CS 6347

Lecture 16

Alternatives to MLE
Course Project

- Pick a group (1-4) students
- Write a brief proposal and email it to me and Baoye
- Do the project
  - Collect/find a dataset
  - Build a graphical model
  - Solve approximately/exactly some inference or learning task
- Demo the project for the class (~20 mins during last 2-3 weeks)
  - Show your results
- Turn in a short write-up describing your project and results (due May 2)
Course Project

• Meet with me and/or Travis about two times (more if needed)
  – We’ll help you get started and make sure you picked a hard/easy enough goal

• For one person:
  – Pick a small data set (or generate synthetic data)
  – Formulate a learning/inference problem using MRFs, CRFs, Bayesian networks
  – Example: SPAM filtering with a Bayesian network using the UCI spambase data set (or other data sets)
  – Compare performance across data sets and versus naïve algorithms
Course Project

• For four people:
  – Pick a more complex data set
  – The graphical model that you learn should be more complicated than a simple Bayesian network
  – Ideally, the project will involve both learning and prediction using a CRF or an MRF (or a Bayesian network with hidden variables)
  – Example: simple binary image segmentation or smallish images
  – Be ambitious but cautious, you don’t want to spend a lot of time formatting the data or worrying about feature selection
Course Project

- Lots of other projects are possible
  - Read about, implement, and compare different approximate MAP inference algorithms (loopy BP, tree-reweighted belief propagation, max-sum diffusion)
  - Compare different approximate MLE schemes on synthetic data
  - Perform a collection of experiments to determine when the MAP LP is tight across a variety of pairwise, non-binary MRFs
  - If you are stuck, have a vague idea, ask me about it!
Course Project

• What you need to do now
  – Find some friends (you can post on Piazza if you need friends)
  – Pick a project
  – Email me and Baoye (with all of your group members cc’d) by 3/20

• Grade will be determined based on the demo, final report, and project difficulty
Recap

• Last week:
  – MLE for MRFs and CRFs

• Today:
  – Alternatives to MLE: Pseudolikelihood, piecewise likelihood, discriminative based learning
Alternatives to MLE

• Exact MLE estimation is intractable
  – To compute the gradient of the log-likelihood, we need to compute marginals of the model

• Alternatives include
  – Pseudolikelihood approximation to the MLE problem that relies on computing only local probabilities
  – For structured prediction problems, we could avoid likelihoods entirely by minimizing a loss function that measures our prediction error
Pseudolikelihood

- Consider a log-linear MRF $p(x|\theta) = \frac{1}{Z(\theta)} \prod_c \exp\langle \theta, f_c(x_c) \rangle$

- By the chain rule, the joint distribution factorizes as
  
  $p(x|\theta) = \prod_i p(x_i|x_1, \ldots, x_{i-1}, \theta)$

- This quantity can be approximated by conditioning on all of the other variables (called the pseudolikelihood)
  
  $p(x|\theta) \approx \prod_i p(x_i|x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n, \theta)$
Pseudolikelihood

• Using the independence relations from the MRF

\[ p(x|\theta) \approx \prod_i p(x_i|x_{N(i)}, \theta) \]

• Only requires computing local probability distributions (typically much easier)
  
  – Does not require knowing \( Z(\theta) \)
Pseudolikelihood

- For samples $x^1, \ldots, x^M$

$$\log \ell_{PL}(\theta) = \sum_{m} \sum_{i} \log p(x^m_i | x^m_{N(i)}, \theta)$$

- This approximation is called the pseudolikelihood
  
  - If the data is generated from a model of this form, then in the limit of infinite data, maximizing the pseudolikelihood recovers the true model parameters
  
  - Can be much more efficient to compute than the log likelihood
Pseudolikelihood

\[
\log \ell_{PL}(\theta) = \sum_m \sum_i \log p(x_i^m | x_N(i), \theta)
\]

\[
= \sum_m \sum_i \log \frac{p(x_i^m, x_N(i)^m | \theta)}{\sum_{x_i'} p(x_i', x_N(i)^m | \theta)}
\]

\[
= \sum_m \sum_i \left[ \log p(x_i^m, x_N(i)^m | \theta) - \log \sum_{x_i'} p(x_i', x_N(i)^m | \theta) \right]
\]

\[
= \sum_m \sum_i \left[ \left\langle \theta, \sum_{c \supset i} f_c(x_c^m) \right\rangle - \log \sum_{x_i'} \exp \left\langle \theta, \sum_{c \supset i} f_c(x_i', x_c^{m \setminus i}) \right\rangle \right]
\]
\[
\log \ell_{PL}(\theta) = \sum_m \sum_i \log p(x^m_i | x^m_{N(i)}, \theta)
\]

\[
= \sum_m \sum_i \log \frac{p(x^m_i, x^m_{N(i)} | \theta)}{\sum_{x'_i} p(x'_i, x^m_{N(i)} | \theta)}
\]

\[
= \sum_m \sum_i \left[ \log p(x^m_i, x^m_{N(i)} | \theta) - \log \sum_{x'_i} p(x'_i, x^m_{N(i)} | \theta) \right]
\]

\[
= \sum_m \sum_i \left[ \theta, \sum_{C \ni i} f_C(x^m_C) \right] - \log \sum_{x'_i} \exp \left( \theta, \sum_{C \ni i} f_C(x'_i, x^m_{C \setminus i}) \right)
\]

Only involves summing over \( x_i \)!
Pseudolikelihood

\[
\log \ell_{PL}(\theta) = \sum_{m} \sum_{i} \log p(x_i^m | x_{N(i)}^m, \theta)
= \sum_{m} \sum_{i} \log \frac{p(x_i^m, x_{N(i)}^m | \theta)}{\sum_{x'_i} p(x'_i, x_{N(i)}^m | \theta)}
= \sum_{m} \sum_{i} \left[ \log p(x_i^m, x_{N(i)}^m | \theta) - \log \sum_{x'_i} p(x'_i, x_{N(i)}^m | \theta) \right]
= \sum_{m} \sum_{i} \left[ \theta, \sum_{C \supset i} f_C(x_C^m) \right] - \log \sum_{x'_i} \exp \left[ \theta, \sum_{C \supset i} f_C(x'_i, x_{C \setminus i}^m) \right]
\]

Concave in \( \theta \)!
Consistency of Pseudolikelihood

- Pseudolikelihood is a consistent estimator
  - That is, in the limit of large data, it is exact if the true model belongs to the family of distributions being modeled

\[
\nabla_\theta \ell_{PL} = \sum_{m} \sum_{i} \left[ \sum_{c \supset i} f_c(x_c^m) - \frac{\sum_{x_i'} \exp(\theta, \sum_{c \supset i} f_c(x_i', x_c^m)) \sum_{c \supset i} f_c(x_i', x_c^m)}{\sum_{x_i'} \exp(\theta, \sum_{c \supset i} f_c(x_i', x_c^m))} \right]
\]

\[
= \sum_{m} \sum_{i} \left[ \sum_{c \supset i} f_c(x_c^m) - \sum_{x_i'} p(x_i'|x_{N(i)}^m, \theta) \sum_{c \supset i} f_c(x_i', x_c^m) \right]
\]

Can check that the gradient is zero in the limit of large data if \( \theta = \theta^* \)
Structured Prediction

- Suppose we have a CRF, \( p(x|y, \theta) = \frac{1}{Z(\theta, y)} \prod_C \exp(\langle \theta, f_C(x_C, y) \rangle) \)

- If goal is to compute \( \arg\max_x p(x|y) \), then MLE may be overkill
  - We only care about classification error, not about learning the correct marginal distributions as well

- Recall that the classification error is simply the expected number of incorrect predictions made by the learned model on samples from the true distribution

- Instead of maximizing the likelihood, we can minimize the classification error over the training set
Structured Prediction

- For samples \((x^1, y^1), \ldots, (x^M, y^M)\), the (unnormalized) classification error is

\[
\sum_m 1\{x^m \in \arg\max_x p(x|y^m, \theta)\}
\]

- The classification error is zero when \(p(x^m|y^m, \theta) \geq p(x|y^m, \theta)\) for all \(x\) and \(m\) or equivalently

\[
\langle \theta, \sum_c f_c(x^m_c, y^m) \rangle \geq \langle \theta, \sum_c f_c(x_c, y^m) \rangle
\]
Structured Prediction

• In the exact case, this can be thought of as having a linear constraint for each possible $x$ and each $y^1, \ldots, y^M$

$$\left(\theta, \sum_C [f_C(x^m_C, y^m) - f_C(x_C, y^m)] \right) \geq 0$$

• Any $\theta$ that simultaneously satisfies each of these constraints will guarantee that the classification error is zero

  – As there are exponentially many constraints, finding such a $\theta$ (if one even exists) is still a challenging problem

  – If such a $\theta$ exists, we say that the problem is separable
Structured Perceptron Algorithm

- In the separable case, a straightforward algorithm can be designed for this task
- Choose an initial $\theta$
- Iterate until convergence
  - For each $m$
    - Choose $x' \in \arg\max_x p(x | y^m, \theta)$
    - Set $\theta = \theta + \sum_c [f_c(x_c^m, y^m) - f_c(x'_c, y^m)]$
Other Alternatives

- Piecewise likelihood uses the observation that $Z(\theta)$ is a convex function of $\theta$

  \[ Z \left( \sum_T \alpha_T \theta_T \right) \leq \sum_T \alpha_T Z(\theta_T) \]

- If $Z(\theta_T)$ corresponds to a tree-structured distribution, then the upper bound can be computed in polynomial time

- To do learning, we minimize the upper bound over $\theta_1, \ldots, \theta_T$

- Instead of using arbitrary $T$, the piecewise likelihood constructs an upper bound on the objective function by summing over $\theta | C$ obtained by zeroing out all components of $\theta$ except for those over the clique $C$ (not always possible)