Advanced Machine Learning - Exercise 2

March 27, 2018

1. In this exercise you will implement the Loopy Belief Propagation (LBP), for a 2D image denoising task. In this task, we are given an image with a patch which contains noisy pixels, that need to be recovered. To do so, we define a Markov Network over a 2D grid graph (corresponding to the image pixels) as follows. The variables $y_1, \dots, y_n$ correspond to the observed pixel values (i.e., the noisy image). The variables $x_1, \dots, x_n$ correspond to the original (unobserved) pixel values (-1.0 to indicate black pixels and 1.0 to indicate white pixels).

   We consider the following Markov network over the x, y variables. Define the following pairwise interaction between x_i and y_i, which reflects the fact that x_i is likely to be close to y_i (this is sometimes known as the data term):

   $$\phi(x_i, y_i) = e^{\alpha x_i y_i}$$

   Also, consider the pairwise interaction between x_i and x_j where i, j are neighboring pixels in the 2D grid:

   $$\phi(x_i, x_j) = e^{\beta x_i x_j}$$

   This reflects the fact that nearby pixels in the original image are likely to be similar, and is known as the smoothness term. The overall Markov network is then:

   $$p(x_1, \dots, x_n, y_1, \dots, y_n) = \prod_{i=1\dots n} \phi(x_i, y_i) \prod_{ij \in E} \phi(x_i, x_j) \qquad (1)$$

   Your goal is to use BP to approximate the MAP problem:

   $$\max_{x_1, \dots, x_n} p(x_1, \dots, x_n | y_1, \dots, y_n) \qquad (2)$$

   This is the problem of interest since the y_i are observed and we would like to infer the x_i.

   (a) (no need to submit) Show $p(x_1, \dots, x_n | y_1, \dots, y_n)$ is a Markov network over variables $x_1, \dots, x_n$, with factors $\psi_{i,j}(x_i, x_j) = \phi_i(x_i, x_j)$ and $\psi_i(x_i) = \phi_i(x_i, y_i)$. The updates for LBP are (convince yourself that these work for the tree case, since they contain the singleton term that we didn't use in class):

$$m_{ij}(x_j) \leftarrow \max_{x_i} \psi_i(x_i)\psi_{ij}(x_i, x_j) \prod_{k\in N(i)\setminus j} m_{ki}(x_i) \tag{3}$$

And then at convergence the approximate MAP is given by:

$$x_i^* = \arg\max_{x_i} \psi_i(x_i) \prod_{k\in N(i)} m_{ki}(x_i) \tag{4}$$

(b) (no need to submit) Since you are updating message in a loopy fashion, they may grow to infinity or decay to zero. One way to avoid this is to normalize each message as follows:

$$m_{ij}(x_j) = \frac{m'_{ij}(x_j)}{\sum_{\bar{x}_j} m'_{ij}(\bar{x}_j)} \tag{5}$$

Convince yourself that this does not change the output of the algorithm (namely the approximated MAP assignemnt)

(c) Implement the max-product algorithm on the above network. There are many options for choosing the sequence of messages. Here's a simple one you can try: choose a certain order on the variables (say row by row in the part of the image you want to recover), and then for each node update all messages from it to its neighbors). You can also choose a convergence criterion, such as message not changing by much.

Submission Guidelines:

- The image to be completed in this exercise is downloadable from the course website under the name "digit.png"
- The output of your code should be a reasonable cleaned version of the noisy image.
- Add your IDs and the path to your code's folder in the following Google Form: `https://goo.gl/5UeWTx`
- Make sure your code's folder has the proper read permissions.
- Your code should run by the command

```
>>> python mrf_denoising.py input_fname output_fname
```

where `input_fname` is the name of the input image and `output_fname` is the output image. For example, invoking

```
>>> python mrf_denoising.py digit.png clean_digit.png
```

will clean the image in the file `digit.png` and save the file `clean_digit.png`.

- For numerical stability, consider working in log-space.
- Change the values of α and β to tune the results.
- You may use `mrf_denoising.py` available on the course Moodle.
- Your code should be readable and well-documented.

2. Show that the tree-width of a 2D grid graph of size (m, m) is **at most** m.

3. Consider the sum-product message update on a tree graph. But, consider the case where all messages are updated simultaneously. Namely:

$$m_{ij}^{t+1}(x_j) = \sum_{x_i} \phi_{ij}(x_i, x_j) \prod_{k \in N(i) \setminus j} m_{ki}^t(x_i) \tag{6}$$

   for all ij, x_j at each iteration. Show that this converges to the true marginals at iteration $t = n$.

4. In this question you will show that LBP generates pairwise and singleton marginals that satisfy the property from Question 7 of Exercise 1. The question refers to sum-proudct LBP. Namely, the algorithm which on trees calculates exact marginals. Consider the pairwise MRF:

$$p(x_1, \ldots, x_n) \propto \prod_{ij \in E} \phi_{ij}(x_i, x_j) \tag{7}$$

   where E are edges of a given graph. Define the following "pseudo-marginals" $\mu_{ij}(x_i, x_j)$ and $\mu_i(x_i)$:

$$\mu_i(x_i) \propto \prod_{k \in N(i)} m_{ki}(x_i) \tag{8}$$

$$\mu_{ij}(x_i, x_j) \propto \phi_{ij}(x_i, x_j) \prod_{k \in N(i) \setminus j} m_{ki}(x_i) \prod_{k \in N(j) \setminus j} m_{kj}(x_j) \tag{9}$$

   (a) Show that for tree graphs the above pairwise pseud-marginals $\mu_{ij}(x_i, x_j)$ are indeed the correct marginals of the MRF (you can use the fact that the messages are in that case the correct sum of assignments in the corresponding sub-tree).

   (b) Show that for any graph E (possibly non-tree) it holds that:

$$p(x_1, \ldots, x_n) \propto \prod_i \mu_i(x_i) \prod_{ij \in E} \frac{\mu_{ij}(x_i, x_j)}{\mu_i(x_i)\mu_j(x_j)} \tag{10}$$

   Note this is exactly the property shown in Q6 in Ex1, but here we are considering non-tree graphs as well. And μ_{ij} and μ_i are not the marginals of p.

   (c) Note that generally the pairwise and singleton marginals might not be consistent. Namely it isn't true that $\sum_{x_j} \mu_{ij}(x_i, x_j) = \mu_i(x_i)$. Show that at a fixed point of LBP this property does hold.