# Protein Secondary Structure Prediction using HMM

Name: Feng Liu
ID: 5002603

## Introduction

In this article, I will design a three-state Hidden Marcov Model of protein secondary structure, with three-state indicating helix, sheets and loops in the protein, and train the HMM model using maximum likelihood learning and Baum-Welch algorithms with . Then I will apply the trained HMM models to some test samples.

## Design of the HMM model

Assuming the protein secondary structure conforms a Hidden Marcov Model and assuming there are three states in the HMM model, state '1', state '2' and state '3' indicating helix, sheets and loops separately. Thus, the set of hidden states should be S={1,2,3}.

Assuming there are 20 possible observation symbols, so the set of observation symbols should be V={1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20}. Every observation symbols represents one kind of amino acid residue in the protein sequences.

In the following part, I will train the HMM model with the sequences provided in the "protein-secondary-structure.train" file and get the state transition probability matrix A and emission probability matrix B. And using sequences in the provided in "protein-secondary-structure.test" to test the HMM I have trained.

## Methods

$2.1 Training with maximum likelihood learning

1. For every training sequence, calculate the number of times of transition between every two states occurs in the sequence, store it in "seq(n).num_trans" matrix. For example, seq(1).num_trans(1,3) is the value of number of times of transition from state 1 to state 3 in sequence 1.
2. For every training sequence, calculate the number of emission times of every state, and store it in "seq(n).num_emit" matrix. For example, seq(1).num_emit(1,15) is the value of number of times of state 1 emits observation symbol 15 (Proline) in sequence 1.

3. For every training sequence, calculate the state transition probability matrix seq(n).A.
4. For every training sequence, calculate the emission probability matrix seq(n).B.
5. Calculate state transition probability matrix A and the emission probability matrix B by averaging seq(n).A and seq(n).B for all sequences.
6. To avoid 0 probabilities, before calculating, set all values in seq(n).num_trans and seq(n).num_emit to 1.

$2.2 Training with Baum-Welch algorithm

1. First choose an state transition probability matrix A and emission probability matrix B
2. Applying forward algorithm, calculating the forward variable by the following step

    M is the number of observation symbols per state, in our case 20;
    N is the number of total states, in our case 3
    T is the sequence length

*forward variable*:[4]

$$\alpha_t(i) = P(o_1 o_2 \cdots o_t, q_t = S_i|\lambda).$$

$\alpha_t(i)$ can be solved inductively:

1. initialization:

$$\alpha_1(i) = \pi_i b_i(o_1), \quad 1 \le i \le N$$

and

2. induction:

$$\alpha_{t+1}(j) = [\sum_{i=1}^{N} \alpha_t(i) a_{ij}] b_j(o_{t+1}),$$
$$1 \le t \le T-1, 1 \le j \le N.$$

3. Applying backward algorithm, calculating backward variable by the following step:

*backward variable*:[5]

$$\beta_t(i) = P(o_{t+1} o_{t+2} \cdots o_T | q_t = S_i, \lambda).$$

$\beta_t(i)$ can be solved inductively:

1. initialization:

$$\beta_T(i) = 1, \quad 1 \le i \le N$$

   and
2. induction:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j),$$
$$1 \le t \le T-1, 1 \le i \le N.$$

4. Calculating new parameters of the HMM model by the following step:

*joint event*:[6]

$$\begin{aligned}\xi_t(i,j) &= P(q_t = S_i, q_{t+1} = S_j | O, \lambda) \\ &= \frac{\alpha_t(i) a_{ij} b_j(o_{t+1}) \beta_{t+1}(j)}{P(O|\lambda)},\end{aligned}$$

*state variable*:[7]

$$\begin{aligned}\gamma_t(i) &= P(q_t = S_i | O, \lambda) \\ &= \sum_{j=1}^{N} \xi_t(i,j),\end{aligned}$$

*parameter updating equations*:

1. state transition probability:

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}, \quad 1 \leq i \leq N, 1 \leq j \leq N, \quad (26)$$

2. symbol emission probability:

$$\bar{b}_j(k) = \frac{\sum_{t=1, o_t=v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)}, \quad 1 \leq j \leq N, 1 \leq k \leq M, \quad (27)$$

3. initial state probability:

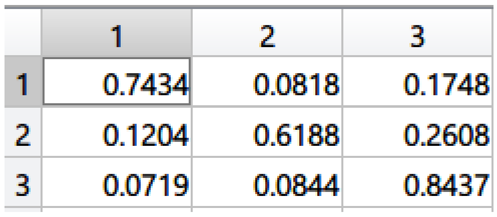
$$\bar{\pi}_i = \gamma_1(i), \quad 1 \leq i \leq N. \quad (28)$$

5. Calculate P(O| $\lambda$ )
6. Repeat step 2 to step 5 until P(O| $\lambda$ ) becomes stable
7. To calculate A and B, just use the average of all seq(n).A and seq(n).B

$2.3
Using the Viterbi algorithm implemented in Homework 2 to apply the HMM trained on the test sequences.

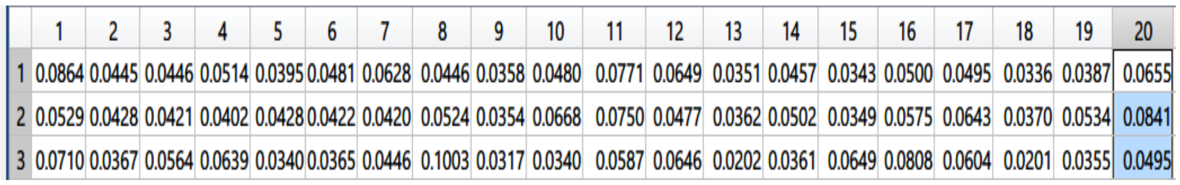
**Results**

$3.1 Maximum likelihood:
A=



| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 0.7434 | 0.0818 | 0.1748 |
| 2 | 0.1204 | 0.6188 | 0.2608 |
| 3 | 0.0719 | 0.0844 | 0.8437 |

B=

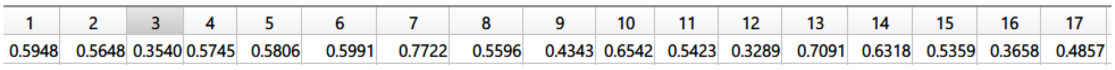


| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.0864 | 0.0445 | 0.0446 | 0.0514 | 0.0395 | 0.0481 | 0.0628 | 0.0446 | 0.0358 | 0.0480 | 0.0771 | 0.0649 | 0.0351 | 0.0457 | 0.0343 | 0.0500 | 0.0495 | 0.0336 | 0.0387 | 0.0655 |
| 2 | 0.0529 | 0.0428 | 0.0421 | 0.0402 | 0.0428 | 0.0422 | 0.0420 | 0.0524 | 0.0354 | 0.0668 | 0.0750 | 0.0477 | 0.0362 | 0.0502 | 0.0349 | 0.0575 | 0.0643 | 0.0370 | 0.0534 | 0.0841 |
| 3 | 0.0710 | 0.0367 | 0.0564 | 0.0639 | 0.0340 | 0.0365 | 0.0446 | 0.1003 | 0.0317 | 0.0340 | 0.0587 | 0.0646 | 0.0202 | 0.0361 | 0.0649 | 0.0808 | 0.0604 | 0.0201 | 0.0355 | 0.0495 |

Apply it on the test sequences.

1. Choose initial state probability pie=[0 0 1]:

   After calculating, for all the sequences, the accuracy rate is:

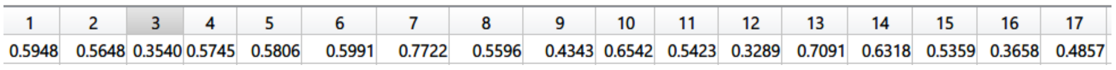


| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.5948 | 0.5648 | 0.3540 | 0.5745 | 0.5806 | 0.5991 | 0.7722 | 0.5596 | 0.4343 | 0.6542 | 0.5423 | 0.3289 | 0.7091 | 0.6318 | 0.5359 | 0.3658 | 0.4857 |

   Average accuracy rate is 0.5463

2. Choose initial state probability pie=[0.3333 0.3333 0.3333]

   After calculating, for all the sequences, the accuracy rate is:

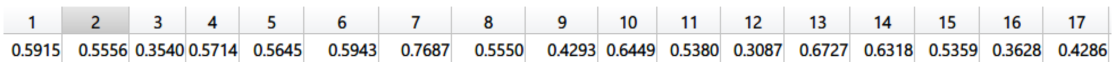


| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.5948 | 0.5648 | 0.3540 | 0.5745 | 0.5806 | 0.5991 | 0.7722 | 0.5596 | 0.4343 | 0.6542 | 0.5423 | 0.3289 | 0.7091 | 0.6318 | 0.5359 | 0.3658 | 0.4857 |

   Average accuracy rate is 0.5463

3. Choose initial state probability pie=[0.8 0.1 0.1]

   After calculating, for all the sequences, the accuracy rate is:

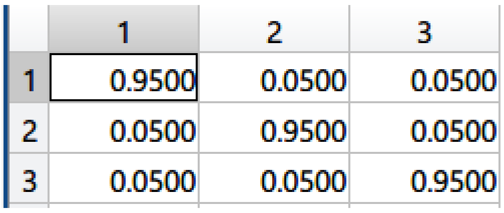


| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0.5915 | 0.5556 | 0.3540 | 0.5714 | 0.5645 | 0.5943 | 0.7687 | 0.5550 | 0.4293 | 0.6449 | 0.5380 | 0.3087 | 0.6727 | 0.6318 | 0.5359 | 0.3628 | 0.4286 |

   Average accuracy rate is 0.5368

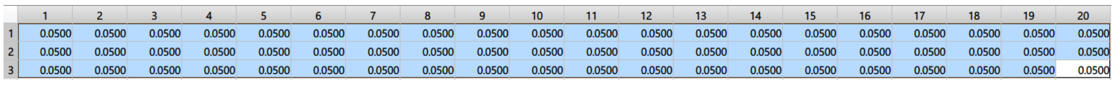
## $3.2 Training with Baum-Welch algorithm

1. Let A=



| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 0.9500 | 0.0500 | 0.0500 |
| 2 | 0.0500 | 0.9500 | 0.0500 |
| 3 | 0.0500 | 0.0500 | 0.9500 |

Let B=



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 |
| 2 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 |
| 3 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 | 0.0500 |

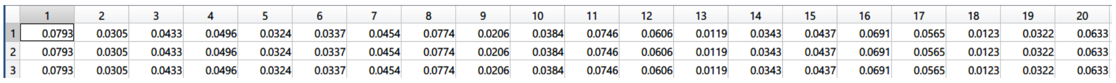
Let pie=[0.3333 0.3333 0.3333]

Then after training

A =

| | | |
|---|---|---|
| 0.8225 | 0.0433 | 0.0433 |
| 0.0433 | 0.8225 | 0.0433 |
| 0.0433 | 0.0433 | 0.8225 |

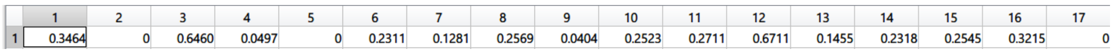
B=



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.0793 | 0.0305 | 0.0433 | 0.0496 | 0.0324 | 0.0337 | 0.0454 | 0.0774 | 0.0206 | 0.0384 | 0.0746 | 0.0606 | 0.0119 | 0.0343 | 0.0437 | 0.0691 | 0.0565 | 0.0123 | 0.0322 | 0.0633 |
| 2 | 0.0793 | 0.0305 | 0.0433 | 0.0496 | 0.0324 | 0.0337 | 0.0454 | 0.0774 | 0.0206 | 0.0384 | 0.0746 | 0.0606 | 0.0119 | 0.0343 | 0.0437 | 0.0691 | 0.0565 | 0.0123 | 0.0322 | 0.0633 |
| 3 | 0.0793 | 0.0305 | 0.0433 | 0.0496 | 0.0324 | 0.0337 | 0.0454 | 0.0774 | 0.0206 | 0.0384 | 0.0746 | 0.0606 | 0.0119 | 0.0343 | 0.0437 | 0.0691 | 0.0565 | 0.0123 | 0.0322 | 0.0633 |

Apply it on test sequences,

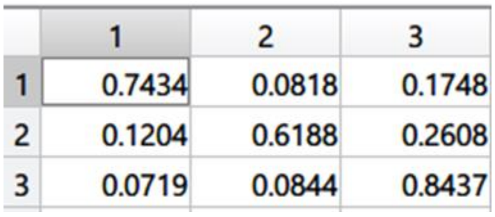
Accuracy rate for the 17 sequences are



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.3464 | 0 | 0.6460 | 0.0497 | 0 | 0.2311 | 0.1281 | 0.2569 | 0.0404 | 0.2523 | 0.2711 | 0.6711 | 0.1455 | 0.2318 | 0.2545 | 0.3215 | 0 |

Average accuracy rate is 0.2263

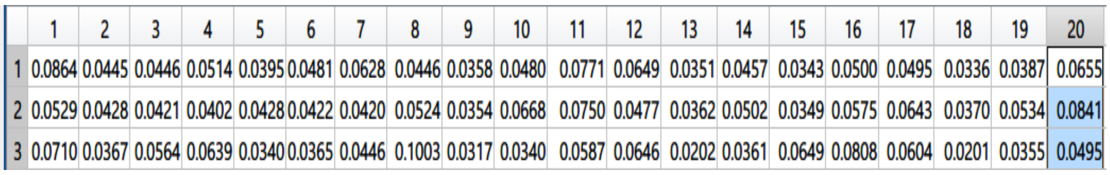
2. Let A=



| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 0.7434 | 0.0818 | 0.1748 |
| 2 | 0.1204 | 0.6188 | 0.2608 |
| 3 | 0.0719 | 0.0844 | 0.8437 |

(the same with A in ML training)

Let B=



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.0864 | 0.0445 | 0.0446 | 0.0514 | 0.0395 | 0.0481 | 0.0628 | 0.0446 | 0.0358 | 0.0480 | 0.0771 | 0.0649 | 0.0351 | 0.0457 | 0.0343 | 0.0500 | 0.0495 | 0.0336 | 0.0387 | 0.0655 |
| 2 | 0.0529 | 0.0428 | 0.0421 | 0.0402 | 0.0428 | 0.0422 | 0.0420 | 0.0524 | 0.0354 | 0.0668 | 0.0750 | 0.0477 | 0.0362 | 0.0502 | 0.0349 | 0.0575 | 0.0643 | 0.0370 | 0.0534 | 0.0841 |
| 3 | 0.0710 | 0.0367 | 0.0564 | 0.0639 | 0.0340 | 0.0365 | 0.0446 | 0.1003 | 0.0317 | 0.0340 | 0.0587 | 0.0646 | 0.0202 | 0.0361 | 0.0649 | 0.0808 | 0.0604 | 0.0201 | 0.0355 | 0.0495 |

(then same with A in ML training)

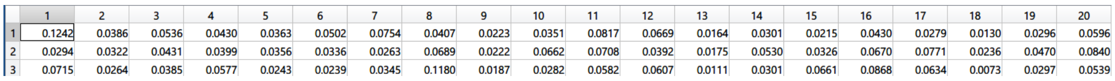
Let pie=[0 0 1]

Then after training:

A =

| | | |
|---|---|---|
| 0.8029 | 0.0437 | 0.0625 |
| 0.0779 | 0.7516 | 0.0796 |
| 0.0415 | 0.0702 | 0.7974 |

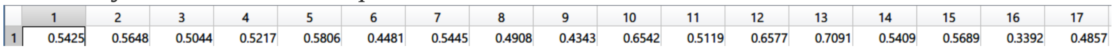
B=



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.1242 | 0.0386 | 0.0536 | 0.0430 | 0.0363 | 0.0502 | 0.0754 | 0.0407 | 0.0223 | 0.0351 | 0.0817 | 0.0669 | 0.0164 | 0.0301 | 0.0215 | 0.0430 | 0.0279 | 0.0130 | 0.0296 | 0.0596 |
| 2 | 0.0294 | 0.0322 | 0.0431 | 0.0399 | 0.0356 | 0.0336 | 0.0263 | 0.0689 | 0.0222 | 0.0662 | 0.0708 | 0.0392 | 0.0175 | 0.0530 | 0.0326 | 0.0670 | 0.0771 | 0.0236 | 0.0470 | 0.0840 |
| 3 | 0.0715 | 0.0264 | 0.0385 | 0.0577 | 0.0243 | 0.0239 | 0.0345 | 0.1180 | 0.0187 | 0.0282 | 0.0582 | 0.0607 | 0.0111 | 0.0301 | 0.0661 | 0.0868 | 0.0634 | 0.0073 | 0.0297 | 0.0539 |

Applying on the test sequences

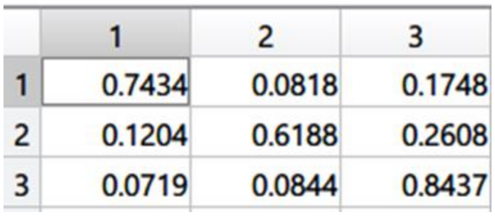
Accuracy rate for the 17 sequences are



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.5425 | 0.5648 | 0.5044 | 0.5217 | 0.5806 | 0.4481 | 0.5445 | 0.4908 | 0.4343 | 0.6542 | 0.5119 | 0.6577 | 0.7091 | 0.5409 | 0.5689 | 0.3392 | 0.4857 |

Average accuracy rate is 0.5353

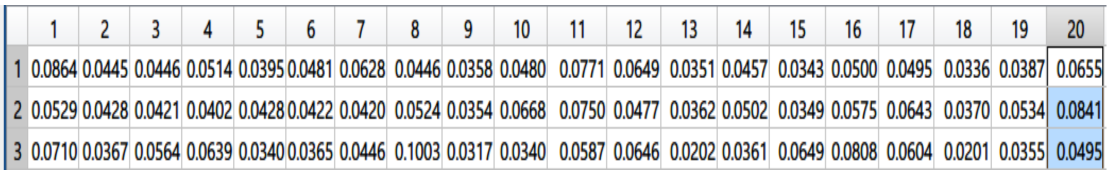
3. Let A=



| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 0.7434 | 0.0818 | 0.1748 |
| 2 | 0.1204 | 0.6188 | 0.2608 |
| 3 | 0.0719 | 0.0844 | 0.8437 |

(the same with A in ML training)

Let B=

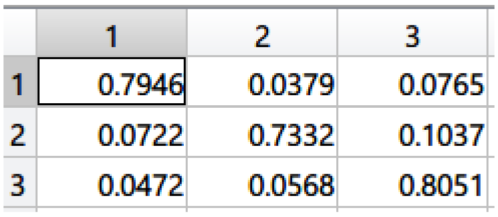


| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.0864 | 0.0445 | 0.0446 | 0.0514 | 0.0395 | 0.0481 | 0.0628 | 0.0446 | 0.0358 | 0.0480 | 0.0771 | 0.0649 | 0.0351 | 0.0457 | 0.0343 | 0.0500 | 0.0495 | 0.0336 | 0.0387 | 0.0655 |
| 2 | 0.0529 | 0.0428 | 0.0421 | 0.0402 | 0.0428 | 0.0422 | 0.0420 | 0.0524 | 0.0354 | 0.0668 | 0.0750 | 0.0477 | 0.0362 | 0.0502 | 0.0349 | 0.0575 | 0.0643 | 0.0370 | 0.0534 | 0.0841 |
| 3 | 0.0710 | 0.0367 | 0.0564 | 0.0639 | 0.0340 | 0.0365 | 0.0446 | 0.1003 | 0.0317 | 0.0340 | 0.0587 | 0.0646 | 0.0202 | 0.0361 | 0.0649 | 0.0808 | 0.0604 | 0.0201 | 0.0355 | 0.0495 |

(then same with A in ML training)

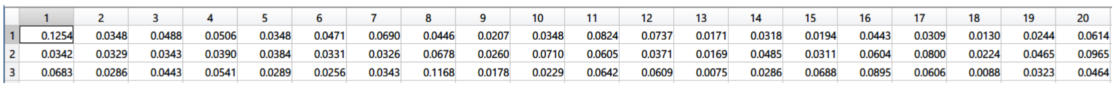
Let pie=[0.3333 0.3333 0.3333]

Then after training:
A=



| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 0.7946 | 0.0379 | 0.0765 |
| 2 | 0.0722 | 0.7332 | 0.1037 |
| 3 | 0.0472 | 0.0568 | 0.8051 |

B=

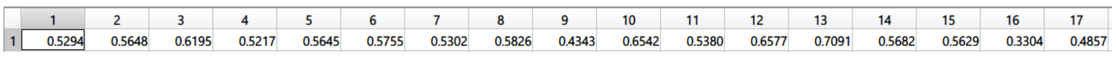


| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.1254 | 0.0348 | 0.0488 | 0.0506 | 0.0348 | 0.0471 | 0.0690 | 0.0446 | 0.0207 | 0.0348 |
| 2 | 0.0342 | 0.0329 | 0.0343 | 0.0390 | 0.0384 | 0.0331 | 0.0326 | 0.0678 | 0.0260 | 0.0710 |
| 3 | 0.0683 | 0.0286 | 0.0443 | 0.0541 | 0.0289 | 0.0256 | 0.0343 | 0.1168 | 0.0178 | 0.0229 |

| | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.0824 | 0.0737 | 0.0171 | 0.0318 | 0.0194 | 0.0443 | 0.0309 | 0.0130 | 0.0244 | 0.0614 |
| 2 | 0.0605 | 0.0371 | 0.0169 | 0.0485 | 0.0311 | 0.0604 | 0.0800 | 0.0224 | 0.0465 | 0.0965 |
| 3 | 0.0642 | 0.0609 | 0.0075 | 0.0286 | 0.0688 | 0.0895 | 0.0606 | 0.0088 | 0.0323 | 0.0464 |

Applying it on the test sequences:

Accuracy rate for the 17 sequences are



| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.5294 | 0.5648 | 0.6195 | 0.5217 | 0.5645 | 0.5755 | 0.5302 | 0.5826 | 0.4343 | 0.6542 | 0.5380 | 0.6577 | 0.7091 | 0.5682 | 0.5629 | 0.3304 | 0.4857 |

Average accuracy rate is 0.5546

**Discussion:**

1. We can see when using the HMM model on the test sequences, the general accuracy results are not ideal. This could be the result of non-typical training samples or non-typical test samples; in addition, the difference between our assumption- protein secondary structures conform a HMM model, and the reality.
2. Initial state probabilities distribution could have an influence on the result. From $3.1, we can see when using different pie in the model, the result could be different.
3. When training with Baum-Welch algorithm, the result could be greatly affected by the A, B and pie we choose at first, thus the accuracy rate for training with Baum-Welch could be better or worse than the Maximum likelihood training. This is because Baum-Welch doesn't yield a global optimal result, but a local optimal result, so it is decided by the initial parameters we choose. When the parameters chose are not reasonable, the result could be really bad.

4. From $3.2.2 and $3.2.3, we can see that when using the Maximum likelihood training result as the input for Baum-Welch training, it could yield a similar or better result. So it is a good idea to combine the two methods together to find a better result, when the states of the training samples are known to us.