Inference and Representation

David Sontag

New York University

Lecture 9, Nov. 21, 2016

Acknowledgements: Partially based on slides by Eric Xing at CMU and Andrew McCallum at UMass Amherst

Today: learning undirected graphical models

1. Reminder of Markov random fields (MRFs)
2. *Conditional* random fields (CRFs)
3. Learning MRFs
    a. Reminder of exponential families
    b. Feature-based (log-linear) representation of MRFs
    c. Maximum likelihood estimation
    d. Maximum entropy view
4. Getting around complexity of inference
    a. Using approximate inference within learning
    b. Pseudo-likelihood

Reminder of Markov random fields (MRFs)

- An alternative representation for joint distributions is as an **undirected graphical model** or **Markov network**
- As in BNs, we have one node for each random variable
- Rather than CPDs, we specify (non-negative) **potential functions** over sets of variables associated with cliques C of the graph,

$$p(x_1, \dots, x_n) = \frac{1}{Z} \prod_{c \in C} \phi_c(\mathbf{x}_c)$$

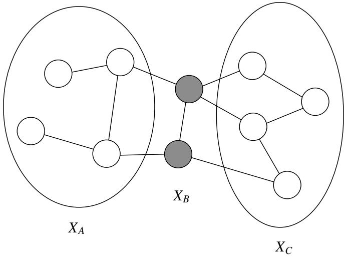
  Z is the **partition function** and normalizes the distribution:

$$Z = \sum_{\hat{x}_1, \dots, \hat{x}_n} \prod_{c \in C} \phi_c(\hat{\mathbf{x}}_c)$$

- Like CPD's, $\phi_c(\mathbf{x}_c)$ can be represented as a table, but it is *not normalized*

Reminder of conditional independence in Markov networks

- Let G be the undirected graph where we have one edge for every pair of variables that appear together in a potential
- Conditional independence is given by **graph separation**!



- $X_{\mathbf{A}} \perp X_{\mathbf{C}} \mid X_{\mathbf{B}}$ if there is no path from $a \in \mathbf{A}$ to $c \in \mathbf{C}$ after removing all variables in $\mathbf{B}$
- **Markov blanket** of a variable in MRFs is precisely its **neighbors**

Motivation for Conditional random fields (CRFs)

Goal: model *structured outputs* **Y** as a function of observed variables **X**

Computer vision

Image segmentation

input: image

output: segmentation

Stereopsis

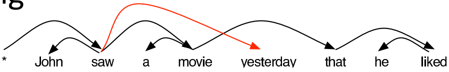
input: two images

output: disparity

Natural language processing

Parsing

output: dependency parse



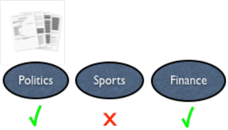
input: sentence

Motivation for Conditional random fields (CRFs)

Goal: model *structured outputs* **Y** as a function of observed variables **X**

- Input: $x \in \mathcal{X}$
- Output: label $y \in \mathcal{Y}$
- Examples:
  - Multi-label prediction:

$\boldsymbol{x}$:

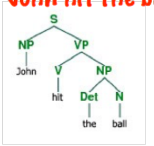


$\mathbf{y}$:

  - Natural language parsing:

$\boldsymbol{x}$: John hit the ball

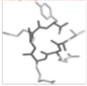
$\mathbf{y}$:



  - Protein side-chain placement:

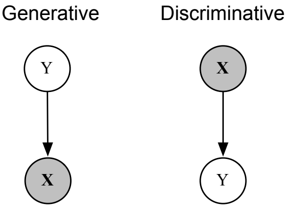
$\boldsymbol{x}$: KDLMHNKCYHFFM

$\mathbf{y}$:



Discriminative versus generative models

- Recall that these are **equivalent** models of $p(\mathbf{Y}, \mathbf{X})$:



- However, suppose all we need for prediction is $p(\mathbf{Y} \mid \mathbf{X})$
- In the left model, we need to estimate *both* $p(\mathbf{Y})$ and $p(\mathbf{X} \mid \mathbf{Y})$
- In the right model, it suffices to estimate just the **conditional distribution** $p(\mathbf{Y} \mid \mathbf{X})$
  - We never need to estimate $p(\mathbf{X})$!
  - Would need $p(\mathbf{X})$ if $\mathbf{X}$ is only partially observed
  - Called a **discriminative** model because it is only useful for discriminating $\mathbf{Y}$'s labels

Conditional random fields (CRFs)

- **Conditional random fields** are undirected graphical models of conditional distributions $p(\mathbf{Y} \mid \mathbf{X})$
  - $\mathbf{Y}$ is a set of **target variables**
  - $\mathbf{X}$ is a set of **observed variables**
- We typically show the graphical model using just the $\mathbf{Y}$ variables
- Potentials are a function of $\mathbf{X}$ and $\mathbf{Y}$
- Can still use all the tools we've learned so far to model this joint distribution over $\mathbf{Y}$

Formal definition of CRFs

- A CRF is a Markov network on variables $\mathbf{X} \cup \mathbf{Y}$, which specifies the conditional distribution

$$P(\mathbf{y} \mid \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{c \in C} \phi_c(\mathbf{x}_c, \mathbf{y}_c)$$

with partition function

$$Z(\mathbf{x}) = \sum_{\hat{\mathbf{y}}} \prod_{c \in C} \phi_c(\mathbf{x}_c, \hat{\mathbf{y}}_c).$$

- As before, two variables in the graph are connected with an undirected edge if they appear together in the scope of some factor
- The only difference with a standard Markov network is the normalization term – before marginalized over $\mathbf{X}$ and $\mathbf{Y}$, now only over $\mathbf{Y}$

Parameterization of CRFs

- Factors may depend on a large number of variables
- We typically parameterize each factor as a log-linear function,

$$\phi_c(\mathbf{x}_c, \mathbf{y}_c) = \exp\{\mathbf{w} \cdot \mathbf{f}_c(\mathbf{x}_c, \mathbf{y}_c)\}$$

- $\mathbf{f}_c(\mathbf{x}_c, \mathbf{y}_c)$ is a feature vector
- $\mathbf{w}$ is a weight vector which is typically learned

Example #1 (NLP): Part-of-speech tagging

United flies some large jet

$$\Downarrow$$

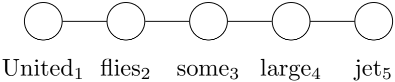
$$\begin{tikzcd}
\mathrm{N} \arrow[r] \arrow[d] & \mathrm{V} \arrow[r] \arrow[d] & \mathrm{D} \arrow[r] \arrow[d] & \mathrm{A} \arrow[r] \arrow[d] & \mathrm{N} \arrow[d] \\
\mathrm{United}_1 & \mathrm{flies}_2 & \mathrm{some}_3 & \mathrm{large}_4 & \mathrm{jet}_5
\end{tikzcd}$$

Example #1 (NLP): Graphical model formulation of POS tagging

given:

- a sentence of length n and a tag set $\mathcal{T}$
- one variable for each word, takes values in $\mathcal{T}$
- edge potentials $\theta(i-1, i, t', t)$ for all $i \in n$, $t, t' \in \mathcal{T}$

example:



$$\mathcal{T} = \{A, D, N, V\}$$

Example #1 (NLP): Features for POS tagging

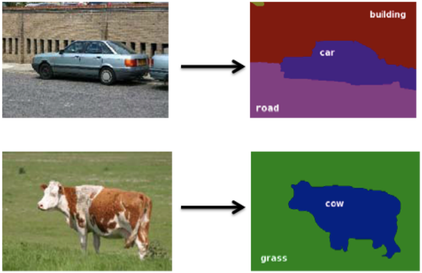
- Parameterization as log-linear model:
  - Weights $\mathbf{w} \in \mathbb{R}^d$. Feature vectors $\mathbf{f}_c(\mathbf{x}, \mathbf{y}_c) \in \mathbb{R}^d$.
  - $\phi_c(\mathbf{x}, \mathbf{y}_c; \mathbf{w}) = \exp(\mathbf{w} \cdot \mathbf{f}_c(\mathbf{x}, \mathbf{y}_c))$
- Edge potentials: Fully parameterize ($\mathcal{T} \times \mathcal{T}$ features and weights), i.e.

$$\theta_{i-1,i}(t', t) = w^T_{t',t}$$

  where the superscript "T" denotes that these are the weights for the transitions
- Node potentials: Introduce features for the presence or absence of certain attributes of each word (e.g., initial letter capitalized, suffix is "ing"), for each possible tag ($\mathcal{T} \times$ #attributes features and weights) *This part is*

  conditional on the input sentence!
- Edge potential same for all edges. Same for node potentials.

Example #2 (vision): Image segmentation



- Problem: Given an image $\mathbf{X} \in \mathbb{R}^{m \times n \times 3}$, produce a labeling $\mathbf{Y} \in \{1, \ldots, k\}^{m \times n}$.
- The labels $1, \ldots, k$ could correspond to e.g. $\{$*grass, sky, tree*$\}$.

Example #2 (vision): Image segmentation

- Approach: Define a grid-structured CRF to model $P(\mathbf{Y}|\mathbf{X})$, where potentials are based on the intuition that neighboring pixels with similar colors should probably have the same label.
- Pairwise potentials over labels for neighboring pixels $i, i+1$:

$$\phi_{i,i+1}(y_i, y_{i+1}) = \exp\left(-\mathbb{1}_{y_i=y_{i+1}} \|x_i - x_{i+1}\| + \mathbb{1}_{y_i \neq y_{i+1}} \|x_i - x_{i+1}\|\right)$$

- Single node potentials over labels, depending e.g. on part of the image in neighborhood of the pixel
- x_i represents the 3-dimensional RGB for pixel i
- Then find the MAP solution for Y:

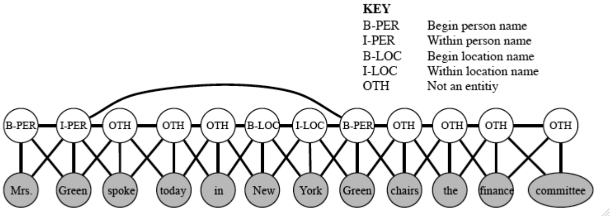
$$Y^* = \text{argmax}_Y P(\mathbf{Y}|\mathbf{X})$$

Example #3 (NLP): named-entity recognition

- Given a sentence, determine the people and organizations involved and the relevant locations:
  "Mrs. Green spoke today in New York. Green chairs the finance committee."
- Entities sometimes span multiple words. Entity of a word not obvious without considering its *context*
- CRF has one variable X_i for each word, which encodes the possible labels of that word
- The labels are, for example, "B-person, I-person, B-location, I-location, B-organization, I-organization"
  - Having beginning (B) and within (I) allows the model to segment adjacent entities

Example #3 (NLP): named-entity recognition

The graphical model looks like (called a *skip-chain CRF*):



There are three types of potentials:

- $\phi^1(Y_t, Y_{t+1})$ represents dependencies between neighboring target variables [analogous to transition distribution in a HMM]
- $\phi^2(Y_t, Y_{t'})$ for all pairs t, t' such that $x_t = x_{t'}$, because if a word appears twice, it is likely to be the same entity
- $\phi^3(Y_t, X_1, \cdots, X_T)$ for dependencies between an entity and the word sequence [e.g., may have features taking into consideration capitalization]

Notice that the graph structure changes depending on the sentence!

Today: learning undirected graphical models

1. Reminder of Markov random fields (MRFs)
2. *Conditional* random fields (CRFs)
3. Learning MRFs
   a. Reminder of exponential families
   b. Feature-based (log-linear) representation of MRFs
   c. Maximum likelihood estimation
   d. Maximum entropy view
4. Getting around complexity of inference
   a. Using approximate inference within learning
   b. Pseudo-likelihood

Reminder of the exponential family

- Recall the definition of probability distributions in the exponential family:
$$p(\mathbf{x};\eta) = h(\mathbf{x})\exp\{\eta\cdot\mathbf{f}(\mathbf{x}) - \ln Z(\eta)\}$$

  $\mathbf{f}(\mathbf{x})$ are called the *sufficient statistics*

- For example, when p is a Gaussian distribution,
$$p(\mathbf{x};\mu,\Sigma) = \frac{1}{(2\pi)^{k/2}|\Sigma|^{1/2}}\exp\left(-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)\right)$$

  then $\mathbf{f}(\mathbf{x}) = [x_1, x_2, \ldots, x_k, x_1^2, x_1x_2, x_1x_3, \ldots, x_2^2, x_2x_3, \ldots]$

- In the exponential family, there is a one-to-one correspondance between distributions $p(\mathbf{x};\eta)$ and marginal vectors $E_p[\mathbf{f}(\mathbf{x})]$
- The expectation of $\mathbf{f}(\mathbf{x})$ gives the first and second-order (non-central) moments, from which one can solve for μ and Σ

Properties of exponential families

The derivative of the log-partition function is equal to the expectation of the sufficient statistic vector (i.e. the distribution's marginals):

$$
\begin{aligned}
\partial_{\eta_i} \ln Z(\eta) &= \partial_{\eta_i} \ln \sum_{\mathbf{x}} \exp\{\eta \cdot \mathbf{f}(\mathbf{x})\} \\
&= \frac{1}{\sum_{\mathbf{x}} \exp\{\eta \cdot \mathbf{f}(\mathbf{x})\}} \partial_{\eta_i} \sum_{\mathbf{x}} \exp\{\eta \cdot \mathbf{f}(\mathbf{x})\} \\
&= \frac{1}{\sum_{\mathbf{x}} \exp\{\eta \cdot \mathbf{f}(\mathbf{x})\}} \sum_{\mathbf{x}} \partial_{\eta_i} \exp\{\eta \cdot \mathbf{f}(\mathbf{x})\} \\
&= \frac{1}{\sum_{\mathbf{x}} \exp\{\eta \cdot \mathbf{f}(\mathbf{x})\}} \sum_{\mathbf{x}} \exp\{\eta \cdot \mathbf{f}(\mathbf{x})\} \partial_{\eta_i} \eta \cdot \mathbf{f}(\mathbf{x}) \\
&= \frac{1}{\sum_{\mathbf{x}} \exp\{\eta \cdot \mathbf{f}(\mathbf{x})\}} \sum_{\mathbf{x}} \exp\{\eta \cdot \mathbf{f}(\mathbf{x})\} f_i(\mathbf{x}) \\
&= \sum_{\mathbf{x}} \frac{\exp\{\eta \cdot \mathbf{f}(\mathbf{x})\}}{\sum_{\hat{\mathbf{x}}} \exp\{\eta \cdot \mathbf{f}(\hat{\mathbf{x}})\}} f_i(\mathbf{x}) = \sum_{\mathbf{x}} p(\mathbf{x}) f_i(\mathbf{x}) = E_p[f_i(\mathbf{x})].
\end{aligned}
$$

Recall: ML estimation in Bayesian networks

- Maximum likelihood estimation: $\max_\theta \ell(\theta; \mathcal{D})$, where

$$\begin{aligned}\ell(\theta; \mathcal{D}) = \log p(\mathcal{D}; \theta) &= \sum_{\mathbf{x} \in \mathcal{D}} \log p(\mathbf{x}; \theta) \\ &= \sum_i \sum_{\hat{\mathbf{x}}_{pa(i)}} \sum_{\substack{\mathbf{x} \in \mathcal{D}: \\ \mathbf{x}_{pa(i)} = \hat{\mathbf{x}}_{pa(i)}}} \log p(x_i \mid \hat{\mathbf{x}}_{pa(i)})\end{aligned}$$

- In Bayesian networks, we have the closed form ML solution:

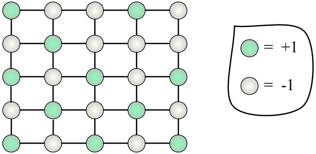
$$\theta^{ML}_{x_i | \mathbf{x}_{pa(i)}} = \frac{N_{x_i, \mathbf{x}_{pa(i)}}}{\sum_{\hat{x}_i} N_{\hat{x}_i, \mathbf{x}_{pa(i)}}}$$

  where $N_{x_i, \mathbf{x}_{pa(i)}}$ is the number of times that the (partial) assignment $x_i, \mathbf{x}_{pa(i)}$ is observed in the training data

- We were able to estimate each CPD independently because the objective **decomposes** by variable and parent assignment

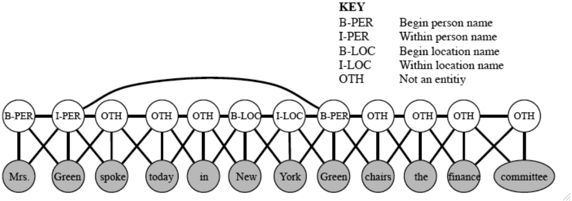
Parameter estimation in Markov networks

- How do we learn the parameters of an Ising model?



$$p(x_1, \cdots, x_n) = \frac{1}{Z} \exp\left(\sum_{i<j} w_{i,j} x_i x_j - \sum_i u_i x_i \right)$$

- What about for a skip-chain CRF?



Bad news for Markov networks

- The global normalization constant $Z(\theta)$ kills decomposability:

$$
\begin{aligned}
\theta^{ML} &= \arg\max_{\theta} \sum_{\mathbf{x}\in\mathcal{D}} \log p(\mathbf{x};\theta) \\
&= \arg\max_{\theta} \sum_{\mathbf{x}\in\mathcal{D}} \left(\sum_{c} \log \phi_c(\mathbf{x}_c;\theta) - \log Z(\theta) \right) \\
&= \arg\max_{\theta} \left(\sum_{\mathbf{x}\in\mathcal{D}} \sum_{c} \log \phi_c(\mathbf{x}_c;\theta) \right) - |\mathcal{D}| \log Z(\theta)
\end{aligned}
$$

- The log-partition function prevents us from decomposing the objective into a sum over terms for each potential
- Solving for the parameters becomes much more complicated

What are the parameters?

- Parameterize $\phi_c(\mathbf{x}_c; \theta)$ using a log-linear parameterization:
  - Single weight vector $\mathbf{w} \in \mathbb{R}^d$ that is used globally
  - For each potential c, a vector-valued **feature function** $\mathbf{f}_c(\mathbf{x}_c) \in \mathbb{R}^d$
  - Then, $\phi_c(\mathbf{x}_c; \mathbf{w}) = \exp(\mathbf{w} \cdot \mathbf{f}_c(\mathbf{x}_c))$
- Example: discrete-valued MRF with only edge potentials, where each variable takes k states
  - Let $d = k^2|E|$, and let $w_{i,j,x_i,x_j} = \log \phi_{ij}(x_i, x_j)$
  - Let $f_{i,j}(x_i, x_j)$ have a 1 in the dimension corresponding to (i, j, x_i, x_j) and 0 elsewhere
- The joint distribution is in the *exponential family*!

$$p(\mathbf{x}; \mathbf{w}) = \exp\{\mathbf{w} \cdot \mathbf{f}(\mathbf{x}) - \log Z(\mathbf{w})\},$$

  where $f(\mathbf{x}) = \sum_c f_c(\mathbf{x}_c)$ and $Z(\mathbf{w}) = \sum_{\mathbf{x}} \exp\{\sum_c \mathbf{w} \cdot f_c(\mathbf{x}_c)\}$
- This formulation allows for parameter sharing

Log-likelihood for log-linear models

$$
\begin{aligned}
\theta^{ML} &= \arg\max_{\theta} \left(\sum_{\mathbf{x}\in\mathcal{D}} \sum_{c} \log \phi_c(\mathbf{x}_c; \theta) \right) - |\mathcal{D}| \log Z(\theta) \\
&= \arg\max_{\mathbf{w}} \left(\sum_{\mathbf{x}\in\mathcal{D}} \sum_{c} \mathbf{w} \cdot \mathbf{f}_c(\mathbf{x}_c) \right) - |\mathcal{D}| \log Z(\mathbf{w}) \\
&= \arg\max_{\mathbf{w}} \ \mathbf{w} \cdot \left(\sum_{\mathbf{x}\in\mathcal{D}} \sum_{c} \mathbf{f}_c(\mathbf{x}_c) \right) - |\mathcal{D}| \log Z(\mathbf{w})
\end{aligned}
$$

- The first term is linear in $\mathbf{w}$
- The second term is also a function of $\mathbf{w}$:

$$
\log Z(\mathbf{w}) = \log \sum_{\mathbf{x}} \exp \left(\mathbf{w} \cdot \sum_{c} \mathbf{f}_c(\mathbf{x}_c) \right)
$$

Log-likelihood for log-linear models

$$\log Z(\mathbf{w}) = \log \sum_{\mathbf{x}} \exp\left(\mathbf{w} \cdot \sum_c \mathbf{f}_c(\mathbf{x}_c)\right)$$

- $\log Z(\mathbf{w})$ does not decompose
  - No closed form solution; even *computing* likelihood requires inference
- Letting $\mathbf{f}(\mathbf{x}) = \sum_c \mathbf{f}_c(\mathbf{x}_c)$, we showed (for all exponential families) that:

$$\nabla_{\mathbf{w}} \log Z(\mathbf{w}) = \mathbb{E}_{p(\mathbf{x};\mathbf{w})}[\mathbf{f}(\mathbf{x})] = \sum_c \mathbb{E}_{p(\mathbf{x}_c;\mathbf{w})}[\mathbf{f}_c(\mathbf{x}_c)]$$

- Thus, the gradient of the log-partition function can be computed by *inference*, computing marginals with respect to the current parameters $\mathbf{w}$
- Similarly, you can show that 2nd derivative of the log-partition function gives the second-order moments, i.e.

$$\nabla^2 \log Z(\mathbf{w}) = \left(\mathbb{E}_{p(\mathbf{x};\mathbf{w})}[f^i(\mathbf{x})f^j(\mathbf{x})]\right)_{ij} = \mathrm{cov}[\mathbf{f}(\mathbf{x})]$$

- Since covariance matrices are always positive semi-definite, this proves that $\log Z(\mathbf{w})$ is convex (so $-\log Z(\mathbf{w})$ is concave)

Solving the maximum likelihood problem in MRFs

$$\ell(\mathbf{w};\mathcal{D}) = \frac{1}{|\mathcal{D}|}\mathbf{w}\cdot\left(\sum_{\mathbf{x}\in\mathcal{D}}\sum_{c}\mathbf{f}_c(\mathbf{x}_c)\right) - \log Z(\mathbf{w})$$

- First, note that the weights $\mathbf{w}$ are unconstrained, i.e. $\mathbf{w}\in\mathbb{R}^d$
- The objective function is jointly concave. Apply any **convex optimization** method to learn!
- Can use gradient ascent, **stochastic gradient ascent**, quasi-Newton methods such as limited memory BFGS (L-BFGS)
- Let's study some properties of the ML solution!

$$\begin{aligned}
\frac{d}{dw_k}\ell(\mathbf{w};\mathcal{D}) &= \frac{1}{|\mathcal{D}|}\sum_{\mathbf{x}\in\mathcal{D}}\sum_{c}(\mathbf{f}_c(\mathbf{x}_c))_k - \sum_{c}\mathbb{E}_{p(\mathbf{x}_c;\mathbf{w})}[(\mathbf{f}_c(\mathbf{x}_c))_k] \\
&= \sum_{c}\frac{1}{|\mathcal{D}|}\sum_{\mathbf{x}\in\mathcal{D}}(\mathbf{f}_c(\mathbf{x}_c))_k - \sum_{c}\mathbb{E}_{p(\mathbf{x}_c;\mathbf{w})}[(\mathbf{f}_c(\mathbf{x}_c))_k]
\end{aligned}$$

The gradient of the log-likelihood

$$\frac{\partial}{\partial w_k}\ell(\mathbf{w};\mathcal{D}) = \sum_c \frac{1}{|\mathcal{D}|}\sum_{\mathbf{x}\in\mathcal{D}}(\mathbf{f}_c(\mathbf{x}_c))_k - \sum_c \mathbb{E}_{p(\mathbf{x}_c;\mathbf{w})}[(\mathbf{f}_c(\mathbf{x}_c))_k]$$

- Difference of expectations!
- Consider the earlier pairwise MRF example. This then reduces to:

$$\frac{\partial}{\partial w_{i,j,\hat{x}_i,\hat{x}_j}}\ell(\mathbf{w};\mathcal{D}) = \left(\frac{1}{|\mathcal{D}|}\sum_{\mathbf{x}\in\mathcal{D}} 1[x_i=\hat{x}_i, x_j=\hat{x}_j]\right) - p(\hat{x}_i,\hat{x}_j;\mathbf{w})$$

- Setting derivative to zero, we see that for the maximum likelihood parameters $\mathbf{w}^{ML}$, we have

$$p(\hat{x}_i,\hat{x}_j;\mathbf{w}^{ML}) = \frac{1}{|\mathcal{D}|}\sum_{\mathbf{x}\in\mathcal{D}} 1[x_i=\hat{x}_i, x_j=\hat{x}_j]$$

  for all edges $ij \in E$ and states $\hat{x}_i, \hat{x}_j$
- Model marginals for ML solution equal the empirical marginals!
- Called **moment matching**, and is a property of maximum likelihood learning in exponential families

Gradient ascent requires repeated marginal inference, which in many models is **hard**!

We will return to this shortly.

Maximum entropy (MaxEnt)

- We can approach the modeling task from an entirely different point of view
- Suppose we know some expectations with respect to a (fully general) distribution $p(\mathbf{x})$:

$$\text{(true)} \ \sum_{\mathbf{x}} p(\mathbf{x}) f_i(\mathbf{x}), \qquad \text{(empirical)} \ \frac{1}{|\mathcal{D}|} \sum_{\mathbf{x} \in \mathcal{D}} f_i(\mathbf{x}) = \alpha_i$$

- Assuming that the expectations are consistent with one another, there may exist **many** distributions which satisfy them. Which one should we select?

  The most uncertain or flexible one, i.e., the one with maximum entropy.

- This yields a new optimization problem:

$$\max_p H(p(\mathbf{x})) = -\sum_{\mathbf{x}} p(\mathbf{x}) \log p(\mathbf{x})$$

$$\text{s.t.} \qquad \sum_{\mathbf{x}} p(\mathbf{x}) f_i(\mathbf{x}) = \alpha_i$$

$$\sum p(\mathbf{x}) = 1 \quad \text{(strictly concave w.r.t. } p(\mathbf{x})\text{)}$$

What does the MaxEnt solution look like?

- To solve the MaxEnt problem, we form the Lagrangian:

$$L = -\sum_{\mathbf{x}} p(\mathbf{x}) \log p(\mathbf{x}) - \sum_i \lambda_i \left(\sum_{\mathbf{x}} p(\mathbf{x}) f_i(\mathbf{x}) - \alpha_i \right) - \mu \left(\sum_{\mathbf{x}} p(\mathbf{x}) - 1 \right)$$

- Then, taking the derivative of the Lagrangian,

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

- And setting to zero, we obtain:

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

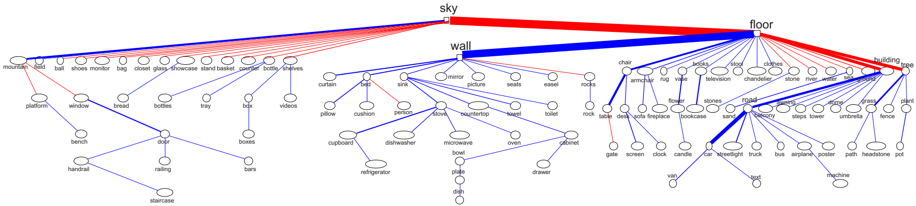
- From the constraint $\sum_{\mathbf{x}} p(\mathbf{x}) = 1$ we obtain $e^{1+\mu} = \sum_{\mathbf{x}} e^{-\sum_i \lambda_i f_i(\mathbf{x})} = Z(\lambda)$
- We conclude that the maximum entropy distribution has the form (substituting $w_i = -\lambda_i$)

$$p^*(\mathbf{x}) = \frac{1}{Z(\mathbf{w})} \exp(\sum_i w_i f_i(\mathbf{x}))$$

Equivalence of maximum likelihood and maximum entropy

- Feature constraints + MaxEnt $\Rightarrow$ exponential family!
- We have seen a case of convex duality:
  - In one case, we assume exponential family and show that ML implies model expectations must match empirical expectations
  - In the other case, we assume model expectations must match empirical feature counts and show that MaxEnt implies exponential family distribution
- Can show that one is the dual of the other, and thus both obtain the same value of the objective at optimality (no duality gap)
- Besides providing insight into the ML solution, this also gives an alternative way to (approximately) solve the learning problem

Chow-Liu algorithm for MRF structure learning



- Let's try to learn the structure of a tree-structured MRF:

$$\max_T \max_{\theta_T} \sum_{\mathbf{x}\in\mathcal{D}} \log p_T(\mathbf{x};\theta_T).$$

- Because of moment matching, for a fixed tree T, the maximum likelihood parameters, i.e.

$$\theta_T^{ML} = \arg\max_{\theta_T} \sum_{\mathbf{x}\in\mathcal{D}} \log p_T(\mathbf{x};\theta_T).$$

have $p_T(x_i, x_j; \theta_T^{ML}) = \hat{p}(x_i, x_j)$, the latter computed from the data $\mathcal{D}$

Chow-Liu algorithm for MRF structure learning

- For the special case of trees, the mapping $\mu \to \theta$ has a simple closed-form solution:

$$p_T(\mathbf{x}) = \prod_{(i,j) \in T} \frac{p_T(x_i, x_j)}{p_T(x_i) p_T(x_j)} \prod_{j \in V} p_T(x_j)$$

- Substituting $\hat{p}_T(\mathbf{x})$ into $\sum_{\mathbf{x} \in \mathcal{D}} \log p_T(\mathbf{x}; \theta_T)$, this then gives the following optimization problem:

$$\max_T \sum_{\mathbf{x} \in \mathcal{D}} \log \left[\prod_{(i,j) \in T} \frac{\hat{p}(x_i, x_j)}{\hat{p}(x_i) \hat{p}(x_j)} \prod_{j \in V} \hat{p}(x_j) \right]$$

  which can be solved using a maximum spanning tree algorithm
- For general graphs, solving the maximum entropy problem is itself intractable

How can we get around the complexity of inference during learning?

Monte Carlo methods

- Recall the original learning objective

$$\ell(\mathbf{w}; \mathcal{D}) = \frac{1}{|\mathcal{D}|} \mathbf{w} \cdot \left(\sum_{\mathbf{x} \in \mathcal{D}} \sum_c \mathbf{f}_c(\mathbf{x}_c) \right) - \log Z(\mathbf{w})$$

- Use any of the sampling approaches (e.g., Gibbs sampling) that we discussed in previous lectures
- All we need for learning (i.e., to compute the derivative of $\ell(\mathbf{w}, \mathcal{D})$) are **marginals** of the distribution
- No need to ever estimate $\log Z(\mathbf{w})$

Using approximations of the log-partition function

- We can substitute the original learning objective

$$\ell(\mathbf{w}; \mathcal{D}) = \frac{1}{|\mathcal{D}|} \mathbf{w} \cdot \Big(\sum_{\mathbf{x} \in \mathcal{D}} \sum_{c} \mathbf{f}_c(\mathbf{x}_c) \Big) - \log Z(\mathbf{w})$$

  with one that uses a tractable approximation of the log-partition function:

$$\tilde{\ell}(\mathbf{w}; \mathcal{D}) = \frac{1}{|\mathcal{D}|} \mathbf{w} \cdot \Big(\sum_{\mathbf{x} \in \mathcal{D}} \sum_{c} \mathbf{f}_c(\mathbf{x}_c) \Big) - \log \tilde{Z}(\mathbf{w})$$

- It is possible to come up with a *convex relaxation* that provides an upper bound on the log-partition function,

$$\log Z(\mathbf{w}) \leq \log \tilde{Z}(\mathbf{w})$$

  (e.g., tree-reweighted belief propagation, log-determinant relaxation)
- Using this, we obtain a *lower bound* on the learning objective

$$\ell(\mathbf{w}; \mathcal{D}) \geq \tilde{\ell}(\mathbf{w}; \mathcal{D})$$

- Again, to compute the derivatives we only need *pseudo-marginals* from the variational inference algorithm

Pseudo-likelihood

- Alternatively, can we come up with a *different* objective function (i.e., a different *estimator*) which succeeds at learning while avoiding inference altogether?
- Pseudo-likelihood method (Besag 1971) yields an exact solution if the data is generated by a model in our model family $p(\mathbf{x}; \theta^*)$ and $|\mathcal{D}| \to \infty$ (i.e., it is **consistent**)
- Note that, via the chain rule,

$$p(\mathbf{x}; \mathbf{w}) = \prod_i p(x_i | x_1, \ldots, x_{i-1}; \mathbf{w})$$

- We consider the following approximation:

$$p(\mathbf{x}; \mathbf{w}) \approx \prod_i p(x_i | x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n; \mathbf{w}) = \prod_i p(x_i | x_{-i}; \mathbf{w})$$

where we have added conditioning over additional variables

Pseudo-likelihood

- The pseudo-likelihood method replaces the likelihood,

$$\ell(\theta; \mathcal{D}) = \frac{1}{|\mathcal{D}|} \log p(\mathcal{D}; \theta) = \frac{1}{|\mathcal{D}|} \sum_{m=1}^{|\mathcal{D}|} \log p(\mathbf{x}^m; \theta)$$

  with the following approximation:

$$\ell_{PL}(\mathbf{w}; \mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{m=1}^{|\mathcal{D}|} \sum_{i=1}^{n} \log p(x_i^m \mid x_{N(i)}^m; \mathbf{w})$$

  (we replaced x_{-i} with $x_{N(i)}$, i's Markov blanket)

- For example, suppose we have a pairwise MRF. Then,

$$p(x_i^m \mid x_{N(i)}^m; \mathbf{w}) = \frac{1}{Z(x_{N(i)}^m; \mathbf{w})} e^{\sum_{j \in N(i)} \theta_{ij}(x_i^m, x_j^m)}, \quad Z(x_{N(i)}^m; \mathbf{w}) = \sum_{\hat{x}_i} e^{\sum_{j \in N(i)} \theta_{ij}(\hat{x}_i, x_j^m)}$$

- More generally, and using the log-linear parameterization, we have:

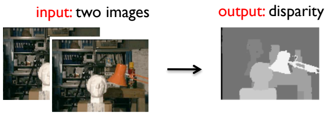
$$\log p(x_i^m \mid x_{N(i)}^m; \mathbf{w}) = \mathbf{w} \cdot \sum_{c : i \in c} f_c(x_c^m) - \log Z(x_{N(i)}^m; \mathbf{w})$$

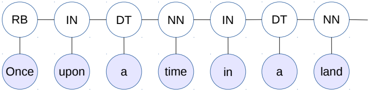
Pseudo-likelihood

- This objective only involves summation over x_i and is tractable
- Has many small partition functions (one for each variable and each setting of its neighbors) instead of one big one
- It is still concave in $\mathbf{w}$ and thus has no local maxima
- Assuming the data is drawn from a MRF with parameters $\mathbf{w}^*$, can show that as the number of data points gets large, $\mathbf{w}^{PL} \rightarrow \mathbf{w}^*$

Density estimation for CRFs

- Suppose we want to predict a set of variables $\mathbf{Y}$ given some others $\mathbf{X}$, e.g., stereo vision or part-of-speech tagging:





- We concentrate on predicting $p(\mathbf{Y}|\mathbf{X})$, and use a **conditional** loss function

$$loss(\mathbf{x}, \mathbf{y}, \hat{\mathcal{M}}) = -\log \hat{p}(\mathbf{y} \mid \mathbf{x}).$$

- Since the loss function only depends on $\hat{p}(\mathbf{y} \mid \mathbf{x})$, suffices to estimate the conditional distribution, not the joint

Density estimation for CRFs

$$\text{CRF:} \quad p(\mathbf{y} \mid \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{c \in C} \phi_c(\mathbf{x}, \mathbf{y}_c), \quad Z(\mathbf{x}) = \sum_{\hat{\mathbf{y}}} \prod_{c \in C} \phi_c(\mathbf{x}, \hat{\mathbf{y}}_c)$$

- Empirical risk minimization with CRFs, i.e. $\min_{\hat{\mathcal{M}}} \mathbf{E}_{\mathcal{D}} \left[loss(\mathbf{x}, \mathbf{y}, \hat{\mathcal{M}}) \right]$:

$$\begin{aligned}
\mathbf{w}^{ML} &= \arg\min_{\mathbf{w}} \frac{1}{|\mathcal{D}|} \sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{D}} -\log p(\mathbf{y} \mid \mathbf{x}; \mathbf{w}) \\
&= \arg\max_{\mathbf{w}} \sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{D}} \left(\sum_c \log \phi_c(\mathbf{x}, \mathbf{y}_c; \mathbf{w}) - \log Z(\mathbf{x}; \mathbf{w}) \right) \\
&= \arg\max_{\mathbf{w}} \ \mathbf{w} \cdot \left(\sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{D}} \sum_c \mathbf{f}_c(\mathbf{x}, \mathbf{y}_c) \right) - \sum_{(\mathbf{x},\mathbf{y}) \in \mathcal{D}} \log Z(\mathbf{x}; \mathbf{w})
\end{aligned}$$