CS 466 – Introduction to Bioinformatics
Lectures 21-23

Mohammed El-Kebir

November 11, 2020

Document history:

- 11/07/2018: Initial version
- 11/08/2018: Fixed various typos and inconsistencies.
- 11/11/2018: Forward algorithm.
- 11/12/2018: Backward algorithm.
- 11/14/2018: Derivation of marginal probability using backward algorithm, posterior decoding, learning problem.
- 11/16/2018: Updated Baum-Welch to use marginal probability rather than joint probability. Updated reasoning for base case of backward algorithm.
- 11/6/2019: Fixed typo in derivation.
- 11/11/2020: Table has n columns, typo in Equation (38).

Contents

Note: These notes are based on Ref. [1] and the lectures notes by Serafim Batzoglou on this topic. For the sake of a clear exposition, I omitted termination probabilities.

1 Hidden Markov Models

Fair bet casino. Suppose you are in a casino, betting on whether the outcome of a coin flip is heads or tails. The crooked dealer changes between a fair and biased coin with probability 0.10. With the fair coin, heads and tails have equal probability of 0.5. With the biased coin, however, the probability of heads is 0.75 and the probability of tails is 0.25. Can we catch the crooked dealer?

Markov model. To answer this question, let's resort to math. We start by defining a Markov model $\mathcal{M}$ as a pair (Q, A) consisting of a set Q of states and transition probabilities $A = [a_{s,t}]$ between all pairs $(s, t) \in Q \times Q$ of states. For our purposes, we will consider a finite state set Q. Coming back to our example, we have $Q = \{\text{fair, biased}\}$. Matrix A is the *transition matrix* with nonnegative entries and each row s of A summing to 1, i.e. $a_{s,t} \geq 0$ for all pairs $(s, t) \in Q \times Q$, and $\sum_{t=1}^{|Q|} a_{s,t} = 1$ for all states $s \in Q$. For instance, in the fair bet casino we have that $a_{\text{fair,biased}} = a_{\text{biased,fair}} = 0.1$ and that $a_{\text{fair,fair}} = a_{\text{biased,biased}} = 0.9$.

The main use of a Markov model (or chain) is to evaluate the probability $\Pr(\boldsymbol{\pi})$ of a *state path* $\boldsymbol{\pi} = \pi_1, \ldots, \pi_n$, where $\pi_i \in Q$ for each $i \in [n]$. Using the chain rule[1], we have

$$\Pr(\boldsymbol{\pi}) = \Pr(\pi_1, \ldots, \pi_n) = \Pr(\pi_n \mid \pi_{n-1}, \ldots, \pi_1) \Pr(\pi_{n-1} \mid \pi_{n-2}, \ldots, \pi_1) \ldots \Pr(\pi_1). \tag{1}$$

The key property that we use in computing this probability is the *Markov property*, which states that the next state depends only on the current state and not any of the previous states. That is, $\Pr(\pi_i \mid \pi_{i-1}, \ldots, \pi_1) = \Pr(\pi_i \mid \pi_{i-1}) = a_{\pi_{i-1},\pi_i}$. Thus, we have

$$\begin{aligned}
\Pr(\boldsymbol{\pi}) &= \Pr(\pi_1, \ldots, \pi_n) = \Pr(\pi_n \mid \pi_{n-1}, \ldots, \pi_1) \Pr(\pi_{n-1} \mid \pi_{n-2}, \ldots, \pi_1) \ldots \Pr(\pi_1) && (2)\\
&= \Pr(\pi_n \mid \pi_{n-1}) \Pr(\pi_{n-1} \mid \pi_{n-2}) \ldots \Pr(\pi_2 \mid \pi_1) \Pr(\pi_1) && (3)\\
&= \Pr(\pi_1) \prod_{i=2}^{n} a_{\pi_{i-1},\pi_i} && (4)\\
&= a_{0,\pi_1} \prod_{i=2}^{n} a_{\pi_{i-1},\pi_i}. && (5)
\end{aligned}$$

Observe that $a_{0,s}$ is the initial probability $\Pr(\pi_1 = s)$ of starting in state $s \in Q$. That is, $a_{0,s} \geq 0$ for all $s \in Q$ and $\sum_{s=1}^{|Q|} a_{0,s} = 1$. Thus, A is an $(|Q| + 1) \times |Q|$ matrix. Setting $\pi_0 = 0$, we rewrite the above equation as

$$\Pr(\boldsymbol{\pi}) = \prod_{i=1}^{n} a_{\pi_{i-1},\pi_i}. \tag{6}$$

Hidden Markov model. Given a state path $\boldsymbol{\pi}$ and Markov model $\mathcal{M} = (Q, A)$ it is trivial to compute $\Pr(\boldsymbol{\pi})$. In practice, we are not given $\boldsymbol{\pi} = \pi_1, \ldots, \pi_n$ but rather a sequence $\mathbf{x} = x_1, \ldots, x_n$ of symbols that were emitted in each state π_i. More formally, we are given a set Σ of *symbols* and *emission probabilities* $E = [e_{s,k}]$ whose entries $e_{s,k}$ indicate the

[1] Repeated application of $\Pr(A, B) = \Pr(A \mid B) \Pr(B)$.

probability of emitting symbol k in state s. A *hidden Markov model* (HMM) is a four-tuple $\mathcal{M} = (Q, A, \Sigma, E)$. It is convenient to think of an HMM $\mathcal{M} = (Q, A, \Sigma, E)$ as generative model, given $\mathcal{M}$ and a state path $\boldsymbol{\pi}$, we generate a symbol sequence $\mathbf{x} = x_1, \ldots, x_n$ using emission probabilities E. That is, the probability of generating symbol s in state π_i is precisely $e_{\pi_i,s}$. Thus, the joint probability of a symbol sequence $\mathbf{x}$ and state path $\boldsymbol{\pi}$ is computed as

$$\begin{aligned}
\Pr(\mathbf{x}, \boldsymbol{\pi}) &= \Pr(x_1, \pi_1, \ldots, x_n, \pi_n) && (7)\\
&= \Pr(x_n\pi_n \mid x_{n-1}, \pi_{n-1}, \ldots, x_1, \pi_1) \ldots \Pr(x_2\pi_2 \mid x_1\pi_1) \Pr(x_1\pi_1) && (8)\\
&= \Pr(x_n\pi_n \mid \pi_{n-1}) \ldots \Pr(x_2\pi_2 \mid \pi_1) \Pr(x_1 \mid \pi_1) \Pr(\pi_1) && (9)\\
&= \prod_{i=1}^{n} e_{\pi_i,x_i} \cdot a_{\pi_{i-1},\pi_i}. && (10)
\end{aligned}$$

In practice, the state path $\boldsymbol{\pi}$ that generated the symbol sequence $\mathbf{x}$ is *hidden* to us (hence the name).

Three questions. Given $\mathcal{M} = (Q, A, \Sigma, E)$, we are interested in the following three questions.

1. What is the most probable state path $\boldsymbol{\pi}^*$ that generated a given symbol sequence $\mathbf{x}$? That is, find $\boldsymbol{\pi}^* = \arg\max_{\pi} \Pr(\mathbf{x}, \boldsymbol{\pi})$.
2. What is the probability $\Pr(\mathbf{x}) = \sum_{\boldsymbol{\pi}} \Pr(\mathbf{x}, \boldsymbol{\pi})$?
3. What is the posterior probability that the i-th observation came from state s given the observed symbol sequence $\mathbf{x}$? That is, compute $\Pr(\pi_i = s \mid \mathbf{x})$.

Using the fair bet casino example, question 1 would be to determine for each coin flip whether the crooked dealer used the fair or biased coin. An example of question 2, would be to compute the probability of observing n times heads in a row (not caring about whether the dealer was cheating or not), i.e. $\Pr(x_1 = H, \ldots x_n = H)$. Finally, question 3 seeks to answer whether the dealer cheated at time i given n heads/tails observations $\mathbf{x}$, i.e. $\Pr(\pi_i = B \mid \mathbf{x})$. We will solve these three questions using different algorithms. The key observation is that the Markov property leads to optimal substructure in each of these questions, thus enabling the use of dynamic programming.

2 Viterbi Algorithm: Most Probable State Path

We are interested in identifying $\boldsymbol{\pi}^* \in Q^n$ with maximum likelihood $\Pr(\mathbf{x}, \boldsymbol{\pi}^*)$. From (7), we immediately observe optimal substructure. Thus, we define $\mathbf{x}_i = x_1, \ldots, x_i$, $\boldsymbol{\pi}_i^* = \pi_1^*, \ldots, \pi_i^*$. Let $v[s, i]$ denote the probability $\Pr(\mathbf{x}_i, \boldsymbol{\pi}_{i-1}^*, \pi_i = s)$, i.e. the probability of the most probable state path of the first i observations with final state $\pi_i = s$. Clearly,

$$\Pr(\mathbf{x}, \boldsymbol{\pi}^*) = \max_{s \in Q} \Pr(\mathbf{x}_n, \boldsymbol{\pi}_{n-1}^*, \pi_n = s) = \max_{s \in Q} v[s, n]. \tag{11}$$

Initially, $v[s,1] = a_{0,s}e_{s,x_1}$ for all states $s \in Q$. We derive the step $i > 1$ as follows.

$$\begin{aligned}
v[s,i] &= \Pr(\mathbf{x}_i, \boldsymbol{\pi}^*_{i-1}, \pi_i = s) && (12)\\
&= \Pr(x_1, \ldots, x_{i-1}, x_i, \pi^*_1, \ldots, \pi^*_{i-1}, \pi_i = s) && (13)\\
&= \Pr(x_i, \pi_i = s \mid x_1, \ldots, x_{i-1}, \pi^*_1, \ldots, \pi^*_{i-1}) \Pr(x_1, \ldots, x_{i-1}, \pi^*_1, \ldots, \pi^*_{i-1}) && (14)\\
&= \Pr(x_i, \pi_i = s \mid \pi^*_{i-1}) \Pr(x_1, \ldots, x_{i-1}, \pi^*_1, \ldots, \pi^*_{i-1}) && (15)\\
&= \max_{t \in Q} \Pr(x_i, \pi_i = s \mid \pi_{i-1} = t) \Pr(x_1, \ldots, x_{i-1}, \pi^*_1, \ldots, \pi^*_{i-2}, \pi_{i-1} = t) && (16)\\
&= \max_{t \in Q} \Pr(x_i, \mid \pi_i = s, \pi_{i-1} = t) \Pr(\pi_i = s \mid \pi_{i-1} = t) \cdot v[t, i-1] && (17)\\
&= \max_{t \in Q} \Pr(x_i, \mid \pi_i = s) \cdot a_{t,s} \cdot v[t, i-1] && (18)\\
&= \max_{t \in Q} e_{s,x_i} \cdot v[t, i-1] \cdot a_{t,s} && (19)\\
&= e_{s,x_i} \max_{t \in Q} v[t, i-1] \cdot a_{t,s}. && (20)
\end{aligned}$$

Thus, we have the following recurrence.

$$v[s,i] = \begin{cases} a_{0,s}e_{s,x_1}, & \text{if } i = 1,\\ e_{s,x_i} \max_{t \in Q} v[t, i-1] \cdot a_{t,s}, & \text{if } i > 1. \end{cases} \tag{21}$$

We view v as an $|Q| \times n$ table. From the recurrence, we see that the rest of this table can be filled out column-by-column. The first column $i = 1$ corresponds to the base case. Each entry in column $i > 1$ requires lookups of all $|Q|$ entries in the preceding column $i-1$. Thus, the running time is $O(n|Q|^2)$. We obtain the final maximum likelihood $\Pr(\mathbf{x}, \boldsymbol{\pi}^*)$ and hidden state π_n from the completed table by scanning through the last column n and identifying the state $s \in Q$ with largest likelihood $v[s,n]$ and setting $\pi_n = s$. We obtain the corresponding maximum likelihood state path $\boldsymbol{\pi}^*$ by back tracking from (π_n, n). This algorithm is known as the Viterbi algorithm.

3 Forward Algorithm

We are interested in the probability $\Pr(\mathbf{x})$, the probability of observing a given symbol sequence $\mathbf{x} = x_1, \ldots, x_n$ marginalizing over all combinations of hidden states $\boldsymbol{\pi} = \pi_1, \ldots, \pi_n$. That is,

$$\Pr(\mathbf{x}) = \sum_{\boldsymbol{\pi}} \Pr(\mathbf{x}, \boldsymbol{\pi}) = \sum_{(\pi_1, \ldots, \pi_n)} \Pr(x_1, \pi_1, \ldots, x_n, \pi_n) = \sum_{(\pi_1, \ldots, \pi_n)} \prod_{i=1}^{n} e_{\pi_i, x_i} \cdot a_{\pi_{i-1}, \pi_i}. \tag{22}$$

There are $|Q|^n$ possible state paths $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_n)$. Thus, computing $\Pr(\mathbf{x})$ by brute force takes $O(n|Q|^n)$ time, where the linear factor n is the time required to compute $\Pr(\mathbf{x}, \boldsymbol{\pi})$. Can we do better than exponential time?

Alternatively, we can approximate $\Pr(\mathbf{x})$ by the probability $\Pr(\mathbf{x}, \boldsymbol{\pi}^*)$ where $\boldsymbol{\pi}^*$ is the state path identified by the Viterbi algorithm in $O(n|Q|^2)$ time. The underlying assumption is that most of the probability mass is contributed by the most probable state path $\boldsymbol{\pi}^*$.

However, this is an approximation. We will describe an *exact* algorithm for computing $\Pr(\mathbf{x})$ with the same running time of $O(n|Q|^2)$ as the Viterbi algorithm.

Recall that $\mathbf{x}_i = x_1, \ldots, x_i$, $\boldsymbol{\pi}_i = \pi_1, \ldots, \pi_i$. Let $f[s, i]$ denote the probability of observing $\mathbf{x}_i = (x_1, \ldots, x_i)$ *and* that $\pi_i = s$. That is,

$$\Pr(\mathbf{x}_i, \pi_i = s) = \Pr(x_1, \ldots, x_i, \pi_i = s) = f[s, i]. \tag{23}$$

Clearly,

$$\Pr(\mathbf{x}) = \Pr(x_1, \ldots, x_n) = \sum_{s \in Q} \Pr(x_1, \ldots, x_n, \pi_n = s) = \sum_{s \in Q} f[s, n]. \tag{24}$$

Recalling that $\mathbf{x}_1 = x_1$ is the singleton sequence, we have that $f[s, 1] = \Pr(x_1, \pi_1 = s) = a_{0,s} e_{s,x_1}$ initially. We derive the step $i > 1$ as follows.

$$\begin{aligned}
f[s, i] &= \Pr(\mathbf{x}_i, \pi_i = s) = \Pr(x_1, \ldots, x_i, \pi_i = s) && (25)\\
&= \sum_{(\pi_1, \ldots, \pi_{i-1})} \Pr(x_1, \ldots, x_{i-1}, x_i, \pi_1, \ldots, \pi_{i-1}, \pi_i = s) && (26)\\
&= \sum_{(\pi_1, \ldots, \pi_{i-1})} \Pr(x_1, \ldots, x_{i-1}, \pi_1, \ldots, \pi_{i-1}, \pi_i = s) \cdot e_{s,x_i} && (27)\\
&= \sum_{t \in Q} \sum_{(\pi_1, \ldots, \pi_{i-2})} \Pr(x_1, \ldots, x_{i-1}, \pi_1, \ldots, \pi_{i-2}, \pi_{i-1} = t) \cdot a_{t,s} \cdot e_{s,x_i} && (28)\\
&= \sum_{t \in Q} \Pr(x_1, \ldots, x_{i-1}, \pi_{i-1} = t) \cdot a_{t,s} \cdot e_{s,x_i} && (29)\\
&= \sum_{t \in Q} f[t, i-1] \cdot a_{t,s} \cdot e_{s,x_i} && (30)\\
&= e_{s,x_i} \sum_{t \in Q} f[t, i-1] \cdot a_{t,s}. && (31)
\end{aligned}$$

Hence, we have the following recurrence.

$$f[s, i] = \begin{cases} a_{0,s} e_{s,x_1}, & \text{if } i = 1, \\ e_{s,x_i} \sum_{t \in Q} \{f[t, i-1] \cdot a_{t,s}\}, & \text{if } i > 1. \end{cases} \tag{32}$$

We compute f in exactly the same way as v, thus requiring $O(n|Q|^2)$ time. The probability $\Pr(\mathbf{x})$ that we seek is obtained by summing the entries in the last column, i.e. $\Pr(\mathbf{x}) = \sum_{s \in Q} f[s, n]$.

4 Backward Algorithm

How do we compute the posterior probability $\Pr(\pi_i = s \mid \mathbf{x})$? From probability theory, we have

$$\Pr(\pi_i = s \mid \mathbf{x}) = \frac{\Pr(\mathbf{x}, \pi_i = s)}{\Pr(\mathbf{x})}. \tag{33}$$

We know how to compute $\Pr(\mathbf{x})$ using the forward algorithm, but how do we compute $\Pr(\mathbf{x}, \pi_i = s)$? We have that

$$\begin{aligned}
\Pr(\mathbf{x}, \pi_i = s) &= \Pr(x_1, \ldots, x_n, \pi_i = s) && (34)\\
&= \Pr(x_1, \ldots, x_i, \pi_i = s) \Pr(x_{i+1}, \ldots, x_n \mid x_1, \ldots, x_i, \pi_i = s) && (35)\\
&= f[s, i] \cdot \Pr(x_{i+1}, \ldots, x_n \mid \pi_i = s). && (36)
\end{aligned}$$

Let $b[s, i]$ denote the probability $\Pr(x_{i+1}, \ldots, x_n \mid \pi_i = s)$. Thus,

$$\Pr(\mathbf{x}, \pi_i = s) = f[s, i] \cdot b[s, i]. \tag{37}$$

Recall that the set Ω is the sample space, the space of all possible events. Initially,

$$b[s, n] = \Pr(x_{n+1}, \ldots, x_n \mid \pi_n = s) = \Pr(\Omega \mid \pi_n = s) = \frac{\Pr(\Omega, \pi_n = s)}{\Pr(\pi_n = s)} = \frac{\Pr(\pi_n = s)}{\Pr(\pi_n = s)} = 1. \tag{38}$$

As for the step, we have

$$\begin{aligned}
b[s, i] &= \Pr(\mathbf{x}_{i+1} \mid \pi_i = s) = \Pr(x_{i+1}, \ldots, x_n \mid \pi_i = s) && (39)\\
&= \sum_{(\pi_{i+1}, \ldots, \pi_n)} \Pr(x_{i+1}, \ldots, x_n, \pi_{i+1}, \ldots, \pi_n \mid \pi_i = s) && (40)\\
&= \sum_{t \in Q} \sum_{(\pi_{i+2}, \ldots, \pi_n)} \Pr(x_{i+1}, \ldots, x_n, \pi_{i+1} = t, \ldots, \pi_n \mid \pi_i = s) && (41)\\
&= \sum_{t \in Q} \sum_{(\pi_{i+2}, \ldots, \pi_n)} \Pr(x_{i+1}, \pi_{i+1} = t, x_{i+2}, \ldots, x_n, \pi_{i+2}, \ldots, \pi_n \mid \pi_i = s) && (42)\\
&= \sum_{t \in Q} \sum_{(\pi_{i+2}, \ldots, \pi_n)} \Pr(x_{i+1}, \pi_{i+1} = t \mid \pi_i = s) \Pr(x_{i+2}, \ldots, x_n, \pi_{i+2}, \ldots, \pi_n \mid \pi_i = s, x_{i+1}, \pi_{i+1} = t) && (43)\\
&= \sum_{t \in Q} \sum_{(\pi_{i+2}, \ldots, \pi_n)} \Pr(x_{i+1}, \pi_{i+1} = t \mid \pi_i = s) \Pr(x_{i+2}, \ldots, x_n, \pi_{i+2}, \ldots, \pi_n \mid \pi_{i+1} = t) && (44)\\
&= \sum_{t \in Q} a_{s,t} \cdot e_{t, x_{i+1}} \Pr(x_{i+2}, \ldots, x_n \mid \pi_{i+1} = t) && (45)\\
&= \sum_{t \in Q} a_{s,t} \cdot e_{t, x_{i+1}} \cdot b[t, i+1]. && (46)
\end{aligned}$$

Thus, we have the following recurrence.

$$b[s, i] = \begin{cases} 1, & \text{if } i = n, \\ \sum_{t \in Q} a_{s,t} \cdot e_{t, x_{i+1}} \cdot b[t, i+1], & \text{if } 1 \leq i < n. \end{cases} \tag{47}$$

Hence, the posterior probability is given by

$$\Pr(\pi_i = s \mid \mathbf{x}) = \frac{\Pr(\mathbf{x}, \pi_i = s)}{\Pr(\mathbf{x})} = \frac{f[s, i] \cdot b[s, i]}{\sum_{s \in Q} f[s, n]}. \tag{48}$$

This is the backward algorithm. We note that we can solve the data likelihood problem using the backward algorithm as follows.

$$\begin{aligned}
\Pr(\mathbf{x}) &= \Pr(x_1, \ldots, x_n) && (49)\\
&= \sum_{s \in Q} \Pr(x_1, \ldots, x_n, \pi_1 = s) && (50)\\
&= \sum_{s \in Q} \Pr(\pi_1 = s) \Pr(x_1, \ldots, x_n \mid \pi_1 = s) && (51)\\
&= \sum_{s \in Q} \Pr(\pi_1 = s) \Pr(x_2 \mid \pi_1 = s) \Pr(x_2, \ldots, x_n \mid \pi_1 = s) && (52)\\
&= \sum_{s \in Q} \Pr(\pi_1 = s) \Pr(x_2 \mid \pi_1 = s) \Pr(\mathbf{x}_2 \mid \pi_1 = s) && (53)\\
&= \sum_{s \in Q} a_{0,s} \cdot e[s, x_1] \cdot b[s, 1]. && (54)
\end{aligned}$$

In the above derivation, Equation (52) follows from the Markov property, i.e. given the hidden state π_i the random variable x_i is independent of all other variables.

5 Posterior Decoding

In the previous section, we learned how to use the Forward-Backward algorithm to compute the posterior probability $\Pr(\pi_i = s \mid \mathbf{x})$. Let $\hat{\pi}_i$ denote the hidden state that maximizes this posterior probability, i.e.

$$\hat{\pi}_i = \arg\max_{s \in Q} \Pr(\pi_i = s \mid \mathbf{x}) = \arg\max_{s \in Q} \frac{f[s, i] \cdot b[s, i]}{\sum_{t \in Q} f[t, n]}. \tag{55}$$

Thus, $\hat{\pi}_i$ is the most likely state at each position, which is the desired quantity in certain applications where we only care about the hidden state at a specific position. Since $\hat{\pi}_i$ is computed independently from the hidden states at the other positions, it may differ from π_i^* obtained using the Viterbi algorithm. We call $\hat{\boldsymbol{\pi}} = \hat{\pi}_1, \ldots, \hat{\pi}_n$ the *posterior decoding*. Since there may be a position i such that $\hat{\pi}_i \neq \pi_i^*$, it does not necessarily hold that $\hat{\boldsymbol{\pi}}$ equals $\boldsymbol{\pi}^* = \arg\max_{\boldsymbol{\pi} \in Q^n} \Pr(\mathbf{x}, \boldsymbol{\pi})$ obtained using the Viterbi algorithm, i.e. $\Pr(\mathbf{x}, \hat{\boldsymbol{\pi}}) \neq \Pr(\mathbf{x}, \boldsymbol{\pi}^*)$. In particular, the posterior decoding may contain state transitions from $\hat{\pi}_i$ to $\hat{\pi}_{i+1}$ with zero probability, i.e. $\Pr(\hat{\pi}_{i+1} \mid \hat{\pi}_i) = a_{i,i+1} = 0$, and thus $\Pr(\mathbf{x}, \hat{\boldsymbol{\pi}}) = 0$. This means that the state path of the posterior decoding may be invalid.

6 Learning Problem

Tables 1 and 2 summarize the algorithms that we have seen so far. What these algorithms have in common is that they take a parameterized HMM as input with transition probabilities $A = [a_{s,t}]$ and emission probabilities $E = [e_{s,k}]$. What do we do if these two matrices are not given to us? In that case, we have to *learn* the two matrices from a training set. We distinguish two learning problems.

Type	Probability	Method	Solution
Joint	$\max_{\boldsymbol{\pi}\in Q^n} \Pr(\mathbf{x}, \boldsymbol{\pi})$	Viterbi algorithm	$\Pr(\mathbf{x}, \boldsymbol{\pi}^*) = \max_{s\in Q} v[s, n]$
Marginal	$\Pr(\mathbf{x})$	Forward algorithm, or backward algorithm	$\Pr(\mathbf{x}) = \sum_{s\in Q} f[s, n]$, $\Pr(\mathbf{x}) = \sum_{s\in Q} a_{0,s} \cdot e[s, x_1] \cdot b[s, 1]$
Posterior	$\Pr(\pi_i = s \mid \mathbf{x})$	Forward algorithm, and backward algorithm	$\hat{\pi}_i = \arg\max_{s\in Q} \frac{f[s,i]\cdot b[s,i]}{\sum_{t\in Q} f[t,n]}$

Table 1: Hidden Markov Models – Three different probabilities

Type	Recurrence
Viterbi	$v[s,i] = \begin{cases} a_{0,s}e_{s,x_1}, & \text{if } i = 1, \\ e_{s,x_i} \max_{t\in Q} v[t, i-1] \cdot a_{t,s}, & \text{if } i > 1. \end{cases}$
Forward	$f[s,i] = \begin{cases} a_{0,s}e_{s,x_1}, & \text{if } i = 1, \\ e_{s,x_i} \sum_{t\in Q} f[t, i-1] \cdot a_{t,s}, & \text{if } i > 1. \end{cases}$
Backward	$b[s,i] = \begin{cases} 1, & \text{if } i = n, \\ \sum_{t\in Q} a_{s,t} \cdot e_{t,x_{i+1}} \cdot b[t, i+1], & \text{if } 1 \le i < n. \end{cases}$

Table 2: Hidden Markov Models – Three recurrences that each can be computed in $O(n|Q|^2)$ time and $O(n|Q|)$ space.

1. Estimation of A and E given a set $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\}$ of observed symbols and corresponding set $\{\boldsymbol{\pi}^{(1)}, \ldots, \boldsymbol{\pi}^{(N)}\}$ of state paths that generated each instance.

2. Estimation of A and E given only a set $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\}$ of observed symbols without the underlying state paths.

We start with the first learning problem. Given set $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\}$ of observed symbols and corresponding set $\{\boldsymbol{\pi}^{(1)}, \ldots, \boldsymbol{\pi}^{(N)}\}$ of state paths, let $x_i^{(j)}$ denote the i-th observation of instance j and let $\pi_i^{(j)}$ denote the state that generated the i-th observation of instance j. Let $n(j)$ denote the number of observations/states in instance $j \in [N]$. Moreover, let $\tilde{a}_{0,s}$ denote the number of instances that start in state $s \in Q$, i.e.

$$\tilde{a}_{0,s} = |\{j \in [N] \mid \pi_1^{(j)} = s\}| \qquad \forall s \in Q. \tag{56}$$

Let $\tilde{a}_{s,t}$ denote the number of times we transition from state s to state t in the training set, i.e.

$$\tilde{a}_{s,t} = |\{(j,i) \mid j \in [N], i \in [n(j)-1], \pi_i^{(j)} = s, \pi_{i+1}^{(j)} = t\}| \qquad \forall s,t \in Q. \tag{57}$$

Finally, let $\tilde{e}_{s,k}$ denote the number of times we observe symbol $k \in \Sigma$ in state $s \in Q$, i.e.

$$\tilde{e}_{s,k} = |\{(j,i) \mid j \in [N], i \in [n(j)], \pi_i^{(j)} = s, x_i^{(j)} = k\}| \qquad \forall s \in Q, k \in \Sigma. \tag{58}$$

Assuming independence between instances in the training set, the maximum likelihood estimators of $A = [a_{s,t}]$ and $E = [e_{s,k}]$ for the joint probability $\Pr(\mathbf{x}^{(1)}, \boldsymbol{\pi}^{(1)}, \ldots, \mathbf{x}^{(N)}, \boldsymbol{\pi}^{(N)}) =$

$\prod_{j=1}^{N} \Pr(\mathbf{x}^{(j)}, \boldsymbol{\pi}^{(j)})$ are given by

$$a_{s,0} = \frac{\tilde{a}_{0,s}}{N}, \qquad \forall s \in Q, \tag{59}$$

$$a_{s,t} = \frac{\tilde{a}_{s,t}}{\sum_{u \in Q} \tilde{a}_{s,u}}, \qquad \forall s, t \in Q, \tag{60}$$

$$e_{s,k} = \frac{\tilde{e}_{s,k}}{\sum_{\ell \in \Sigma} \tilde{e}_{s,\ell}}, \qquad \forall s \in Q, k \in \Sigma. \tag{61}$$

The main drawback is that limited data (i.e. small N) may lead to a probability of 0 (overfitting). To overcome this, we can add pseudocounts $\epsilon > 0$ and thus consider $\tilde{a}_{s,0} + \epsilon, \tilde{a}_{s,t} + \epsilon$ and $\tilde{e}_{s,k} + \epsilon$ in the numerators.

Let's consider the second learning problem, where we are only given set $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\}$ of observed symbols and the underlying states are hidden to us.

Baum-Welch algorithm. To learn $A = [a_{s,t}]$ and $E = [e_{s,k}]$, we use Expectation Maximization (EM). The idea is to guess initial matrices $\theta = (A, E)$, and then iteratively refine these matrices so as to maximize the marginal probability $\Pr(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)} \mid \theta) = \prod_{j=1}^{N} \Pr(\mathbf{x}^{(j)} \mid \theta)$.

Let's start with a single training instance j. How do we find $\theta^* = (A^*, E^*)$ such that $\Pr(\mathbf{x}^{(j)} \mid \theta^*)$ is maximum? This is a hard inference problem that we will approximate heuristically using an Expectation Maximization algorithm. We start by deriving the probability $\Pr(\pi_i^{(j)} = s, \pi_{i+1}^{(j)} = t \mid \mathbf{x}^{(j)}, \theta)$ that $a_{s,t}$ is used in instance $\mathbf{x}^{(j)}$ at position i as follows.

$$\Pr(\pi_i^{(j)} = s, \pi_{i+1}^{(j)} = t \mid \mathbf{x}^{(j)}, \theta) = \frac{\Pr(\pi_i^{(j)} = s, \pi_{i+1}^{(j)} = t, \mathbf{x}^{(j)} \mid \theta)}{\Pr(\mathbf{x}^{(j)} \mid \theta)} \tag{62}$$

$$= \frac{\Pr(\pi_i^{(j)} = s, \pi_{i+1}^{(j)} = t, \mathbf{x}^{(j)} \mid \theta)}{\sum_{s \in Q} f^{(j)}[s, n(j)]}. \tag{63}$$

The denominator follows from the forward algorithm (we could have also used the backward algorithm). Let's focus on the numerator – we will drop the instance index (j) for clarity.

$$\Pr(\pi_i = s, \pi_{i+1} = t, \mathbf{x} \mid \theta) = \Pr(x_1, \ldots, x_i, \pi_i = s, \pi_{i+1} = t, x_{i+1}, \ldots, x_n \mid \theta) \tag{64}$$

$$= \Pr(\pi_{i+1} = t, x_{i+1}, \ldots, x_n \mid x_1, \ldots, x_i, \pi_i = s, \theta) \tag{65}$$
$$\cdot \Pr(x_1, \ldots, x_i, \pi_i = s \mid \theta)$$

$$= \Pr(\pi_{i+1} = t, x_{i+1}, \ldots, x_n \mid \pi_i = s, \theta) \cdot f[s, i] \tag{66}$$

$$= \Pr(x_{i+2}, \ldots, x_n \mid \pi_{i+1} = t, \theta) \Pr(x_{i+1} \mid \pi_{i+1} = t, \theta) \tag{67}$$
$$\Pr(\pi_{i+1} = t \mid \pi_i = s, \theta) \cdot f[s, i]$$

$$= b[t, i+1] \cdot e_{t,x_{i+1}} \cdot a_{s,t} \cdot f[s, i] \tag{68}$$

Thus, the expected number of times of transitioning from s to t at any position $i < n(j)$ in instance $j \in [N]$ is given by

$$\frac{1}{\sum_{s \in Q} f^{(j)}[s, n(j)]} \sum_{i=1}^{n(j)-1} b^{(j)}[t, i+1] \cdot e_{t,x_{i+1}} \cdot a_{s,t} \cdot f^{(j)}[s, i]. \tag{69}$$

Across all training instances, we have

$$\mathbb{E}(a_{s,t}) = \tilde{a}_{s,t} = \sum_{j=1}^{N} \frac{1}{\sum_{s \in Q} f^{(j)}[s, n(j)]} \sum_{i=1}^{n(j)-1} b^{(j)}[t, i+1] \cdot e_{t,x_{i+1}} \cdot a_{s,t} \cdot f^{(j)}[s, i]. \tag{70}$$

Similarly, we can derive

$$\mathbb{E}(e_{s,k}) = \tilde{e}_{s,k} = \sum_{j=1}^{N} \frac{\sum_{i=1}^{n(j)} \mathbf{1}(x_i^{(j)} = k) \cdot f^{(j)}[s, i] \cdot b^{(j)}[s, i]}{\sum_{s \in Q} f^{(j)}[s, n(j)]}. \tag{71}$$

where indicator function $\mathbf{1}(x_i^{(j)} = k) = 1$ if $x_i^{(j)} = k$ and $\mathbf{1}(x_i^{(j)} = k) = 0$ otherwise, and

$$\mathbb{E}(a_{0,s}) = \tilde{a}_{0,s} = \sum_{j=1}^{N} \frac{a_{0,s} \cdot e_{s,x_1^{(j)}}}{\sum_{s \in Q} f^{(j)}[s, n(j)]}. \tag{72}$$

Equations (70), (71) and (72) correspond to the expectation (E) step and take $O(Nm|Q|^2)$ time and $O(m|Q|^2)$ space to compute (where $m = \max_{j \in [N]} n(j)$). In the maximization (M) step, we compute the new values of $\theta = (A, E)$ given $\tilde{a}_{s,0}, \tilde{a}_{s,t}$ and $\tilde{e}_{s,k}$ using Equations (59), (60) and (61). We repeat this procedure until convergence of the marginal likelihood

$$\Pr(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)} \mid \theta) = \prod_{j=1}^{N} \Pr(\mathbf{x}^{(j)} \mid \theta) = \prod_{j=1}^{N} \sum_{s \in Q} f^{(j)}[s, n]. \tag{73}$$

This algorithm is known as the Baum-Welch algorithm.

References

[1] Richard Durbin, Sean R. Eddy, Anders Krogh, and Graeme Mitchison. *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*. Cambridge University Press, 1998.