Nigel Chan Terng Tseng

1002027

ISTD

ML HW 4

Question 1:

For Emission Parameters using the formula count(u -> o)/count(u), where u is the state and o is the observation from its respective state:

u\o	a	B	c
X	1/6	3/6	2/6
Y	2/6	0	4/6
Z	1/6	2/6	3/6

For Transmission Parameters using the formula count(u,v)/count(u), where u is the initial state and v is the state being transitioned to:

u\v	X	Y	Z	STOP
START	2/5	0	3/5	0
X	0	3/6	2/6	1/6
Y	1/6	0	1/6	4/6
Z	3/6	3/6	0	0

Question 2:

To find the most probable state sequence, we need to calculate the scores of each path from the beginning and store the score from start to that node. Let **j** be the number of positions starting from 0(START) to n+1(STOP), and t be the tag/label at that position(this defines the node).

To find the probability score up to a certain node, we model it as P(j, t)

$$\pi(j+1, u) = \max_v \{\pi(j, v) \times b_u(x_{j+1}) \times a_{v,u}\}$$

Using the equation given in the slides, we will model it such that pi(j+1, u) is the current node we look at, and the other half of the equation is finding the max score from taking the previous node, multiplying it by the current node's emission parameter, and then multiplying it by the transition parameter from the previous node to current node

At j = 0, we only have P(0, START), which will equate to 1 as position 0 can have be the tag START.

At j = 1, to enumerate and find the scores of the next node with an emission parameter of b, we have P(1, X), P(1, Y) and P(1, Z). where we get

P(1, X) = 1 * 3/6 * 2/5 = 1/5

P(1, Y) = 1 * 0 * 0 = 0

P(1, Z) = 1 * 2/6 * 3/5 = 1/5

Given that from START to Y, we get a zero, we will hence take away this path, and continue calculating wrt P(1, X) and P(1, Z)

At j = 2, we first take into account the previous node score P(1, X) = 1/5, and enumerate the next nodes from that node:

P(2, X) = 1/5 * 3/6 * 0 = 0

P(2, Y) = 1/5 * 0 * 1/6 = 0

P(2, Z) = 1/5 * 2/6 * 2/6 = 1/45

Since we know that previous node P(1, Y) is equal to zero, we know that calculating P(2, X), P(2, Y) and P(2, Z) from this node would equate to zero.

Now we look at the previous node of P(1, Z) = 1/5, and enumerate accordingly:

P(2, X) = 1/5 * 3/6 * 3/6 = 1/20

P(2, Y) = 1/5 *0 * 1/6 = 0

P(2, Z) = 1/5 * 2/6 * 0 = 0

To find which previous node gave the highest, we then find the max (as seen in the formula)

For P(2, X) = max{0, 0, 1/20} = 1/20

For P(2, Z) = max{1/45, 0, 0} = 1/45

At j = 3, we just need to check the transition state of the previous node to STOP.

For P(2, X),

P(3, STOP) = 1/20 * 1/6 = 1/120

For P(2, Z),

P(3, STOP) = 1/45* 0 = 0

Hence. Max{P(3, STOP)} = max{1/120, 0} = 1/120

By looking at the probabilities calculated, we can infer that the best optimal path in this case would be START -> Z -> X -> STOP

Question 3:

Given the prior knowledge about certain tags or certain observations, to modify our Viterbi algorithm, we can take away certain labels at each position i, where these labels are the ones that have been ruled out as not possible when we look at the label with respect to its emission parameter x_i for that position. Thus, we only need to do a check for each position to filter out the non-possible labels in the new algorithm to deal with our new decoding task.

Question 4:

$$
\begin{aligned}
&p(x_1, x_2, \ldots, x_{j-1}, y_j = u, x_j, x_{j+1}, \ldots, x_n; \theta) \\
&= p(x_1, x_2, \ldots, x_{j-1}, y_j = u; \theta) \\
&\qquad\qquad p(x_j, x_{j+1}, \ldots, x_n | y_j = u; \theta)
\end{aligned}
$$

To begin, we can model the current problem after the equations learnt in lecture. In this case however, we will have some slight changes where z will be the label, and x and y will be the emission parameters:

$$P(x_1, x_2, \ldots, x_{i-1}, y_1, y_2, \ldots, y_{i-1}, z_i = u, x_i, x_{i+1}, \ldots, x_n, y_i, y_{i+1}, \ldots, y_n ;\theta)$$

We then split the equation into two, giving the following:

$$P(x_1, x_2, \ldots, x_{i-1}, y_1, y_2, \ldots, y_{i-1}, z_i = u; \theta) * P(x_i, x_{i+1}, \ldots, x_n, y_i, y_{i+1}, \ldots, y_n | z_i = u ;\theta)$$

These two equations are the forward and backward probabilities and can thus be written as such:

$$\alpha_u(i) * \beta_u(i)$$

where $\alpha_u(i)$ is the first equation and $\beta_u(i)$is the second equation

Forward Probability

For the first part, to calculate the sum of all scores of all the paths from START to the node u at i, we first find the equations for Forward Probabilities $\alpha_u(i)$:

Given that START has no emission parameters, $\alpha_u(1)$ is simply the transition state from START to u:

$$\alpha_u(1) = a_{START,u}$$

For the next part, we want to find $\alpha_u(i + 1)$. In this case, we can take the original formula, and simply multiply it by the additional emission parameter that has been introduced into this question. Assuming that both emission parameters x and y are independent of each other, we get:

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

where $b_v(y_i)$ is the second emission parameter introduced and $a_{v,u}$ is the transition parameter from the previous node v to current node u.

Backward Probability

For the next part, we want to calculate the sum of all scores of all paths from node u at i, to STOP. Now we find the Backward Probabilities $\beta_u(i)$:

Likewise to the previous equation, to find $\beta_u(n)$, we factor in the second emission parameter into the equation to get:

$$\beta_u(n) = a_{u,STOP} * b_u(x_n) * b_u(y_n)$$

where $b_u(y_n)$ is the second emission parameter and $a_{u,STOP}$ is the transition parameter from u to STOP.

For the last part, once again we factor in the second emission parameter into the formula for $\beta_u(i)$:

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

where $b_u(y_i)$ is the second emission parameter and $a_{u,v}$ is the transition parameter from current node u to next node v.

DONE

For the time complexity of this algorithm, the added second emission parameter does not change the overall complexity of the computation. As such, the time complexity is the same as the normal Forward Backward algorithm and is given as $O(nt^2)$.

n is derived from the number of positions in total.

the first t is derived from the number of possible nodes u at any given position

and the second t is derived from the corresponding number of nodes v that are associated with any of the previously mentioned nodes u, in the algorithm. Hence, $O(nt^2)$.