Problem Set 5

10-601 Fall 2012

Due: Friday Nov 30th, by 4pm

Please write both your Andrew ID and name on the assignment.

TAs: Selen and Brendan

1 Bayes Net questions [Brendan]

1.a Legal Bayes Nets [10 points]

Prove that in every Bayesian network there is at least one node with **no** incoming edge.

1.b Stochastic inference [10 points]

Prove that when performing stochastic inference, if not all nodes have been sampled then there is always at least one **unsampled node** that either

- Does not have any parent, or
- All its parents have been already sampled

2 Hidden Markov Models [30 points] [Selen]

Hidden Markov Models (HMMs) are probabilistic models that are used in a wide variety of sequence analysis problems. We define an HMM for K classes of hidden states and T data points. Let the data set be $\mathbf{X} = \{x_1, \ldots, x_T\}$, where each x_i a discrete observed variable. Hidden state variables are $\mathbf{Z} = \{z_1, \ldots, z_T\}$, where each hidden state is $z_t \in \{1..K\}$.

The transition probabilities are given by a $K \times K$ matrix $\mathbf{A}$, where $a_{kj} = P(z_t = k | z_{t-1} = j)$. The initial state variable z_1 is special since it does not have a parent node. Its distribution can be represented by a vector of probabilities π where $P(z_1) = \pi_{z_1}$. Finally, the emission distribution for a hidden state class k is parametrized by $\vec{\phi}_{.k}$, where $\phi_{xk} = P(x_i = x | z_i = k)$. Let $\mathbf{\Theta} = \{\mathbf{A}, \pi, \phi\}$.

2.a The full likelihood of a data set

If we have a data set $\mathbf{X} = \{x_1, \ldots, x_T\}$, write the following expressions in terms of the parameters.

1. **[2 points]** Write down the the full likelihood of observed and latent variables, $P(\mathbf{X}, \mathbf{Z} | \mathbf{\Theta})$.
2. **[2 points]** Write down the the likelihood of the data set, $P(\mathbf{X} | \mathbf{\Theta})$.

2.b Expectation-Maximization (EM) for Maximum Likelihood Learning

Our goal is to estimate $\mathbf{A}$ and ϕ that maximizes the likelihood of the data set $P(\mathbf{X} | \mathbf{\Theta})$.

1. **[3 points]** We can use the EM algorithm to compute $P(\mathbf{X} | \mathbf{\Theta})$:
   - In the E step, we use the current parameters and compute the posterior distribution of the latent variables $P(\mathbf{Z} | \mathbf{X}, \mathbf{\Theta}^{\text{old}})$.

- In the M step, we find the new parameter values by solving an optimization problem:

$$\Theta^{\text{new}} = \text{argmax}_{\mathbf{\Theta}} Q(\mathbf{\Theta}, \mathbf{\Theta}^{\text{old}}) \tag{1}$$

where

$$Q(\Theta, \Theta^{\text{old}}) = \sum_{\mathbf{Z}} P(\mathbf{Z}|\mathbf{X}, \mathbf{\Theta}^{\text{old}}) \ln P(\mathbf{X}, \mathbf{Z}|\mathbf{\Theta}) \tag{2}$$

Assume that we can compute $P(\mathbf{X}, \mathbf{Z}|\mathbf{\Theta})$ in $O(1)$. What is the time complexity of $P(\mathbf{X}|\mathbf{\Theta})$ if we use the above procedure?

2. **[8 points]** In class, we learned how to compute:

$$\alpha(z_t) = P(x_1, \ldots, x_t, z_t) \tag{3}$$
$$\beta(z_t) = P(x_{t+1}, \ldots, x_T|z_t) \tag{4}$$

Show that

$$\xi(z_{t-1}, z_t) = P(z_{t-1}, z_t|\mathbf{X}) \tag{5}$$
$$= \frac{\alpha(z_{t-1})P(x_t|z_t)P(z_t|z_{t-1})\beta(z_t)}{p(\mathbf{X})} \tag{6}$$

How can you use one of the α or β definitions to compute $P(\mathbf{X})$?

3. **[5 points]** Can we say that if any elements of the parameters π or $\mathbf{A}$ for a hidden Markov model are initially set to 0, then they will remain zero in all subsequent updates of the EM algorithm? If yes, show your steps. If no, explain.

2.c A coin game [10 points]

TA's Brendan and Selen play a coin toss game to illustrate how we can use HMMs for sequence analysis problems. Brendan starts tossing first, and they take turns. The game finishes when "THT" appears, and the winner is the one who last flips the coin. At each timestep, they can flip the coin many times, and the stopping rules are as follows:
a. At his turn, each time Brendan flips the coin, he also flips an extra biased coin ($P(H) = 0.4$.) He stops only if the extra coin lands H, otherwise he keeps flipping the fair and extra coins. The flips of the extra biased coin are not recorded.
b. At her turn, Selen flips the (fair) coin until T appears (all of her flips are recorded).
You are given a sequence of recorded coin flips, you would like to infer the winner and the flips of each player.

1. **[5 points]** Describe an HMM to model this game.

2. **[5 points]** How would you use this HMM model to infer the (most probable) winner and the (most probable) flips of each player?

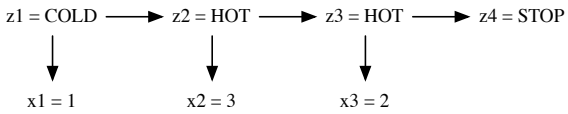
3 HMM Programming [Brendan]

Anthropologists in the future are trying to study global warming in our present day, but have lost all temperature records. However, they have my diary of how many ice cream cones I've eaten every day, and want to reconstruct the temperatures from that.[1] (Note that, compared to Problem 2, we now include an explicit STOP state to make the stochastic process well-defined.)

We model this as a Hidden Markov Model, with three latent states {COLD, HOT, STOP} and three discrete observation types {ONEIC, TWOIC, THREEIC}. (I only eat 1, 2, or 3 cones.) Cold days tend

[1] This example is adapted from Jason Eisner: http://cs.jhu.edu/~jason/papers/#eisner-2002-tnlp

to follow cold days, and hot days tend to follow hot days; and we eat more ice cream on hot days. Generation proceeds until it reaches a STOP state. Let T be the number of observations (same thing as the number of state variables, not including the STOP state). For example, one possible generated chain, three timesteps long, consists of latent states $(z_1 ... z_4) = (\text{COLD}, \text{HOT}, \text{HOT}, \text{STOP})$ and observations $(x_1 ... x_3) = (\text{ONEIC}, \text{THREEIC}, \text{TWOIC})$. Note the output space is discrete: the numbers don't matter; they could just as well be meaningless symbols.



Given a dataset of $x_1 \dots x_T$, and known transition and emission parameters, we are interested in inferring the most likely path $z_1 \dots z_T$, which we'll explicitly write out for clarity:

$$\arg\max_{z_1 \dots z_T} \; P(x_1 \dots x_T, z_t \dots z_T, z_{T+1} = \text{STOP}) \tag{7}$$

$$= P(z_1) P(x_1 \mid z_1) \left(\prod_{t=2}^{T} p(z_t \mid z_{t-1}) p(x_t | z_t) \right) p(z_{T+1} = \text{STOP} \mid z_T) \tag{8}$$

The transition and emission parameters A and ϕ are provided in the starter code on the website. See below for submission details.

1. **[5 points]** Implement an exhaustive best-path algorithm: enumerate every possible path, compute its log-probability, and choose the highest-scoring one.

   [Debugging hint: check the examples $x = \{1, 2, 1\}$ and $x = \{3, 2, 3\}$. The second datapoint is equally likely under either state, but the most likely path states are different due to the HMM chain dependencies: the HMM gives you temporal smoothing, where you consider both past and future for a better estimate of the present.]

   Report the most-likely path for this example dataset (*smallX* in the starter code):

   $$\vec{x} = [1, 1, 3, 1, 2, 3, 3]$$

2. **[2 points]** Find and report a dataset for which the most-likely path is all HOT's, that differs from *smallX* to the minimal extent possible.

3. **[5 points]** Report the log joint probability (natural logarithm, please!) that these two data-path pairs attain; i.e. for each, where $\hat{z}$ is the max-likely path, report:

   $$\log p(x_1 \dots x_T, \hat{z}_1 \dots \hat{z}_T, z_{T+1} = \text{STOP})$$

4. **[20 points]** Implement the Viterbi algorithm to compute the best-path. Test it on small examples; it should always agree with the exhaustive solution.

   [Hint: *simdata.m* will simulate new examples for more thorough testing, if you like. You will not of course always recover the correct path, but should be reasonably close, especially if A is sharper. In any case, both Viterbi and the exhaustive algorithm should agree. We will test your implementations with similar simulated data.]

   Report the Viterbi algorithm's solution to the the dataset *bigX*. Is it feasible to run the exhaustive algorithm on this dataset?

3.a Submitting your code

Write your solution in either Matlab or Python, and fill out the stubs given (we provide starter code for both languages). Submit both hardcopy and electronically:

- Print out your implementation. **All your code must less than one page long, in 10-point font. It must all be on the same page.**
- Copy your code to Andrew AFS, e.g. by using *scp* or an SFTP program to *unix.andrew.cmu.edu*, to the directory:

  `/afs/andrew.cmu.edu/usr10/brendano/10601/ps5_submit/YOURANDREWID/`

  Only copy the files you need: if you write in Matlab, don't copy the Python files, and vice versa. Please copy all the individual files so that it is directly runnable.

  Try copying a file into the directory before the deadline to make sure everything is working. You should be able to delete the file as well.

  Make sure your implementations have filled out the three given functions *logjointprob*, *exhaustive_bestpath*, and *viterbi_bestpath*.We will use automated scripts to call those functions.

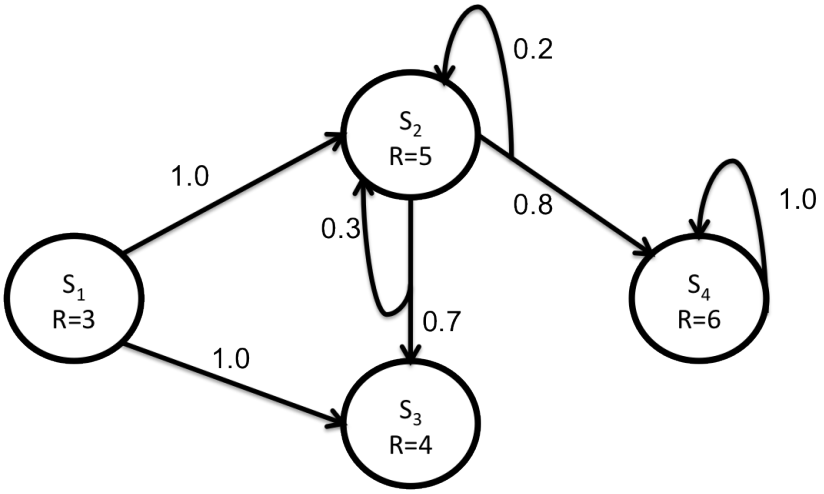
  Your code must be runnable on *unix.andrew.cmu.edu*, and is not allowed to use any external libraries (so we can run it reliably).

4 Markov Decision Processes [20 points] [Selen]

1. [10 points] A standard (first-order) MDP is described by a set of states S, a set of actions A, a transition function T, and a reward function R where $T(s; a; s')$ gives the probability of transitioning to s' after taking action a in state s, and $R(s)$ gives the immediate reward of being in state s. In a k-order MDP, probability of transitioning into a state s' given that an action a was taken in state s depends on the previous $k-1$ states. Formally, the transition function T is described as $T(s_{k-1}, ..., s_1, s, a, s') = P(s', a, s, s_1, ..., s_{k-1})$ where $P(s', a, s, s_1, ..., s_{k-1})$ is the probability of transitioning to state s' given that action a was taken in state s, and the previous $k-1$ states are $(s_{k-1}, ..., s_1)$.

Given a k-order MDP M = (S; A; T; R) describe how to construct a standard (first-order) MDP $M' = (S'; A'; T'; R')$ that is equivalent to M, meaning that a solution to M' can be easily converted into a solution to M. Describe S', A', T', R' and give a brief justification for your construction.

2. [10 points] Consider the MDP given in the figure below. R denotes rewards, and the numbers next to arrows denote probabilities of outcomes. The discount factor is $\gamma = 0.8$.
a. [5 points] Write down the numerical value of $J(S_2)$ after the first and second iterations of Value Iteration (in other words compute $J^1(S_2)$ and $J^2(S_2)$).
Initial value functions are $J^0(S_1) = 0, J^0(S_2) = 0, J^0(S_3) = 0, J^0(S_4) = 0$.
b. [5 points] Write down $J^*(S_2)$, the optimal value of state S_2.


0.2
S2
R=5
1.0
0.3
0.8
1.0
S1
R=3
S4
R=6
0.7
1.0
S3
R=4