Loopy Belief Propagation on MRFs and Binary Image Denoising

David Doria

February 23, 2011

1 Introduction

The goal of this code is to provide a base implementation of Loopy Belief Propagation on MRFs in ITK. We use Binary Image Denoising as an example problem to demonstrate this code.

2 Loopy Belief Propagation

We have implemented the following message update rules:

- Sum-Product

$$m_{ij}(l) = \sum_{p \in L} \left[e^{-B(L(p),L(l))} e^{-U(L(p))} \prod_{k=N(i)\backslash j} m_{ki}(L(p)) \right] \tag{1}$$

- Max-Product

$$m_{ij}(l) = \max_{p \in L} \left[e^{-B(L(p),L(l))} e^{-U(L(p))} \prod_{k=N(i)\backslash j} m_{ki}(L(p)) \right] \tag{2}$$

- Min-Sum

$$m_{ij}(l) = \min_{p \in L} \left[B(L(p),L(l)) + U(L(p)) + \sum_{k=N(i)\backslash j} m_{ki}(L(p)) \right] \tag{3}$$

(Details can be found in my tutorial "Belief Propagation on MRFs").

3 Binary Denoising

The binary denoising problem takes a binary image as input and outputs a binary image that has had the "noise removed". To solve this problem using the BP framework presented, we must simply specify appropriate cost functions.

3.1 Label set

In this problem, the label set L is $0, 1$, and hence $|L| = 2$. That is, every node (pixel) can either take the value 0 or 1 (a binary image).

3.2 Unary Cost

In this problem, the input image is called the "observations". The cost of assigning a node a particular label is related to if the label agrees with the observations. One such function is:

$$Unary(node, label) = \begin{cases} .2 & \text{if } observation(node) = label \\ .8 & \text{otherwise} \end{cases} \tag{4}$$

This function encourages the resulting labeling to be the same as the observations. If a node's label is the same as its original observation, the cost is .2. If a node's label is different from its original observation, the cost is .8.

3.3 Binary Cost

In a denoising problem, typically "smoothness" is the goal. That is, a node is likely to take the same label as its neighbors. The binary cost should encourage this smoothness. One such function to achieve this is:

$$Binary(label1, label2) = \begin{cases} 0 & \text{if } label1 = label2 \\ 1 & \text{otherwise} \end{cases} \tag{5}$$

This means if a node's label is the same as its neighbor, the cost is 0. If neighboring labels are different, the cost is 1.

4 Code Structure

- LoopyBP class

  This is an abstract base class to provide the interface for a loopy belief propagation algorithm. The functions UnaryCost and BinaryCost must be implemented by a subclass. LoopyBP performs the actual message passing algorithm.

- BinaryDenoising class

  This is a subclass of LoopyBP. It contains the implementations of the cost functions specific to the binary denoising problem.

- Message class

  The Message class contains the label that the message is "talking" about and the value of the message.

- MessageVector class

  When two nodes talk to each other, the actually send a vector of messages, one for each label. This class encapsulates this vector of Messages.

- UpdateSchedule class

  This is an abstract base class for scheduling algorithms. It's core function is to provide the next MessageVector to process.

- RasterUpdateScheduler class

  This is a basic scheduling algorithm - it simply traverses the MRF passing messages along the way. The LoopyBP class contains this scheduler object.