CS 475 Machine Learning: Homework 8
Graphical Models
Due: Thursday December 6, 2012, 12:00pm
50 Points Total Version 1.0

1 Programming (25 points)

In this assignment you will implement the sum product algorithm for calculating marginal probabilities in a linear chain MRF (more specifically, a factor graph). You will implement sum product on top of potential functions that we give you (there is no learning in the homework, only exact inference).

1.1 The Factor Graph

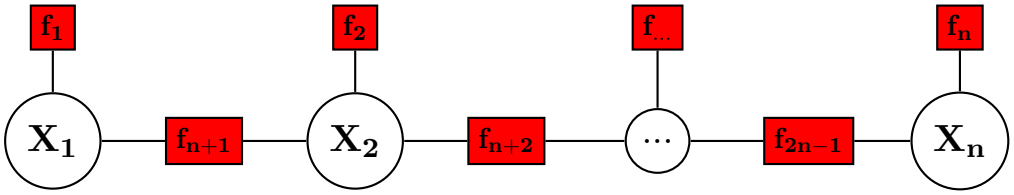

Figure 1: The factor graph that will be used in this assignment

Each of the variables in our chain (the x_i) are k-ary discrete variables.

1.2 Sum Product Algorithm

The following is adapted from section 8.4.4 of Bishop. For more details and diagrams, see the book (available online here).

Our goal is to find the marginal probability of a node in our factor graph. Due to the linear structure of our factor graph, the sum product algorithm lets us write this marginal probability as:

$$p(x) = \prod_{s \in N(x)} \left[\sum_{X_s} F_s(x, X_s) \right] \tag{1}$$

where:

$N(x)$ is the set of all factor node neighbors of x

$F_s(x, X_s)$ represents the product of all the factors "downstream" of f_s

Part of the above equation will be used many times, so for the sake of computation and intuition, we will define the sum term as a "message" μ:

$$\mu_{f_s \to x}(x) := \sum_{X_s} F_s(x, X_s) \tag{2}$$

If you look at the diagram in the book, you will see the recursive nature of $F_s(x, X_s)$, so we need a base case:

$$\mu_{f \to x}(x) := f(x) \text{ iff the only neighbor of f is x}$$

The sum product algorithm defines the message from a variable node to a factor node as the product of the messages it receives from its "downstream" factors:

$$\mu_{x \to f_s} := \prod_{l \in N(x) \setminus f_s} \mu_{f_l \to x}(x) \tag{3}$$

As before, there is a base case for this equation:

$$\mu_{x \to f}(x) := 1 \text{ iff the only neighbor of x is f}$$

And we're done: we can find marginal probabilities using equation 1. While seemingly simple, the details may be a bit opaque. To help clarify them, you will implement Sum Product on the chain factor graph above.

1.3 Implementation

For the factor graph in this assignment, there are unary (f_1 to f_n) and binary (f_{n+1} to f_{2n-1}) factors, and each factor f_i is associated with a potential function ψ_i. Each unary factor will have a potential function $\psi_i(a)$ which returns a real non-negative value corresponding to the potential when variable x_i takes value a. Each factor node between two variable nodes will have a potential function $\psi_i(a, b)$ which returns a value corresponding to the potential when variable x_{i-n} takes value a and node x_{i-n+1} takes value b. Recall that every x_i can take values from 1 to k.

We will provide you code that gives you values of $\psi_i(a)$ and $\psi_i(a, b)$. They will be in the class `cs475.sum_product.ChainMRFPotentials` and have the signatures:

```
public double potential(int i, int a)
public double potential(int i, int a, int b)
```

There are n nodes in this chain, so the value of `i` must be between 1 and n (inclusive) in the first method and between $n + 1$ and $2n - 1$ (inclusive) in the second method. Since every x_i can take values from 1 to k (inclusive), you must only call this function with values for `a` and `b` between 1 and k (inclusive). You will be able to get values for n and k by calling the following functions in `cs475.sum_product.ChainMRFPotentials`:

```
public int chainLength()  // returns n
public int numXValues()   // returns k
```

These values will be read into `cs475.sum_product.ChainMRFPotentials` from a text file that must be provided in the constructor:

```
public ChainMRFPotentials(String data_file)
```

We are providing you with a sample of this data file, `sample_mrf_potentials.txt`. The format is `"n k"` on the first line and either `"i a potential"` or `"i a b potential"` on subsequent lines. Feel free to try out new chains to get different probability distributions, just make sure it contains all the needed potential values.

Your code will work by calculating these messages given the value of the potential functions between the variable nodes in the chain. For details on how to do this, you can refer to your notes from class, see Bishop's examples in the book.

1.4 What You Need to Implement

We have provided you with a class, `ChainMRF`, with one method left blank that you will need to implement:

```
public class SumProduct {
    public static double[] marginalProbability(int x_i, ChainMRFPotentials p) {
        // TODO
    }
}
```

This should return a double array where the jth element is the probability that $x_i = j$. The length of this array should be $k+1$ and you should leave the 0 index as 0. These are probabilities so don't forget to normalize to sum to 1.

1.5 How We Will Run Your Code

We will run your code by providing you with a single command line argument which is the data file:

```
java cs475.sum_product.SumProductTester mrf_potentials.txt
```

Note that we will use new data files with different values of n and k, so make sure your code works for any reasonable input.

Your output should just be the results of the print statements in the code given. **Do not print anything else in the version you hand in.**

2 Analytical Questions (25 points)

1 (8 points) Suppose you are given a markov model with m variables $(x_1, x_2, \ldots x_m)$ where each variable is assumed to have k states $(s_1, s_2 \ldots, s_k)$. You are given a $k \times k$ transition matrix T with $T_{ij} = P(x_{t+1} = s_i | x_t = s_j)$ for $t = 1, ..., m-1$. If we observe $x_1 = s_1$, please derive the probability distribution of $P(x_m)$?

2 (8 points) Consider the following Gaussian Hidden Markov Models, we have k hidden states and each state has a Gaussian distribution as the output. Show the marginal distribution of the output is the Gaussian Mixture. Given some data generated by the HMM above, what is the difference between fitting Gaussian Mixture Models or Gaussian HMM?

3 (9 points) You are working on a new machine learning system. After spending a lot of time collecting data, writing features and fitting a model to the data, you are disappointed in the terrible performance. You are sure there isn't a bug, but don't know what to try. You think the problem may be one of a bias and variance tradeoff.

1. What experiment can you run to verify if this is the case?
2. How will you determine if the problem is bias or variance?
3. Name one way to improve your system is the problem is variance.
4. Name one way to improve your system is the problem is bias.

3 What to Submit

In each assignment you will submit two things.

1. **Code:** Your code as a zip file named `library.zip`. **You must submit source code (.java files)**. We will run your code using the exact command lines described above, so make sure it works ahead of time. Remember to submit all of the source code, including what we have provided to you.
2. **Writeup:** Your writeup as a **PDF file** (compiled from latex) containing answers to the analytical questions asked in the assignment. Make sure to include your name in the writeup PDF and use the provided latex template for your answers.

To submit your assignment, visit the "Homework" section of the website (http://www.cs475.org/.)

4 Questions?

Remember to submit questions about the assignment to the appropriate group on the class discussion board: http://bb.cs475.org.