Probabilistic Graphical Models 10-708
Homework 2: Due February 24, 2014 at 4 pm

Directions. This homework assignment covers the material presented in Lectures 4-8. You must complete all four problems to obtain full credit. To submit your assignment, please upload a pdf file containing your writeup and a zip file containing your code to Canvas by 4 pm on Monday, February 24th. We highly encourage that you type your homework using the LaTeX template provided on the course website, but you may also write it by hand and then scan it.

1 Fundamentals [25 points]

Consider the Bayesian network shown in Figure 1, which illustrates the influence of genetics and environment on an individual's overall health. Specifically, this model encodes a set of independence relationships among the following variables: Genetics (G), Environment (E), Health (H), Disease1 (D_1), Disease2 (D_2), Disease3 (D_3), Symptom1 (S_1), Symptom2 (S_2). Use this model to answer the questions in Parts 1 and 2 below.

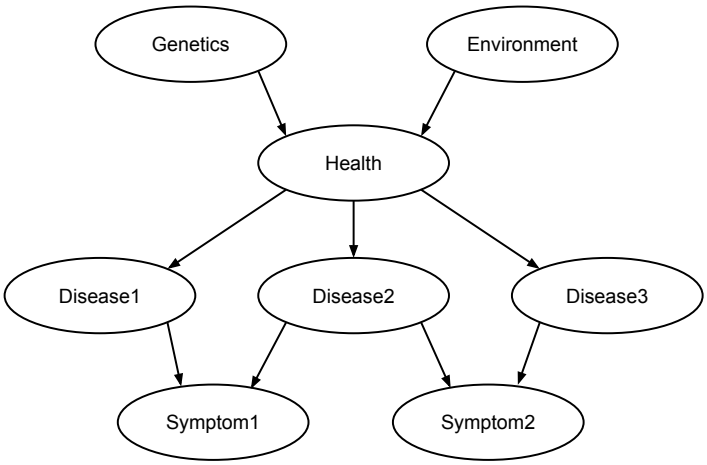

Figure 1: A Bayesian network that represents a joint distribution over the variables Genetics, Environment, Health, Disease1, Disease2, Disease3, Symptom1, and Symptom2.

Part 1: Variable Elimination in Bayesian Networks [13 points]

1. Write down an expression for $P(S_1)$ in terms of the un-factorized joint distribution.
2. Starting with the expression you gave in the last question, write down an expression for $P(S_1)$ in terms of the factorized joint distribution and simplify it by pushing summations into the product of terms whenever possible.

3. What variable elimination ordering does the resulting expression correspond to?
4. Walk through an execution of the variable elimination algorithm for evaluating $P(S_1)$ using the ordering you specified in the last question. Start by writing out the initial set of factors and drawing the moralized graph. Then go through each step of the elimination algorithm and (1) specify the intermediate factors that are generated by the product and sum operations, in that order, (2) write down the new set of factors that remain after the entire step has been executed, and (3) draw the new graph induced by eliminating the variable. For clarity, please use $\phi(\cdot)$ to denote initial factors, $\psi(\cdot)$ to denote intermediate factors generated by a product operation, and $\tau(\cdot)$ to denote intermediate factors generated by a sum operation.
5. Assuming each variable can take m values, what is the size of the largest intermediate factor generated during this execution of the variable elimination algorithm, i.e. how many entries does this factor have?
6. What is the computational complexity (in terms of the number of possible assignments, m, and the number of variables in the network, n) of this execution of variable elimination?
7. Can you do better than this? Specifically, is there another elimination ordering that yields a lower-complexity algorithm? If so, give the best such ordering and evaluate its computational complexity.

Part 2: Variable Elimination with Clique Trees [6 points]

A *clique tree*, also known as a *junction tree*, is a very useful data structure that serves as a "graphical flowchart" of the factor manipulation process for the variable elimination algorithm (and, as we will later see, for the message passing algorithm). We define a clique tree $\mathcal{T}$ for a set of factors Φ over variables $\mathcal{X}$ as an undirected graph whose notes i are each associated with a cluster $\boldsymbol{C}_i \subseteq \mathcal{X}$ and whose edges ij are each associated with a sepset $\boldsymbol{S}_{ij} = \boldsymbol{C}_i \cap \boldsymbol{C}_j$. Clique trees must satisfy two important properties: *family preservation* and *running intersection*. Before doing this question, please carefully read Section 10.1 of Koller and Friedman so that you obtain a good understanding of what a clique tree is and how it plays a role in variable elimination.

1. Draw the best clique tree for the Bayesian network of Figure 1, i.e. the one induced by the variable elimination ordering with the lowest computational complexity. What is the size of the largest intermediate factor generated in the corresponding execution of variable elimination? What is the computational complexity of this variable elimination run?
2. Draw the worst clique tree for the Bayesian network of Figure 1, i.e. the one induced by the variable elimination ordering with the highest computational complexity. What is the size of the largest intermediate factor generated in the corresponding execution of variable elimination? What is the computational complexity of this variable elimination run?

Part 3: From Variable Elimination to Message Passing [6 points]

Before doing this question, please review the lecture notes and/or read Section 10.2 of Koller and Friedman, paying careful attention to Section 10.2.2, so that you understand how to perform sum-product message passing using clique trees. Note that the sum-product algorithm described in the textbook is equivalent to the junction tree algorithm with Shafer-Shenoy updates introduced in class. From now on, we will simply refer to this as sum-product message passing.

Suppose you design a clique tree $\mathcal{T}$ for the Bayesian network of Figure 1 and run sum-product message passing to calibrate $\mathcal{T}$. Denote the calibrated beliefs over each cluster as $\beta_i(\boldsymbol{C}_i)$ and the calibrated beliefs over each sepset as $\mu_{ij}(\boldsymbol{S}_{ij})$.

1. What exactly is a "message" in the sum-product message passing algorithm?

2. Assume there are N cliques in $\mathcal{T}$. How many messages must be sent in order to fully calibrate the clique tree?
3. Give an expression for the full joint distribution in terms of the cluster and/or sepset beliefs.
4. Given an expression for $P(S_1)$ in terms of the cluster and/or sepset beliefs.

2 Sum-Product Message Passing on Clique Trees [25 points]

Please note that the questions in this section are adapted from Exercises 10.5 and 10.12 in the Koller and Friedman textbook.

Part 1: Conditional Probabilities from Clique Trees [10 points]

Let $\mathcal{T}$ be a clique tree defined by a set of initial factors Φ, and let $\boldsymbol{C}_r$ be a root of $\mathcal{T}$. Assume $\boldsymbol{C}_j$ is an arbitrary clique in the tree and $\boldsymbol{C}_i$ is its upward neighbor (closer to the root). Define β_j as the potential at clique $\boldsymbol{C}_j$ after the upward pass of the sum-product message passing algorithm has been executed, in which messages are passed up from the leaves to the root. Show that β_j correctly encodes the original unnormalized conditional measure $\tilde{P}_\Phi(\boldsymbol{C}_j - \boldsymbol{S}_{ij} \mid \boldsymbol{S}_{ij})$. In other words, letting $\boldsymbol{X} = \boldsymbol{C}_j - \boldsymbol{S}_{ij}$ and $\boldsymbol{S} = \boldsymbol{S}_{ij}$ for simplicity, show that the following holds:

$$\beta_j(\boldsymbol{X}|\boldsymbol{S}) = \frac{\beta_j(\boldsymbol{X}, \boldsymbol{S})}{\beta_j(\boldsymbol{S})} = \ldots = \frac{\tilde{P}_\Phi(\boldsymbol{X}, \boldsymbol{S})}{\tilde{P}_\Phi(\boldsymbol{S})} = \tilde{P}_\Phi(\boldsymbol{X}|\boldsymbol{S})$$

Hint: Think about how to write $\beta_j(\boldsymbol{X}, \boldsymbol{S})$ and $\beta_j(\boldsymbol{S})$ in terms of the initial factors in Φ.

Part 2: Message Passing with Evidence [15 points]

Let $\mathcal{T}$ be a clique tree over a set of discrete-valued variables $\mathcal{X}$ that is defined by a set of initial factors Φ. Let $\boldsymbol{X} = \overline{\boldsymbol{x}}$ be some observed assignment to a subset of the variables. Consider a setting in which we are unsure about a particular observation $X_i = \overline{x}_i$ for some variable in the observed set $X_i \in \boldsymbol{X}$. Suppose we want to compute the value of the unnormalized marginal probability of the evidence for all possible values of X_i. More precisely, let $\boldsymbol{X}_{-i} = \boldsymbol{X} - \{X_i\}$ and let $\overline{\boldsymbol{x}}_{-i}$ be the observed assignments to $\boldsymbol{X}_{-i}$. We want to compute $\tilde{P}_\Phi(X_i = x_i, \boldsymbol{X}_{-i} = \overline{\boldsymbol{x}}_{-i})$ for every variable $X_i \in \boldsymbol{X}$ and every possible assignment x_i. The goal of this problem will be to design a variant of the sum-product message passing algorithm that can perform this task without requiring more messages than the standard two-pass calibration.

Hint: In the standard sum-product message passing algorithm, we deal with evidence by reducing our original factors over the observations before passing any messages. We do this by incorporating a set of evidence potentials of the form $\delta(x_i, \overline{x}_i)$ into our original set of factors Φ. In this setting, we cannot do this because we need to come up with a scheme that allows us to compute the marginals over all possible assignments to each observed variable. To achieve this, consider reducing the factors during (rather than before) the message passing process.

1. Redefine the messages $\delta_{i \to j}$ such that after performing sum-product message passing, the beliefs at each clique in the calibrated tree will be conditioned on every variable in the evidence set $\boldsymbol{X}$ except the ones that are also in the scope of that clique.
2. Show how the marginal distributions $\tilde{P}_\Phi(X_i = x_i, \boldsymbol{X}_{-i} = \overline{\boldsymbol{x}}_{-i})$ can be computed from the calibrated clique tree beliefs after running sum-product message passing using the messages defined in the previous step.
3. Justify the correctness of your algorithm.

3 Maximum Likelihood vs. Bayesian Parameter Estimation [25 points]

Consider the following generative classification model with K classes defined by prior class probabilities $p(C_k) = \pi_k$ and class-conditional densities $p(x|C_k)$ where x is the observed feature vector. Suppose we have a training data set $D = \{x_n, y_n\}_{n=1}^N$, where each y_n is a binary target vector of length K that uses the 1-of-K coding scheme (a vector with one 1 and the rest 0s).

Assume that the data points are drawn independently from this model, and the class-conditional densities are given by Gaussian distributions (all with the same covariance matrix), so that

$$p(x|C_k) = N(x|\mu_k, \Sigma)$$

1. [5 pts] Show that the maximum-likelihood estimation for the prior probabilities is given by

$$\widehat{\pi}_k = \frac{N_k}{N} \quad \forall k$$

   where N_k is the number of data points assigned to class C_k.

2. [6 pts] Assuming that Σ is known, show that the maximum likelihood estimate for the mean of the Gaussian distribution for class C_k is given by

$$\widehat{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} y_{nk} x_n \quad \forall k$$

3. [7 pts] Assuming that Σ is known, and assuming that the mean μ has a standard normal prior, i.e., $\mu_k \sim N(\mu_k|0, I) \quad \forall k$, show that the posterior distribution $p(\mu_k|D)$ is also a normal distribution, and show that the posterior mean is:

$$\widehat{\mu}_k = (\Sigma + N_k I)^{-1} \sum_{n=1}^{N} y_{nk} x_n \quad \forall k$$

4. [7 pts] Under the same prior as the previous question, show that the maximum a posteriori (MAP) estimate of μ is:

$$\widehat{\mu}_k = (\Sigma + N_k I)^{-1} \sum_{n=1}^{N} y_{nk} x_n \quad \forall k$$

   Note: the MAP estimate is defined as follows:

$$\widehat{\mu}^{\text{MAP}} = \arg\max_{\mu} \log p(\mu|D)$$

4 An HMM for the Stock Market [25 points]

The Hidden Markov Model (HMM) has been a workhorse in the financial industry for decades, and is still a popular model today. In this question, we will attempt to use an HMM to infer hidden market states based on observations of day-to-day changes in the S&P 500, which is a major stock index computed as a weighted average of the stock prices of 500 leading companies in the US.

HMMs are state space models composed of a sequence of latent variables (states) that each generate an observed variable (emissions). The latent variables form a Markov chain, which means that the state at each time point, Z_t, is conditionally independent of all prior states given the state at the previous time point, Z_{t-1}. The HMM graphical model is shown in Figure 2.

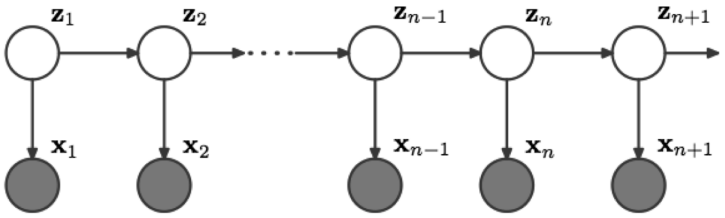

Figure 2: The graphical model for an HMM. Each emission X_t is conditioned on the corresponding latent state Z_t. The latent variables form a Markov chain.

The joint probability distribution is:

$$P(\boldsymbol{X}, \boldsymbol{Z}; \theta) = P(Z_1; \pi) \left[\prod_{t=2}^{T} p(Z_t | Z_{t-1}; A) \right] \left[\prod_{t=1}^{N} P(X_t | Z_t; E) \right]$$

where $\boldsymbol{X} = \{X_1, ..., X_T\}$ is the sequence of emissions, $\boldsymbol{Z} = \{Z_1, ..., Z_T\}$ is the sequence of states, and $\theta = \{\pi, A, E\}$ is the set of model parameters. Here π is the categorical parameter for $P(Z_1)$, A is the state transition matrix, and E is the matrix of emission probabilities. Whenever appropriate, we will omit π, A, E for brevity since we assume they are fixed.

Part 1: Derive the Forward-Backward Algorithm [8 points]

In this part, we are going to derive the forward-backward algorithm for an HMM using a factor graph. One way to do this is to put a factor on each edge of the graphical model, in which case we'd have factors of the form $\phi_t(Z_t, Z_{t-1})$ and $\psi_t(Z_t, X_t)$. But since $\boldsymbol{X}$ is observed, we can absorb the ψ factors into ϕ and write

$$\phi_t(Z_t, Z_{t-1}) := \phi_t(Z_t, Z_{t-1}, X_t) = P(Z_t | Z_{t-1}) P(X_t | Z_t)$$

without increasing the complexity (i.e. the tree-width) of the factor graph. We also define

$$\phi_1(Z_1) := \phi_1(Z_1, X_1) = P(Z_1) P(X_1 | Z_1)$$

1. Draw the factor graph corresponding to these definitions of the factors.

2. Based on the standard message passing algorithm on a factor graph, we define the following two messages:

$$m_{Z_{t-1} \to \phi_t}(Z_{t-1}) = m_{\phi_{t-1} \to Z_{t-1}}(Z_{t-1})$$
$$m_{\phi_t \to Z_t}(Z_t) = \sum_{Z_{t-1}} \phi_t(Z_{t-1}, Z_t)\, m_{Z_{t-1} \to \phi_t}(Z_{t-1})$$

   It can be shown that $\alpha(Z_t) := P(X_1, ..., X_t, Z_t) = m_{\phi_t \to Z_t}(Z_t)$ (to verify this, you can try it yourself). Now consider the reverse direction. Write down messages $m_{Z_{t-1} \to \phi_{t-1}}(Z_{t-1})$ and $m_{\phi_t \to Z_{t-1}}(Z_{t-1})$, and then show that

$$m_{\phi_t \to \phi_{t-1}}(Z_{t-1}) = \sum_{Z_t} \phi_t(Z_t, Z_{t-1})\, m_{\phi_{t+1} \to \phi_t}(Z_t) \tag{1}$$

3. Given $\beta(Z_t) := m_{\phi_{t+1} \to \phi_t}(Z_t)$, we have the following recursive property

$$\beta(Z_t) = \sum_{Z_{t+1}} \beta(Z_{t+1}) P(X_{t+1} | Z_{t+1}) P(Z_{t+1} | Z_t)$$

   Show that this property holds when we define $\beta(Z_t) = P(X_{t+1}, ..., X_T | Z_t)$.

output symbol	description	% daily change
1	large drop	$< -2\%$
2	small drop	-2% to -1%
3	no change	-1% to 1%
4	small rise	1% to 2%
5	large rise	$> 2\%$

Table 1: The output symbols for the HMM model. The change in the S&P 500 index is measured as (closing index today − closing index yesterday) / (closing index yesterday)

Part 2: Perform HMM Inference for Real [17 points]

For this problem, we will use a simple HMM with $Q = 3$ hidden states and a sequence length of $T = 100$ trading days. The choice of a small Q reduces the model complexity, which helps in performing inference over the hidden states. You can roughly think of these hidden states as "bull market", "bear market", and "stable market". The emission model is a multinomial distribution with $O = 5$ observed symbols. See Table 1 for details on the output states. The parameters of this HMM have already been estimated for you using the EM algorithm on a much longer portion of data (from 01/01/2010 to 07/31/2013). You will use this fully parameterized model to carry out inference over a sequence of 100 trading days (starting on 08/01/2013), and then you will perform forward prediction of the output values over the next 28 days. All of the data used in this problem was downloaded from Yahoo! finanace.

In "hmm_params.mat" you will find the following parameters:

- *transition*: the transition probability matrix, where $transition(i, j) = P(Z_{t+1} = j | Z_t = i)$
- *prior*: the prior distribution over Z_1, where $prior(i) = P(Z_1 = i)$
- *emission*: the emission probability matrix, where $emission(i, j) = P(X_t = j | Z_t = i)$
- *price_change*: the observations, where $price\_change(i) = \frac{price(i) - price(i-1)}{price(i-1)}$

Here are the details of your tasks:

1. [5 pts] Implement the Forward-Backward algorithm and run it using the observations for time points $t = 1, ..., 100$. Report the inferred distributions over the hidden states at $t = 1, ..., 100$ by plotting the probabilities $P(Z_t = i | X_1, ..., X_{100})$ for $i = 1, 2, 3$ over $t = 1, ..., 100$. Make sure you label the 3 time series (one for each hidden state) in your plot.
2. [5 pts] Implement the Viterbi algorithm and find the most likely sequence of hidden states $Z_1, ..., Z_{100}$ for the same time period. Report the most likely hidden states over $t = 1, ..., 100$ by plotting these values as a time series.
3. [5 pts] Now, let's see how well this model performs. Predict the output symbols for time points $t = 101, ..., 128$ by carrying out the following steps for each time point t:
   (a) Run the forward algorithm to estimate $P(Z_{t-1} | X_{t-101}, ..., X_{t-1})$. Note that $X_{t-101}, ..., X_{t-1}$ are the ground truth observations. (Alternatively, you can compute $P(Z_t | X_1, ..., X_{t-1})$, but clarify if you take this approach.)
   (b) Compute $P(Z_t)$ from $P(Z_{t-1})$ using the transition matrix. Generate the output value X_t by sampling a state z_t from $P(Z_t)$ and then from $P(X_t | Z_t = z_t)$.

   Compare your predictions with the ground truth observations at time points $t = 101, ..., 128$. What's the percentage of these values that your model predicts correctly? Report the average and variance over 100 runs.

4. [2 pts] One of your TAs is considering investing in the S&P 500. Would you recommend that he/she use this model to guide their investment decisions? Why or why not? Answer this question carefully as your TA's money is on the line!