SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

01.112 Machine Learning, Fall 2018
Homework 4

Due Friday 23 Nov 2018, 5pm

Sample Solutions

In this homework, we would like to look at the Hidden Markov Model (HMM), one of the most influential models used for structured prediction in machine learning.

1. (10 pts) Assume that we have the following training data available for us to estimate the model parameters:

State sequence	Observation sequence
$(\mathbf{X}, \mathbf{Y}, \mathbf{Z}, \mathbf{X})$	$(\mathbf{b}, \mathbf{c}, \mathbf{a}, \mathbf{b})$
$(\mathbf{X}, \mathbf{Z}, \mathbf{Y})$	$(\mathbf{a}, \mathbf{b}, \mathbf{a})$
$(\mathbf{Z}, \mathbf{Y}, \mathbf{X}, \mathbf{Z}, \mathbf{Y})$	$(\mathbf{b}, \mathbf{c}, \mathbf{a}, \mathbf{b}, \mathbf{c})$
$(\mathbf{Z}, \mathbf{X}, \mathbf{Y})$	$(\mathbf{c}, \mathbf{b}, \mathbf{a})$

Clearly state what are the parameters associated with the HMM. Under the maximum likelihood estimation (MLE), what would be the values for the optimal model parameters? Clearly show how each parameter is estimated exactly.

Answer The transition probabilities are estimated as:

$$a_{u,v} = \frac{\text{Count}(u; v)}{\text{Count}(u)}$$

	X	Y	Z	STOP
START	0.5	0.0	0.5	0.0
X	0.0	0.4	0.4	0.2
Y	0.2	0.0	0.2	0.6
Z	0.4	0.6	0.0	0.0

The emission probabilities are estimated as:

$$b_u(o) = \frac{\text{Count}(u \to o)}{\text{Count}(u)}$$

	a	b	c
X	0.4	0.6	0.0
Y	0.4	0.0	0.6
Z	0.2	0.6	0.2

2. (10 pts) Now, consider during the evaluation phase, you are given the following new observation sequence. Using the parameters you just estimated from the data, find the most probable state sequence using the Viterbi algorithm discussed in class. Clearly present the steps that lead to your final answer.

State sequence	Observation sequence
$(?, ?)$	$(\mathbf{b}, \mathbf{b})$

Answer

- Base case:

$$\pi(0, \texttt{START}) = 1, \quad \text{otherwise} \quad \pi(0, v) = 0 \ \text{ if } \ v \neq \texttt{START} \tag{1}$$

- Moving forward:

  $k = 1$

$$\pi(1, X) = a_{\texttt{START},X} \times b_X(b) = 0.5 \times 0.6 = 0.3 \tag{2}$$
$$\pi(1, Y) = a_{\texttt{START},Y} \times b_Y(b) = 0.0 \times 0.0 = 0.0 \tag{3}$$
$$\pi(1, Z) = a_{\texttt{START},Z} \times b_Z(b) = 0.5 \times 0.6 = 0.3 \tag{4}$$

  $k = 2$

$$\begin{aligned}\pi(2, X) &= \max_{u \in \mathcal{T}} \{\pi(1, u) \times a_{u,X} \times b_X(b)\} \\ &= \max\{0.3 \times 0.0 \times 0.6, \quad 0.0 \times 0.2 \times 0.6, \quad 0.3 \times 0.4 \times 0.6\} \\ &= 0.072 \end{aligned} \tag{5}$$

$$\begin{aligned}\pi(2, Y) &= \max_{u \in \mathcal{T}} \{\pi(1, u) \times a_{u,Y} \times b_Y(b)\} \\ &= \max\{0.3 \times 0.4 \times 0.0, \quad 0.0 \times 0.0 \times 0.0, \quad 0.3 \times 0.6 \times 0.0\} \\ &= 0.0 \end{aligned} \tag{6}$$

$$\begin{aligned}\pi(2, Z) &= \max_{v \in \mathcal{T}} \{\pi(1, v) \times a_{v,Z} \times b_Z(b)\} \\ &= \max\{0.3 \times 0.4 \times 0.6, \quad 0.0 \times 0.2 \times 0.6, \quad 0.3 \times 0.0 \times 0.6\} \\ &= 0.072 \end{aligned} \tag{7}$$

  $k = 3$

$$\begin{aligned}\pi(3, \texttt{STOP}) &= \max_{v \in \mathcal{T}} \{\pi(2, v) \times a_{v,\texttt{STOP}}\} \\ &= \max\{0.072 \times 0.2, 0.0 \times 0.6, 0.072 \times 0.0\} \\ &= 0.0144 \end{aligned} \tag{8}$$

- Backtracking:

$$y_2^* = \arg\max_{v \in \mathcal{T}} \{\pi(2, v) \times a_{v,\texttt{STOP}}\} = X \tag{9}$$

$$y_1^* = \arg\max_{v \in \mathcal{T}} \{\pi(1, v) \times a_{v,Y}\} = Z \tag{10}$$

Therefore, the optimal sequence is: Z, X.

3. (20 pts) The Viterbi algorithm discussed in class can be used for computing the single most probable state sequence for a new observation sequence. Specifically, in the Viterbi algorithm we are interested in finding the optimal sequence using the following formula:

$$(s_1^*, s_2^*) = \arg\max_{s_1, s_2} P(s_1, s_2 | o_1 = \mathbf{b}, o_2 = \mathbf{b})$$

However, sometimes we are interested in finding the k most probable state sequences for a given observation sequence. This is sometimes called top-k decoding. Clearly describe how to modify the Viterbi algorithm to support top-k decoding.

Answer **Note this is a very formal answer. As long as your explanations and ideas are correct, you will get the credits for this question.**

The key idea of finding the k most probable state sequences is that we redefine $\pi(s, v)$ in the Viterbi algorithm

- from "the score of the most probable path/sequence from time 0 to time s, assuming the state at time s is v"
- to "the scores of the k most probable paths/sequences from time 0 to time s, assuming the state at time s is v".

Thus $\pi(s, v)$ is now an array of size k. We use $\pi(s, v)[t]$ to denote the t-th element in the array. We need to make sure the array is sorted in descending order (or in ascending order; in that case the algorithm below would be slightly different).

We also introduce the array $\rho(s, v)$ which stores the following:

- "the k most probable paths/sequences from time 0 to time s, assuming the state at time s is v".

In other words, the $\rho(s, v)$ array stores the corresponding paths for the scores stored in $\pi(s, v)$. Specifically, the path $\rho(s, v)[t]$ has a score $\pi(s, v)[t]$. Note that storing the complete paths will simplify the pseudocode presented below. In practice, however, it is possible to only store the scores and then use a similar backtracking algorithm to recover the paths later (you can think about this!).

Here comes the algorithm:

- Base cases:

$$\pi(0, \texttt{START})[0] = 1.0 \quad \rho(0, \texttt{START})[0] = \texttt{START}$$
$$\pi(0, \texttt{START})[t] = 0.0 \quad \rho(0, \texttt{START})[0] = null \ \ (1 \leq t \leq k-1)$$
$$\pi(0, \texttt{START}) = 0.0 \quad \rho(0, \texttt{START})[0] = null \ \ (\forall v \neq \texttt{START}, 1 \leq t \leq k-1)$$

- Moving forward recursively : for any $s \in \{1, \cdots, n\}$, and for any v, we compute the following:

$$\pi(s, v)[0, 1, \cdots, k-1] = \underset{u, t' \in \{0,1,\cdots,k-1\}}{\arg\,\text{sort}} \quad \pi(s-1, u)[t'] \cdot a_{u,v} \cdot b_v(x_s) \tag{11}$$

  Note that this sort operation sort all the possible values that appear to its right, and then return a sorted list with values in descending order.
  What does this mean? It means at time s, for any u and t', we first compute the values for the following terms:

$$\pi(s-1, u)[t'] \cdot a_{u,v} \cdot b_v(x_s) \tag{12}$$

  Next we sort these values (in descending order), and then pick the k highest values and store them to the array $\pi(s, v)$.
  Now we also need to store the paths to $\rho(s, v)$. How do we do that?
  Since now we have the array $\pi(s, v)$, let's assume we have

$$\pi(s, v)[t']\pi(s-1, u)[t'] \cdot a_{u,v} \cdot b_v(x_s) \tag{13}$$

  then we can set $\rho(s, v)[t]$ as follows :

$$\rho(s, v)[t] = \rho(s-1, u)[t'] + (u \to v) \tag{14}$$

- Finally, for $s = n + 1$, we have:

$$\pi(n+1, \texttt{STOP})[0, 1, \cdots, k-1] = \underset{u, t' \in \{0,1,\cdots,k-1\}}{\arg\,\text{sort}} \quad \pi(n, u)[t'] \cdot a_{u,\texttt{STOP}} \tag{15}$$

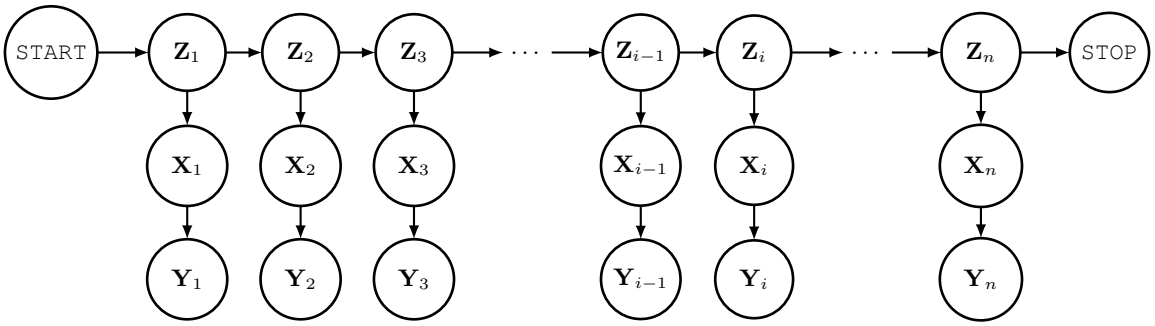
  Similarly, for the array $\pi(n+1, \texttt{STOP})$, if we have

$$\pi(n+1, \texttt{STOP})[t] = \pi(n, u)[t'] \cdot a_{u,\texttt{STOP}} \tag{16}$$

  then we set:

$$\rho(n+1, \texttt{STOP})[t] = \rho(n, u)[t'] + (u \to \texttt{STOP}) \tag{17}$$

- Now, we can simply read off the k most probable state sequences from the array $\rho(n+1, \texttt{STOP})$ in descending order.



4. (20 pts) Now consider a slightly different graphical model which extends the HMM (see above). For each state ($\mathbf{Z}$), there is now an observation pair ($\mathbf{X}$, $\mathbf{Y}$), where $\mathbf{Y}$ sequence is generated from the $\mathbf{X}$ sequence.

   Assume you are given a large collection of observation pair sequence, and a predefined set of possible states, you would like to estimate the most probable state sequence for each observation pair sequence using an EM algorithm similar to the dynamic programming algorithm discussed in class. Clearly define the forward and backward scores in a way analogous to those defined for HMM that we discussed in class, and explain what they mean. Give algorithms for computing the forward and backward scores. Analyze the time complexity associated with your algorithms.

Answer Assume we have a set of possible states $\{0, 1, \ldots, N-1, N\}$ where $0 = \texttt{START}$ and $N = \texttt{STOP}$.

$$\begin{aligned} &P(x_1, \ldots, x_{i-1}, y_1, \ldots, y_{i-1}, z_i = u, x_i, \ldots, x_n, y_i, \ldots, y_n; \theta) \\ &= P(x_1, \ldots, x_{i-1}, y_1, \ldots, y_{i-1}, z_i = u; \theta) \times P(x_i, \ldots, x_n, y_i, \ldots, y_n | z_i = u; \theta) \\ &= \alpha_u(i)\beta_u(i) \end{aligned} \tag{18}$$

where

$$\alpha_u(i) = P(x_1, \ldots, x_{i-1}, y_1, \ldots, y_{i-1}, z_i = u; \theta) \tag{19}$$

$$\beta_u(i) = P(x_i, \ldots, x_n, y_i, \ldots, y_n | z_i = u; \theta) \tag{20}$$

Forward

- Base Case

$$\alpha_u(1) = a_{\texttt{START},u}, \quad \forall \ u \in \{1, \ldots, N-1\} \tag{21}$$

- Recursive Case

  For $i = 2, \ldots, n$:

$$\alpha_u(i+1) = \sum_v \alpha_v(i) \cdot a_{v,u} \cdot b_v(x_i) \cdot c_{x_i}(y_i) \tag{22}$$

where

$$c_x(y) = P(y|x) \tag{23}$$

Backward

- Base case

$$\beta_u(n) = a_{u,\texttt{STOP}} \cdot b_u(x_n) \cdot c_{x_n}(y_n) \quad \forall u = 1, \ldots, N-1 \tag{24}$$

- Recursive Case
  For $i = n-1, \ldots, 1$:

$$\beta_u(i) = \sum_v a_{u,v} \cdot b_u(x_i) \cdot c_{x_i}(y_i) \cdot \beta_v(i+1) \tag{25}$$

At each time step/position, there are N forward (α) and N backward (β) terms to compute. To compute each term, there are $O(N)$ operations. Thus, at each time step/position, there are $O(N^2)$ operations. The length of sentence is n, which is the number of different time steps/positions. Hence, the total time complexity is $O(nN^2)$.