(1) Did you *receive* any help whatsoever from anyone in solving this assignment? Yes / No.

If you answered 'yes', give full details: ______________ (e.g. "Jane explained to me what is asked in Question 3.4")

(2) Did you *give* any help whatsoever to anyone in solving this assignment? Yes / No.

If you answered 'yes', give full details: ______________ (e.g. "I pointed Joe to section 2.3 since he didn't know how to proceed with Question 2".

Collaboration without full disclosure will be handled severely, in compliance with CMU's Policy on Cheating and Plagiarism.

6

(a)

(b)

For result6b1, Segmentation accuracy: 0.740384615384615 [308/416]

For result6b2, Segmentation accuracy: 0.567307692307692 [236/416]

Result6b1 has a better accuracy. Because in result6b1, we are using the same sequence for training and testing, and for result6b2 we are using different sequence for training and testing. The performance is always better when the testing data has same or similar distribution with the training data.

(c)

Yes they are really similar. And yes it's expected. Because the model we get from 6(a) is the result trained by taggedsampleseq1. By examining this sequence we can see it starts with state '0' and end with state '1', also it can be seen from the sequence that in state '0' it's overwhelming likely to generate output a while in state '1' it's far more likely to generate output b. All this is consistent with the information given by samplemod2.

One of the difference is that in the model get from 6(a), the possibility that state "1" generates 'a' is higher than that in samplemod2. And the transition possibility from state 0 to state 1 is higher.

7

(a)

(b)

Yes the learnt model get some pattern in sampleseq2, but sometimes it gives counterpart result. The model get every time might depend. For example, the initial state might be 0 sometime while

other time it could be 1. So the output probability also varies. Because the training is unsupervised, it’s not guaranteed which state might generate which output. Although we can predict the likelihood and probabilities, the relation between 0/1 and a/b might change over time. Also it applies to the initial state. It could be either 1 or 0.
Choose the bestmod7b, compare it with samplemod2. They are extremely similar. Both starts with state 0; end with state 1, more likely to generate a in state 0, more likely to generate b in state 1.
Use bestmod7b to decode sampleseq2, the result is the same with the tags from 5(a).

(c)

```
Segmentation accuracy: 0.675480769230769 [281/416]
Segmentation accuracy: 0.524038461538462 [218/416]
Segmentation accuracy: 0.675480769230769 [281/416]
Segmentation accuracy: 0.516826923076923 [215/416]
Segmentation accuracy: 0.524038461538462 [218/416]
Segmentation accuracy: 0.675480769230769 [281/416]
Segmentation accuracy: 0.524038461538462 [218/416]
Segmentation accuracy: 0.675480769230769 [281/416]
Segmentation accuracy: 0.675480769230769 [281/416]
Segmentation accuracy: 0.545673076923077 [227/416]
```

No, the likelihood and model and accuracy vary each time.
The best accuracy is 0.67548, it’s greater than 0.5216.
The best accuracy is less than result6b1 and greater than result6b2.
The segmentation provided by best model somehow assemble the true tag, but makes mistakes sometimes, when the sequence is in one state for a short period, and then switch to another state, the trained model tend to ignore the switch, and sometimes there is latency reflecting that switch. However when the system stayed in one state for a longer period of time, the model tend to capture the true tag of the symbol better. However there are also some inaccuracies when it comes to capture the boundary of state switch. On average it performs better than just tagging all output as one state.