Homework Assignment 2

Due: February 9, 2017 at 11am

Total points: 100

Deliverables: `hw2.pdf` containing typeset solutions to Problems 1-4.
Source code containing your implementation for Problem 1.
`README` file explaining how to compile and run the source code on Linux or Windows.

Guidelines: All files must be submitted by Dropbox. You can brainstorm with others, but please solve the problems and write up the answers and code by yourself. You may use textbooks (Koller & Friedman, Russell & Norvig, etc.), lecture notes, and standard programming references (e.g., online Java API documentation). Please do NOT use any other resources or references (e.g., example code, online problem solutions, etc.) without asking.

1 Programming: Learning/Inference in Hidden Markov Model [45 points]

The task we will consider in this assignment is optical character recognition (OCR). The dataset we provide consists of a sequence of words, one character per row. The very first character of each word was capitalized in the original data and has been omitted for simplicity. The format of the data is described in `generate_hmm_plots.m`, so please look though that file before continuing. This file provides sample code structure to generate plots.

In this problem you will implement maximum likelihood estimation and the forward-backward algorithm for Hidden Markov Models (HMMs). Let X_t denote the t-th letter in a word and O_t^k the value of the k-th pixel for the t-th character. The result should be a stationary model (one that does not depend on t), i.e., you should have a single distribution $P(X_1)$, a single CPT $P(X_t, X_{t-1})$ and 64 CPTs $P(O_t^k|X_t)$ (of for each pixel k).

1. **Parameter Estimation (MLE/MAP) in HMMs**

   For this first part you will set the parameters of the HMM using maximum likelihood and maximum a posteriori estimation using several values of pseudo-count/hyperparameter α.

   Your task is to fill in the missing code in the file `hmm_learn.m`. Note that `hmm_learn.m` goes over the specifics of what parameters you need to learn. Because we will be comparing HMM to Naïve Bayes, `hmm_learn.m` should also fit a probability model $P(X_t)$ which serves as the class prior for Naïve Bayes.

   To help debug your code and generate results, `generate_hmm_plots.m` will plot the transition model that you learn, and the observation model for the letter 'a'. You should see that the transition model "makes sense", e.g. $P(X_t = u|X_{t-1} = q)$ should be high, and that the observation model looks like a blurry version of the desired letter.

2. **The Forward-Backward Algorithm**

   In this part you will implement the Forward-Backward algorithm for HMMs and compare its performance to a Naïve Bayes approach which classifies each character independently of all others. You have two programming tasks for this part.

   (a) `hmm_fb.m` - In this file, you will implement the Forward-Backward algorithm to compute marginal probabilities $P(X_t|O_1, \ldots, O_T)$. The input to this function is the trained model from `hmm_learn.m` and the pixel data corresponding to a *single word* (not the entire test set). See the file `generate_hmm_plots.m` to see how `hmm_fb.m` is used.

(b) `generate_hmm_plots.m` - Run this file to train the HMM model and evaluate it on the test data. Naïve Bayes will serve is a baseline, but one critical line of code is missing in this file. Remember that Naïve Bayes computes the probability

$$P(X_t|O_t) \propto P(O_t|X_t)P(X_t)$$

Also, remember that $P(X_t)$ was computed in `hmm_learn.m`, and $P(O_t|X_t)$ was computed in `hmm_fb.m`. You need to fill in this line before the code will run.

Writeup: Try several values of the smoothing/pseudo-counts: $\alpha = 0, 1, 2, 4, 8$ and include the plots for the resulting observation model for 'a' and the transition model in the writeup. **Describe** in 1-2 sentences the effect of smoothing. Include a plot of accuracy on the test set vs. smoothing parameter α for HMM and NB. Next, **discuss** (3-4 sentences) how the two algorithms differ in performance, what their performance and errors are and how/why they differ.

2 Kalman Filters [15 points]

Often, we wish to monitor a continuous-state system whose behavior switches unpredictably among a set of k distinct "modes". For examples, an aircraft trying to evade a missile can execute a series of distinct maneuvers that the missile may attempt to track. A Bayesian network representation of such a **switching Kalman filter** model is shown in Figure 1

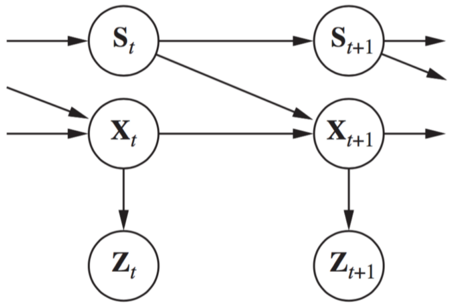

Figure 1: A Bayesian network representation of a switching Kalman filter. The switching variable S_t is a discrete state variable whose value determines the transition model of the continuous state variable $\boldsymbol{X}_t$. For any discrete state i, the transition model $P(\boldsymbol{X}_{t+1}|\boldsymbol{X}_t, S_t = i)$ is a linear Gaussian model, just as in a regular Kalman filter. The transition model for the discrete state $P(S_{t+1}|S_t)$, can be thought of as a matrix, as in a hidden Markov model.

1. Suppose that the discrete state S_t has k possible values and that the prior continuous state estimate $P(\boldsymbol{X}_0)$ is a multivariate Gaussian distribution. Show that the prediction $P(\boldsymbol{X}_1)$ is a **mixture of Gaussians** - that is, a weighted sum of Gaussians such that the weights sum to 1.
2. Show that if the current continuous state estimate $P(\boldsymbol{X}_t|e_{1:t})$ is a mixture of m Gaussians, then in the general case the updated state estimate $P(\boldsymbol{X}_{t+1}|e_{1:t+1})$ will be a mixture of km Gaussians.
3. What aspect of the temporal process do the weights in the Gaussian mixture represent?

The results in (a) and (b) show that the representation of the posterior grows without limit even for switching Kalman filters, which are among the simplest hybrid dynamic models.

3 Graph and Independence Relations [20 points]

For $i = 1, 2, 3$ let X_i be an indicator variable for the event that a coin toss comes up heads (which occurs with probability q). supposing that the X_i are independent, define $Z_4 = X_1 \oplus X_2$ and $Z_5 = X_2 \oplus X_3$ where $\oplus$ denotes addition in modulo two arithmetic.

1. Compute the conditional distribution of (X_2, X_3) given $Z_5 = 0$; then, compute the conditional distribution of (X_2, X_3) given $Z_5 = 1$.
2. Draw a directed graphical model (the graph and conditional probability tables) for these five random variables. What independence relations does the graph imply?
3. Draw an undirected graphical model (the graph and compatibility functions) for these five random variables. What independence relations does the graph imply?
4. Under what conditions on q do we have $Z_5 \perp X_3$ and $Z_4 \perp X_1$? Are either of these marginal independence assertions implied by the graphs in (b) or (c)?

4 BN2O Networks [20 points]

Consider the network shown in Figure 2

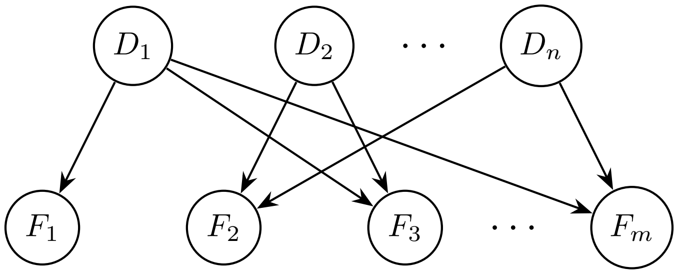

Figure 2

We assueme that all variable are binary, and that the F_i variables in the second layer all have *noisy or* CPDs. Specifically, the CPD of F_i is given by

$$P(f_i^0 | \boldsymbol{Pa}_{F_i}) = (1 - \lambda_{i,0}) \prod_{D_j \in \boldsymbol{Pa}_{F_i}} (1 - \lambda_{i,j})^{d_j}$$

where the $\lambda_{i,j}$ is the noise parameter associated with parent D_j of variable F_i. (This network architecture,, called a *BN2O network*, is characteristic of several medical diagnosis applications, where the D_i variable represent diseases (e.g., flu, pneumonia), and the F_i variables represent medical findings (e.g., coughing, sneezing). For more information about BN2O networks see box 5.C of Koller & Friedman).

Our general task is medical diagnosis: We obtain evidence concerning some of the findings, and we are interested in the resulting posterior probability over some subset of diseases. Since we are only interested in computing the probability of a particular subset of diseases, we wish (for reasons of computational efficiency) to remove from the network those disease variables that are not of interest at the moment.

1. Begin by considering a particular variable F_i, and assume (without loss of generality) that the parents of F_i are $D_1, \ldots, D_k$, and that we wish to maintain only the parents $D_1, \ldots, D_l$ for $l < k$. Show how we can construct a new noisy-or CPD for F_i that preserves the correct joint distribution over $D_1, \ldots, D_l, F_i$.

2. We now remove some fixed set of disease variables $\mathcal{D}$ from the network, executing this pruning procedure for all the finding variables F_i, removing all parents $D_j \in \mathcal{D}$. Is this transformation exact? In other words, if we compute the posterior probability over some variable $D_i \notin \mathcal{D}$, will we get the correct posterior probability (relative to our original model)? Justify your answer.