Written Assignment 2 CMPUT 651
Steven Weikai Lu

Question 1: Compute marginal probability of all observations

We start from x_1 to x_T

Marginal distribution of the first observation is trivial:

$$p(x_1) = \sum_{i \in s_1} p(s_1 = i) p(x_1 | s_1 = i) \tag{1}$$

Taking one step forward, we have:

$$p(x_1, x_2) = \sum_{i \in s_1} p(s_1 = i) p(x_1 | s_1 = i) \sum_{j \in s_2} p(s_2 = j | s_1 = i) p(x_2 | s_2 = j) \tag{2}$$

Hence, for general case, the marginal probability of all observations is a beam of probability of x_i induced from s_i sequentially conditional on previous states:

$$\begin{aligned} p(x_1, x_2, ..., x_T) = &\sum_{i \in s_1} p(s_1 = i) p(x_1 | s_1 = i) \sum_{j \in s_2} p(s_2 = j | s_1 = i) p(x_2 | s_2 = j) \\ &\sum_{l \in s_3} p(s_3 = l | s_2 = j) p(x_3 | s_3 = l) \ldots \sum_{m \in s_T} p(s_T = m | s_{T-1}) p(x_T | s_T = m) \end{aligned} \tag{3}$$

We then define:

$$M[t][j] \triangleq \sum_{1:t-1} p(x_{1:t}, s_t = j) \tag{4}$$

Recursively,

$$M[t][j] = M[t-1][s_t] \sum_{s_t} p(s_t = j | s_{t-1}) p(x_t | s_j) \tag{5}$$

Pseudo-code

Algorithm 1 Compute marginal $P(X_{1:T})$

Result: $P(X_{1:T})$
Initialize 2d-array M;
$M[1][j] = \sum_{i \in s_1} p(s_1 = i) p(x_1 | s_1 = i)$
for $t = 1, 2, \ldots, T$ **do**
  $M[t][j] = M[t-1][s_t] \sum_{s_t} p(s_t = j | s_{t-1}) p(x_t | s_j)$
end
Output $\sum_{j \in s_t} M[t][j]$

Question 2: Compute the most jointly probable sequences of observations and states

Notation: I use lowercase for an instance of x_t and s_t, and uppercase for a sequence of $X_{1:T}$ or $S_{1:T}$.

Assuming fully trained HHM, we are given π_j, $P(s_{t+1} = j|s_t = i)$, $P(x_t|s_t = j)$

Since everything past time-step t do not depend on any $x_i, i \leq t-1$,We define similarly to the lecture:

$$D[t][j] \triangleq max_{S_{1:t-1},X_{1:t-1}} P(X_{1:t-1}, S_{1:t-1}, s_t = j) \cdot max_{x_t} P(x_t|s_t = j) \tag{6}$$

Initialization

For the first time-step, trivially:

$$D[1][j] = max_{s_0} P(x_0, s_1 = j) \cdot max_{x_1} P(x_1|s_1 = j) \tag{7}$$

Note that s_0 and x_0 have nothing to choose from, so it become trivial,

$$\begin{aligned} D[1][j] = max_{\emptyset} P(s_1 = j) \cdot max_{x_1} P(x_1|s_1 = j) \\ = \pi_j \cdot max_{x_1} P(x_1|s_1 = j) \end{aligned} \tag{8}$$

Recursion

As we mentioned above, given current state, current observation is independent of anything comes after, hence we can use the similar recursion structure for all as in the lecture:

$$D[t][j] = max_{S_{1:t-1},X_{1:t-1}} P(X_{1:t-1}, S_{1:t-1}, s_t = j) \cdot max_{x_t} P(x_t|s_t = j) \tag{9}$$

$$\begin{aligned} D[t][j] = max_{s_{t-1},x_{t-1}} max_{S_{1:t-2},X_{1:t-2}} P(X_{1:t-2}, S_{1:t-2}, s_{t-1} = j) \cdot \\ max_{x_{t-1}} P(x_{t-1}|s_{t-1}) \cdot P(s_t|s_{t-1}) \cdot max_{x_t} P(x_t|s_t = j) \end{aligned} \tag{10}$$

$$D[t][j] = max_{s_t} argmax_i D[t-1][i] \cdot P(s_t = j|s_{t-1} = i) \cdot max_{x_t} P(x_t|s_t = j) \tag{11}$$

Termination

Terminates when $t = T$:

$$D[T][j] = max_{s_T} argmax_i D[T-1][i] \cdot P(s_T = j|s_{T-1} = i) \cdot max_{x_T} P(x_T|s_T = j) \tag{12}$$

Backtrack and Output

Finally, we traverse the DP table to output the sequence with maximum likelihood:

$$s_t = argmax_j D[t][j], \forall t \in [1, T] \tag{13}$$

$$x_t = argmax_i P(x_t = i | s_t = j) argmax_j D[t][j], \forall t \in [1, T] \tag{14}$$

Pseudo-code

Algorithm 2 Compute maximum $P(X_{1:T}, S_{1:T})$

Result: max $P(X_{1:T}, S_{1:T})$
Given π_j, $P(s_{t+1} = j | s_t = i)$, $P(x_t | s_t = j)$, initialize 2d-array D;
foreach *possible state* $i \in s_1$ **do**
  $D[1][j] = \pi_j \cdot max_{x_1} P(x_1 | s_1 = j)$
end
for $t = 2, 3, \ldots, T$ **do**
  foreach *possible state* $i \in s_t$ **do**
    $D[t][j] = max_{s_t} argmax_i D[t-1][i] \cdot P(s_t = j | s_{t-1} = i) \cdot max_{x_t} P(x_t | s_t = j)$
  end
end
for $t = 1, 2, \ldots, T$ **do**
  Output: $s_t = argmax_j D[t][j]$, $x_t = argmax_i P(x_t = i | s_t)$
end

Thinking question 1

$$argmax_{s_1,\ldots,s_T,x_1,\ldots,x_T} p(s_1, \ldots, s_T, x_1, \ldots, x_T) \tag{15}$$

$$argmax_{x_1,\ldots,x_T} p(x_1, \ldots, x_T) \tag{16}$$

No they are not the same. Although they share similar recursion structure for the inference, the beam induced by each s_i is different. For (15), we take the maximum probability of all s_i but for (16) we need to aggregate all probable s_i when traversing the network.

Thinking question 2

Yes. One can do this by regarding an HMM as a special class of real-time recurrent neural networks with a linear input-output function. However, this bring to us new questions:

0.1

Is this linear input-output function amplifying the power of BP? If we make some modification on the input-output function would BP be more suitable? But that seems to just bring us back to ”normal” recurrent neural networks.

0.2

When will BP be a reasonable choice for training HMMs? Maybe when data is limited?