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

Machine Learning Department
School of Computer Science
Carnegie Mellon University

Learning MRFs / CRFs

Matt Gormley
Lecture 10
Sep. 30, 2019

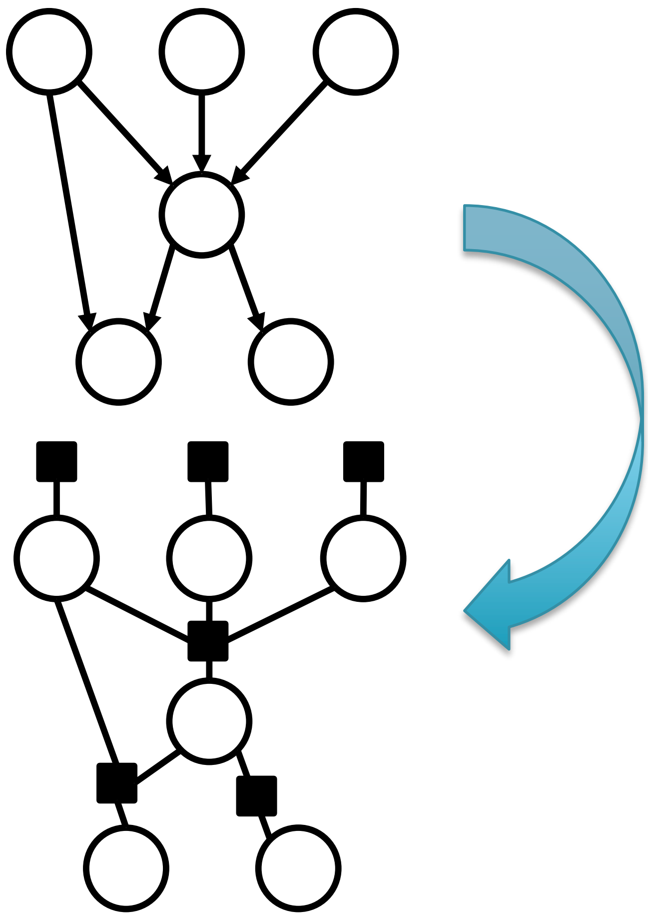
Q&A

Q: How do we convert UGMs to factor graphs again?

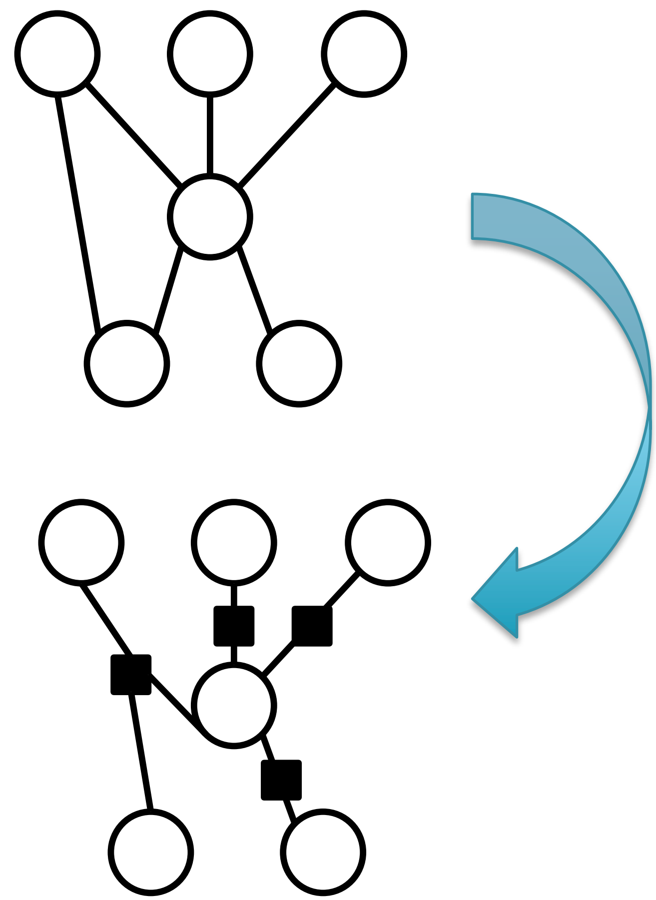
A: Oops! There was a mistake in my slide… see the fix on the next slide.

Converting to Factor Graphs

Each conditional and marginal distribution in a **directed GM** becomes a factor



Each maximal clique in an **undirected GM** becomes a factor



Reminders

- **Homework 1: DAgger for seq2seq**
  - Out: Thu, Sep. 12
  - Due: Thu, Sep. 26 at 11:59pm
- **Homework 2: BP for Syntax Trees**
  - Out: Sat, Sep. 28
  - Due: Sat, Oct. 12 at 11:59pm

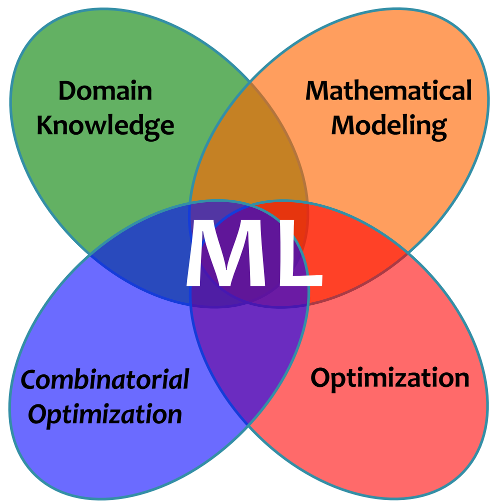
LEARNING FOR MRFS

Machine Learning

The **data** inspires the structures we want to predict

Our **model** defines a score for each structure

It also tells us what to optimize



Learning tunes the parameters of the model

Inference finds {best structure, marginals, partition function} for a new observation

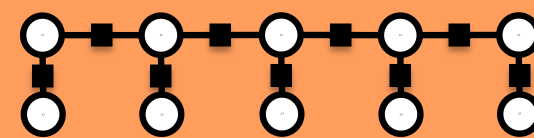
(**Inference** is usually called as a subroutine in learning)

1. Data

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

2. Model

$$p(\boldsymbol{x} \mid \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \prod_{C \in \mathcal{C}} \psi_C(\boldsymbol{x}_C)$$



3. Objective

$$\ell(\theta; \mathcal{D}) = \sum_{n=1}^{N} \log p(\boldsymbol{x}^{(n)} \mid \boldsymbol{\theta})$$

5. Inference

1. Marginal Inference

$$p(\boldsymbol{x}_C) = \sum_{\boldsymbol{x}' : \boldsymbol{x}'_C = \boldsymbol{x}_C} p(\boldsymbol{x}' \mid \boldsymbol{\theta})$$

2. Partition Function

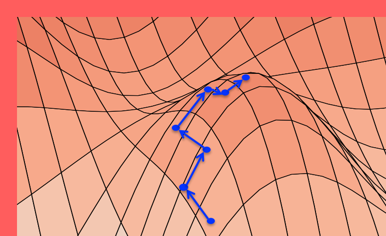
$$Z(\boldsymbol{\theta}) = \sum_{\boldsymbol{x}} \prod_{C \in \mathcal{C}} \psi_C(\boldsymbol{x}_C)$$

3. MAP Inference

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\operatorname{argmax}}\ p(\boldsymbol{x} \mid \boldsymbol{\theta})$$

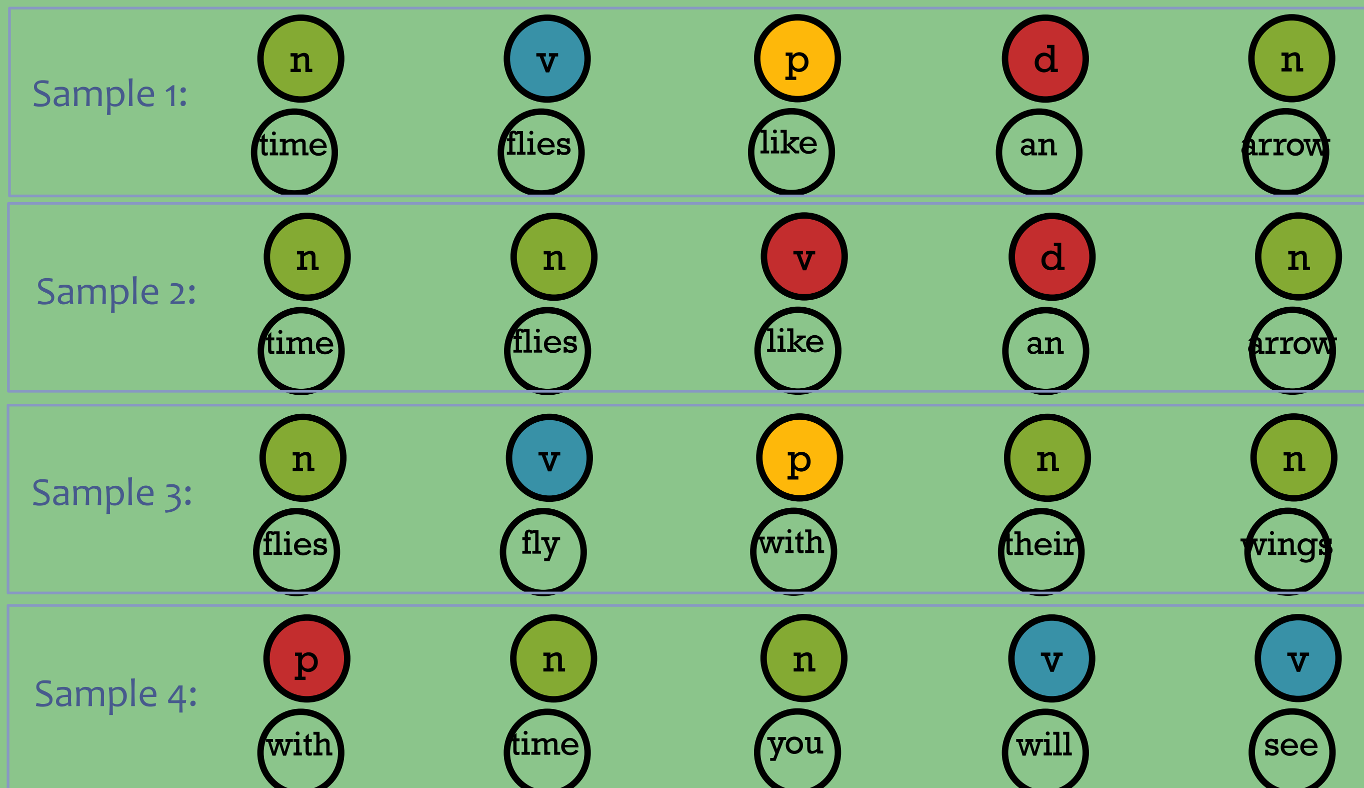
4. Learning

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \ell(\boldsymbol{\theta}; \mathcal{D})$$

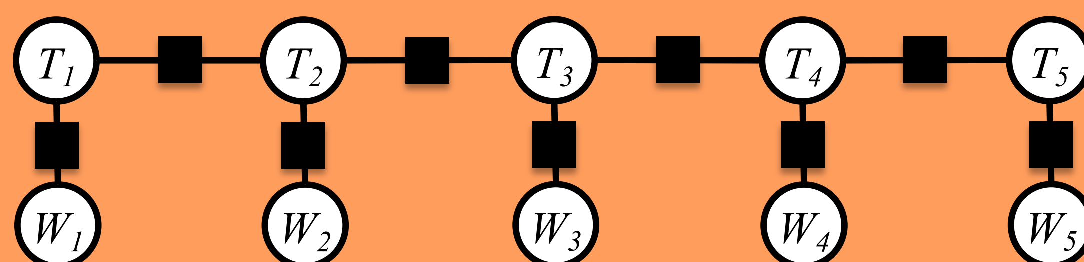


1. Data

Given training examples: $\mathcal{D} = \{\boldsymbol{x}^{(n)}\}_{n=1}^{N}$



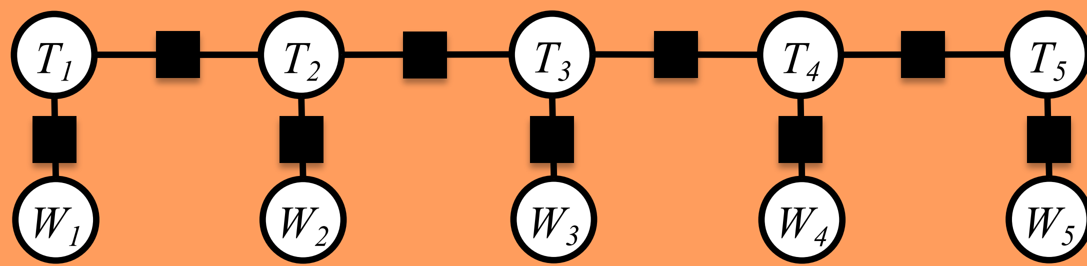
2. Model



2. Model

Define the model to be an MRF:

$$p(\boldsymbol{x} \mid \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \prod_{C \in \mathcal{C}} \psi_C(\boldsymbol{x}_C)$$



3. Objective

Choose the objective to be log-likelihood:

(Assign high probability to the things we observe and low probability to everything else)

$$\ell(\theta; \mathcal{D}) = \sum_{n=1}^{N} \log p(\boldsymbol{x}^{(n)} \mid \boldsymbol{\theta})$$

3. Objective

Choose the objective to be log-likelihood:

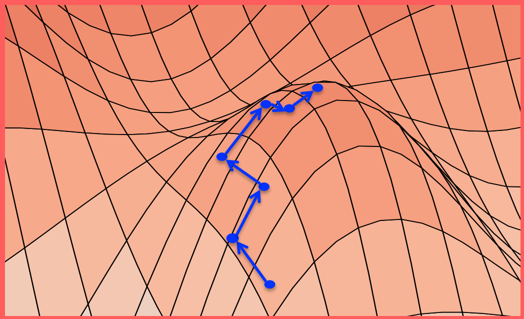
(Assign high probability to the things we observe and low probability to everything else)

$$\ell(\theta; \mathcal{D}) = \sum_{n=1}^{N} \log p(\boldsymbol{x}^{(n)} \mid \boldsymbol{\theta})$$

4. Learning

Tune the parameters to maximize the objective function

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmax}} \, \ell(\boldsymbol{\theta}; \mathcal{D})$$



3. Objective

Choose the objective to be log-likelihood:

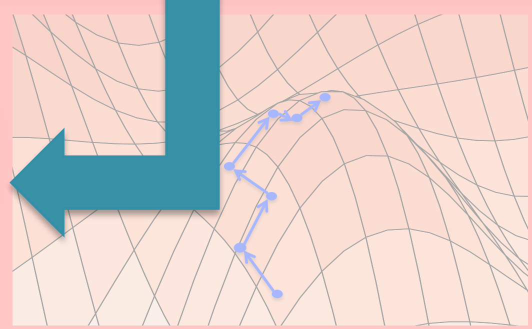
(Assign high probability to the things we observe and low probability to everything else)

Goals for Today's Lecture

1. Consider different parameterizations
2. Optimize this objective function

Tune the parameter function

$$\theta^* = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \ell(\boldsymbol{\theta}; \mathcal{D})$$



5. Inference

Three Tasks:

1. Marginal Inference

Compute marginals of variables and cliques

$$p(x_i) = \sum_{\boldsymbol{x}' : x'_i = x_i} p(\boldsymbol{x}' \mid \boldsymbol{\theta}) \quad \Bigg| \quad p(\boldsymbol{x}_C) = \sum_{\boldsymbol{x}' : \boldsymbol{x}'_C = \boldsymbol{x}_C} p(\boldsymbol{x}' \mid \boldsymbol{\theta})$$

2. Partition Function

Compute the normalization constant

$$Z(\boldsymbol{\theta}) = \sum_{\boldsymbol{x}} \prod_{C \in \mathcal{C}} \psi_C(\boldsymbol{x}_C)$$

3. MAP Inference

Compute variable assignment with highest probability

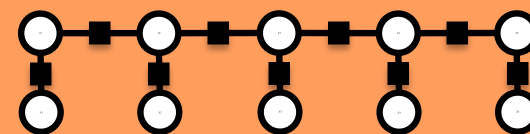
$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\operatorname{argmax}} \; p(\boldsymbol{x} \mid \boldsymbol{\theta})$$

1. Data

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

2. Model

$$p(\boldsymbol{x} \mid \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \prod_{C \in \mathcal{C}} \psi_C(\boldsymbol{x}_C)$$



3. Objective

$$\ell(\theta; \mathcal{D}) = \sum_{n=1}^{N} \log p(\boldsymbol{x}^{(n)} \mid \boldsymbol{\theta})$$

5. Inference

1. Marginal Inference

$$p(\boldsymbol{x}_C) = \sum_{\boldsymbol{x}' : \boldsymbol{x}'_C = \boldsymbol{x}_C} p(\boldsymbol{x}' \mid \boldsymbol{\theta})$$

2. Partition Function

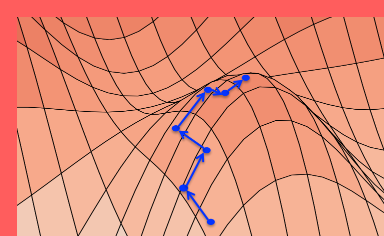
$$Z(\boldsymbol{\theta}) = \sum_{\boldsymbol{x}} \prod_{C \in \mathcal{C}} \psi_C(\boldsymbol{x}_C)$$

3. MAP Inference

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x}}{\operatorname{argmax}}\; p(\boldsymbol{x} \mid \boldsymbol{\theta})$$

4. Learning

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\, \ell(\boldsymbol{\theta}; \mathcal{D})$$



MLE for Undirected GMs

- Today's parameter estimation assumptions:
  1. The graphical model structure is given
  2. Every variable appears in the training examples

Questions

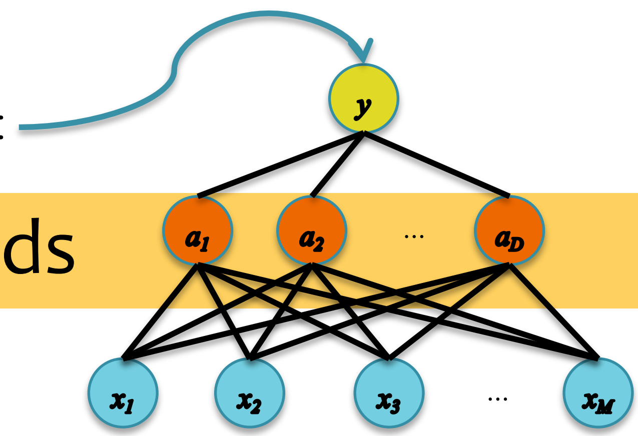
1. What does the **likelihood objective** accomplish?
2. Is likelihood the ***right* objective** function?
3. **How do we optimize** the objective function (i.e. learn)?
4. What **guarantees** does the optimizer provide?
5. (What is the **mapping from data → model**? In what ways can we incorporate our domain knowledge? How does this impact learning?)

Options for MLE of MRFs

- **Setting I:** $\psi_C(\boldsymbol{x}_C) = \theta_{C,\boldsymbol{x}_C}$
  A. MLE by inspection (Decomposable Models)
  B. Iterative Proportional Fitting (IPF)

- **Setting II:** $\psi_C(\boldsymbol{x}_C) = \exp(\boldsymbol{\theta} \cdot \boldsymbol{f}(\boldsymbol{x}_C))$
  C. Generalized Iterative Scaling
  D. Gradient-based Methods

- **Setting III:** $\psi_C(\boldsymbol{x}_C) =$
  E. Gradient-based Methods



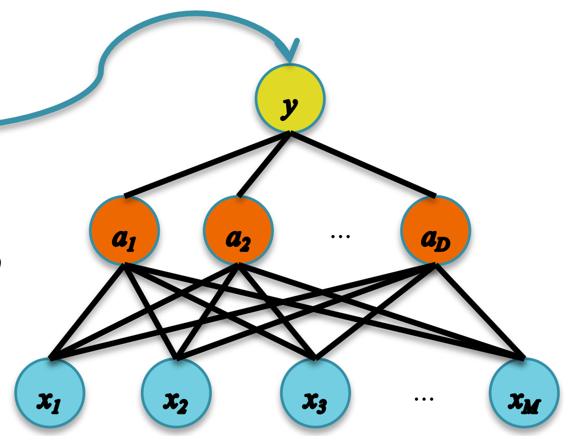
MRF LEARNING (TRIVIAL CASE)

Options for MLE of MRFs

- **Setting I:** $\psi_C(\boldsymbol{x}_C) = \theta_{C,\boldsymbol{x}_C}$
  A. MLE by inspection (Decomposable Models)
  B. Iterative Proportional Fitting (IPF)

- **Setting II:** $\psi_C(\boldsymbol{x}_C) = \exp(\boldsymbol{\theta} \cdot \boldsymbol{f}(\boldsymbol{x}_C))$
  C. Generalized Iterative Scaling
  D. Gradient-based Methods

- **Setting III:** $\psi_C(\boldsymbol{x}_C) =$
  E. Gradient-based Methods



MLE by Inspection

Whiteboard:

- Example 1: linear-chain on three variables
- Example 2: “decomposable” with four variables

MLE by Inspection

- **Definition**: Graph is **decomposable** if it can be recursively subdivided into sets A, B, and S such that S separates A and B.

MLE by Inspection

- **Definition:** Graph is **decomposable** if it can be recursively subdivided into sets A, B, and S such that S separates A and B.
- **Recipe for MLE by Guessing:**
  - Three conditions:
    1. Graphical model is *decomposable*
    2. Potentials defined on *maximal cliques*
    3. Potentials are are parameterized as: $\psi_C(\boldsymbol{x}_C) = \theta_{C,\boldsymbol{x}_C}$
  - **Step 1:** set each clique potential to its empirical marginal
  - **Step 2:** divide out every non-empty intersection between cliques *exactly once*

MLE by Inspection

- **Definition:** Graph is **decomposable** if it can be recursively subdivided into sets A, B, and S such that S separates A and B.
- **Recipe for MLE by Guessing:**
  - Three conditions:
    1. Graphical model is *decomposable*
    2. Potentials defined on *maximal cliques*
    3. Potentials are are parameterized as: $\psi_C(\boldsymbol{x}_C) = \theta_{C,\boldsymbol{x}_C}$
  - **Step 1:** set each clique potential to its empirical marginal
  - **Step 2:** divide out every non-empty intersection between cliques *exactly once*

How is this different than learning tabular Bayesian Networks?

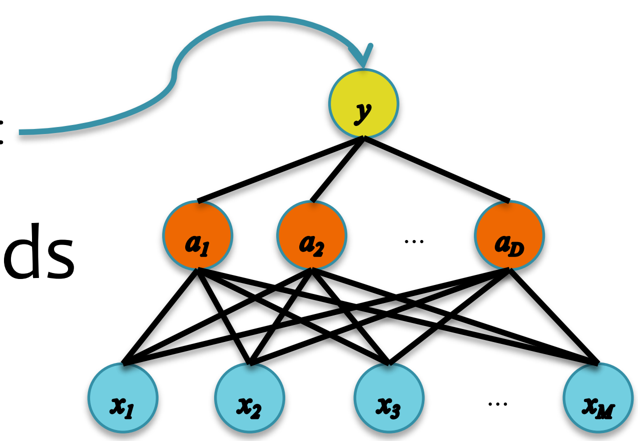
LOG-LINEAR PARAMETERIZATION OF CONDITIONAL RANDOM FIELD

Options for MLE of MRFs

- **Setting I:** $\psi_C(\boldsymbol{x}_C) = \theta_{C,\boldsymbol{x}_C}$
  A. MLE by inspection (Decomposable Models)
  B. Iterative Proportional Fitting (IPF)

- **Setting II:** $\psi_C(\boldsymbol{x}_C) = \exp(\boldsymbol{\theta} \cdot \boldsymbol{f}(\boldsymbol{x}_C))$
  C. Generalized Iterative Scaling
  D. Gradient-based Methods

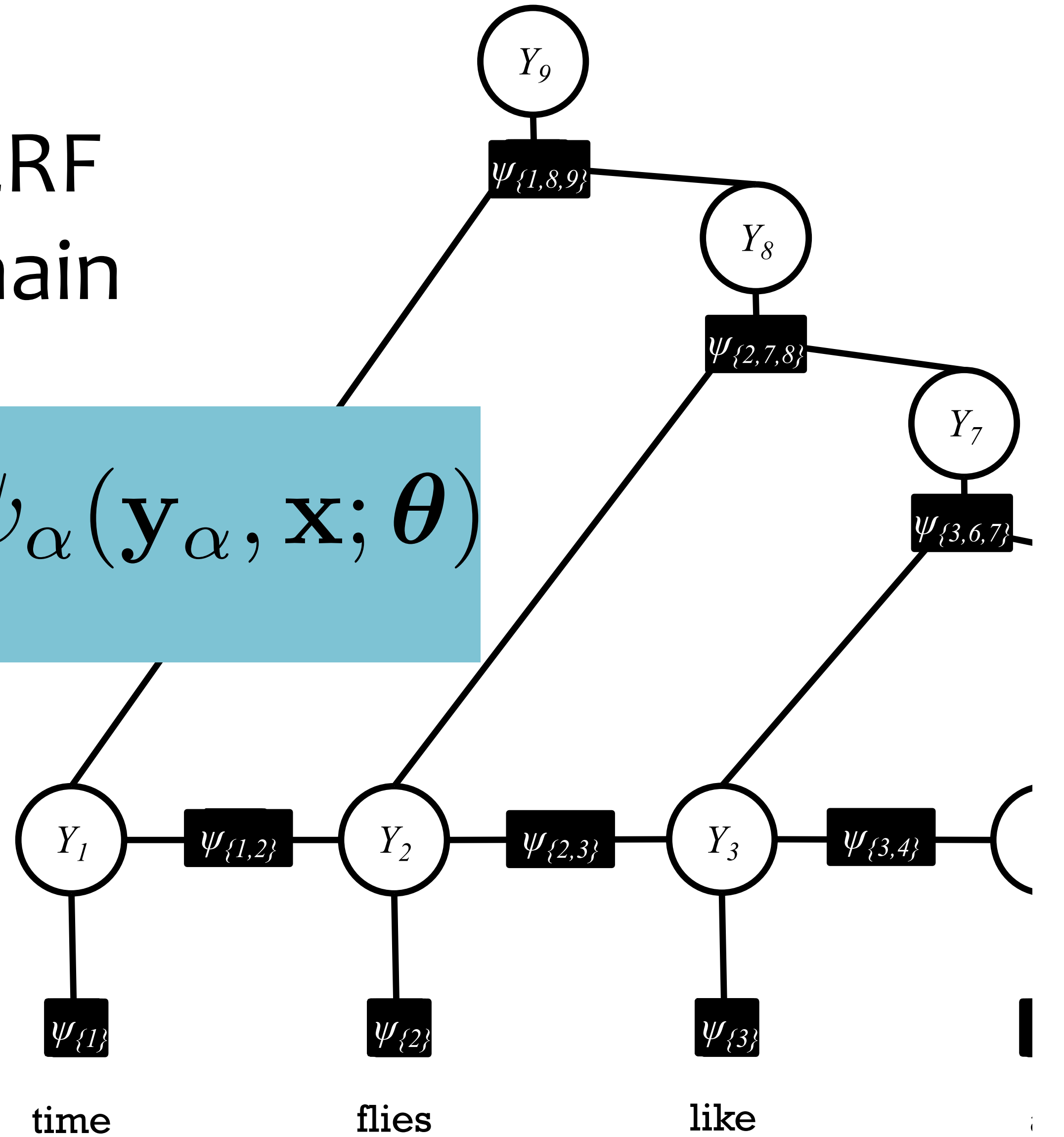
- **Setting III:** $\psi_C(\boldsymbol{x}_C) =$
  E. Gradient-based Methods



General CRF

The topology of the graphical model for a CRF doesn't have to be a chain

$$p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{\alpha} \psi_{\alpha}(\mathbf{y}_{\alpha}, \mathbf{x}; \boldsymbol{\theta})$$



Log-linear CRF Parameterization

$$p_{\boldsymbol{\theta}}(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{\alpha} \psi_{\alpha}(\mathbf{y}_{\alpha}, \mathbf{x}; \boldsymbol{\theta})$$

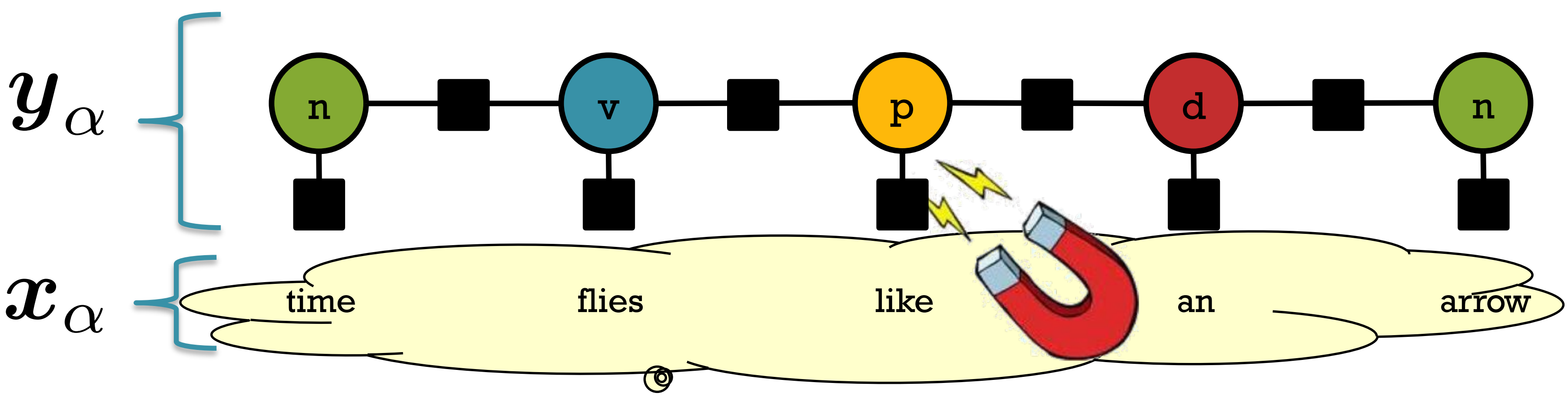
Define each potential function in terms of a fixed set of feature functions:

$$\psi_{\alpha}(\underbrace{\mathbf{y}_{\alpha}}_{\text{Predicted variables}}, \underbrace{\mathbf{x}}_{\text{Observed variables}}; \boldsymbol{\theta}) = \exp(\boldsymbol{\theta} \cdot \mathbf{f}_{\alpha}(\mathbf{y}_{\alpha}, \mathbf{x}))$$

Log-linear CRF Parameterization

Define each potential function in terms of a fixed set of feature functions:

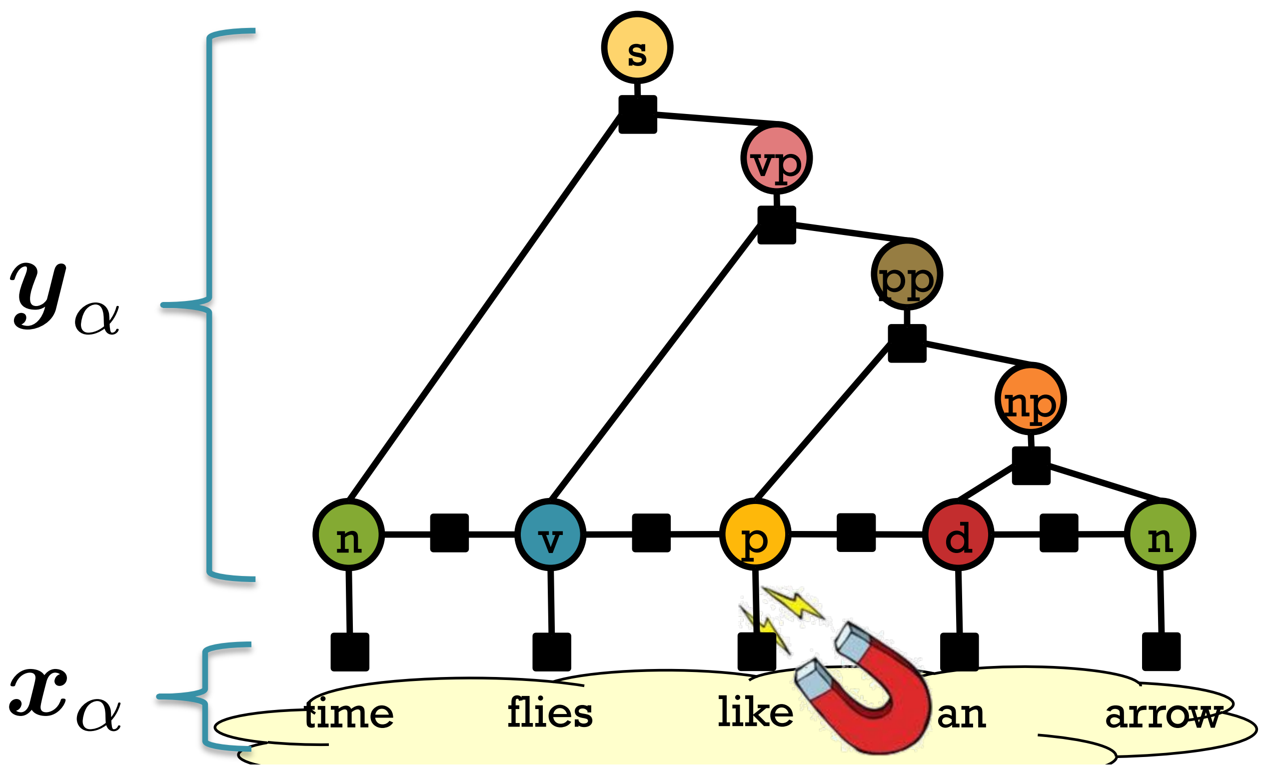
$$\psi_\alpha(\mathbf{y}_\alpha, \mathbf{x}; \boldsymbol{\theta}) = \exp(\boldsymbol{\theta} \cdot \mathbf{f}_\alpha(\mathbf{y}_\alpha, \mathbf{x}))$$



Log-linear CRF Parameterization

Define each potential function in terms of a fixed set of feature functions:

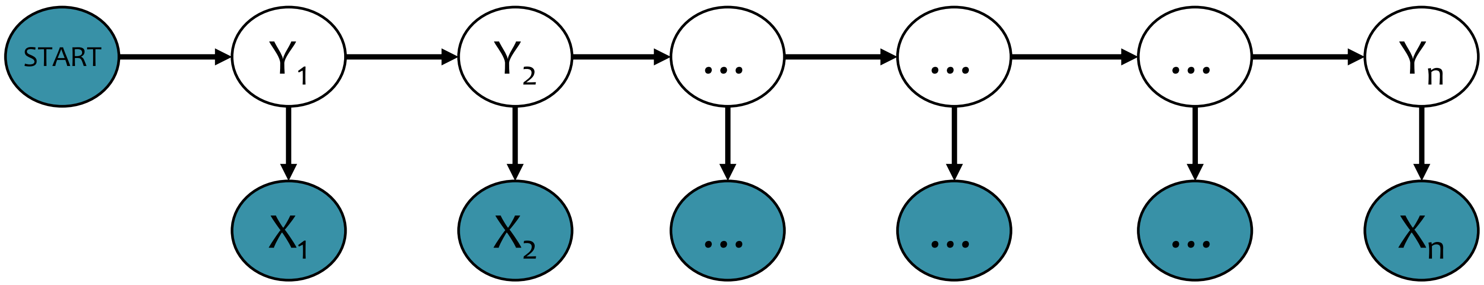
$$\psi_\alpha(\mathbf{y}_\alpha, \mathbf{x}; \boldsymbol{\theta}) = \exp(\boldsymbol{\theta} \cdot \mathbf{f}_\alpha(\mathbf{y}_\alpha, \mathbf{x}))$$



Conditional Random Fields (CRFs) for time series data

LINEAR-CHAIN CRFS

Shortcomings of Hidden Markov Models



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

Conditional Random Field (CRF)

Conditional distribution over tags X_i given words w_i.
The factors and Z are now specific to the sentence w.

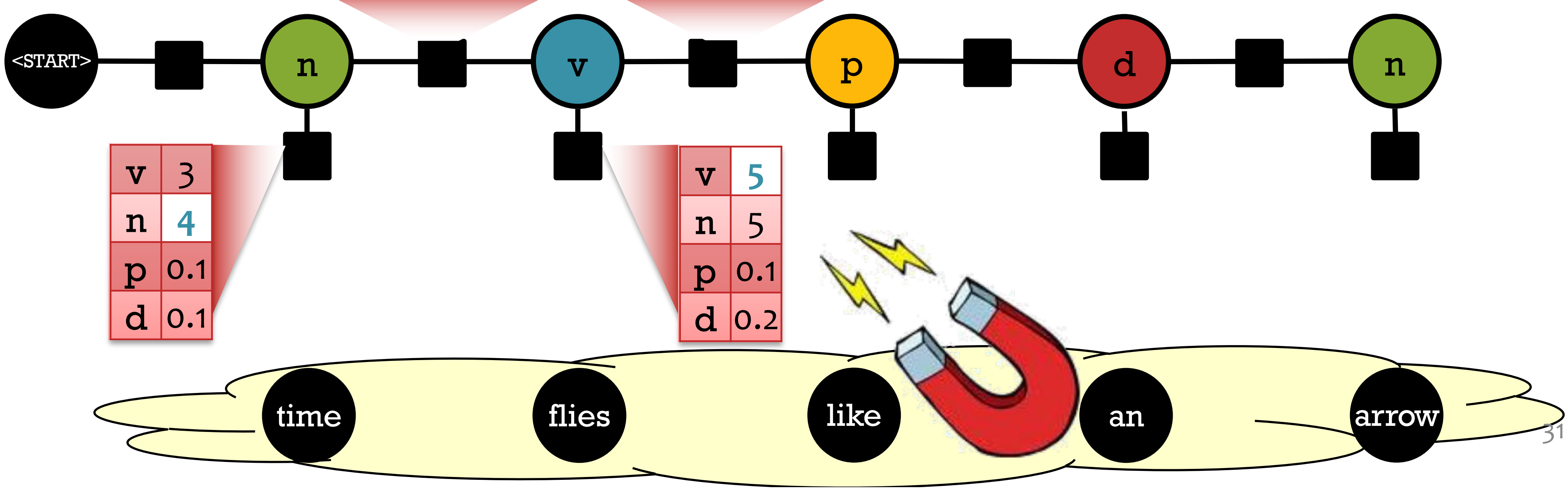
$$p(\text{n, v, p, d, n} \mid \text{time, flies, like, an, arrow}) \quad = \frac{1}{Z}\,(4 * 8 * 5 * 3 * \ldots)$$

	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0

	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0

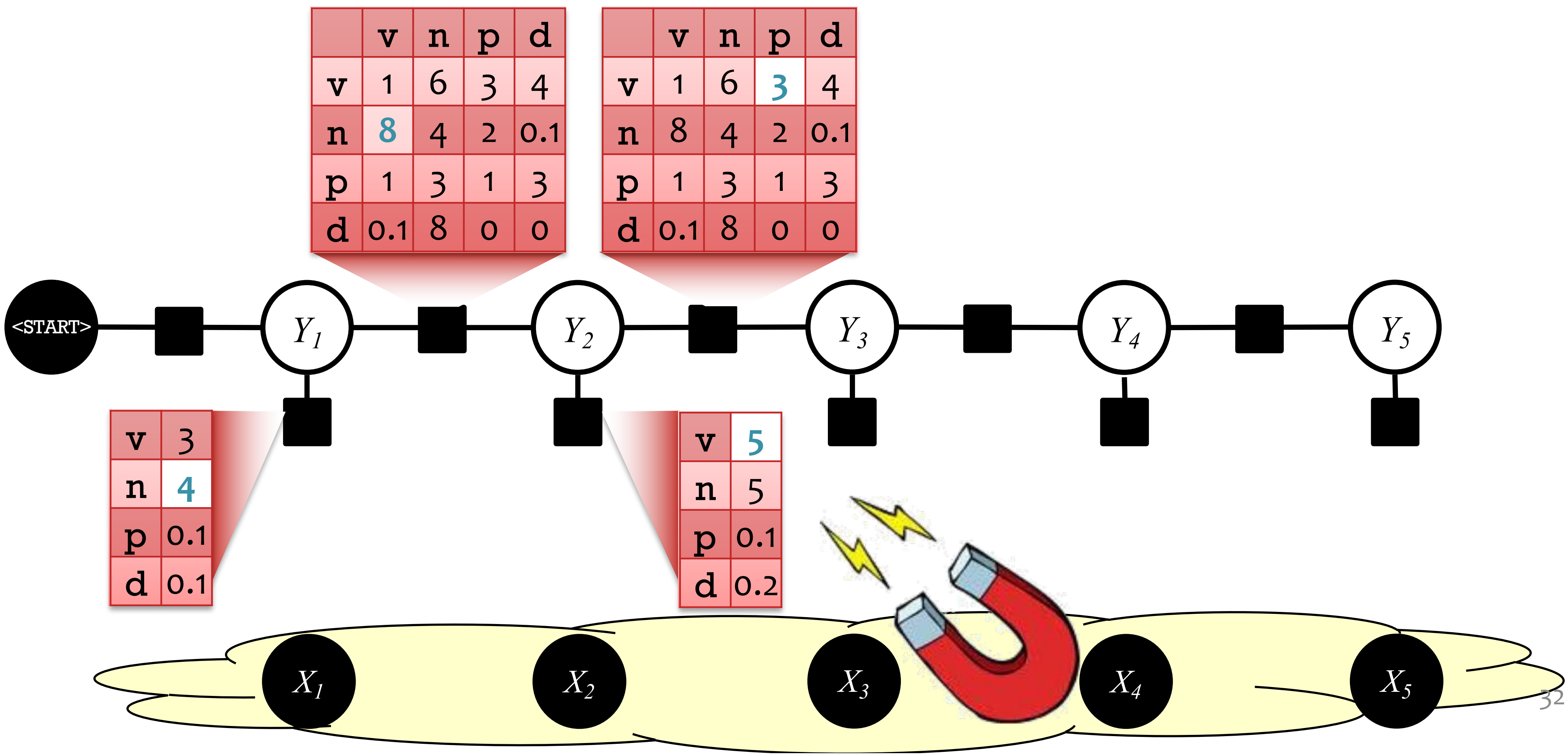
v	3
n	4
p	0.1
d	0.1

v	5
n	5
p	0.1
d	0.2



Conditional Random Field (CRF)

Recall: Shaded nodes in a graphical model are **observed**

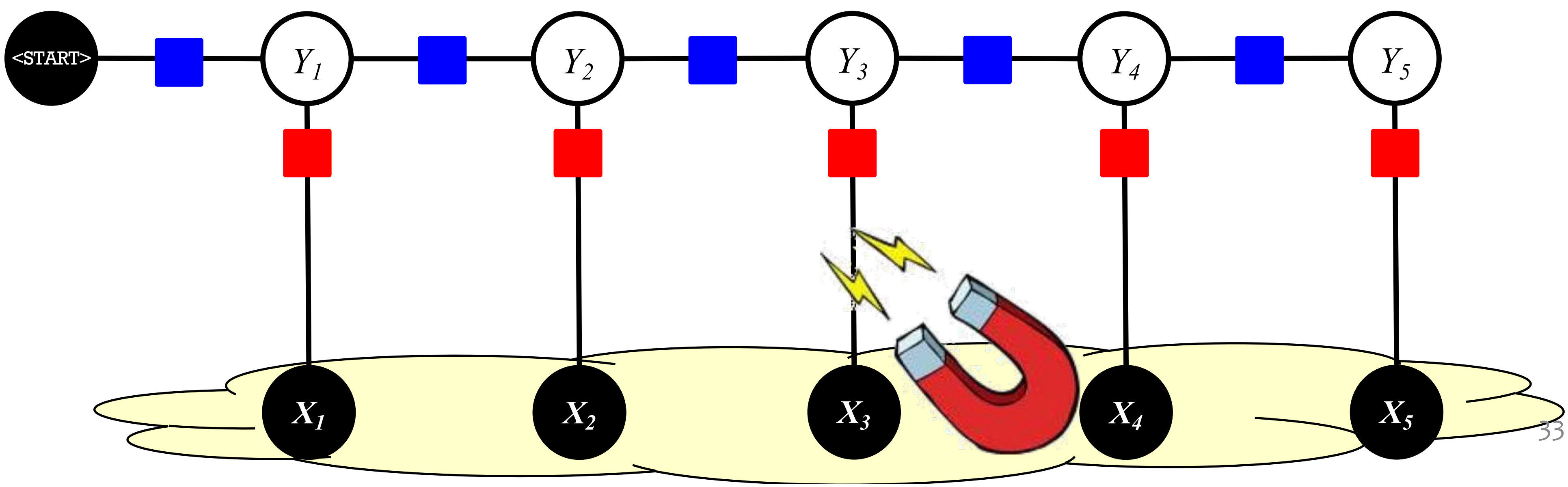


Conditional Random Field (CRF)

This **linear-chain CRF** is just **like an HMM,** except that its factors are **not** necessarily probability distributions

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{k=1}^{K} \psi_{\text{em}}(y_k, x_k)\psi_{\text{tr}}(y_k, y_{k-1})$$

$$= \frac{1}{Z(\mathbf{x})} \prod_{k=1}^{K} \exp(\boldsymbol{\theta} \cdot \mathbf{f}_{\text{em}}(y_k, x_k))\exp(\boldsymbol{\theta} \cdot \mathbf{f}_{\text{tr}}(y_k, y_{k-1}))$$



Exercise

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{k=1}^{K} \psi_{\text{em}}(y_k, x_k) \psi_{\text{tr}}(y_k, y_{k-1})$$

$$= \frac{1}{Z(\mathbf{x})} \prod_{k=1}^{K} \exp(\boldsymbol{\theta} \cdot \mathbf{f}_{\text{em}}(y_k, x_k)) \exp(\boldsymbol{\theta} \cdot \mathbf{f}_{\text{tr}}(y_k, y_{k-1}))$$

Multiple Choice: Which model does the above distribution share the most in common with?

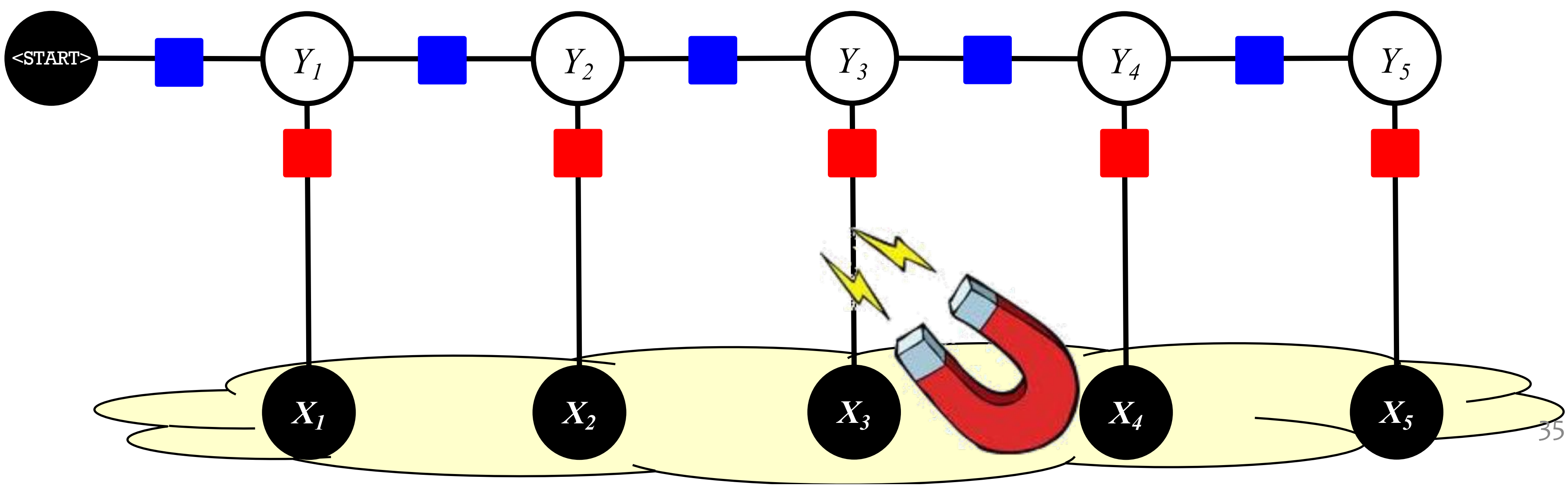
A. Hidden Markov Model
B. Bernoulli Naïve Bayes
C. Gaussian Naïve Bayes
D. Logistic Regression

Conditional Random Field (CRF)

This **linear-chain CRF** is just **like an HMM,** except that its factors are **not** necessarily probability distributions

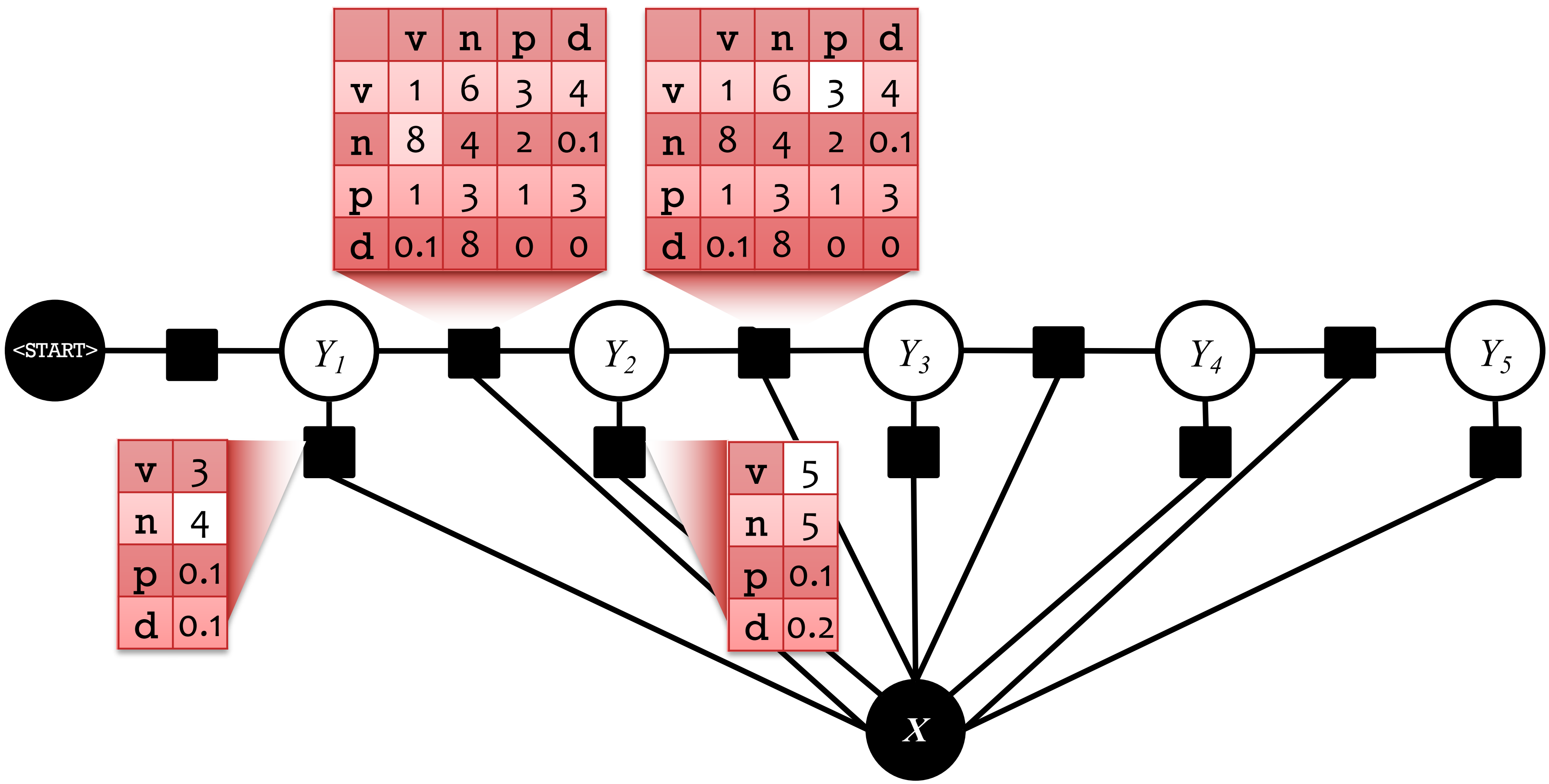
$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{k=1}^{K} \psi_{\text{em}}(y_k, x_k) \psi_{\text{tr}}(y_k, y_{k-1})$$

$$= \frac{1}{Z(\mathbf{x})} \prod_{k=1}^{K} \exp(\boldsymbol{\theta} \cdot \mathbf{f}_{\text{em}}(y_k, x_k)) \exp(\boldsymbol{\theta} \cdot \mathbf{f}_{\text{tr}}(y_k, y_{k-1}))$$



Conditional Random Field (CRF)

- That is the **vector** X
- Because it's observed, we can condition on it for free
- Conditioning is how we converted from the MRF to the CRF (i.e. when taking a slice of the emission factors)



	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0

	v	n	p	d
v	1	6	3	4
n	8	4	2	0.1
p	1	3	1	3
d	0.1	8	0	0

v	3
n	4
p	0.1
d	0.1

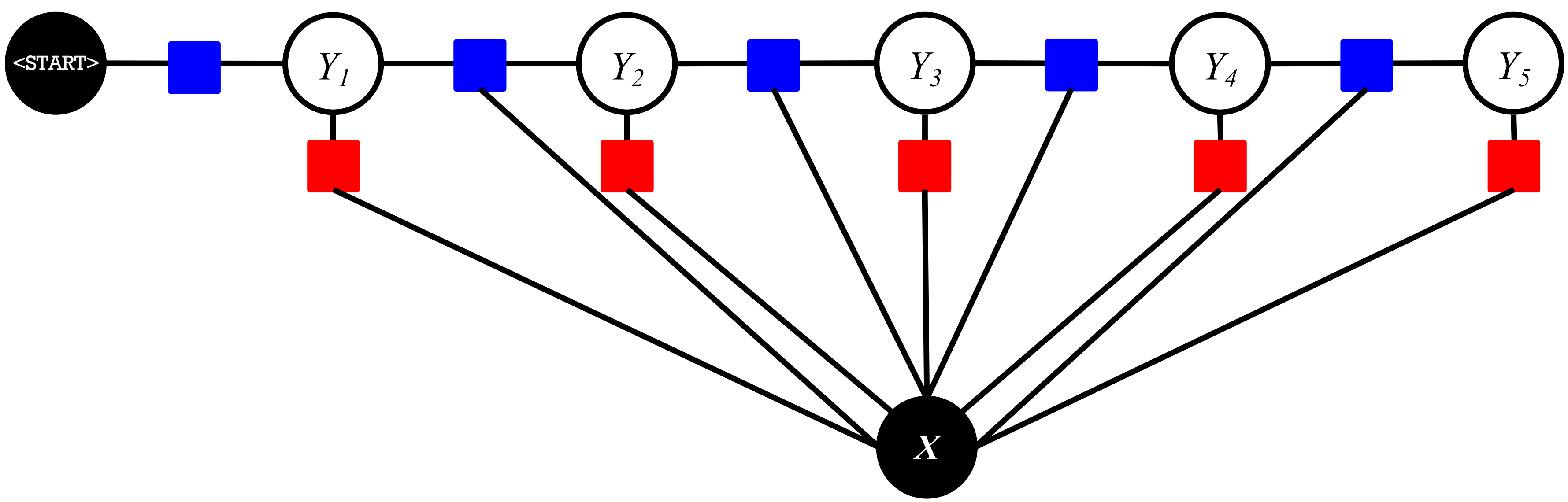
v	5
n	5
p	0.1
d	0.2

Conditional Random Field (CRF)

- This is the **standard** linear-chain CRF definition
- It permits rich, overlapping features of the vector $\boldsymbol{X}$

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{k=1}^{K} \psi_{\text{em}}(y_k, \mathbf{x}) \psi_{\text{tr}}(y_k, y_{k-1}, \mathbf{x})$$

$$= \frac{1}{Z(\mathbf{x})} \prod_{k=1}^{K} \exp(\boldsymbol{\theta} \cdot \mathbf{f}_{\text{em}}(y_k, \mathbf{x})) \exp(\boldsymbol{\theta} \cdot \mathbf{f}_{\text{tr}}(y_k, y_{k-1}, \mathbf{x}))$$

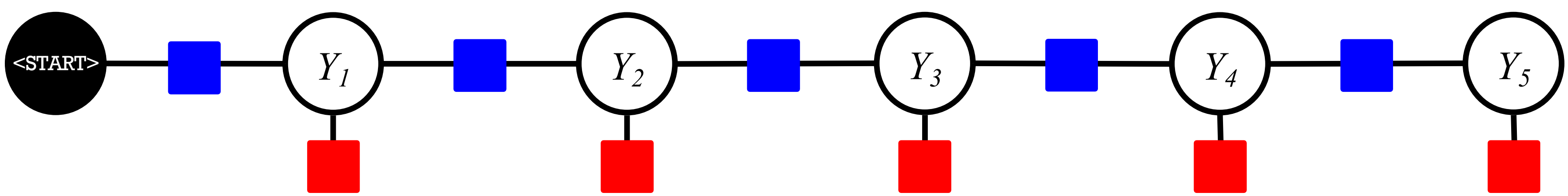


Conditional Random Field (CRF)

- This is the **standard** linear-chain CRF definition
- It permits rich, overlapping features of the vector $\boldsymbol{X}$

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{k=1}^{K} \psi_{\text{em}}(y_k, \mathbf{x}) \psi_{\text{tr}}(y_k, y_{k-1}, \mathbf{x})$$

$$= \frac{1}{Z(\mathbf{x})} \prod_{k=1}^{K} \exp(\boldsymbol{\theta} \cdot \mathbf{f}_{\text{em}}(y_k, \mathbf{x}))\exp(\boldsymbol{\theta} \cdot \mathbf{f}_{\text{tr}}(y_k, y_{k-1}, \mathbf{x}))$$



Visual Notation: Usually we draw a CRF **without** showing the variable corresponding to $\boldsymbol{X}$

LEARNING CRFS

Recipe for Gradient-based Learning

1. Write down the objective function
2. Compute the partial derivatives of the objective (i.e. gradient, and maybe Hessian)
3. Feed objective function and derivatives into black box

4. Retrieve optimal parameters from black box

Optimization Algorithms

What is the black box?

- Newton's method
- Hessian-free / Quasi-Newton methods
  - Conjugate gradient
  - L-BFGS
- Stochastic gradient methods
  - Stochastic gradient descent (SGD)
  - Stochastic meta-descent
  - AdaGrad

Stochastic Gradient Descent

- Suppose we have N training examples s.t. $f(x) = \sum_{i=1}^{N} f_i(x)$.
- This implies that $\nabla f(x) = \sum_{i=1}^{N} \nabla f_i(x)$.

SGD Algorithm:

1. Choose a starting point x.
2. While not converged:
    - Choose a step size t.
    - Choose i so that it sweeps through the training set.
    - Update

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} + t \nabla f_i(\vec{x})$$

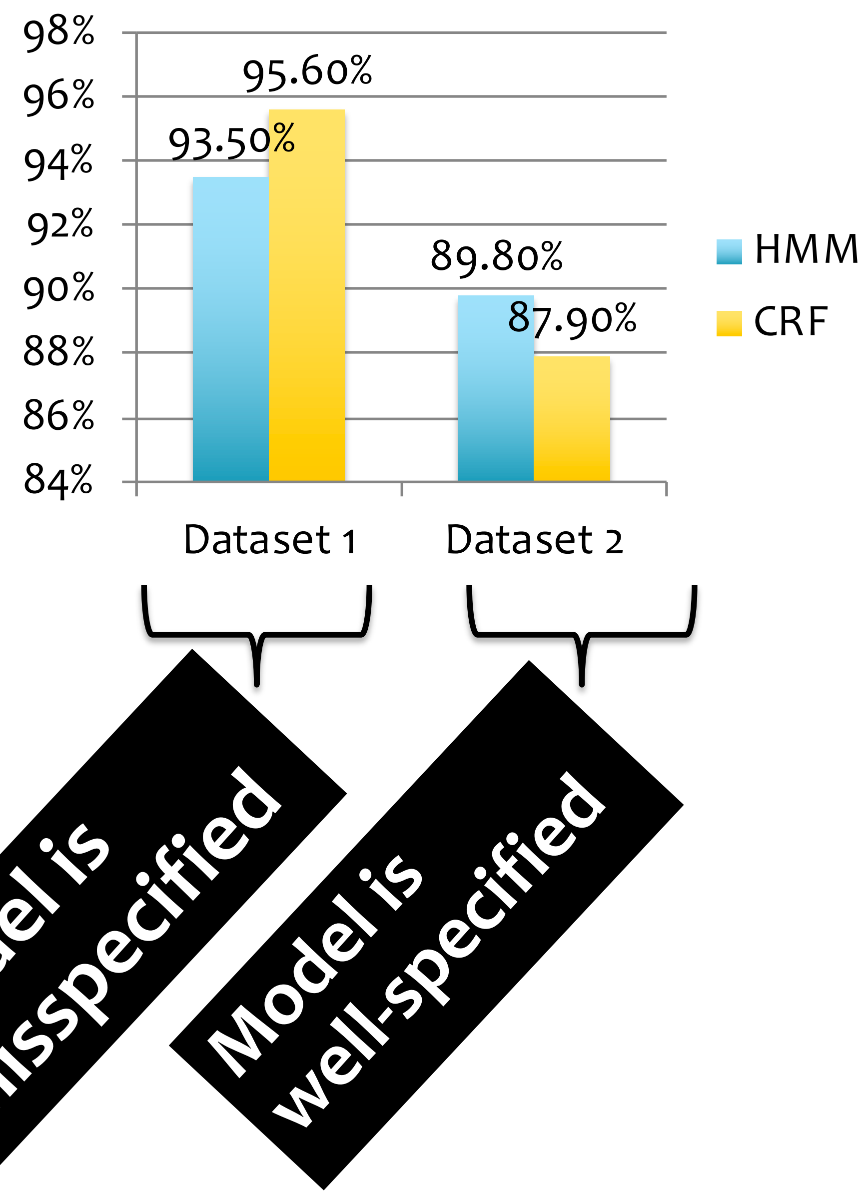
Whiteboard:

- Gradient of MRF log-likelihood for feature-based potentials
- Gradient of CRF log-likelihood for feature-based potentials

Generative vs. Discriminative

Liang & Jordan (ICML 2008) compares **HMM** and **CRF** with **identical features**

- Dataset 1: (Real)
  - WSJ Penn Treebank (38K train, 5.5K test)
  - 45 part-of-speech tags
- Dataset 2: (Artificial)
  - Synthetic data generated from HMM learned on Dataset 1 (1K train, 1K test)
- Evaluation Metric: Accuracy



NEURAL POTENTIAL FUNCTIONS

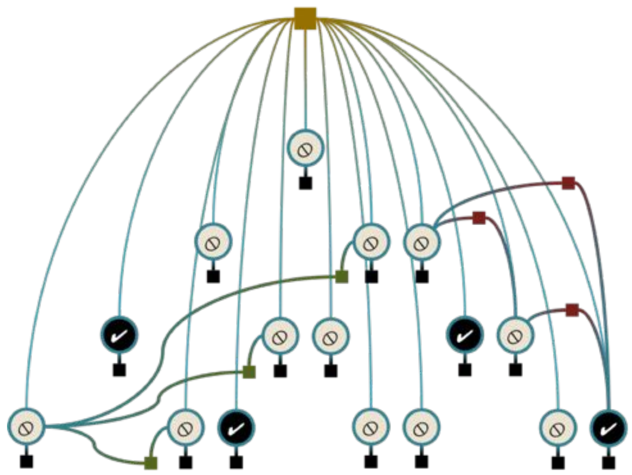
Hybrids of Graphical Models and Neural Networks

This lecture is not about a convergence of the two fields.

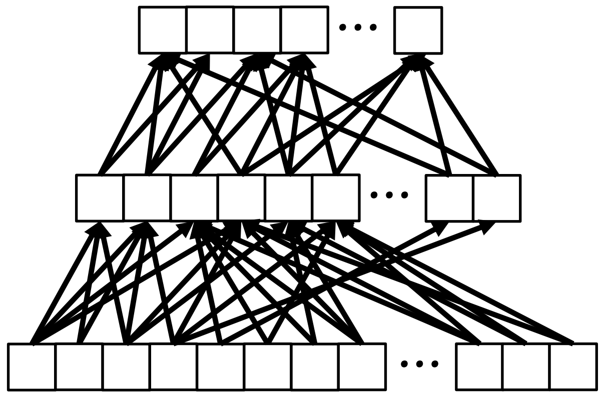
Rather, it is about state-of-the-art collaboration between two complementary techniques.

Motivation:
Hybrid Models

Graphical models let you encode domain knowledge



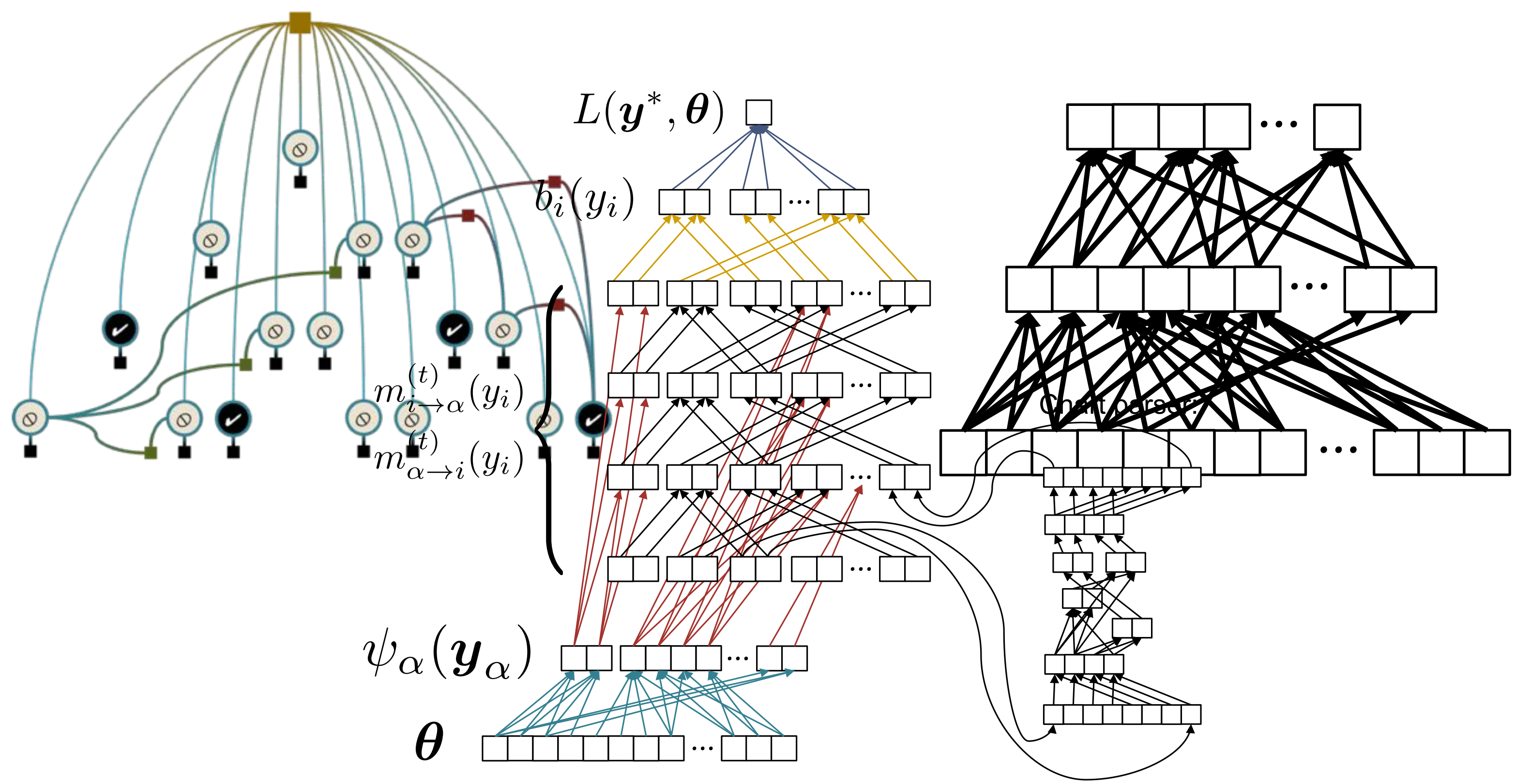
Neural nets are really good at fitting the data discriminatively to make good predictions



Could we define a neural net that incorporates domain knowledge?

Motivation:
Hybrid Models

Key idea: Use a NN to learn features for a GM, then train the entire model by backprop



A Recipe for Neural Networks

1. Given training data:

$$\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^N$$

2. Choose each of these:
   - Decision function

$$\hat{\boldsymbol{y}} = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$$

   - Loss function

$$\ell(\hat{\boldsymbol{y}}, \boldsymbol{y}_i) \in \mathbb{R}$$

Face

Face

Not a face

Examples: Linear regression, Logistic regression, Neural Network

Examples: Mean-squared error, Cross Entropy

A Recipe for Neural Networks

1. Given training data:

$$\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^N$$

2. Choose each of these:
   - Decision function

$$\hat{\boldsymbol{y}} = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$$

   - Loss function

$$\ell(\hat{\boldsymbol{y}}, \boldsymbol{y}_i) \in \mathbb{R}$$

3. Define goal:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^N \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \boldsymbol{y}_i)$$

4. Train with SGD:
(take small steps opposite the gradient)

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta_t \nabla \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \boldsymbol{y}_i)$$

A Recipe for

Background

Today's Lecture

- Suppose our decision function is a graphical model!
- We know how to compute marginal probabilities (inference), but how to do make a prediction, **y**?

2. Choose each of these:
   - Decision function

$$\hat{y} = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$$

   - Loss function

4. Train with SGD: (take small steps opposite the gradient)

$$- \eta_t \nabla \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \boldsymbol{y}_i)$$

- Can we use an MBR decoder as the decision function in this recipe?

MBR DECODING

Minimum Bayes Risk Decoding

- Suppose we given a loss function $l(\boldsymbol{y}', \boldsymbol{y})$ and are asked for a single tagging
- How should we choose just one from our probability distribution $p(\boldsymbol{y}|\boldsymbol{x})$?
- A minimum Bayes risk (MBR) decoder $h(\boldsymbol{x})$ returns the variable assignment with minimum **expected** loss under the model's distribution

$$\begin{aligned} h_{\boldsymbol{\theta}}(\boldsymbol{x}) &= \underset{\hat{\boldsymbol{y}}}{\operatorname{argmin}} \ \mathbb{E}_{\boldsymbol{y} \sim p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{x})}[\ell(\hat{\boldsymbol{y}}, \boldsymbol{y})] \\ &= \underset{\hat{\boldsymbol{y}}}{\operatorname{argmin}} \ \sum_{\boldsymbol{y}} p_{\boldsymbol{\theta}}(\boldsymbol{y} \mid \boldsymbol{x}) \ell(\hat{\boldsymbol{y}}, \boldsymbol{y}) \end{aligned}$$

Minimum Bayes Risk Decoding

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \underset{\hat{\boldsymbol{y}}}{\text{argmin}} \; \mathbb{E}_{\boldsymbol{y} \sim p_{\boldsymbol{\theta}}(\cdot | \boldsymbol{x})}[\ell(\hat{\boldsymbol{y}}, \boldsymbol{y})]$$

Consider some example loss functions:

The ***0-1*** **loss function** returns *1* only if the two assignments are identical and *0* otherwise:

$$\ell(\hat{\boldsymbol{y}}, \boldsymbol{y}) = 1 - \mathbb{I}(\hat{\boldsymbol{y}}, \boldsymbol{y})$$

The MBR decoder is:

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \underset{\hat{\boldsymbol{y}}}{\text{argmin}} \; \sum_{\boldsymbol{y}} p_{\boldsymbol{\theta}}(\boldsymbol{y} \mid \boldsymbol{x})(1 - \mathbb{I}(\hat{\boldsymbol{y}}, \boldsymbol{y}))$$

$$= \underset{\hat{\boldsymbol{y}}}{\text{argmax}} \; p_{\boldsymbol{\theta}}(\hat{\boldsymbol{y}} \mid \boldsymbol{x})$$

which is exactly the MAP inference problem!

Minimum Bayes Risk Decoding

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \operatorname*{argmin}_{\hat{\boldsymbol{y}}} \ \mathbb{E}_{\boldsymbol{y} \sim p_{\boldsymbol{\theta}}(\cdot|\boldsymbol{x})}[\ell(\hat{\boldsymbol{y}}, \boldsymbol{y})]$$

Consider some example loss functions:

The **Hamming loss** corresponds to accuracy and returns the number of incorrect variable assignments:

$$\ell(\hat{\boldsymbol{y}}, \boldsymbol{y}) = \sum_{i=1}^{V} (1 - \mathbb{I}(\hat{y}_i, y_i))$$

The MBR decoder is:

$$\hat{y}_i = h_{\boldsymbol{\theta}}(\boldsymbol{x})_i = \operatorname*{argmax}_{\hat{y}_i} \ p_{\boldsymbol{\theta}}(\hat{y}_i \mid \boldsymbol{x})$$

This decomposes across variables and requires the variable marginals.

BACKPROPAGATION AND BELIEF PROPAGATION

Whiteboard:

- Gradient of MRF log-likelihood with respect to log potentials
- Gradient of MRF log-likelihood with respect to potentials

Factor Derivatives

Log-probability:

$$\log p(\mathbf{y}) = \left[\sum_{\alpha} \log \psi_\alpha(\mathbf{y}_\alpha)\right] - \log \sum_{\mathbf{y}' \in \mathcal{Y}} \prod_{\alpha} \psi_\alpha(\mathbf{y}'_\alpha) \tag{1}$$

Derivatives:

$$\frac{\partial \log p(\mathbf{y})}{\partial \log \psi_\alpha(\mathbf{y}'_\alpha)} = \mathbb{1}(\mathbf{y}_\alpha = \mathbf{y}'_\alpha) - p(\mathbf{y}'_\alpha) \tag{2}$$

$$\frac{\partial \log p(\mathbf{y})}{\partial \psi_\alpha(\mathbf{y}'_\alpha)} = \frac{\mathbb{1}(\mathbf{y}_\alpha = \mathbf{y}'_\alpha) - p(\mathbf{y}'_\alpha)}{\psi_\alpha(\mathbf{y}'_\alpha)} \tag{3}$$