Assignment 1: HMM

DD2380 Artificial Intelligence

Q1: 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.

$$\pi = \begin{bmatrix} 0.5 & 0.5 \end{bmatrix} \qquad A = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix} \qquad B = \begin{bmatrix} 0.9 & 0.1 \\ 0.5 & 0.5 \end{bmatrix}$$

Q2: What is the result of this operation?

r is the probability distribution in the next time step

$$r = \pi A = \begin{bmatrix} 0.5 & 0.5 \end{bmatrix} \cdot \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix} = \begin{bmatrix} 0.5 & 0.5 \end{bmatrix}$$

Q3: What is the result of this operation?

We multiply the r obtained last question with the B matrix. The result is the probability of the observation in the next time step.

$$rB = \begin{bmatrix} 0.5 & 0.5 \end{bmatrix} \cdot \begin{bmatrix} 0.9 & 0.1 \\ 0.5 & 0.5 \end{bmatrix} = \begin{bmatrix} 0.7 & 0.3 \end{bmatrix}$$

Q4: Why is it valid to substitute $O_{1:t}=o_{1:t}$ with $O_t=o_t$ when we condition on the state $X_t=x_i$?

The condition is valid because Ot is conditionally independent of O1:t given Xt. That means, that knowing Ot and the state Xt, Ot-1 (or O1:t) will not give more information.

Q5: How many values are stored in the matrices δ and δ^{idx} respectively

The matrix δ has T rows (length of the sequences of observations) and as many columns as the number of states.

The matrix δ^{idx} has T-1 rows (length of the sequences of observations - 1) and as many columns as the number of states.

Q6: Why we do we need to divide by the sum over the final α values for the di-gamma function?

The di-gamma is defined by: $\gamma_t(i,j) = P(\mathbf{X}_t = \mathbf{x}_i, \mathbf{X}_{t+1} = \mathbf{x}_j | \mathbf{O}_{1:T} = \mathbf{o}_{1:T})$ applying the product rule we obtained:

$$\gamma_t(i,j) = p(X_t = i, X_{t+1} = j \mid O_{1:T}, \lambda) = \{productrule\}$$
$$= \frac{p(X_t = i, X_{t+1} = j, O_{1:T} \mid \lambda)}{p(O_{1:T} \mid \lambda)} = \frac{p(X_t = i, X_{t+1} = j, O_{1:t}, O_{t+1:T} \mid \lambda)}{p(O_{1:T} \mid \lambda)}$$

and being:

$$P(\mathbf{O}_{1:T} = \mathbf{o}_{1:T}) = \sum_{j=1}^{N} P(\mathbf{O}_{1:T} = \mathbf{o}_{1:T}, \mathbf{X}_T = \mathbf{x}_j)$$
$$= \sum_{j=1}^{N} \alpha_T(j)$$

Applying the definition of conditional probability, this equals the joint probability divided by the probability of the observations.

alpha_T(i) is probability of having seen all the observations and being in state i. Summing over i we get the probability of observations.

Question 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:

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

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

For both sequences it converges: T= 1000 converge after 2089 iterations and T = 10000 after 14850 iterations.

The algorithm converges when log(P(O|model)) decreases

```
converge!!!  2089
3 3 0.6964764072237691 0.013355520467963426 0.2901680723082681 0.10146484237721358 0.8120129780896936 0.0865221795330935
7 0.19211580982560375 0.3012794969185915 0.5066046932558034
***************
3 4 0.6887972824578699 0.22515641843057055 0.07537016018577528 0.0106761389257843 0.06786807834759359 0.4120667550524721
4 0.28139193370751686 0.23867323289241826 4.0752407194770376e-48 5.853711322911404e-13 0.35330159172209175 0.64669840827
73223
***************
[[0.9999999999999991, 0.0, 0.0]]
converge!!!  14850
3 3 0.6942811339673699 0.04489865023216916 0.26082021580046066 0.11766839530079967 0.7460771058286327 0.1362544988705687
9 0.15418587384984364 0.2566938178518232 0.5891203082983326
***************
3 4 0.7099994962398879 0.18640862145684559 0.10359164843695491 2.3386631346917574e-07 0.09881201266130091 0.421125255936
0722 0.3121739269713422 0.16788880443128756 0.03211319660009302 0.171325074506396 0.18662833974475376 0.6099333891487566
***************
[[0.0, 1.0000000000000975, 5e-323]]
```

Question 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?

using a random initialization as:

$$a_{ij} \approx 1/N$$
$$b_j(k) \approx 1/M$$
$$\pi_i \approx 1/N$$

We see that we get the correct values of the matrix but with permuted states. To calculate the distance between the matrix we cannot use for example the euclidean distance. We have to use a measure that stays invariant under state permutation

```
A
**********
[[0.8120130752294792, 0.08652202119337503, 0.10146490357714538],
 [0.3012797495265254, 0.5066047634621015, 0.19211548701137243],
 [0.013355490467541739, 0.2901677310334285, 0.6964767784990303]]
B
**********
[[0.0678681327908323,
  0.4120666948025931,
  0.2813918968992333,
  0.2386732755073412],
 [3.0242224802908594e-44,
  1.1120779895959382e-12,
  0.35330164535948466,
  0.6466983546394034],
 [0.6887969727676481,
  0.22515624245477645,
  0.07537028227156276,
  0.010676502506013406]]
PI
**********
[[0.0, 0.0, 1.000000000000009]]
```

Question 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?

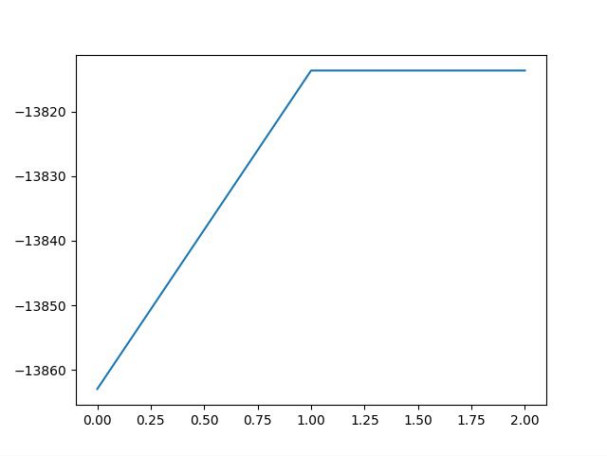
- Using more hidden states we obtain higher likelihood but we need more computer power
- Using less hidden states we obtain smaller likelihood, we need less computer power
- The optimal setting will depend of each problem and the amount of data available. We can measure this using BIC (Bayesian information criterion): It has into account the highest likelihood but it penalizes complexity (increased number of states).
- In the case of the fishing derby, would be the minimum number of hidden states that make us obtain the highest score

Question 10 Initialize your Baum-Welch algorithm with a uniform distribution. How does this affect the learning?

Initialize your Baum-Welch algorithm with a diagonal **A** matrix and $\pi = [0, 0, 1]$. How does this affect the learning?

Initialize your Baum-Welch algorithm with a matrices that are close to the solution. How does this affect the learning?

The Uniform distribution converge after a very few iterations (3 iterations). We are in a local maximum. After 1 iteration the logprob doesn't change. The Matrix A and pi barely change at all and B change a little more +-0.2




```
A
************
[[0.33333333333334125, 0.33333333333334125, 0.33333333333334125],
 [0.33333333333334125, 0.33333333333334125, 0.33333333333334125],
 [0.33333333333334125, 0.33333333333334125, 0.33333333333334125]]
B
************
[[0.24300000000000088,
  0.24800000000000083,
  0.23600000000000096,
  0.27300000000000006],
 [0.24300000000000088,
  0.24800000000000083,
  0.23600000000000096,
  0.27300000000000006],
 [0.24300000000000088,
  0.24800000000000083,
  0.23600000000000096,
  0.27300000000000006]]
PI
************
[[0.3333333333333331, 0.3333333333333331, 0.3333333333333331]]
```

When A is a diagonal matrix, it doesn’t converge (it stops after 3 iterations).

Pi doesn’t change and A is all rows 0 except the last one that is the same as PI. In B only the last row is updated.

```
converge!!!  3
A
***********
[[0, 0, 0], [0, 0, 0], [0.0, 0.0, 1.0]]
B
***********
[[0, 0, 0, 0], [0, 0, 0, 0], [0.2642, 0.2699, 0.2085, 0.2574]]
PI
***********
[[0.0, 0.0, 1.0]]
```

It converges faster