CSE 515, Statistical methods in CS, Spring 2013: Assignment 3
Due: Friday, 24th May, 11:59am

For this assignment, you will turn-in a write-up answering the questions below *and* all your code via catalyst dropbox `https://catalyst.uw.edu/collectit/assignment/dvij/26655/111633`.

1 Programming: Loopy Belief Propagation [80 points]

Task description. The task we will consider in this assignment is binary image denoising. Given a noise-corrupted version of a binary image, you are required to recover the original image. The dataset we provide consists of a set of corrupted images in grayscale.

You will model the denoising problem as an inference problem in a CRF (figure 2) where the observations are the noisy pixels p_i given to you and the random variables are the true binary pixels $Y_i \in \{0,1\}$. The CRF is structured as a grid where every nodes are pixels (the set of nodes is denoted V) and there are edges between every pixel and its neighboring pixels -above,below,right left (the set of edges is denoted E). The node log-potentials are defined to be

$$\theta_{i;0} = \theta^n_{00} p_i + \theta^n_{01}$$
$$\theta_{i;1} = \theta^n_{10} p_i + \theta^n_{11}$$

where p_i is the value of the i-th pixel in the corrupted image and θ^n is a 2×2 matrix. The edge log-potentials are the same for all edges and given by

$$\theta_{ij;kl} = \theta^e_{kl} \quad k,l \in \{0,1\}.$$

The final joint probability distribution over the labels y is

$$P(Y) \propto \exp\left(\sum_{i\in V}\left(\sum_{k\in\{0,1\}} \theta_{i;k}\mathbb{I}\left[Y_i = k\right]\right) + \sum_{(i,j)\in E}\left(\sum_{k,l\in\{0,1\}} \mathbb{I}\left[Y_i = k\right]\mathbb{I}\left[Y_j = l\right]\theta_{ij;kl}\right)\right)$$

You are given the parameters of the model θ^n, θ^e and a script that converts these into the node and edge potentials for the entire network. You are required to implement the function bp in the script `test_bp.m` that takes the node and edge potentials and performs loopy belief propagation on the CRF to compute (approximations to) the node and edge marginals b_i, b_{ij}. This script will visualize your results and compute reconstruction errors. In figure 1, you can see an example of output from the script.

We have provided a file `Data.mat` which has a set of binary images and corresponding noisy versions pre-created for you. If you have the matlab image processing toolbox, you can generate additional datasets using the script `create_binimages.m`.

What to include in the write up: Include 5 examples of the images (original, noisy, reconstructred) produced by the script `test_bp.m`. Also play with the values of the parameters ($p.F$ in the script corresponding to θ^n, $p.G$ corresponding to θ^e) and see if you can improve results significantly. Report the mean reconstruction error over the entire dataset (setting $Ntest = Data.N$ in `test_bp.m`) for each choice of parameters.

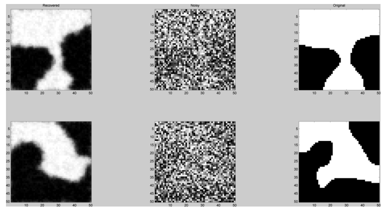

Figure 1: Reconstructed (left), Noisy (center), Original (right)

.

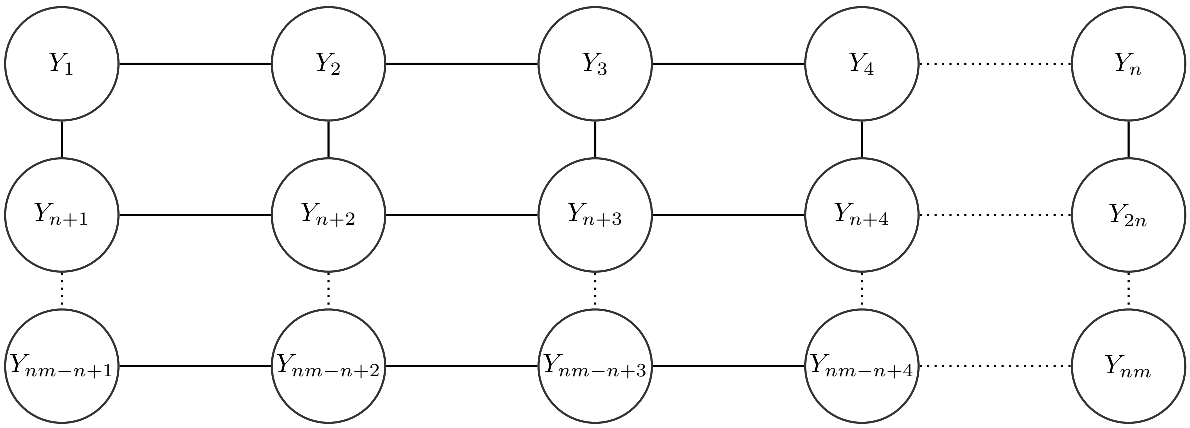

Figure 2: Grid-Structured CRF

2 Expectation Maximization for image segmentation [80 points]

Task description. In this part of the assignment, you will implement an algorithm for unsupervised image segmentation. The algorithm will perform clustering of pixels using the EM algorithm with a Gaussian mixture model, to produce clusters of similar pixels which produces a segmentation of the image. You are given a processed version of the Berkeley image segmentation dataset `http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/resources.html`, which contains color images with ground-truth human generated segmentations. The images have been processed into a matlab data structure with 5 real-valued features for each pixel (3 corresponding to color and 2 to position) in the files `Data.mat`. The file `GroundTruth.mat` contains ground truth human segmentations for each image.

You are required to implement a clustering algorithm to generate a segmentation of the image (where the clusters corresponds to image segments). You will use a Gaussian mixture model to represent the data

$$p(\mathbf{x}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\mathbf{x}; \mu_k, \Sigma) = \sum_{k=1}^{K} \pi_k \frac{\exp\left(-\frac{(\mathbf{x}-\mu_k)^T (\Sigma)^{-1} (\mathbf{x}-\mu_k)}{2}\right)}{\sqrt{(2\pi)^5 \det(\Sigma)}}.$$

where K is the number of clusters, μ_k are the cluster means, and Σ is a (uniform) cluster variance.

We can turn this into a model with a hidden variable z corresponding to the cluster index:

$$\begin{aligned} p(\mathbf{x}, z = k) &= p(\mathbf{x}|z = k)p(z = k) \\ p(\mathbf{x}|z = k) &= \mathcal{N}(\mathbf{x}; \mu_k, \Sigma) \\ p(z = k) &= \pi_k. \end{aligned}$$

Marginalizing out z, we get the initial mixture distribution over $\mathbf{x}$. This allows us to treat the problem of clustering pixels in an image (or equivalently segmenting the image into clusters) as a problem of learning the parameters of the above model where the variable z is hidden and $\mathbf{x}$ is a feature vector representation of each pixel in the image (5 dimensional in this problem). You will need to implement an EM-algorithm to maximize the likelihood of the observed data (feature vectors for every pixel in an image) wrt the model parameters $\{\mu_k, \pi_k\}$. You can set Σ to be a fixed diagonal matrix (you do not need to learn it):

$$\Sigma = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & \frac{S}{m} & 0 \\ 0 & 0 & 0 & 0 & \frac{S}{m} \end{bmatrix}$$

where m is in the range $[1, 40]$ and $S = \sqrt{\frac{\text{Number of pixels}}{\text{Number of clusters}}}$. Finally, to get the clustering out of this, you will compute

$$P(z_i = k|\mathbf{x}_i) = \frac{\pi_k \mathcal{N}(\mathbf{x}_i, \mu_k, \Sigma)}{\sum_j \pi_j \mathcal{N}(\mathbf{x}_i, \mu_j, \Sigma)}.$$

for each pixel i, and assign the pixel i to the cluster $\arg\max_k P(z_i = k|\mathbf{x}_i)$. The file `TestEM.m` will run your EM algorithm on every image in the dataset, and compute accuracy metrics (boundary recall, the fraction of cluster boundaries that occur within a distance of 2 pixels of the human-segmentations) averaged over the dataset.

What to include in the write up: Include examples of the image segmentations produced by your algorithm on some of the dataset images (original,human segmentation,ground segmentation) produced by the script `TestEM.m`. Also run your algorithm for different values of m ($m = 1, 10, 20, 30, 40$) and $K = 10, 15, 20$. Report the resulting boundary recall and precision (computed by `TestEM.m`) averaged over the entire dataset for each case. Also write about any qualitative differences in the results arising from changing parameters and why you think these differences occur.