DD2380 AI Assignment 1

Rohit Kini
rrkini@kth.se

Neil Pradhan
npradhan@kth.se

1. **This problem can be formulated in matrix form. Please specify the initial probability vector π, the transition probability matrix A and the observation probability matrix B.**
   A = $\begin{pmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{pmatrix}$ **B** = $\begin{pmatrix} 0.9 & 0.1 \\ 0.5 & 0.5 \end{pmatrix}$ π = $(0.5 \quad 0.5)$

2. **What is the result of this operation?**
   The result of this operation is we get **P(**O_t**|A,P(**X_t**))**.

3. **What is the result of this operation?**
   The result of this operation is we get **P(**O_t**|A,B,P(**X_t**))**.

4. **Why is it valid to substitute $O_{1:t}$ = $o_{1:t}$ with O_t = o_t when we condition on the state X_t = x_i ?**
   It is valid to do as the current observation is only dependent on current state. It is independent of other observations and states. Thus we can neglect those and substitute $O_{1:t}$ = $o_{1:t}$ with O_t = o_t when we condition on the state X_t = x_i.

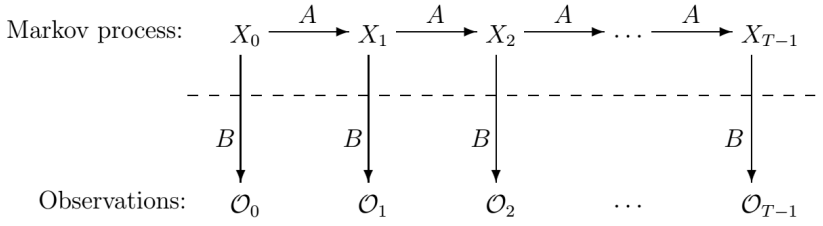

Figure 1: Hidden Markov Model [1]

5. **How many values are stored in the matrices δ and δ^{idx} respectively?**
   For every observation sequence, we have N(Dimension of transition matrix) values stored in δ and δ^{idx} respectively.

6. **Why we do we need to divide by the sum over the final α values for the di-gamma function?**
   The Forward- Backward algorithm for the HMM, is followed by smoothening and we divide by the total probability of the observation sequence to normalize this value and extract only the probability that $X_t = x_i$ where x_i is that particular state.[2]

32nd Conference on Neural Information Processing Systems (NeurIPS 2018), Montréal, Canada.

7. **Train an HMM with the same parameter dimensions as above, i.e. A should be a 3 times 3 matrix, etc. Initialize your algorithm with the following matrices:**

$$\mathbf{A} = \begin{pmatrix} 0.54 & 0.26 & 0.20 \\ 0.19 & 0.53 & 0.28 \\ 0.22 & 0.18 & 0.6 \end{pmatrix} \mathbf{B} = \begin{pmatrix} 0.5 & 0.2 & 0.11 & 0.19 \\ 0.22 & 0.28 & 0.23 & 0.27 \\ 0.19 & 0.21 & 0.15 & 0.45 \end{pmatrix} \pi = (0.3 \quad 0.2 \quad 0.5)$$

Does the algorithm converge? How many observations do you need for the algorithm to converge? How can you define convergence?

Yes the algorithm converges and it will always do as the while loop is designed to run certain amount of iterations, thus it will always converge to some value. The important part is how we define convergence, if we define convergence as algorithm converging to certain value then it does, but it would be of no use as we wont be able to define the optimal solution of the problem. Thus we will define convergence as on how close we are close to our target estimation of λ (A, B π). If we don't have target values for parameters as we do in this case. We can divide current observation sequence in two parts one for learning and other for testing.

8. **Train an HMM with the same parameter dimensions as above, i.e. A is a 3x3 matrix etc. The initialization is left up to you.**
How close do you get to the parameters above, i.e. how close do you get to the generating parameters in Eq. 3.1? What is the problem when it comes to estimating the distance between these matrices? How can you solve these issues?

After Training through the HMM with the provided file and provided initially parameters we get the following result :

$$\mathbf{A} = \begin{pmatrix} 0.688763 & 0.0665491 & 0.244687 \\ 0.149257 & 0.68139 & 0.169353 \\ 0.136962 & 0.25312 & 0.609918 \end{pmatrix}$$

$$\mathbf{B} = \begin{pmatrix} 0.688917 & 0.161974 & 0.0913179 & 0.057791 \\ 0.046281 & 0.467306 & 0.313733 & 0.17268 \\ 0.0265444 & 0.111477 & 0.292561 & 0.569418 \end{pmatrix}$$

π = $(1 \quad 6.51674e-26 \quad 1.44032e-22)$

As we can see the parameters that we get are very close to the original matrix by which this sequence is generated.

9. **Train an HMM with different numbers of hidden states.**
What happens if you use more or less than 3 hidden states? Why?
Are three hidden states and four observations the best choice? If not, why? How can you determine the optimal setting? How does this depend on the amount of data you have?

The best way to select a good model from a set of models of increasing states is the one that maximises the likelihood function. But naturally increasing the hidden states will give a better model but there is a chance of over fitting, hence we can use the Bayesian information criterion for selecting the best model.

10. **Initialize your Baum-Welch algorithm with a uniform distribution. How does this effect the learning?**
Initialize your Baum-Welch algorithm with a diagonal A matrix and π = [0,0,1]. How does this effect the learning?
Initialize your Baum-Welch algorithm with a matrices that are close to the solution. How does this effect the learning?

    - If we initialize the Baum-Welch algorithm with uniform distribution then the values will not be able to get out of local maxima and the method will not converge. So it is better to initialize with random noise in uniform distribution, or normal distribution (with mean as global mean and variance as global variance).
    - After initializing as said we get infeasible solution. Suggesting probabilities turn negative or greater than 1.
    - For this case we took the values we got in question 8 and initialize the algorithm with those values as they were near to the target. The result was again close to the target but not with certain amount of error.

    $$\mathbf{A} = \begin{pmatrix} 0.703261 & 0.0258575 & 0.270882 \\ 0.11986 & 0.737934 & 0.142206 \\ 0.165044 & 0.285026 & 0.54993 \end{pmatrix}$$

    $$\mathbf{B} = \begin{pmatrix} 0.676385 & 0.196548 & 0.0890651 & 0.0380024 \\ 0.0608127 & 0.457246 & 0.289646 & 0.192295 \\ 0.00660545 & 0.0535848 & 0.325779 & 0.61403 \end{pmatrix}$$

    $$\pi = \begin{pmatrix} 1 & 3.18438e-59 & 3.08931e-63 \end{pmatrix}$$

    Generally there is a possibility to get a good result if we initialize $/pi/$ and A uniformly for Ergodic models. And if the model is left-right then we should initialize $/pi/$ as 1 for first state and zero for consecutive ones and A should be preferably a diagonal matrix. We can also use clustering algorithms for correct initialization.

1 References

1. Stamp, Mark. A Revealing Introduction to Hidden Markov Models 1 A Simple Example. 2018.
2. Wikipedia Contributors. "Forward–Backward Algorithm." Wikipedia, Wikimedia Foundation, 22 Feb. 2019, en.wikipedia.org/wiki/Forward