Inference and Representation, Fall 2015

Problem Set 5: Structure learning & Gaussian processes

Due: Tuesday, December 1, 2015 at 3pm (uploaded to NYU Classes.)

Your submission should include a PDF file called "solutions.pdf" with your written solutions, separate output files, and all of the code that you wrote.

Important: *See problem set policy on the course web site.*

For this assignment, you are allowed to use basic graph packages (e.g., for representing and working with undirected graphs, or for finding the maximum spanning tree), but are **not** permitted to use any machine learning, graphical models, or probabilistic inference packages.

1. **Tree factorization.** Let T denote the edges of a tree-structured pairwise Markov random field with vertices V. For the special case of trees, prove that *any* distribution $p_T(\mathbf{x})$ corresponding to a Markov random field over T admits a factorization of the form:

$$p_T(\mathbf{x}) = \prod_{(i,j)\in T} \frac{p_T(x_i, x_j)}{p_T(x_i)p_T(x_j)} \prod_{j\in V} p_T(x_j), \tag{1}$$

where $p_T(x_i, x_j)$ and $p_T(x_i)$ denote pairwise and singleton marginals of the distribution p_T, respectively.

Hint: consider the Bayesian network where you choose an arbitrary node to be a root and direct all edges away from the root. Show that this is equivalent to the MRF. Then, looking at the BN's factorization, reshape it into the required form.

2. **Chow-Liu algorithm.** When trying to do object detection from computer images, *context* can be very helpful. For example, if "car" and "road" are present in an image, then it is likely that "building" and "sky" are present as well (see Figure 1). In recent work, a tree-structured Markov random field (see Figure 2) was shown to be particularly useful for modeling the prior distribution of what objects are present in images and using this to improve object detection [1].

You will replicate some of the results from [1] (it is not necessary to read this paper to complete this assignment). Specifically, you will implement the Chow-Liu algorithm

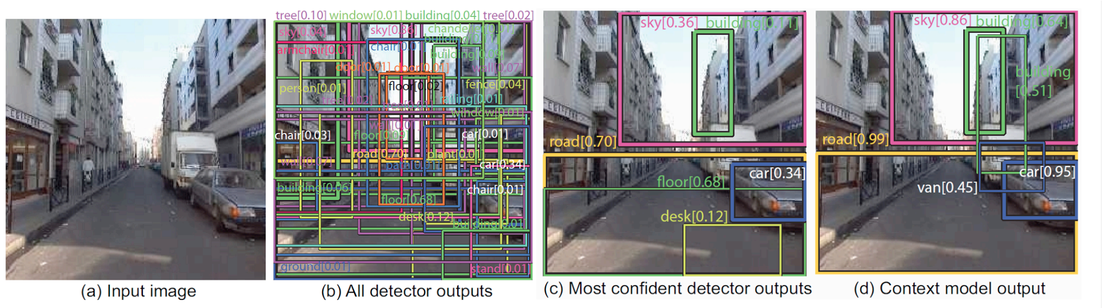

Figure 1: Using context within object detection for computer vision. [1]

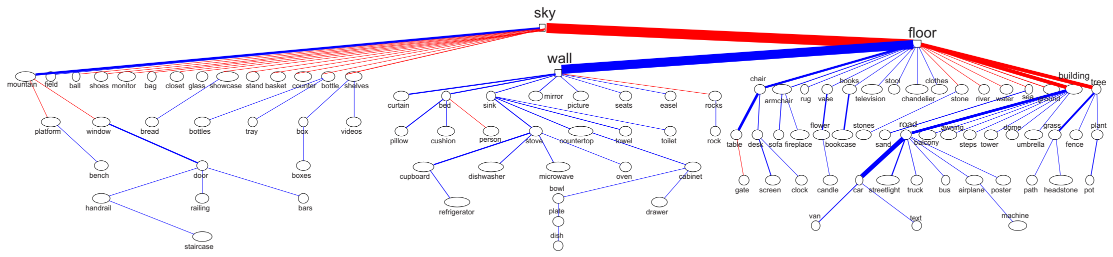

Figure 2: Pairwise MRF of object class presences in images [1]. Red edges denote negative correlations between classes. The thickness of each edge represents the strength of the link. You will be learning this MRF in question 2(a).

(1968) for maximum likelihood learning of tree-structured Markov random fields [2]. See also Murphy's book Section 26.3 for a brief overview.

The goal of learning is to find the tree-structured distribution $p_T(\mathbf{x})$ that maximizes the log-likelihood of the training data $\mathcal{D} = \{\mathbf{x}\}$:

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

Recall from Lecture 10 that for a fixed structure T, the maximum likelihood parameters for a MRF have a property called *moment matching*, meaning that the learned distribution will have marginals $p_T(x_i, x_j)$ equal to the empirical marginals $\hat{p}(x_i, x_j)$ computed from the data $\mathcal{D}$, i.e. $\hat{p}(x_i, x_j) = \mathit{count}(x_i, x_j)/|\mathcal{D}|$ where $\mathit{count}(x_i, x_j)$ is the number of data points in $\mathcal{D}$ with $X_i = x_i$ and $X_j = x_j$. Thus, using the factorization from Eq. 1, the learning task is reduced to solving

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

We can simplify the quantity being maximized over T as follows (let $N = |\mathcal{D}|$):

$$\begin{aligned}
&= \sum_{\mathbf{x}\in\mathcal{D}} \Big(\sum_{(i,j)\in T} \log \left[\frac{\hat{p}(x_i, x_j)}{\hat{p}(x_i)\hat{p}(x_j)} \right] + \sum_{j\in V} \log\left[\hat{p}(x_j)\right] \Big) \\
&= \sum_{(i,j)\in T} \sum_{\mathbf{x}\in\mathcal{D}} \log \left[\frac{\hat{p}(x_i, x_j)}{\hat{p}(x_i)\hat{p}(x_j)} \right] + \sum_{j\in V} \sum_{\mathbf{x}\in\mathcal{D}} \log\left[\hat{p}(x_j)\right] \\
&= \sum_{(i,j)\in T} \sum_{x_i, x_j} N\hat{p}(x_i, x_j) \log \left[\frac{\hat{p}(x_i, x_j)}{\hat{p}(x_i)\hat{p}(x_j)} \right] + \sum_{j\in V} \sum_{x_i} N\hat{p}(x_i) \log\left[\hat{p}(x_j)\right] \\
&= N \Big(\sum_{(i,j)\in T} I_{\hat{p}}(X_i, X_j) - \sum_{j\in V} H_{\hat{p}}(X_j) \Big),
\end{aligned}$$

where $I_{\hat{p}}(X_i, X_j) = \sum_{x_i, x_j} \hat{p}(x_i, x_j) \log \frac{\hat{p}(x_i, x_j)}{\hat{p}(x_i)\hat{p}(x_j)}$ is the empirical *mutual information* of variables X_i and X_j, and $H_{\hat{p}}(X_i)$ is the empirical *entropy* of variable X_i. Since the entropy terms are not a function of T, these can be ignored for the purpose of finding the maximum likelihood tree structure. **We conclude that the maximum likelihood tree can be**

obtained by finding the maximum-weight spanning tree in a complete graph with edge weights $I_{\hat{p}}(X_i, X_j)$ for each edge (i, j).

The Chow-Liu algorithm then consists of the following two steps:

(a) Compute each edge weight based on the empirical mutual information.

(b) Find a maximum spanning tree (MST) via Kruskal or Prim's Algorithm.

(c) Output a pairwise MRF with edge potentials $\phi_{ij}(x_i, x_j) = \frac{\hat{p}(x_i,x_j)}{\hat{p}(x_i)\hat{p}(x_j)}$ for each $(i, j) \in T$ and node potentials $\phi_i(x_i) = \hat{p}(x_i)$.

We have one random variable $X_i \in \{0, 1\}$ for each object type (e.g., "car" or "road") specifying whether this object is present in a given image. For this problem, you are provided with a matrix of dimension $N \times M$ where $N = 4367$ is the number of images in the training set and $M = 111$ is the number of object types. This data is in the file "chowliu-input.txt", and the file "names.txt" specifies the object names corresponding to each column.

Implement the Chow-Liu algorithm described above to learn the maximum likelihood tree-structured MRF from the data provided. Your code should output the MRF in the standard UAI format described here:

`http://www.hlt.utdallas.edu/~vgogate/uai14-competition/modelformat.html`

3. A major computational burden in using Gaussian Processes is inverting the covariance matrix, which is necessary for both parameter inference and for prediction. Is it possible to have a "precision function" $p(x_i, x_j)$ that provides the (i, j)th entry of the precision matrix (inverse covariance matrix), analogous to the covariance function $k(x_i, x_j)$ which takes as input the data points x_i and x_j and computes the (i, j)th entry of the covariance matrix? Please explain why or why not.

4. For four different covariance functions, make side-by-side plots showing for different parameter values (a) the shape of the covariance function, and (b) samples from the Gaussian process with this covariance functions. Make sure to include both compact-/non-compactly supported and differentiable/non-differentiable covariance functions. (You should write code to do this; do not use an existing Gaussian process software package.)

References

[1] Myung Jin Choi, Joseph J. Lim, Antonio Torralba, and Alan S. Willsky. Exploiting hierarchical context on a large database of object categories. In *IEEE Conference on Computer VIsion and Pattern Recognition (CVPR)*, 2010.

[2] C. K. Chow and C. N. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14:462–467, 1968.