CS 6347

Lecture 13

Maximum Likelihood Learning

Maximum Likelihood Estimation

- Given samples $x^1, \dots, x^M$ from some unknown distribution with parameters θ...
  - The log-likelihood of the evidence is defined to be

$$\log l(\theta) = \sum_m \log p(x|\theta)$$

  - Goal: maximize the log-likelihood

MLE for Bayesian Networks

- Given samples $x^1, \dots, x^M$ from some unknown Bayesian network that factors over the directed acyclic graph G
  - The parameters of a Bayesian model are simply the conditional probabilities that define the factorization
  - For each $i \in G$ we need to learn $p(x_i | x_{parents(i)})$, create a variable $\theta_{x_i | x_{parents(i)}}$

$$\log l(\theta) = \sum_m \sum_{i \in V} \log \theta_{x_i^m | x_{parents(i)}^m}$$

MLE for Bayesian Networks

$$\log l(\theta) = \sum_{m} \sum_{i \in V} \log \theta_{x_i^m | x_{parents(i)}^m}$$

$$= \sum_{i \in V} \sum_{m} \log \theta_{x_i^m | x_{parents(i)}^m}$$

$$= \sum_{i \in V} \sum_{x_{parents(i)}} \sum_{x_i} \mathrm{N}_{x_{\mathrm{i}}, x_{\mathrm{parents(i)}}} \log \theta_{x_i | x_{parents(i)}}$$

MLE for Bayesian Networks

$$\log l(\theta) = \sum_m \sum_{i \in V} \log \theta_{x_i^m | x_{parents(i)}^m}$$

$$= \sum_{i \in V} \sum_m \log \theta_{x_i^m | x_{parents(i)}^m}$$

$$= \sum_{i \in V} \sum_{x_{parents(i)}} \sum_{x_i} \mathrm{N}_{x_{\mathrm{i}}, x_{\mathrm{parents(i)}}} \log \theta_{x_i | x_{parents(i)}}$$

$N_{x_i, x_{parents(i)}}$ is the number of times $(x_i, x_{parents(i)})$ was observed in the samples

MLE for Bayesian Networks

$$\log l(\theta) = \sum_m \sum_{i \in V} \log \theta_{x_i^m | x_{parents(i)}^m}$$

$$= \sum_{i \in V} \sum_m \log \theta_{x_i^m | x_{parents(i)}^m}$$

$$= \sum_{i \in V} \sum_{x_{parents(i)}} \sum_{x_i} \mathrm{N}_{x_{\mathrm{i}}, x_{\mathrm{parents(i)}}} \log \theta_{x_i | x_{parents(i)}}$$

Fix $x_{parents(i)}$ solve for $\theta_{x_i | x_{parents(i)}}$ for all x_i

(on the board)

MLE for Bayesian Networks

$$\theta_{x_i|x_{parents(i)}} = \frac{\mathrm{N}_{x_i,x_{\text{parents}(i)}}}{\sum_{x_i'} \mathrm{N}_{x_i',x_{\text{parents}(i)}}} = \frac{\mathrm{N}_{x_i,x_{\text{parents}(i)}}}{\mathrm{N}_{x_{\text{parents}(i)}}}$$

- This is just the empirical conditional probability distribution
  - Worked out nicely because of the factorization of the joint distribution
- Similar to the coin flips result from last time

MLE for MRFs

- Let's compute the MLE for MRFs that factor over the graph G as $p(x) = \frac{1}{Z(\theta)} \prod_C \psi_C(x_C|\theta)$

- The parameters θ control the allowable potential functions

- Again, suppose we have samples $x^1, \dots, x^M$ from some unknown MRF of this form

$$\log l(\theta) = \left[\sum_m \sum_C \log \psi_C(x_C^m|\theta)\right] - M \log Z\,(\theta)$$

MLE for MRFs

- Let's compute the MLE for MRFs that factor over the graph G as $p(x) = \frac{1}{Z(\theta)} \prod_C \psi_C(x_C|\theta)$

- The parameters θ control the allowable potential functions

- Again, suppose we have samples $x^1, \ldots, x^M$ from some unknown MRF of this form

$$\log l(\theta) = \left[\sum_m \sum_C \log \psi_C(x_C^m|\theta)\right] - M \log Z(\theta)$$

$Z(\theta)$ couples all of the potential functions together!

Even computing $Z(\theta)$ by itself was a challenging task…

Conditional Random Fields

- Learning MRFs is quite restrictive
  - Most “real” problems are really conditional models
- Example: image segmentation
  - Represent a segmentation problem as a MRF over a two dimensional grid
  - Each x_i is an binary variable indicating whether or not the pixel is in the foreground or the background
  - How do we incorporate pixel information?
    - The potentials over the edge (i, j) of the MRF should depend on x_i, x_j as well as the pixel information at nodes i and j

Feature Vectors

- The pixel information is called a feature of the model
  - Features will consist of more than just a scalar value (i.e., pixels, at the very least, are vectors of RGBA values)
- Vector of features y (e.g., one vector of features y_i for each $i \in V$)
  - We think of the joint probability distribution as a conditional distribution $p(x|y, \theta)$
- This makes MLE even harder
  - Samples are pairs $(x^1, y^1), \ldots, (x^M, y^M)$
  - The feature vectors can be different for each sample: need to compute $Z(\theta, y^m)$ in the log-likelihood!

Log-Linear Models

- MLE seems daunting for MRFs and CRFs
  - Need a nice way to parameterize the model and to deal with features
- We often assume that the models are log-linear in the parameters
  - Many of the models that we have seen so far can easily be expressed as log-linear models of the parameters
  - Example: represent the s-t cut problem as a log-linear model (on the board)

Log-Linear Models

- Feature vectors should also be incorporated in a log-linear way
  - There is no fixed way to do this: it is up to you to decide how best to incorporate your feature information into the model
- The potential on the clique C should be a log-linear function of the parameters

$$\psi_C(x_C|y,\theta) = \exp(\langle \theta, f_C(x_C, y)\rangle)$$

- Here, f is a feature map that takes a collection of feature vectors and returns a vector
  - What might be a good feature map for image segmentation?

MLE for Log-Linear Models

$$p(x|y,\theta) = \frac{1}{Z(\theta,y)} \prod_C \exp(\langle \theta, f_C(x_C, y) \rangle)$$

$$\log l(\theta) = \sum_m \left[\sum_C \langle \theta, f_C(x_C^m, y^m) \rangle \right] - \log Z(\theta, y^m)$$

$$= \left\langle \theta, \sum_m \sum_C f_C(x_C^m, y^m) \right\rangle - \sum_m \log Z(\theta, y^m)$$

MLE for Log-Linear Models

$$p(x|y,\theta) = \frac{1}{Z(\theta,y)} \prod_C \exp(\langle \theta, f_C(x_C, y) \rangle)$$

$$\log l(\theta) = \sum_m \left[\sum_C \langle \theta, f_C(x_C^m, y^m) \rangle \right] - \log Z(\theta, y^m)$$

$$= \underbrace{\left\langle \theta, \sum_m \sum_C f_C(x_C^m, y^m) \right\rangle}_{\text{Linear in } \theta} - \underbrace{\sum_m \log Z(\theta, y^m)}_{\text{Depends non-linearly on } \theta}$$

Concavity of MLE

We will show that $\log Z(\theta, y)$ is a convex function of θ...

Fix a distribution $q(\mathrm{x}|\mathrm{y})$

$$
\begin{aligned}
D(q||p) &= \sum_x q(x|y) \log \frac{q(x|y)}{p(x|y,\theta)} \\
&= \sum_x q(x|y) \log q(x|y) \; - \sum_x q(x|y) \log p(x|y,\theta) \\
&= -H(q) \; - \sum_x q(x|y) \log p(x|y,\theta) \\
&= -H(q) + \log Z(\theta, y) - \sum_x \sum_C q(x|y) \langle \theta, f_C(x_C, y) \rangle \\
&= -H(q) + \log Z(\theta, y) - \sum_C \sum_{x_C} q_C(x_C|y) \langle \theta, f_C(x_C, y) \rangle
\end{aligned}
$$

Concavity of MLE

$$\log Z(\theta, y) = \max_q \left[H(q) + \sum_C \sum_{x_C} \underbrace{q_C(x_C|y)\langle\theta, f_C(x_C, y)\rangle}_{\text{Linear in } \theta} \right]$$

- If a function $g(x, y)$ is convex in x for each y, then $\max_y g(x, y)$ is convex in y
  - As a result, $\log Z(\theta, y)$ is a convex function of θ

MLE for Log-Linear Models

$$p(x|y,\theta) = \frac{1}{Z(\theta,y)} \prod_C \exp(\langle \theta, f_C(x_C, y) \rangle)$$

$$\log l(\theta) = \sum_m \left[\sum_C \langle \theta, f_C(x_C^m, y^m) \rangle \right] - \log Z(\theta, y^m)$$

$$= \underbrace{\left\langle \theta, \sum_m \sum_C f_C(x_C^m, y^m) \right\rangle}_{\text{Linear in } \theta} - \underbrace{\sum_m \log Z(\theta, y^m)}_{\text{Convex in } \theta}$$

MLE for Log-Linear Models

$$p(x|y,\theta) = \frac{1}{Z(\theta,y)} \prod_C \exp(\langle \theta, f_C(x_C, y) \rangle)$$

$$\log l(\theta) = \sum_m \left[\sum_C \langle \theta, f_C(x_C^m, y^m) \rangle \right] - \log Z(\theta, y^m)$$

$$= \underbrace{\left\langle \theta, \sum_m \sum_C f_C(x_C^m, y^m) \right\rangle - \sum_m \log Z(\theta, y^m)}$$

Concave in θ

Could optimize it using gradient ascent!
(need to compute $\nabla_\theta \log Z(\theta, y)$)

MLE via Gradient Ascent

- What is the gradient of the log-likelihood with respect to θ?

$$\nabla_\theta \log l(\theta) = ?$$

(worked out on board)

MLE via Gradient Ascent

- What is the gradient of the log-likelihood with respect to θ?

$$\nabla_\theta \log l(\theta) = \sum_C \sum_{x_C} p_C(x_C|y,\theta) f_C(x_C, y)$$

  - This is the expected value of the feature maps under the joint distribution
  - To compute/approximate this quantity, we only need to compute/approximate the marginal distributions $p_C(x_C|y,\theta)$
  - This requires performing marginal inference on a different model at each step of gradient ascent!