CPSC 422,
Homework 1

General Instructions:

- Working in Teams: You may work with at most one other person. That person must also be enrolled in CPSC 422 this term. If you are working with a partner, the two of you must submit only one assignment, listing both of your names. Keep in mind that when you work in pairs, each of you must understand the work that you submit.
- Note: You are permitted to derive your solutions by implementing the respective algorithms, even when it is not explicitly required in the question. You are NOT permitted to take existing implementations of the algorithms and use those for your solutions. Such existing algorithms include those available from friends, CD, or off the web. You are also NOT permitted to provide your implementation to anyone else outside your team.
- Important: It is best if you type your assignment. Handwritten work will be marked only if clearly legible, and will still have to be submitted via handin.
- Make sure to include your name (or the names of the two team members if you are working with someone else) and the assignment number, at the top of the assignment. Ensure that each question and sub-question is appropriately labeled and clearly identifiable.
- The assignment is due Fri., Feb. 5, 2pm.

1 [20 Points] Inferring Non-Knowledge from Multi-Digit Errors

Suppose we want to diagnose the errors school students make when adding multi-digit binary numbers. We will only consider adding two two-digit numbers to form a three digit number. That is, problems of the form:

$$\begin{array}{cccc} & A_1 & A_0 \\ + & B_1 & B_0 \\ \hline C_2 & C_1 & C_0 \end{array}$$

Here A, B, and C are all binary digits.

1. Suppose we want to model whether the student knows single-digit binary addition and whether they know how to carry. If they know how, they usually get the right answer, but sometimes make mistakes. If they don't know how to do the mechanism, they just pick the result of the step at random. What variables are needed to model binary addition and the errors students could make? You need to specify, in words, what each of the variables represents. Give a DAG that specifies the dependence of these variables.

2. Assume that 80% of the students know binary addition and 50% know how to carry. Further assume that students make mistakes with a probability of 5% (in either addition or carry) even if they know the underlying mechanism. Give the conditional probability tables for your network.

3. Implement this, using the AISpace Bayes net at http://www.aispace.org/bayes/index.shtml. You will use your network to make a number of inferences. You do not need to submit any code, but explain which variables you instantiated for each question and how, and which variable you queried.

   (a) Assume a student is solving the addition task 11+10 (i.e. the problem from above with $A_1 = 1$, $A_0 = 1$, $B_1 = 1$, and $B_0 = 0$), and their result for C_2 is 1. What is the probability that they know how to carry? What is the probability that they know how to add?

   (b) Assume a student is solving the addition task 11+10, and their result contains $C_2 = 1$ and $C_1 = 1$. What is the probability that they know how to carry? What is the probability that they know how to add?

   (c) What is the probability that a student who knows addition and how to carry correctly adds 11+10? (Hint: you can augment your network by one or more variables to be able to query this probability at a single node)

   (d) What is the probability that a student who knows addition but doesn't know how to carry correctly adds 11+10?

2 [30 points] Approximate Inference by Sampling

In this exercise, you will employ likelihood weighting and rejection sampling for approximate inference in a network that can also be solved by exact inference techniques. The exact same

approximation techniques also apply for more complicated networks that cannot be solved by exact inference.

As an example network, we will use the "Car Starting Problem", one of the sample networks in the AIspace Belief Net applet at `http://aispace.org/bayes/`. (Load the "Car Starting Problem" via `File` → `Load Sample Problem`.) We will perform exact inference in the applet and approximate inference using rejection sampling and likelihood weighting with the samples provided. You can use any programming language and/or plotting tool of your choice for this analysis (note that there are many data points, so MS Excel might not scale handle them well). To get the files needed for this assignment, download and unzip file `hmw1.zip` from the course webpage. Rejection sampling and likelihood weighting are anytime algorithms: they already yield results with a small number of samples, but results continuously improve as the number of samples grows. We will study this characteristic by plotting $P(\text{Spark Adequate}|\text{evidence})$ for different number of samples.

1. Without any evidence, compute the exact probability $P(\text{Spark Adequate} = t)$ using the applet.

2. Without any evidence, compute the approximate probability of $P(\text{Spark Adequate} = t)$ using forward sampling and the samples provided in file `rs_1.txt`.

3. Now we observe that the car cranks but does not start. Use the samples provided in file `rs_1.txt` to compute $P(\text{Spark Adequate} = t|\text{Car Cranks} = t, \text{Car Starts} = f)$, the probability that the Spark Adequate is true given our observations. Generate a graph whose x-axis is the number of used samples N, and whose y-axis is $P(\text{Spark Adequate} = t|\text{Car Cranks} = t, \text{Car Starts} = f)$ as computed based on the first N samples. Start the graph with the first accepted sample in order to avoid divisions by zero; make the x-axis logarithmic so you can see the algorithm's behavior for small sample sizes. What is the algorithm's approximation of $P(\text{Spark Adequate} = t|\text{Car Cranks} = t, \text{Car Starts} = f)$ using 100000 samples?

4. Let p denote the true probability $P(\text{Spark Adequate}|\text{Car Cranks=t,Car Starts=f})$ and let $\hat{p}$ denote the approximation of p using rejection sampling based on n accepted samples. Theoretically, using Hoeffding's inequality, derive the tightest bound ϵ such that $P(|\hat{p}-p| > \epsilon) < 0.05$. How many accepted samples n are there at N=100000? What is the value of ϵ for that n? Augment your plot from the question above with upper and lower confidence bounds $\hat{p} + \epsilon$ and $\hat{p} - \epsilon$, where both $\hat{p}$ and ϵ are computed from the accepted samples up to that point. (Remember that some samples are rejected, so $N \neq n$.)

5. File `lw_1.txt` contains the result of running likelihood weighting for 100000 samples; each line contains twenty numbers: the first nineteen numbers are the samples and the last number is the weight. What is the algorithm's approximation of $P(\text{Spark Adequate} = t|\text{Car Cranks} = t, \text{Car Starts} = f)$ using 100000 samples? Generate a graph equivalent to your graph in the question above; which algorithm seems to converge faster?

6. Now we make a new observation. We check the battery and it turns out that it is very old; thus, we fix the value of Battery Age=very old and generate new samples for likelihood weighting. The results of likelihood weighting for this modified network are given in `lw_2.txt`. Redo the analysis from the three questions above with this new data; discuss how the results change, and why.

7. Why do all samples in `lw_2.txt` either have weight 0.0 or 0.196 ?

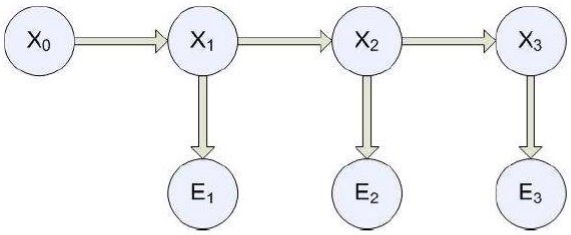

Figure 1: Hidden Markov Model with 4 time steps.

3 [50 points] Smoothing and Most Probable Path in HMMs

Consider the Hidden Markov Model in Figure 1, with transition matrix

X_t	$P(X_{t+1} = \text{true} \mid X_t)$
true	0.4
false	0.7

and observation matrix

X_t	$P(E_t = \text{true} \mid X_t)$
true	0.9
false	0.3

Assume $P(X_0 = \text{true}) = 0.5$ and just 3 time steps, with observations $(e_1, e_2, e_3) = (\text{true}, \text{false}, \text{true})$.

1. Find the smoothed probability estimates $P(X_t = \text{true} \mid e_{1:3})$ for $t = 1, 2, 3$ using the appropriate algorithm covered in class.

2. Find the most likely sequence of states (s_1, s_2, s_3) of variables X_1, X_2, X_3 using the appropriate algorithm covered in class.

3. From the smoothed probability estimates in question 1, give the most likely state for each of X_i, that is, the state m_i which maximizes $P(X_i = m_i \mid e_{1:3})$. Is this "sequence of most likely states"(m_1, m_2, m_3) identical to the "most likely sequence of states" from question 2? In general, what is the difference between the two?

4. Give an example of a practical application where using the "most likely sequence of states" yields a more appropriate answer than the "sequence of most likely states".