Digital Image Processing Laboratory: Markov Random Fields and MAP Image Segmentation

December 11, 2015

1 Introduction

This laboratory explores the use of discrete Markov random fields (MRF) for applications such as segmentation. You should implement your solution to Section 2 in Matlab, **but your implementations to Sections 3 and 4 should be in ANSI C.**

For the purposes of this laboratory, the random field $\{X_s\}_{s\in S}$ will be a discrete valued random field. So, without loose of generality, X_s will take values in the set $\{0, \cdots, M-1\}$. Generally, X_s will represent the class of a pixel in an image.

We will assume that X_s is a MRF with a strictly positive density. We will also make the common assumption, that the Gibbs distribution for X uses only pairwise interactions. In this case, the set of all cliques is given by

$$\mathcal{C} = \{\{i,j\} | \, i \in \partial j \text{ for } i,j \in S\} \tag{1}$$

where ∂j denotes the neighbors of j. For this laboratory, we will assume that X_s takes values on an $N \times N$ rectangular lattice, S, with an 8-point neighborhood structure and circular boundary conditions. Since X is a strictly positive MRF, we know by the Hammersly-Clifford theorem that its discrete density function must have the form of a Gibbs distribution.

$$p(x) = \frac{1}{z} \exp\{-u(x)\} \tag{2}$$

We will further assume that the energy functional, $u(x)$, only has terms corresponding to pairwise interactions, so the energy functional is given by

$$u(x) = \beta \sum_{\{i,j\}\in\mathcal{C}} b_{i,j}\delta(x_i \neq x_j) \ . \tag{3}$$

where $\mathcal{C}$ is the set of all cliques and $b_{i,j}$ is the weighting assigned to each unique pixel pair. For this laboratory, we will assume that $b_{i,j}$ is given by the following function.

$$b_{i,j} = \begin{cases} \frac{1}{2\left(2+\sqrt{2}\right)} & \text{for } j = (i_1, (i_2 \pm 1)\text{mod}N) \\ \frac{1}{2\left(2+\sqrt{2}\right)} & \text{for } j = ((i_1 \pm 1)\text{mod}N, i_2) \\ \frac{1}{4(1+\sqrt{2})} & \text{for } j = ((i_1 \pm 1)\text{mod}N, (i_2 \pm 1)\text{mod}N) \end{cases} .$$

Questions or comments concerning this laboratory should be directed to Prof. Charles A. Bouman, School of Electrical and Computer Engineering, Purdue University, West Lafayette IN 47907; (765) 494-0340; bouman@ecn.purdue.edu

Notice that this function is choosen so that for all i

$$1 = \sum_{j\in\partial i} b_{i,j} \ .$$

The value of β determines the degree of regularity or smoothness in the random field X. When β is large and positive, the spatially adjacent pixels in X are highly correlated; when $\beta = 0$, they are independent; and when β is negative with large magnitude, then spatially adjacent pixels in X are likely to have differing values. Typically, large values of β are used to model the spatial regularity expected in accurate segmentations of real images.

2 Gibbs Sampler Simulation

It is often useful to generate samples from a discrete MRF such as X. There are a number of methods for doing this, but we will limit our experiments to the Gibbs sampler.

Define $p_i(m|x_{\partial s(k)})$ to be the discrete density of the pixel X_i given its neighbors $X_{\partial i}$. Then it is easily shown the this conditional density function is given by

$$p_i(m|x_{\partial s(k)}) = \frac{\exp\left\{-\beta \sum_{j\in\partial i} b_{i,j}\delta(m \neq x_j)\right\}}{\sum_{m'=0}^{M-1} \exp\left\{-\beta \sum_{j\in\partial i} b_{i,j}\delta(m' \neq x_j)\right\}} \ . \tag{4}$$

The Gibbs sampler generates sample form the distribution of the MRF by interatively replacing pixels in the MRF given their assumed distribution. More specifically, the Gibbs sampler is implemented with the following algorithm.

Gibbs Sampler Algorithm:

1. Set $N = \#$ of pixels
2. Initialize $X^{(0)}$
3. Order the N pixels as $N = s(0), \cdots, s(N-1)$
4. Repeat for $k = 0$ to *Iterations*
   (a) $X^{(k+1)} \leftarrow X^{(k)}$
   (b) For $i = 0$ to $N-1$
      generate $W \sim p_{s(i)}\left(x_{s(i)}|X^{(k+1)}_{\partial s(i)}\right)$
      $X^{(k+1)}_{s(i)} \leftarrow W$

Notice that one iteration of the Gibbs sampler represents a single pass through all pixels.

Section Problems:

1. Derive the result of equation 4 by using the form of the Gibbs distribution.
2. Run 100 iteration of the Gibbs sampler using a 8×8 image (i.e. $N = 64$), $M = 2$, and $\beta = 8$. Initialize the Gibbs sampler by generating $X^{(0)}$ using i.i.d. pixel values uniformly distributed between the values $\{0, 1\}$.
3. Repeat the experiment of part 1 four times.
4. Print out the results for each of your experiments.

3 MAP Segmentation using ICM

In this section, you will perform approximate MAP segmentation of an image under the assumption that each class is modeled by a Gaussian distribution with known mean and variance. Therefore, the conditional distribution of y_s given x_s has the form

$$p_{y_s|x_s}(y_s|x_s) \sim \mathcal{N}(\mu_{x_s}, \sigma_{x_s})$$

for $x_s \in \{0, \cdots, M-1\}$, and the log likelihood of the image, y, given an assumed segmentation, x, is given by

$$\log p_{y|x}(y|x) = \sum_{s \in S} \log p_{y_s|x_s}(y_s|x_s) \ .$$

We also assume that the segmentation, x, is an MRF with a Gibbs distribution of the form

$$p_x(x) = \frac{1}{z} \exp\left\{ -\beta \sum_{\{i,j\} \in \mathcal{C}} b_{i,j} \delta(x_i \neq x_j) \right\} \ . \tag{5}$$

Putting these together, we find that the MAP estimate of the segmentation is given by

$$\hat{x} = \arg \max_{x \in \{0,1\}^N} \left\{ \log p_{y|x}(y|x) + \log p(x) \right\} \tag{6}$$

$$= \arg \min_{x \in \{0,1\}^N} \left\{ \sum_{s \in S} - \log p_{y|x}(y_s|x_s) + \beta \sum_{\{i,j\} \in \mathcal{C}} b_{i,j} \delta(x_i \neq x_j) \right\} \ . \tag{7}$$

So the MAP estimat minimizes the following cost functional.

$$c(x) = \sum_{s \in S} - \log p_{y|x}(y_s|x_s) + \beta \sum_{\{i,j\} \in \mathcal{C}} b_{i,j} \delta(x_i \neq x_j) \tag{8}$$

In general, the optimization of this type of functional is difficult, so we are force to accept a suboptimal solution. There have been many approaches proposed for approxately solving this optimization problem. One commonly used method is known as iterative conditional modes (ICM). The ICM method iteratively minimizes the functional with respect to a single pixel in much the same manner as used by iterative coordinate descent (ICD). However, ICM typically optimizes over a discrete valued random field, rather than the continuous random field used in ICD.

Define the notation for the negative log likelihood as

$$l(y|k) = \frac{1}{2\sigma_k^2}(y-\mu_k)^2 + \frac{1}{2}\log\left(2\pi\sigma_k^2\right) \ .$$

Then the ICM algorithm is given by

Standard ICD Algorithm:

1. Set M = desired number of iterations
2. Select desired values of μ_k and σ_k for $k \in \{0, \cdots, M-1\}$
3. For each $i \in S$ /* Initialize with ML estimate */

$$x_s \leftarrow \arg \min_{k\in\{0,\cdots,M-1\}} l(y_s|k)$$

4. For $m = 0$ to $M-1$
   (a) For each $i \in S$

$$x_s \leftarrow \arg \min_{k\in\{0,\cdots,M-1\}} \left\{ l(y_s|k) + \beta \sum_{i\in\partial s} b_{s,i}\delta(k \neq x_i) \right\}$$

Section Problems:

1. Does the value of the cost function of (8) converge with repeated application of the ICM algorithm? If so prove your assertion? If not give a counter example.
2. Does the ICM algorithm converge to the MAP estimate for this problem?
3. Use the matlab script `mk_data.m` to generate an image with $\mu_0 = \mu_1 = 127$, $\sigma_0 = 10$, and $\sigma_1 = 20$.
4. Compute the ML segmentation (i.e. $\beta = 0$). Print out the result.
5. Compute the MAP segmentation using 20 iterations of ICD optimization with $\beta = 8$. Print out the optimization result for interations 2, 5, 10, and 20. Use a free boundary condition, so remove an cliques that connect across opposite edges of the image.
6. Plot the cost functional of (8) as a function of the iteration number for the ICM optimization.

4 Segmentation of Complex Images using Mixture Distributions

The method of Section 3 is somewhat limited because of the assumption that $p_{y_s|x_s}$ is a Gaussian distribution. More generally, any distribution can be assumed for $p_{y_s|x_s}$, so the method is really quite general. However, whatever distribution is choosen, it is often necessary to estimate the parameters of the distrubtion from real data.

A good choice for this distribution is a multivariate Gaussian mixture (MGM). The MGM distribution has the following advantages:

- The MGM distribution can be adjusted to fit any empirical distribution if the number of mixture components is choosen to be sufficiently large.
- The parameters of the MGM can be estimated using the EM algorithm.
- The MGM distribution is easy to compute.

In this section, you will use the SMAP software package, including the cluster program for estimation of MGM parameters, to segment a natural image. The SMAP algorithm is more sophisticated then standard MAP segmentation, but it is similar in concept. More importantly, this software package allows each class to be modeled by a MGM.

Section Problems:

1. Download and unpack the SMAP software from the labs webpage. Compile the ANSI C code for both the cluster program and the SMAP segmentation program.
2. Run the example in `SMAP.X.X.X/segmentation/example3`. This example contains a Matlab script that allows the user to select three regions of the included natural image. The script then writes the 3-color image data from these regions into files. The data in each file is used to train a MGM using the EM algorithm. The resulting parameters are then used to segment the image.
3. Print out the list of parameters for each of the three MGM's.
4. Print out the original image.
5. Print out the segmented image.