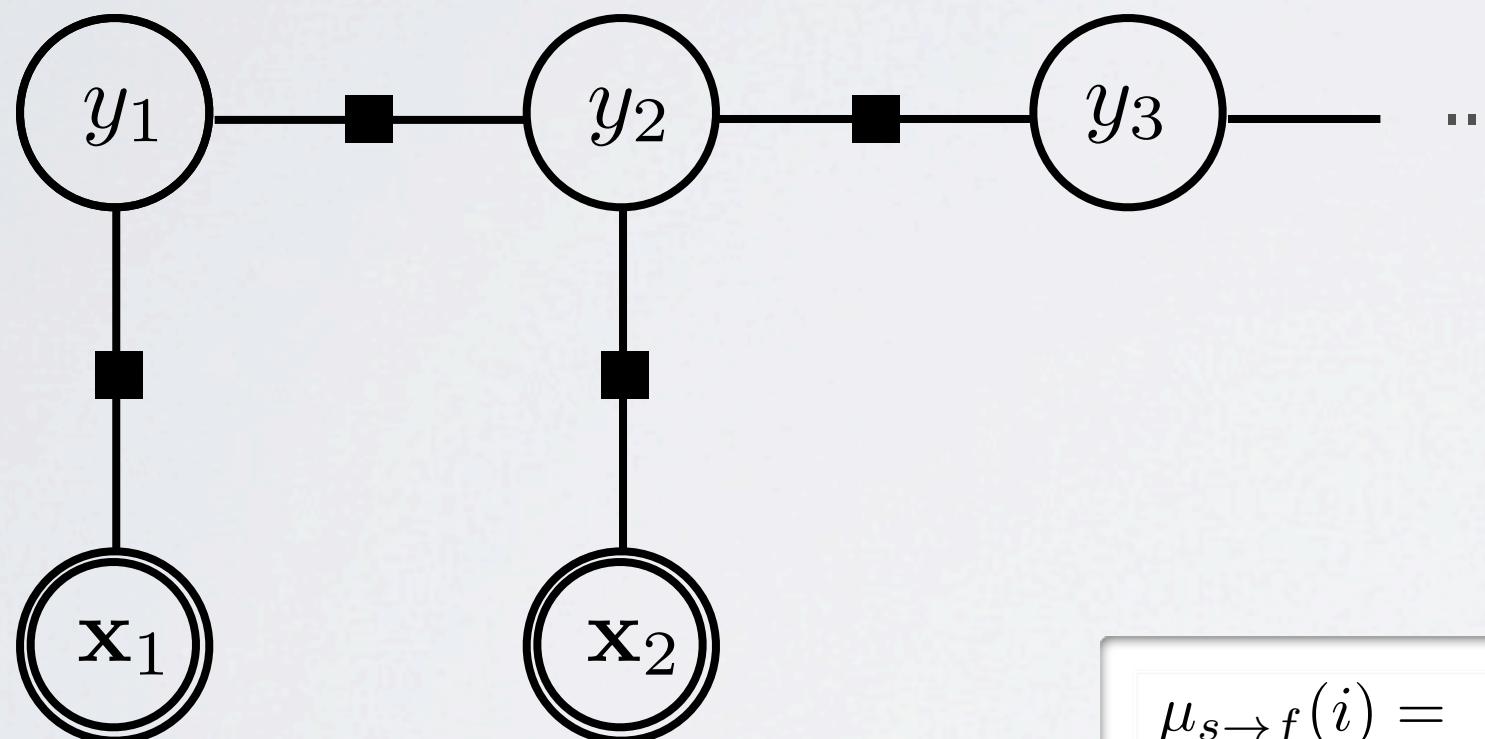
$$\mu_{f \rightarrow s}(i) = \sum_{\mathbf{z} \text{ is unobs. and } z_s = i} \phi_f(\mathbf{z}) \prod_{s' \in \text{Ne}(f) \setminus s} \mu_{s' \rightarrow f}(z_{s'})$$



# BELIEF PROPAGATION

**Topics:** belief propagation, message passing

- Example: our (simplified) linear chain CRF



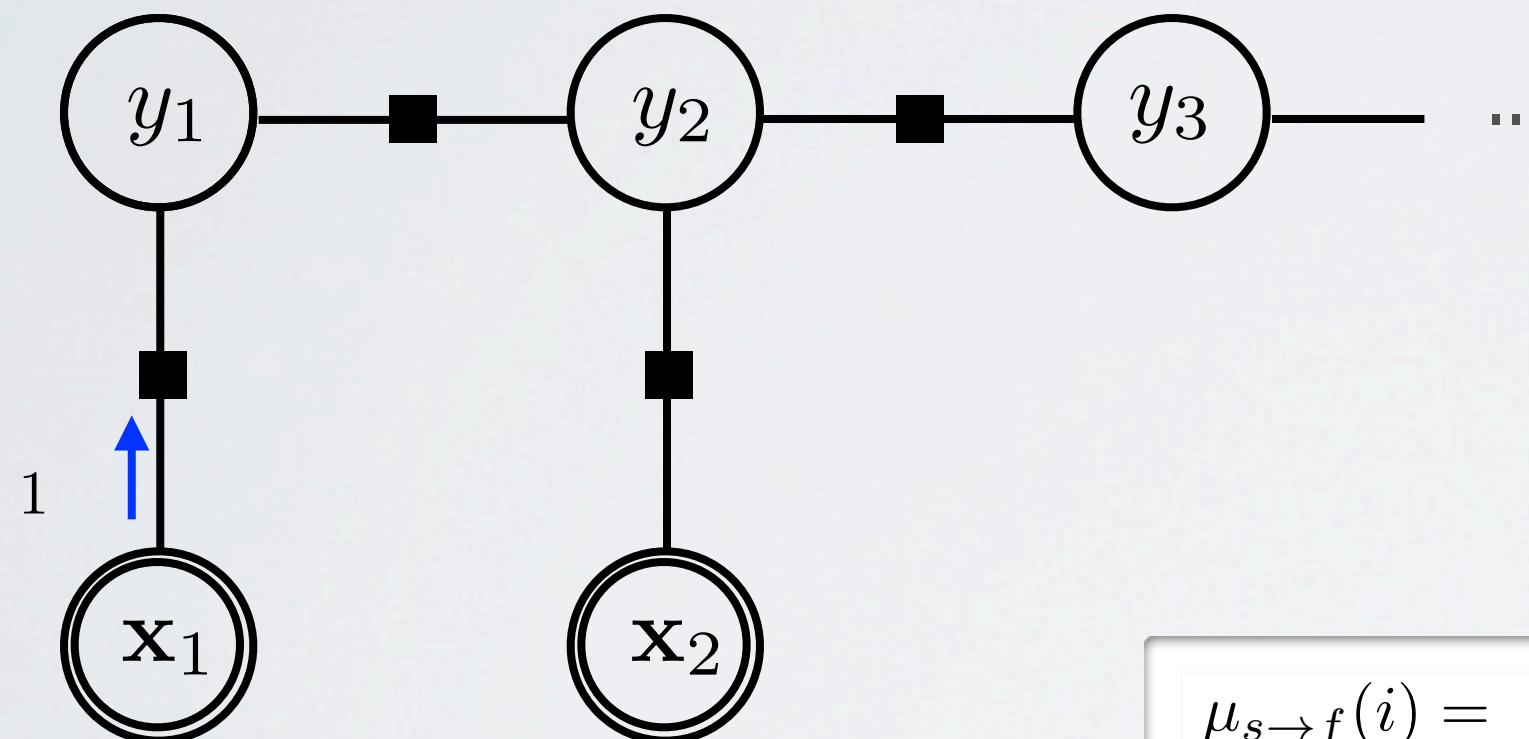
$$\mu_{s \rightarrow f}(i) = \prod_{f' \in \text{Ne}(s) \setminus f} \mu_{f' \rightarrow s}(i)$$

$$\mu_{f \rightarrow s}(i) = \sum_{\substack{\mathbf{z} \text{ is unobs.} \\ \text{and } z_s = i}} \phi_f(\mathbf{z}) \prod_{s' \in \text{Ne}(f) \setminus s} \mu_{s' \rightarrow f}(z_{s'})$$

# BELIEF PROPAGATION

**Topics:** belief propagation, message passing

- Example: our (simplified) linear chain CRF



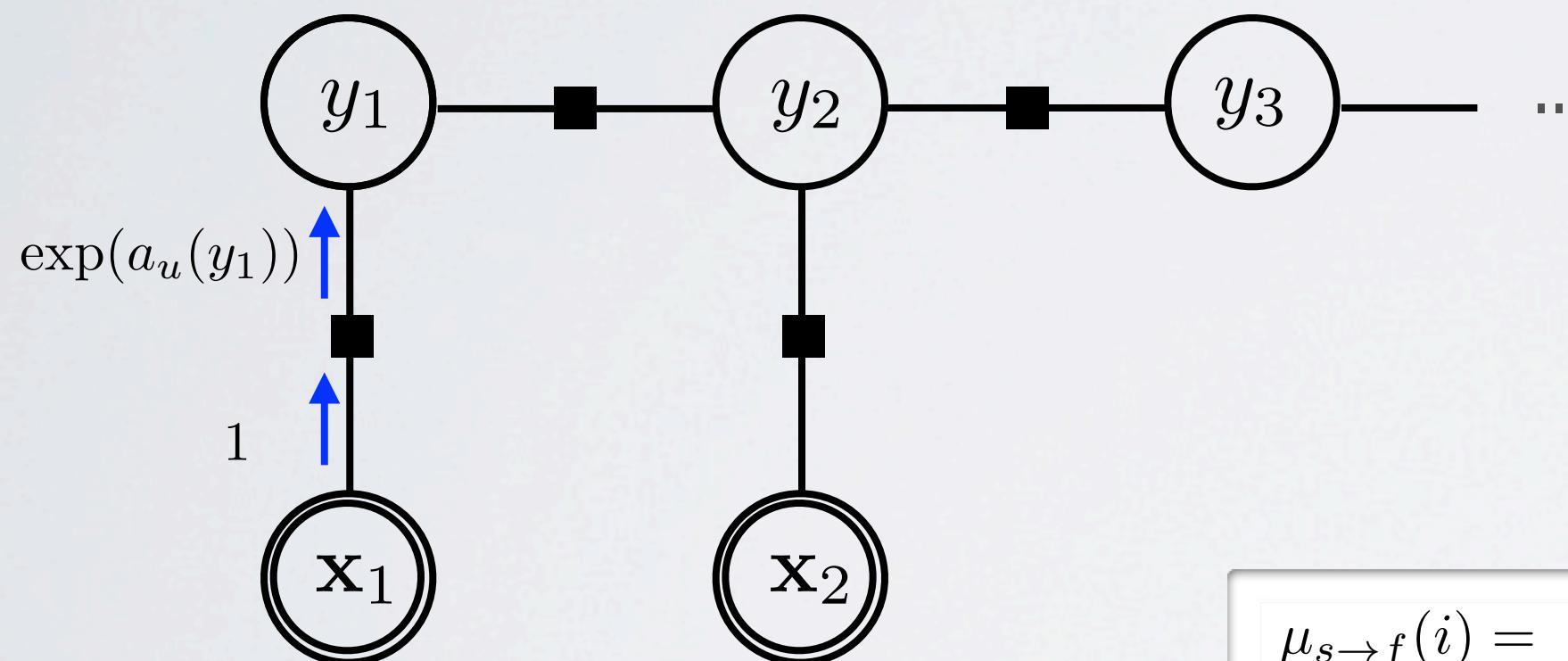
$$\mu_{s \rightarrow f}(i) = \prod_{f' \in \text{Ne}(s) \setminus f} \mu_{f' \rightarrow s}(i)$$

$$\mu_{f \rightarrow s}(i) = \sum_{\substack{\mathbf{z} \text{ is unobs.} \\ \text{and } z_s = i}} \phi_f(\mathbf{z}) \prod_{s' \in \text{Ne}(f) \setminus s} \mu_{s' \rightarrow f}(z_{s'})$$

# BELIEF PROPAGATION

**Topics:** belief propagation, message passing

- Example: our (simplified) linear chain CRF



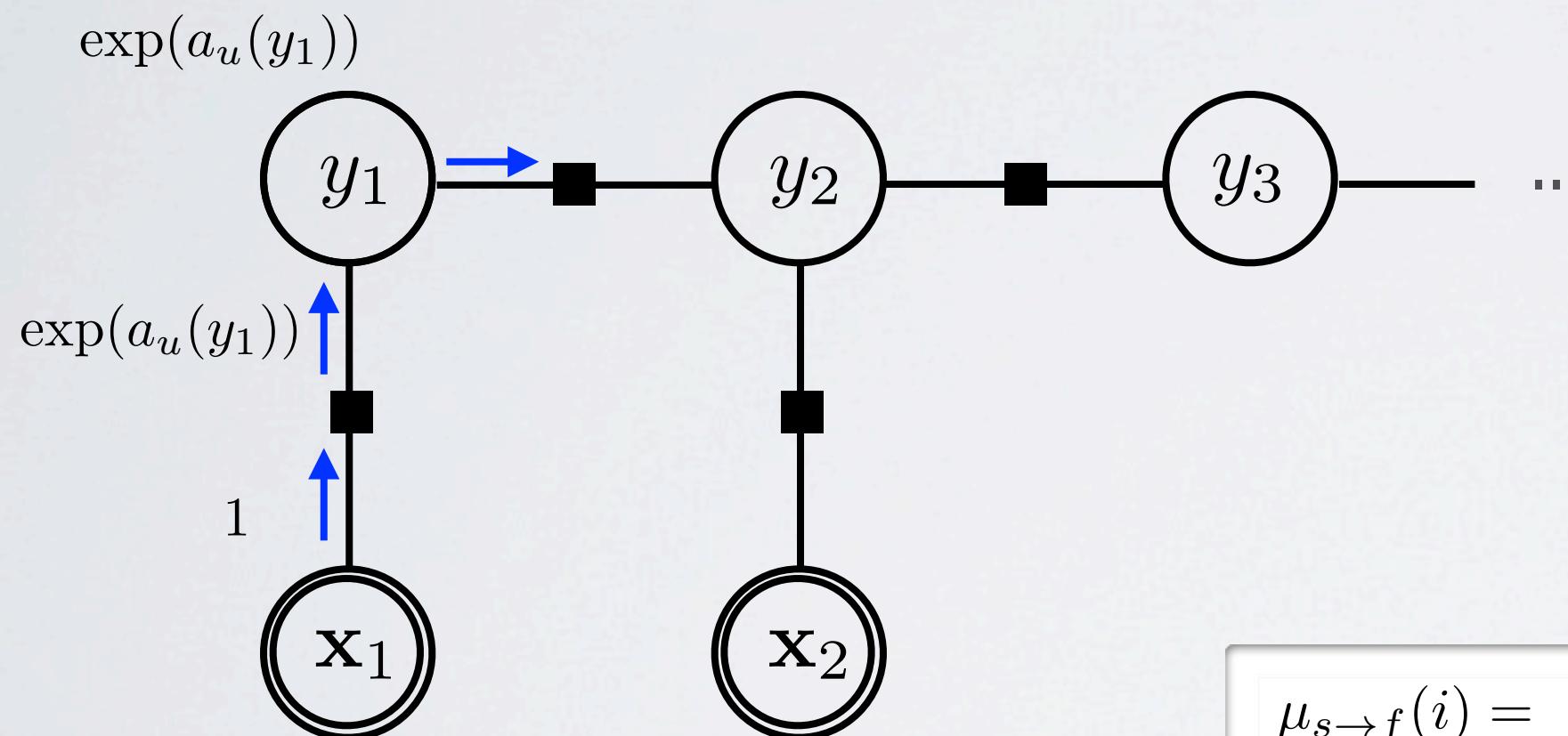
$$\mu_{s \rightarrow f}(i) = \prod_{f' \in \text{Ne}(s) \setminus f} \mu_{f' \rightarrow s}(i)$$

$$\mu_{f \rightarrow s}(i) = \sum_{\substack{\mathbf{z} \text{ is unobs.} \\ \text{and } z_s = i}} \phi_f(\mathbf{z}) \prod_{s' \in \text{Ne}(f) \setminus s} \mu_{s' \rightarrow f}(z_{s'})$$

# BELIEF PROPAGATION

**Topics:** belief propagation, message passing

- Example: our (simplified) linear chain CRF



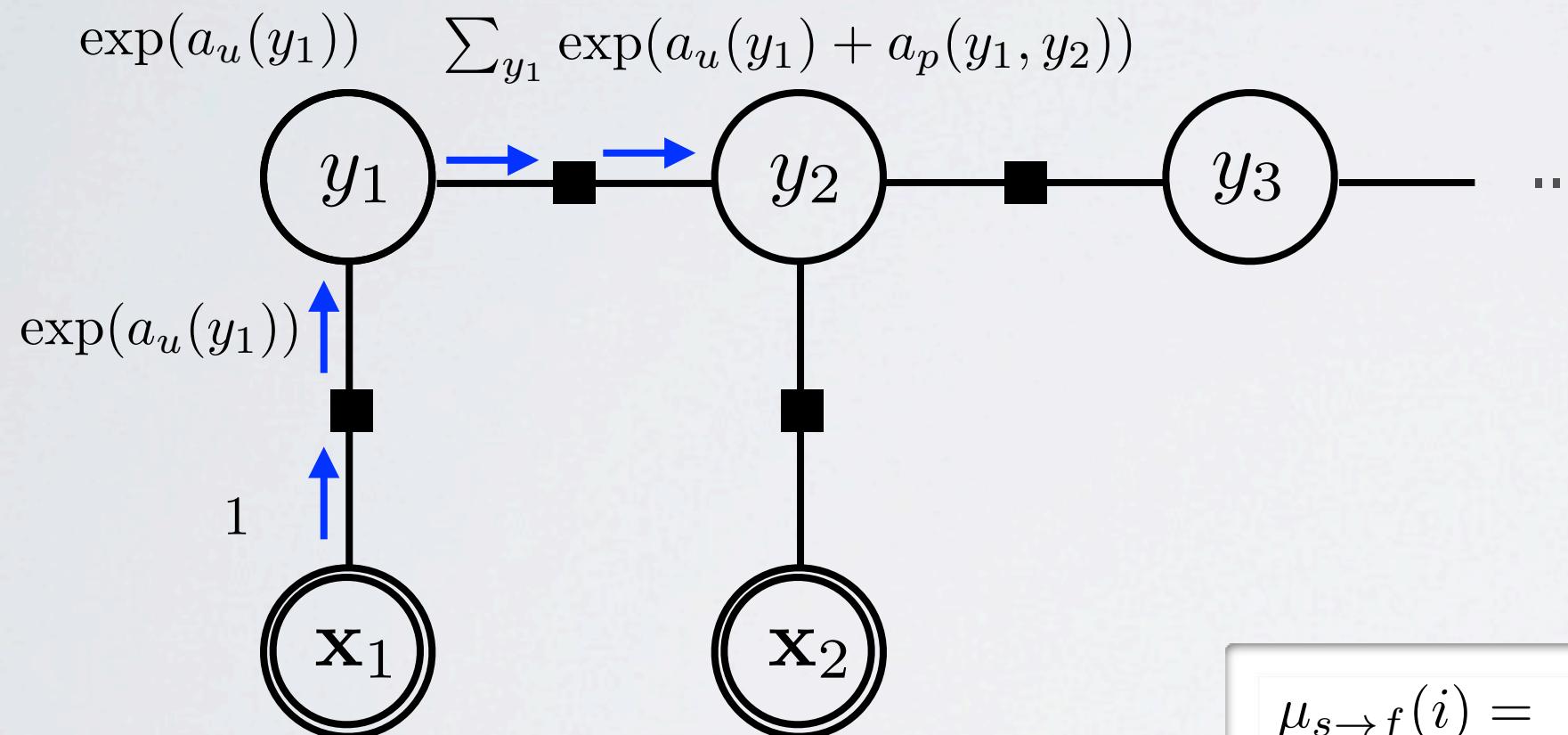
$$\mu_{s \rightarrow f}(i) = \prod_{f' \in \text{Ne}(s) \setminus f} \mu_{f' \rightarrow s}(i)$$

$$\mu_{f \rightarrow s}(i) = \sum_{\substack{\mathbf{z} \text{ is unobs.} \\ \text{and } z_s = i}} \phi_f(\mathbf{z}) \prod_{s' \in \text{Ne}(f) \setminus s} \mu_{s' \rightarrow f}(z_{s'})$$

# BELIEF PROPAGATION

**Topics:** belief propagation, message passing

- Example: our (simplified) linear chain CRF



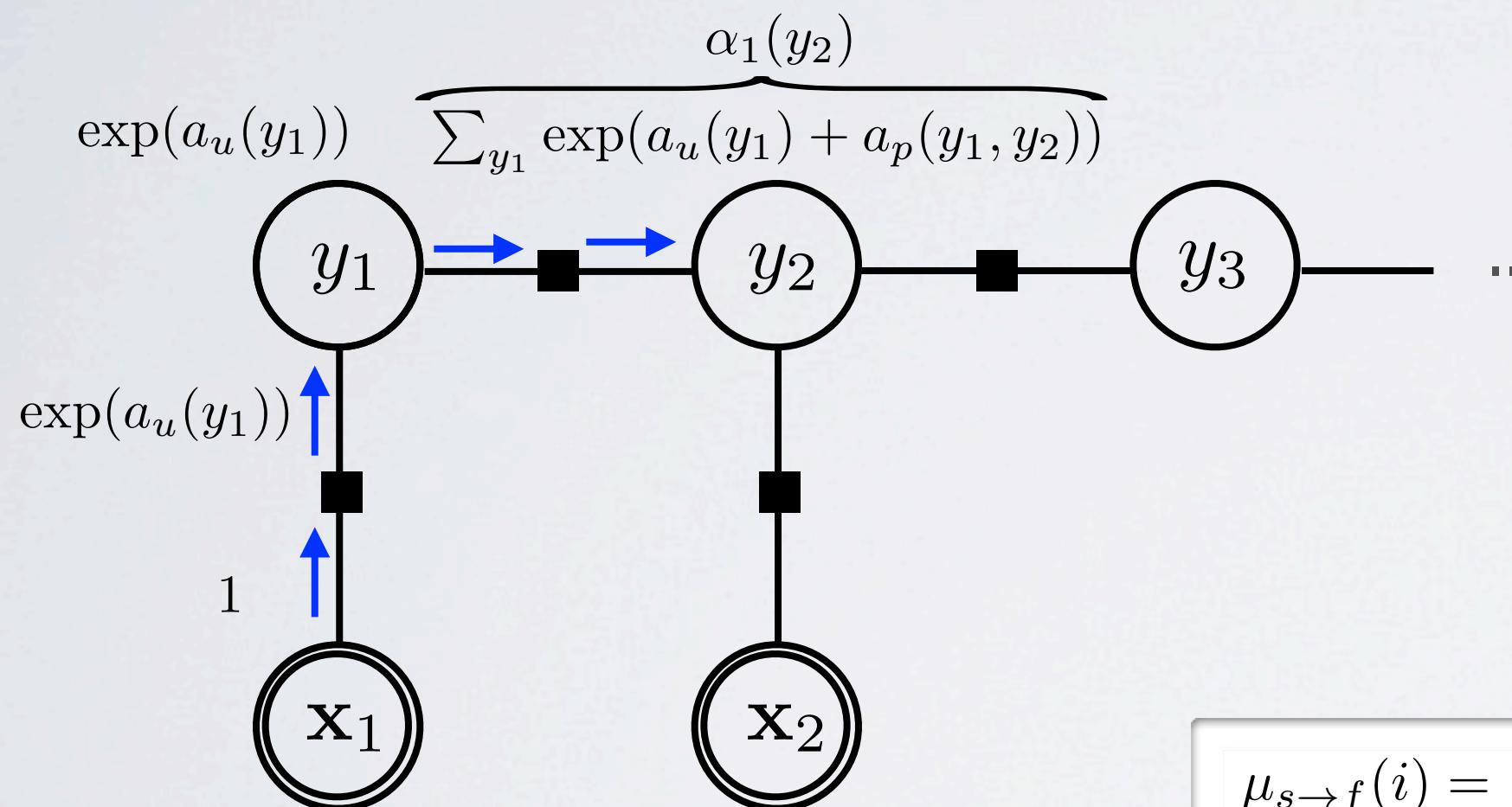
$$\mu_{s \rightarrow f}(i) = \prod_{f' \in \text{Ne}(s) \setminus f} \mu_{f' \rightarrow s}(i)$$

$$\mu_{f \rightarrow s}(i) = \sum_{\substack{\mathbf{z} \text{ is unobs.} \\ \text{and } z_s = i}} \phi_f(\mathbf{z}) \prod_{s' \in \text{Ne}(f) \setminus s} \mu_{s' \rightarrow f}(z_{s'})$$

# BELIEF PROPAGATION

**Topics:** belief propagation, message passing

- Example: our (simplified) linear chain CRF



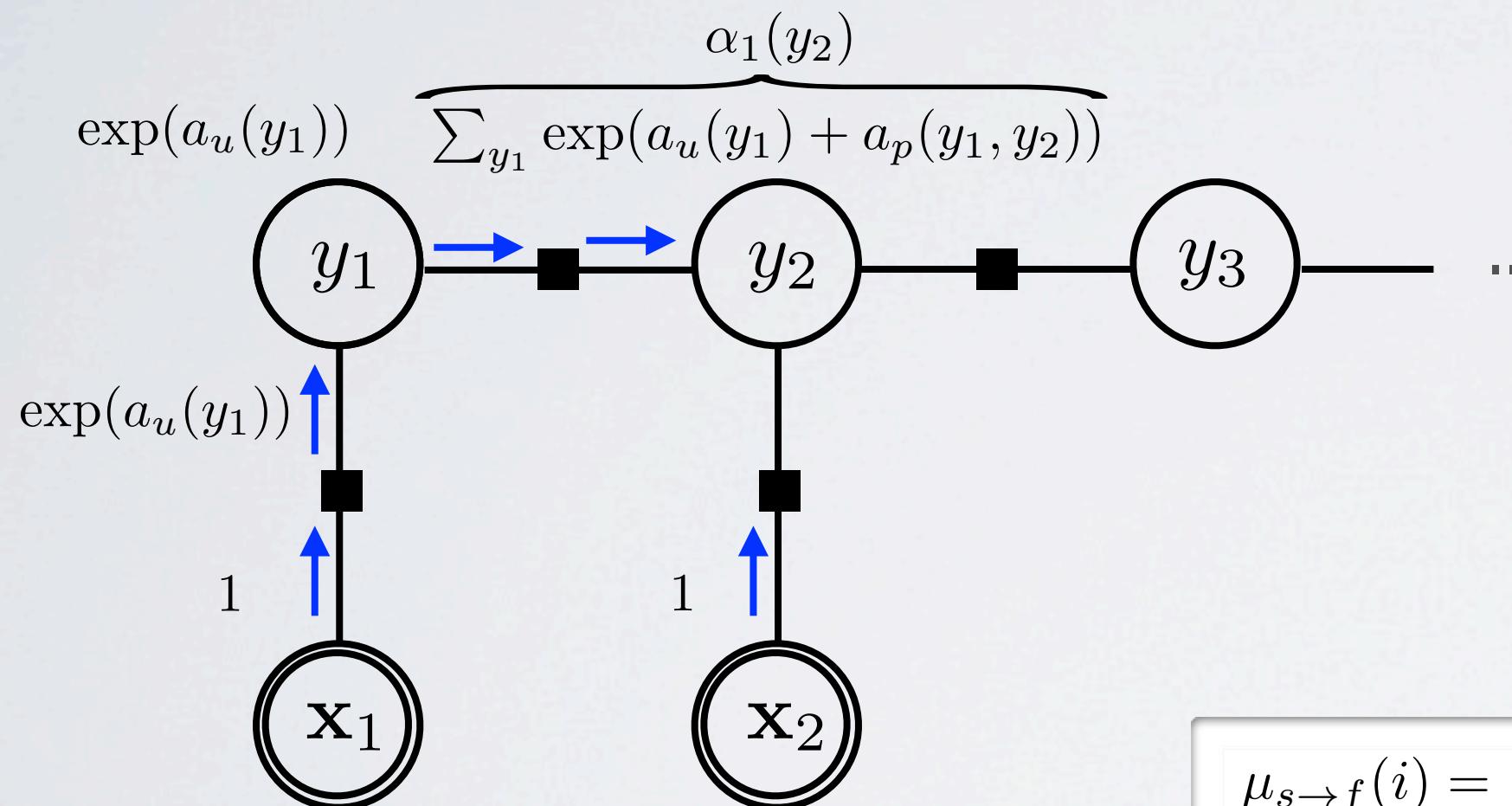
$$\mu_{s \rightarrow f}(i) = \prod_{f' \in \text{Ne}(s) \setminus f} \mu_{f' \rightarrow s}(i)$$

$$\mu_{f \rightarrow s}(i) = \sum_{\substack{\mathbf{z} \text{ is unobs.} \\ \text{and } z_s = i}} \phi_f(\mathbf{z}) \prod_{s' \in \text{Ne}(f) \setminus s} \mu_{s' \rightarrow f}(z_{s'})$$

# BELIEF PROPAGATION

**Topics:** belief propagation, message passing

- Example: our (simplified) linear chain CRF



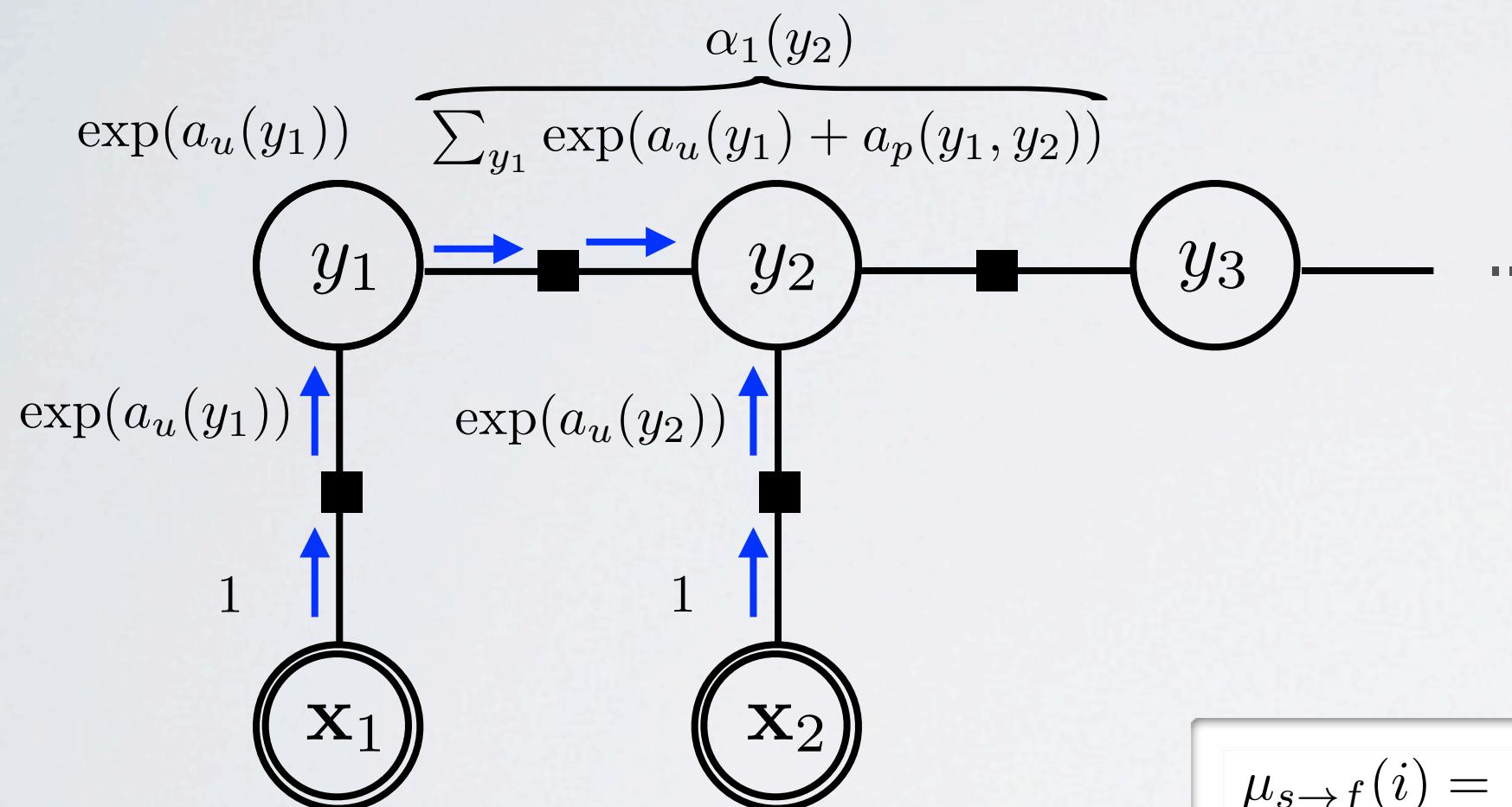
$$\mu_{s \rightarrow f}(i) = \prod_{f' \in \text{Ne}(s) \setminus f} \mu_{f' \rightarrow s}(i)$$

$$\mu_{f \rightarrow s}(i) = \sum_{\substack{\mathbf{z} \text{ is unobs.} \\ \text{and } z_s = i}} \phi_f(\mathbf{z}) \prod_{s' \in \text{Ne}(f) \setminus s} \mu_{s' \rightarrow f}(z_{s'})$$

# BELIEF PROPAGATION

**Topics:** belief propagation, message passing

- Example: our (simplified) linear chain CRF



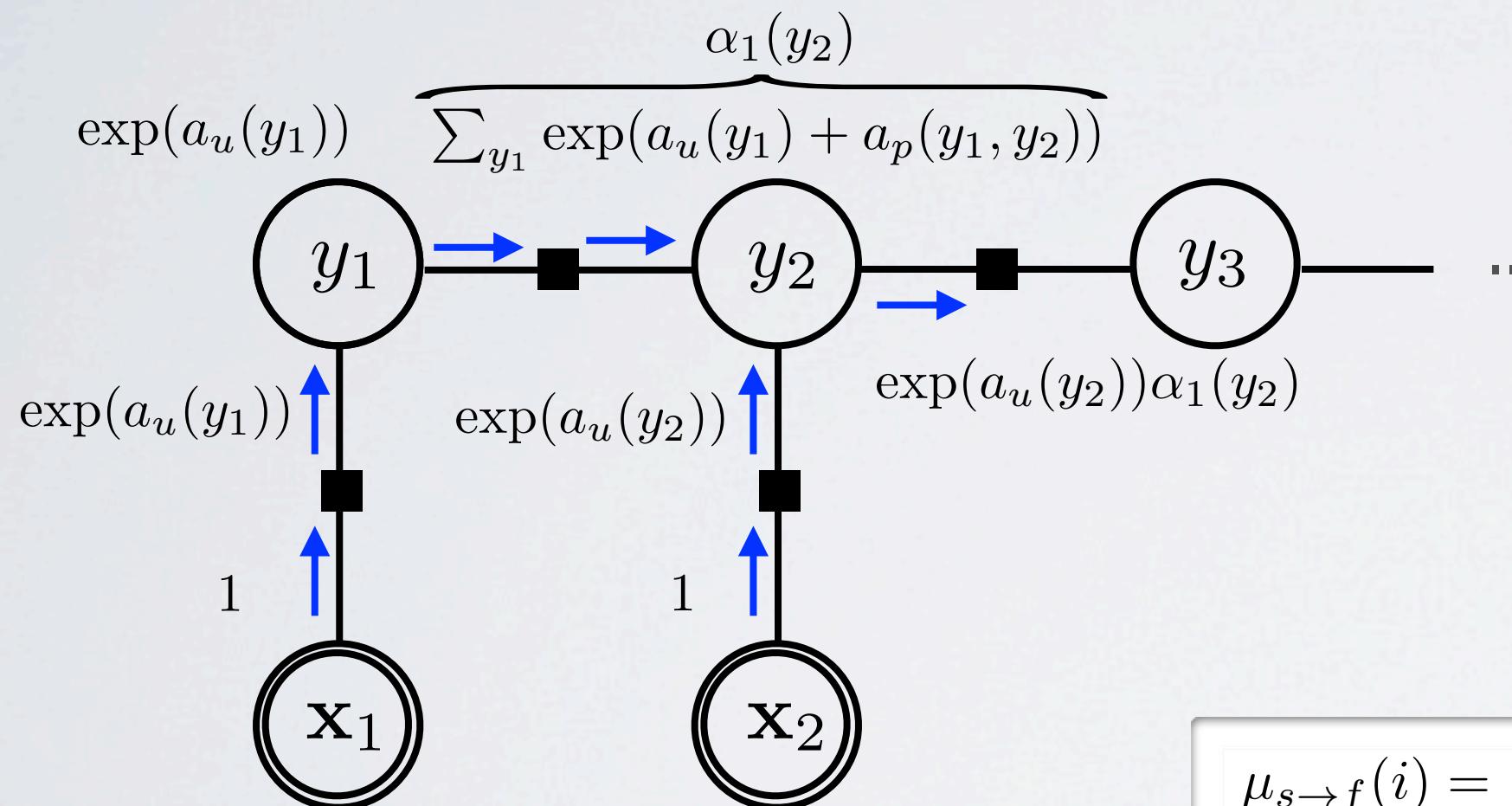
$$\mu_{s \rightarrow f}(i) = \prod_{f' \in \text{Ne}(s) \setminus f} \mu_{f' \rightarrow s}(i)$$

$$\mu_{f \rightarrow s}(i) = \sum_{\substack{\mathbf{z} \text{ is unobs.} \\ \text{and } z_s = i}} \phi_f(\mathbf{z}) \prod_{s' \in \text{Ne}(f) \setminus s} \mu_{s' \rightarrow f}(z_{s'})$$

# BELIEF PROPAGATION

**Topics:** belief propagation, message passing

- Example: our (simplified) linear chain CRF



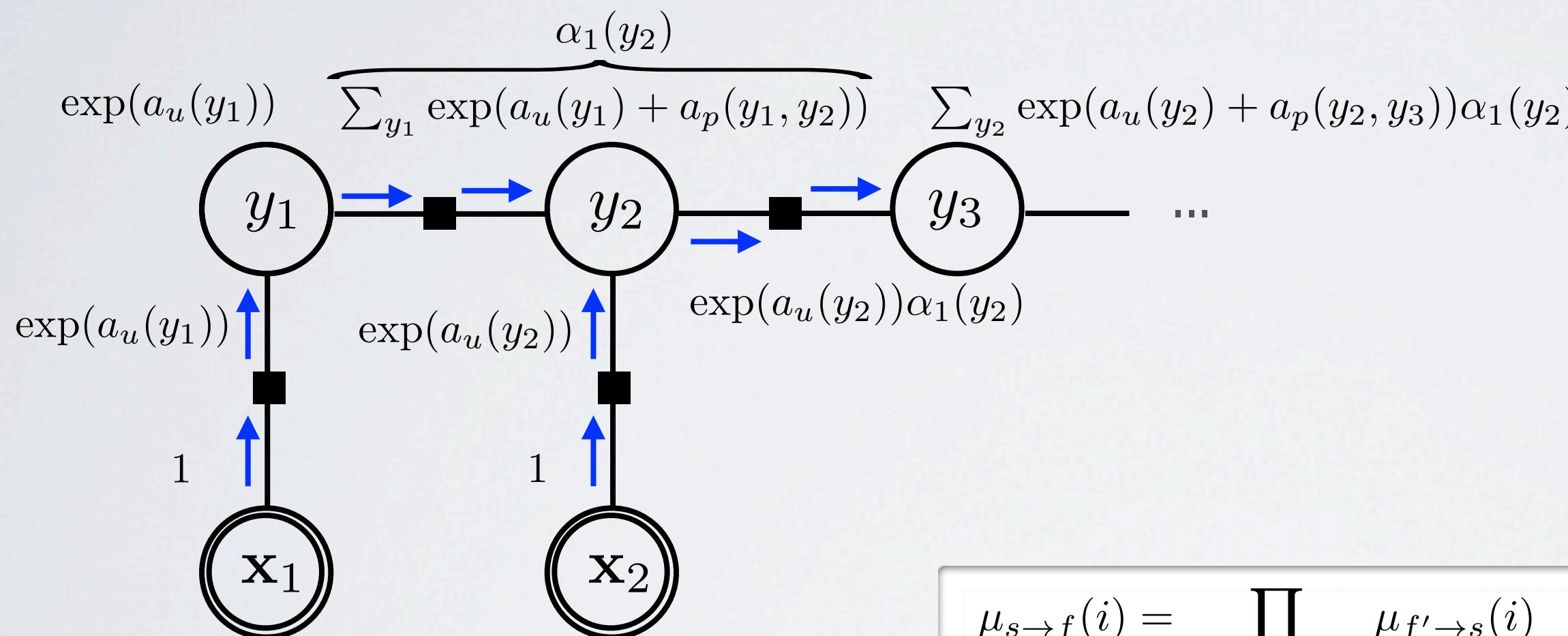
$$\mu_{s \rightarrow f}(i) = \prod_{f' \in \text{Ne}(s) \setminus f} \mu_{f' \rightarrow s}(i)$$

$$\mu_{f \rightarrow s}(i) = \sum_{\substack{\mathbf{z} \text{ is unobs.} \\ \text{and } z_s = i}} \phi_f(\mathbf{z}) \prod_{s' \in \text{Ne}(f) \setminus s} \mu_{s' \rightarrow f}(z_{s'})$$

# BELIEF PROPAGATION

**Topics:** belief propagation, message passing

- Example: our (simplified) linear chain CRF



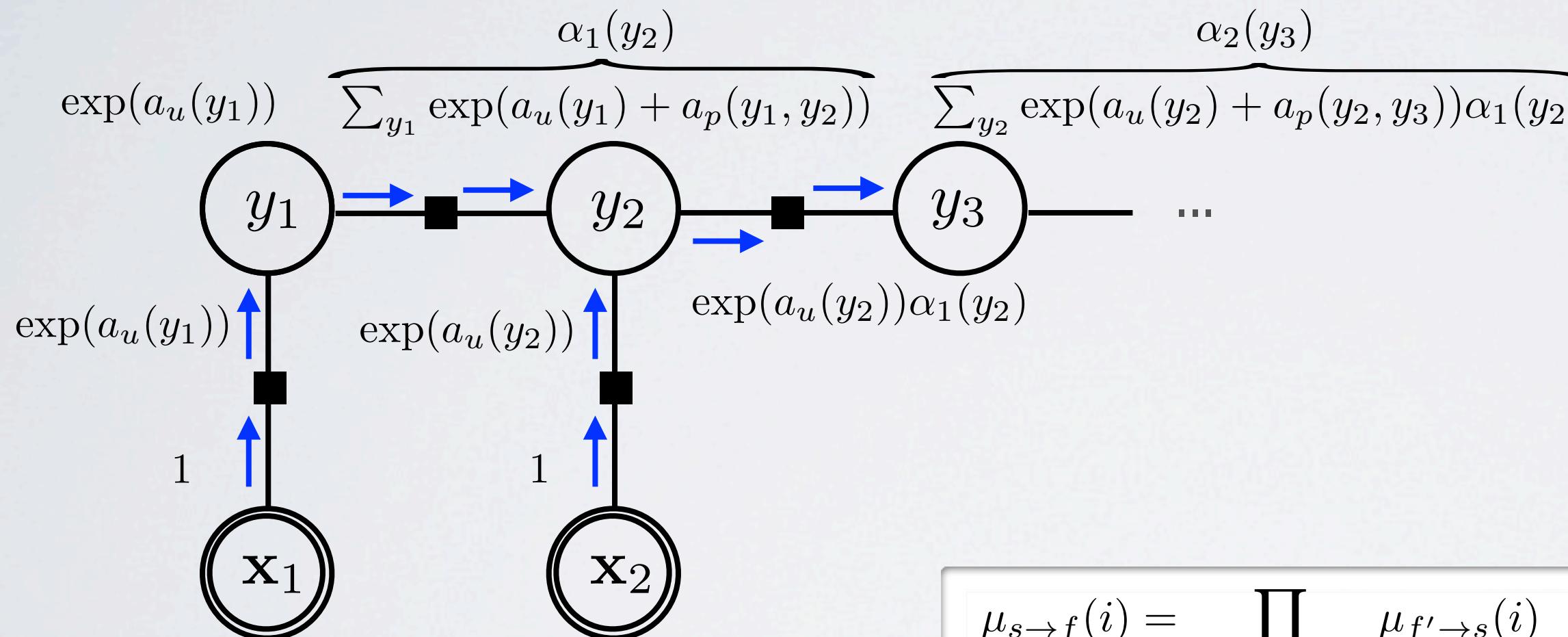
$$\mu_{s \rightarrow f}(i) = \prod_{f' \in \text{Ne}(s) \setminus f} \mu_{f' \rightarrow s}(i)$$

$$\mu_{f \rightarrow s}(i) = \sum_{\substack{\mathbf{z} \text{ is unobs.} \\ \text{and } z_s = i}} \phi_f(\mathbf{z}) \prod_{s' \in \text{Ne}(f) \setminus s} \mu_{s' \rightarrow f}(z_{s'})$$

# BELIEF PROPAGATION

**Topics:** belief propagation, message passing

- Example: our (simplified) linear chain CRF



$$\mu_{s \rightarrow f}(i) = \prod_{f' \in \text{Ne}(s) \setminus f} \mu_{f' \rightarrow s}(i)$$

$$\mu_{f \rightarrow s}(i) = \sum_{\substack{\mathbf{z} \text{ is unobs.} \\ \text{and } z_s = i}} \phi_f(\mathbf{z}) \prod_{s' \in \text{Ne}(f) \setminus s} \mu_{s' \rightarrow f}(z_{s'})$$

# (LOOPY) BELIEF PROPAGATION

**Topics:** belief propagation, message passing

- On a linear chain graph, belief propagation is the same a forward-backward
  - ▶ forward pass of message passing computes the  $\alpha_k(y_{k+1})$
  - ▶ backward pass of message passing computes the  $\beta_k(y_{k-1})$
- For numerical stability, passing log-messages is preferred
- Can do inference on other types of structures
  - ▶ belief propagation is also exact on arbitrary trees
  - ▶ on a graph with loops, (loopy-)belief propagation can be used to do approximate inference (but can diverge, if not careful)
  - ▶ many general purpose libraries are publicly available

# LINEAR CHAN CRF

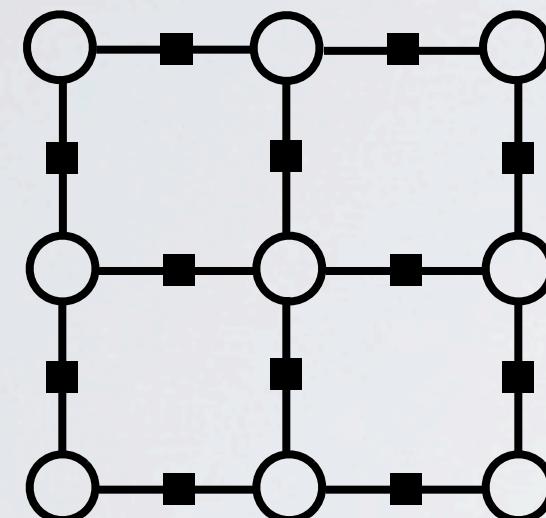
**Topics:** other variations on linear chain CRF

- We could add lateral connections between labels that are at 2 positions away  $\phi_f(y_k, y_{k+2})$
- We could add lateral connections for triplets of labels  $\phi_f(y_k, y_{k+1}, y_{k+2})$
- The idea is to add connections between things to model their dependency more directly
- The connectivity can even change between examples

# GENERAL CRF

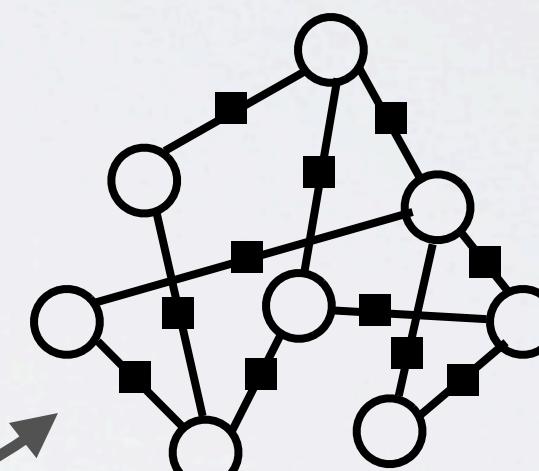
## Topics: CRFs in general

- We don't have to restrict the CRF structure to linear chains



Grid structure  
(pixels in image)

$$p(\mathbf{y}|\mathbf{X}) = \frac{1}{Z(\mathbf{X})} \prod_f \Psi_f(\mathbf{y}, \mathbf{X})$$



General pair-wise structure  
(webpages sharing a link)

- We could also have  $n$ -ary factors, with  $n > 2$

# (LOOPY) BELIEF PROPAGATION

**Topics:** CRFs in general

- Marginals can be approximated with:

$$p(y_k | \mathbf{X}) = \frac{\exp(\log \phi_f(y_k) + \sum_{f' \in \text{Ne}(k) \setminus f} \log \mu_{f' \rightarrow k}(y_k))}{\sum_{y'_k} \exp(\log \phi_f(y'_k) + \sum_{f' \in \text{Ne}(k) \setminus f} \log \mu_{f' \rightarrow k}(y'_k))}$$

- In general, an approximated marginal is computed by
  1. summing all the log-factors that involve only the  $y_k$  variables of interest
  2. summing all the log-messages coming into the  $y_k$  variables from other factors
  3. exponentiating
  4. renormalizing