01.112 Machine Learning
Homework 4

Joel Huang, 1002530

November 23, 2018

Hidden Markov Models

Model Parameters

Parameters associated with the HMM

The HMM is parameterized by:

1. $\mathcal{T}$, the set of states, including the `START` and `STOP` states. $\mathcal{T} = 0$ (`START`), $\ldots, |\mathcal{T}| - 2$ (`STOP`).
2. $\mathcal{O}$, the set of observation symbols.
3. $a_{u,v}$, the transition parameters, which are the probabilities of transitioning from state u to v.

$$a_{u,v} = p(y_{next} = v \,|\, y_{curr} = u) \tag{1}$$

where $u \in [0, \ldots, |\mathcal{T}| - 2]$, $v \in [1, \ldots, |\mathcal{T}| - 1]$.

4. $b_u(o)$, the emission parameters, which are the probabilities of emitting symbol o given state u.

$$b_u(o) = p(x = o \,|\, y = u) \tag{2}$$

Computing optimal model parameters

- States $\mathcal{T} = \{$`START`, X, Y, Z, `STOP`$\}$
- Observations $\mathcal{O} = \{a, b, c\}$
- Transition parameters:

$$a_{\texttt{START},X} = \frac{Count(\texttt{START}; X)}{Count(\texttt{START})} = \frac{2}{4} = 0.5 \tag{3}$$

$$a_{\texttt{START},Z} = \frac{Count(\texttt{START}; Z)}{Count(\texttt{START})} = \frac{2}{4} = 0.5 \tag{4}$$

$$a_{X,Y} = \frac{Count(X; Y)}{Count(X)} = \frac{2}{5} = 0.4 \tag{5}$$

$$a_{X,Z} = \frac{Count(X; Z)}{Count(X)} = \frac{2}{5} = 0.4 \tag{6}$$

$$a_{Y,X} = \frac{Count(Y; X)}{Count(Y)} = \frac{1}{5} = 0.2 \tag{7}$$

$$a_{Y,Z} = \frac{Count(Y; Z)}{Count(Y)} = \frac{1}{5} = 0.2 \tag{8}$$

$$a_{Z,X} = \frac{Count(Z; X)}{Count(Z)} = \frac{2}{5} = 0.4 \tag{9}$$

$$a_{Z,Y} = \frac{Count(Z; Y)}{Count(Z)} = \frac{3}{5} = 0.6 \tag{10}$$

$$a_{X,\texttt{STOP}} = \frac{Count(X; \texttt{STOP})}{Count(X)} = \frac{1}{5} = 0.2 \tag{11}$$

$$a_{Y,\texttt{STOP}} = \frac{Count(Y; \texttt{STOP})}{Count(Y)} = \frac{3}{5} = 0.6 \tag{12}$$

- Emission parameters:

$$b_X(a) = \frac{Count(X \to a)}{Count(X)} = \frac{2}{5} = 0.4 \tag{13}$$

$$b_Y(a) = \frac{Count(Y \to a)}{Count(Y)} = \frac{2}{5} = 0.4 \tag{14}$$

$$b_Z(a) = \frac{Count(Z \to a)}{Count(Z)} = \frac{1}{5} = 0.2 \tag{15}$$

$$b_X(b) = \frac{Count(X \to b)}{Count(X)} = \frac{3}{5} = 0.6 \tag{16}$$

$$b_Z(b) = \frac{Count(Z \to b)}{Count(Z)} = \frac{3}{5} = 0.6 \tag{17}$$

$$b_Y(c) = \frac{Count(Y \to c)}{Count(Y)} = \frac{3}{5} = 0.6 \tag{18}$$

$$b_Z(c) = \frac{Count(Z \to c)}{Count(Z)} = \frac{1}{5} = 0.2 \tag{19}$$

Viterbi Algorithm

Formulation

1. Our objective is to ouput the sequence of tags $y_1, \ldots, y_n$ with the highest likelihood:

$$\max_{y_1,\ldots,y_n} P(y_1, \ldots, y_n) \tag{20}$$

Let $y_1, \ldots, y_k$ be a subset of $y_1, \ldots, y_n$, where $v = y_k$. Define $\pi(k, v)$ as the sequence of tags up to y_k that has the highest likelihood:

$$\pi(k, v) = \max_{y_1,\ldots,y_k} P(y_1, \ldots, y_k) \max_{y_1,\ldots,y_k} \left\{ \prod_{i=1}^{k} a_{y_{i-1},y_i} \cdot \prod_{i=1}^{k} b_{y_i}(x_i) \right\} \tag{21}$$

Table of $\pi(k, v)$ for next state v

iteration k	0	1	...	n
$v = \texttt{START}$	1	0	...	0
$v = y_1$	0	$\pi(1, y_1)$	...	$\pi(n, y_1)$
$v = y_2$	0	$\pi(1, y_2)$	...	$\pi(n, y_2)$
...	⋮	⋮	⋱	⋮
$v = y_n$	0	$\pi(1, y_n)$	...	$\pi(n, y_n)$

For example, for iteration $k = 4$, and next state $v = y_2$, we find the maximum probability of the over all possible states.

$$\pi(4, y_2) = \max_{u \in y_0,\ldots,y_{n+1}} \{\pi(3, u) \cdot a_{u,y_2} \cdot b_{y_2}(x_4)\}, \tag{22}$$

- $\pi(3, u)$, where u is the state which survives after taking the maximum probability over all possible states when $k = 3$,
- a_{u,y_2}, the probability of transitioning from state u to state y_2,
- $b_{y_2}(x_4)$, the probability of emitting observation x_4 given state y_2.

2. Base case, $k = 0$.

$$\pi(k = 0, v) = \begin{cases} 1, & \text{if } v = \texttt{START} \\ 0, & \text{otherwise} \end{cases} \tag{23}$$

3. Recursive, for $k = 1, \ldots, n$, the best probability of the best previous path × transition × emission.

$$\pi(k, v) = \max_u \{ \pi(k-1, u) \cdot a_{u,v} \cdot b_v(x_k) \} \tag{24}$$

4. Termination case.

$$\pi(k = (n+1), \texttt{STOP}) = \max_v \{ \pi(n, v) \cdot a_{v,\texttt{STOP}} \} \tag{25}$$

Computation

1. Input: $x = \{b, b\}$
2. $k = 0$, base case:

$$\pi(0, \texttt{START}) = 1 \tag{26}$$

3. $k = 1$:

$$\begin{aligned} \pi(1, X) &= \pi(0, \texttt{START}) \cdot a_{\texttt{START},X} \cdot b_X(b) \\ &= 1 \times 0.5 \times 0.6 = 0.3 \\ \pi(1, Y) &= \pi(0, \texttt{START}) \cdot a_{\texttt{START},Y} \cdot b_Y(b) \\ &= 1 \times 0 \times 0 = 0 \\ \pi(1, Z) &= \pi(0, \texttt{START}) \cdot a_{\texttt{START},Z} \cdot b_Z(b) \\ &= 1 \times 0.5 \times 0.6 = 0.3 \end{aligned} \tag{27}$$

4. $k = 2$:

$$\begin{aligned} \pi(2, X) &= \max_{X,Y,Z} \{ \pi(1, X) \cdot a_{X,X} \cdot b_X(b), \\ &\qquad \pi(1, Y) \cdot a_{X,Y} \cdot b_Y(b), \\ &\qquad \pi(1, Z) \cdot a_{X,Z} \cdot b_Z(b) \} \\ &= \max\{0,\ 0,\ 0.072\} = 0.072 \\ \pi(2, Y) &= \max\{0,\ 0,\ 0\} = 0 \\ \pi(2, Z) &= \max\{0.072,\ 0,\ 0\} = 0.072 \end{aligned} \tag{28}$$

5. $k = 3$, termination case:

$$\begin{aligned} \pi(3, \texttt{STOP}) &= \max_v \{ \pi(2, v) \cdot a_{v,\texttt{STOP}} \} \\ &= \max\{\pi(2, X) \cdot a_{X,\texttt{STOP}} \\ &\qquad \pi(2, Y) \cdot a_{Y,\texttt{STOP}}, \\ &\qquad \pi(2, Z) \cdot a_{Z,\texttt{STOP}}\} \\ &= \max\{0.072 \times 0.2,\ 0,\ 0\} = 0.0144 \end{aligned} \tag{29}$$

6. We then backtrack, finding the best values for each state:

$$\begin{aligned} y_2 &: \arg\max_v \{ 0.0144,\ 0,\ 0 \} = X \\ y_1 &: \arg\max_v \{ 0,\ 0,\ 0.072 \} = Z \end{aligned} \tag{30}$$

So the most probable sequence is:

$$y_0, \ldots, y_{n+1} = \texttt{START}, Z, X, \texttt{STOP} \tag{31}$$

Top-k decoding

Currently, at each $\pi(i, u)$ we only store one parent. However suppose we stored the top k predecessors. So each $\pi(i, u)$ corresponds not only to a most likely value and the node which it transitioned from, but a list of the top k nodes it could have transitioned from and their values in sorted order. Therefore, in order to perform top-k decoding, we must store the top k optimal sub-paths at each node, instead of just the top sub-path with the highest probability.

Formulation

1. Base case, $i = 0$. This is unchanged from the vanilla Viterbi algorithm.

$$\pi(i = 0, v) = \begin{cases} 1, & \text{if } v = \texttt{START} \\ 0, & \text{otherwise} \end{cases} \tag{32}$$

2. Recursive, for $i = 1, \ldots, n$, the k best probabilities of the previous paths $\times$ transition $\times$ emission. Here, in contrast to the vanilla Viterbi algorithm which carries out a max operation to sieve out the best preceding node, we take the top k preceding nodes and store them.

   We define a k-max operator, which selects the k highest elements in the set, then sorts them in descending order.

$$\pi(i, v) = \text{k-max}_u \{ \pi(i-1, u) \cdot a_{u,v} \cdot b_v(x_i) \} \tag{33}$$

3. Termination case.

$$\pi(i = (n+1), \texttt{STOP}) = \text{k-max}_v \{ \pi(n, v) \cdot a_{v,\texttt{STOP}} \} \tag{34}$$

4. Backtracking. We can visualize the backtracking using a matrix A, of size $k \times n$.

	1	…	n
1			
⋮			
k			

   At the termination point, we have $\pi(n, v) \cdot a_{v,\texttt{STOP}}$ for all $v \in \mathcal{T}$. Since $\pi(n+1, \texttt{STOP})$ is now a list of the k highest probabilities of paths, we check which paths from the n-th layer led to it. For each v, if $\pi(n, v) \cdot a_{v,\texttt{STOP}} \in \pi(n+1, \texttt{STOP})$, then fill the n-th column of A with the states v that contributed to these highest probabilities.

   Similarly, for the i-th layer, compute $\pi(i-1, u) \cdot a_{u,v} \cdot b_v(x_i)$ and check which path from the $(i-1)$-th layer led to it, and fill the i-th column of A with the states u that contributed to the highest probabilities. Repeat this till the first $(i = 1)$ layer, where we instead compute $\pi(1, u) \cdot a_{\texttt{START},v} \cdot b_v(x_i)$. Reading the rows of A then gives us each top-k most probable sequences.

Forward-backward algorithm

Define two observation spaces $\{X_1, \ldots, X_n\}$, and $\{Y_1, \ldots, Y_n\}$. Define a state space $\{Z_0, \ldots, Z_{n+1}\}$ where $Z_0 = \texttt{START}$, $Z_{n+1} = \texttt{STOP}$. Parameterize the model by $a_{u,v}$, $b_u(o)$, and $c_o(e)$, where $c_o(e)$ is the additional emission probability introduced. Unfolding the joint probability using newly defined α and β:

$$\begin{aligned} & P(x_1, \ldots, x_n, y_1, \ldots, y_n, z_i = u) \\ &= P(x_1, \ldots, x_{i-1}, y_1, \ldots, y_{i-1}, z_i = u) \cdot \\ & \quad P(x_1, \ldots, x_n, y_1, \ldots, y_n | z_i = u) \\ &= P(x_1, \ldots, x_{i-1}, z_i = u) \cdot \sum_{j=1}^{i} P(y_j | x_j) \cdot \\ & \quad P(x_1, \ldots, x_{i-1} | z_i = u) \cdot \sum_{k=i}^{n} P(y_k | x_k) \\ &= \alpha_u(i) \cdot \beta_u(i) \end{aligned} \tag{35}$$

Forward algorithm

1. Base case:

$$\alpha_u(1) = a_{\texttt{START},u} \tag{36}$$

2. Recursive case:

$$\alpha_u(i) = \sum_v \alpha_v(i-1) \cdot a_{v,u} \cdot b_u(x_i) \cdot c_{x_i}(y_i) \tag{37}$$

Backward algorithm

1. Base case:

$$\beta_u(n) = a_{u,\texttt{STOP}} \cdot b_u(x_n) \cdot c_{x_n}(y_n) \tag{38}$$

2. Recursive case:

$$\beta_u(i) = \sum_v \beta_v(i+1) \cdot a_{v,u} \cdot b_u(x_i) \cdot c_{x_i}(y_i) \tag{39}$$

For this algorithm, time complexity will be $O(n|T|^2)$ for both forward and backward algorithms. We need to visit each node in the layer once, giving $|T|^2$ total operations. We also traverse from `START` to `STOP`, over n total layers. Hence the total number of operations is $n \cdot |T|^2$.