CS 6347

Lecture 14-15

More Maximum Likelihood

Recap

- Last week: Introduction to maximum likelihood estimation
  - MLE for Bayesian networks
    - Optimal CPTs correspond to empirical counts
- Today: MLE for CRFs
- Announcements:
  - HW 3 is available and due 3/11

Maximum Likelihood Estimation

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

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

  - Goal: maximize the log-likelihood

MLE for MRFs

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

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

MLE for MRFs

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

- 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

Log-Linear Models

- Feature vectors should also be incorporated in a log-linear way
- The potential on the clique C should be a log-linear function of the parameters

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

where

$$\langle\theta, f_C(x_C,y)\rangle = \sum_k \theta_k \cdot f_C(x_C,y)_k$$

- Here, f is a feature map that takes a collection of feature vectors and returns a vector the same size as θ

UTD

Log-Linear MRFs

- Over complete representation: one parameter for each clique C and choice of x_C

$$p(x|\theta) = \frac{1}{Z}\prod_C \exp(\theta_C(x_C))$$

  - $f_C(x_C)$ is a 0-1 vector that is indexed by C and x_C whose only non-zero component corresponds to $\theta_C(x_C)$

- One parameter per clique

$$p(x|\theta) = \frac{1}{Z}\prod_C \exp(\theta_C f_C(x_C))$$

  - $f_C(x_C)$ is a vector that is indexed ONLY by C whose only non-zero component corresponds to θ_C

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

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

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

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

Linear in θ

Depends non-linearly on θ

Concavity of MLE

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

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

$$D(q||p) = \sum_x q(x|y) \log \frac{q(x|y)}{p(x|y,\theta)}$$

$$= \sum_x q(x|y) \log q(x|y) \; - \sum_x q(x|y) \log p(x|y,\theta)$$

$$= -H(q) \; - \sum_x q(x|y) \log p(x|y,\theta)$$

$$= -H(q) + \log Z(\theta, y) - \sum_x \sum_C q(x|y) \langle \theta, f_C(x_C, y) \rangle$$

$$= -H(q) + \log Z(\theta, y) - \sum_C \sum_{x_C} q_C(x_C|y) \langle \theta, f_C(x_C, y) \rangle$$

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

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

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

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

Linear in θ

Convex in θ

MLE for Log-Linear Models

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

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

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

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?

$$\nabla_\theta \log Z(\theta, y^m) = ?$$

(worked out on board)

MLE via Gradient Ascent

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

$$\nabla_\theta \log Z(\theta, y^m) = \sum_C \sum_m \sum_{x_C} p_C(x_C | y^m, \theta) f_C(x_C, y^m)$$

This is the expected value of the feature maps under the joint distribution

MLE via Gradient Ascent

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

$$\nabla_\theta \log l(\theta) = \sum_C \sum_m \left(f_C(x_C^m, y^m) - \sum_{x_C} p_C(x_C | y^m, \theta) f_C(x_C, y^m) \right)$$

  - 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!

Moment Matching

- Let $f(x^m, y^m) = \sum_C f_C(x_C^m, y^m)$
- Setting the gradient with respect to θ equal to zero and solving gives

$$\sum_m f(x^m, y^m) = \sum_m \sum_x p(x|y^m, \theta) f(x, y^m)$$

- This condition is called moment matching and when the model is an MRF instead of a CRF this reduces to

$$\frac{1}{M} \sum_m f(x^m) = \sum_x p(x|\theta) f(x)$$

Moment Matching

- To better understand why this is called moment matching, consider a log-linear MRF

$$p(x) = \frac{1}{Z} \prod_{C} \exp(\theta_C(x_C))$$

- That is, $f_C(x_C)$ is a vector that is indexed by C and x_C whose only non-zero component corresponds to $\theta_C(x_C)$

- The moment matching condition becomes

$$\frac{1}{M} \sum_{m} \delta(x_C = x_C^m) = p_C(x_C|\theta), \qquad \text{for all } C, x_C$$

Regularization in MLE

- Recall that we can also incorporate prior information about the parameters into the MLE problem
  - This involved solving an augmented MLE

$$\prod_m p(x^m|\theta)p(\theta)$$

  - What types of priors should we choose for the parameters?

Regularization in MLE

- Recall that we can also incorporate prior information about the parameters into the MLE problem
  - This involved solving an augmented MLE

$$\prod_m p(x^m|\theta)p(\theta)$$

  - What types of priors should we choose for the parameters?
    - Gaussian prior: $p(\theta) \propto \exp(-\frac{1}{2}x^T\Sigma^{-1}x^T + \mu^T x)$
    - Uniform over $[0,1]$

Regularization in MLE

- Recall that we can also incorporate prior information about the parameters into the MLE problem
  - This involved solving an augmented MLE

$$\prod_m p(x^m|\theta) \exp(-\frac{1}{2}\theta^T D\theta^T)$$

Gaussian prior with a diagonal covariance matrix all of whose entries are equal to λ

  - What types of priors should we choose for the parameters?
    - Gaussian prior: $p(\theta) \propto \exp(-\frac{1}{2}\theta^T\Sigma^{-1}\theta^T + \mu^T\theta)$
    - Uniform over $[0,1]$

Regularization in MLE

- Recall that we can also incorporate prior information about the parameters into the MLE problem
  - This involved solving an augmented MLE

$$\log \prod_m p(x^m|\theta) \exp(-\frac{1}{2}\theta^T D \theta^T) = \left[\sum_m \log p(x^m|\theta)\right] - \frac{\lambda}{2}\sum_k \theta_k^2$$

$$= \left[\sum_m \log p(x^m|\theta)\right] - \frac{\lambda}{2}||\theta||_2^2$$

Regularization in MLE

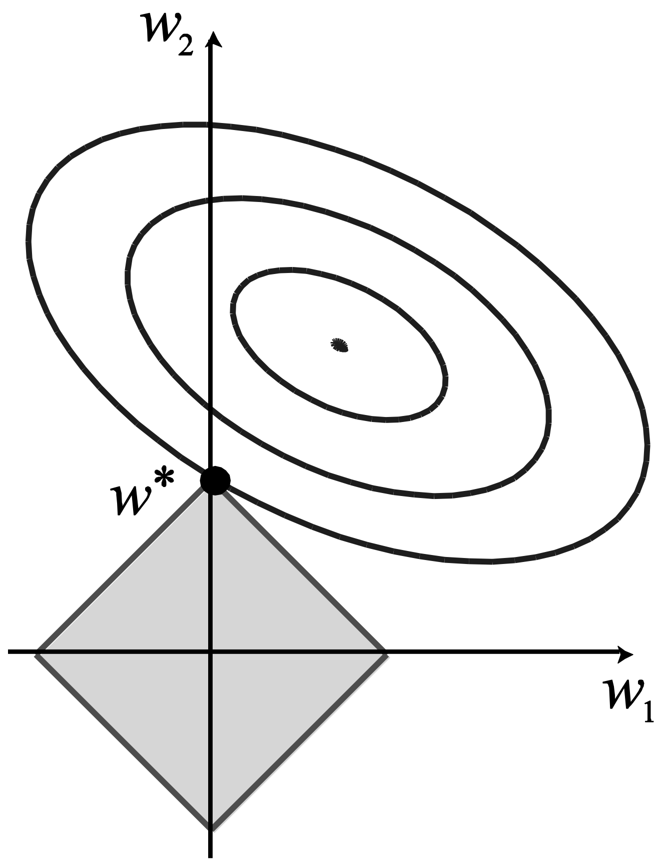
- Recall that we can also incorporate prior information about the parameters into the MLE problem
  - This involved solving an augmented MLE

$$\log \prod_m p(x^m|\theta) \exp(-\frac{1}{2}\theta^T D \theta^T) = \left[\sum_m \log p(x^m|\theta)\right] - \frac{\lambda}{2}\sum_k \theta_k^2$$

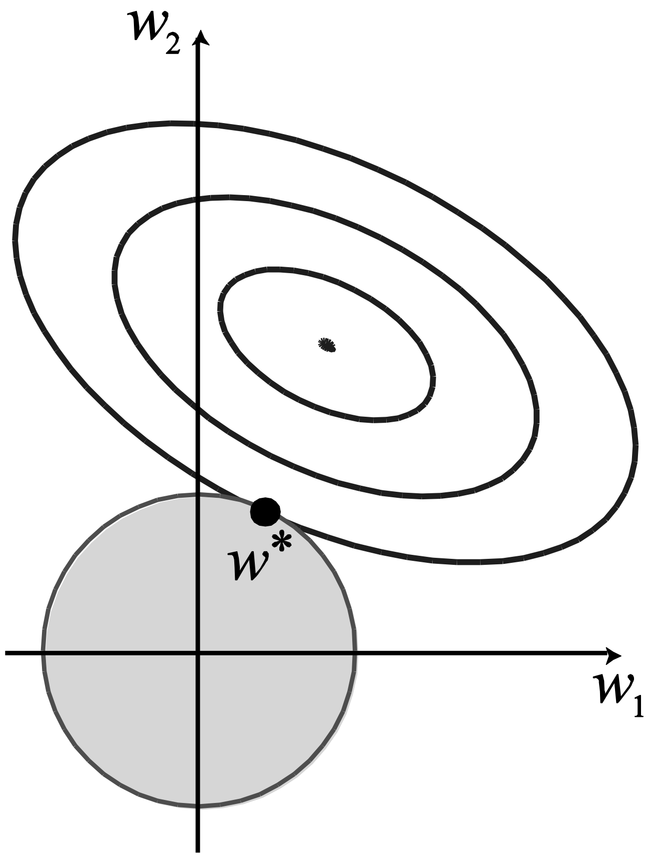
$$= \left[\sum_m \log p(x^m|\theta)\right] - \frac{\lambda}{2}||\theta||_2^2$$

Known as ℓ_2 regularization

Regularization



ℓ_1



ℓ_2

Duality and MLE

$$\log Z(\theta, y) = \max_{q} \left[H(q) + \sum_{C} \sum_{x_C} q_C(x_C | y) \langle \theta, f_C(x_C, y) \rangle \right]$$

$$\log l(\theta) = \left\langle \theta, \sum_{m} \sum_{C} f_C(x_C^m, y^m) \right\rangle - \sum_{m} \log Z(\theta, y^m)$$

Plugging the first into the second gives:

$$\log l(\theta) = \left\langle \theta, \sum_{m} \sum_{C} f_C(x_C^m, y^m) \right\rangle - \sum_{m} \max_{q^m} \left[H(q^m) + \sum_{C} \sum_{x_C} q_C^m(x_C | y^m) \langle \theta, f_C(x_C, y^m) \rangle \right]$$

Duality and MLE

$$\max_{\theta} \log l(\theta) = \max_{\theta} \min_{q^1,\ldots,q^M} \left[\left\langle \theta, \sum_C \sum_m \left(f_C(x_C^m, y^m) - \sum_{x_C} q_C^m(x_C|y^m) f_C(x_C, y^m) \right) \right\rangle - \sum_m H(q^m) \right]$$

- This is called a minimax or saddle-point problem
- Recall that we ended up with similar looking optimization problems when we constructed the Lagrange dual function
- When can we switch the order of the max and min?
  - The function is linear in theta, so there is an advantage to swapping the order

Sion's Minimax Theorem

Let X be a compact convex subset of R^n and Y be a convex subset of R^m

Let f be a real-valued function on $X \times Y$ such that

- $f(x,\cdot)$ is a continuous concave function over Y for each $x \in X$
- $f(\cdot, y)$ is a continuous convex function over X for each $y \in Y$

then

$$\sup_y \min_x f(x,y) = \min_x \sup_y f(x,y)$$

UTD

Duality and MLE

$$\max_{\theta} \min_{q^1,\ldots,q^M} \left[\left\langle \theta, \sum_C \sum_m \left(f_C(x_C^m, y^m) - \sum_{x_C} q_C^m(x_C|y^m) f_C(x_C, y^m) \right) \right\rangle - \sum_m H(q^m) \right]$$

is equal to

$$\min_{q^1,\ldots,q^M} \max_{\theta} \left[\left\langle \theta, \sum_C \sum_m \left(f_C(x_C^m, y^m) - \sum_{x_C} q_C^m(x_C|y^m) f_C(x_C, y^m) \right) \right\rangle - \sum_m H(q^m) \right]$$

Solve for θ?

Maximum Entropy

$$\max_{q^1,\ldots,q^M} \sum_m H(q^m)$$

such that the moment matching condition is satisfied

$$\sum_m f(x^m, y^m) = \sum_m \sum_x q^m(x|y^m) f(x, y^m)$$

and $q^1, \ldots, q^m$ are discrete probability distributions

- Instead of maximizing the log-likelihood, we could maximize the entropy over all approximating distributions that satisfy the moment matching condition

UTD

MLE in Practice

- We can compute the partition function in linear time over trees using belief propagation
  - We can use this to learn the parameters of tree-structured models
- What if the graph isn't a tree?
  - Use variable elimination to compute the partition function (exact but slow)
  - Use importance sampling to approximate the partition function (can also be quite slow; maybe only use a few samples?)
  - Use loopy belief propagation to approximate the partition function (can be bad if loopy BP doesn't converge quickly)

MLE in Practice

- Practical wisdom:
  - If you are trying to perform some prediction task (i.e., MAP inference to do prediction), then it is better to learn the "wrong model"
  - Learning and prediction should use the same approximations
- What people actually do:
  - Use a few iterations of loopy BP or sampling to approximate the marginals
  - Approximate marginals give approximate gradients (recall that the gradient only depended on the marginals)
  - Perform approximate gradient descent and hope it works

MLE in Practice

- Other options
  - Replace the true entropy with the Bethe entropy and solve the approximate dual problem
  - Use fancier optimization techniques to solve the problem faster
    - e.g., the method of conditional gradients