Assignment 4

EQ2341 Pattern Recognition and Machine Learning

Names
Oriol Closa and Clara Escorihuela Altaba

Date
January 23, 2022

1 Introduction

This assignment aims at implementing and verifying the backward algorithm. During the estimation of a state sequence, it is important to consider the enitre observable sequence, from $t = 0$ to $t = T$. The forward algorithm allows us to estimate the conditional probabilities until time t. However, it is possible that the probability a state i differs until time t and until time T. Here is where the backward algorithm takes relevance.

2 Code implementation

2.1 Verifying the backward algorithm

The following section shows the implementation of the backward algorithm based on the methodology exposed by Mark Stamp in «A Revelion Introduction to Hidden Markov Models»[1] along with the theory introduced in the course literature[2]. The barckward or β-pass algorithm is an iterative algorithm very similar to the Forward Algorithm, but in this case, the starting points is in the end and we work back toward the beginning.

In order to do that we must define the $beta$ variable.

For $t = 0, 1, ...T - 1$ and $i = 0, 1, ..., N - 1$ define.

$$\beta_t(i) = P(O_{t+1}, O_{t+2}, ..., O_{T-1}|x_t = q_i, \lambda) \tag{1}$$

1. For $i = 0, 1, ..., N - 1$.

$$\beta_t(i) = 1 \tag{2}$$

2. For $T - 2, T - 3, ..., 0$ and $i = 0, 1, ..., N - 1$ compute.

$$\beta_t(i) = \left(\sum_{j=0}^{N-1} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) \right) \tag{3}$$

2.2 Finite and infinite HMM testing

The following lines describe the test experiments in order to verify the behaviour of the β-pass algorithm. In particular we have tested with two different HMM models, with finite and infinite durations, with the expectation of obtaining the same results.

The finite HMM example has been acquired from the previous assignment, while the infinite example has been acquired from page 121 of the class literature belonging to the problems section right after the backward explanation. Figure 1 represents the code for the `main` function where both `forward` and `backward` algorithms are called due to the latter needing the scaling values from the first.

```
q = np.array([1.0, 0.0])
a = np.array([np.array([0.9, 0.1, 0.0]), np.array([0.0, 0.9, 0.1])])
obs = np.array([-0.2, 2.6, 1.3])

g1 = GaussD(means=[0.0], stdevs=[1.0])
g2 = GaussD(means=[3.0], stdevs=[2.0])
prob_obs = [g.prob(obs) for g in [g1, g2]]
mc = MarkovChain(initial_prob=q, transition_prob=a, dist_obs=prob_obs, obs=obs, scaling=True)
h = HMM(mc, [g1, g2])
a_ts, c_ts = mc.forward()
b_ts = mc.backward()
log_prob = h.logprob(c_ts)
print("A_hat =", a_ts, "C_ts =", c_ts, "B_hat =", b_ts)
print("LogProbability = ", log_prob)

q = np.array([1.0, 0.0, 0.0])
a = np.array([np.array([0.3, 0.7, 0.0]), np.array([0.0, 0.5, 0.5]), np.array([0.0, 0.0, 1.0])])
obs = np.array([1, 2, 4, 4, 1])
g3 = GaussD(means=[6.0], stdevs=[3.0])
prob_obs = [g.prob(obs) for g in [g1, g2, g3]]

mc = MarkovChain(initial_prob=q, transition_prob=a, dist_obs=prob_obs, obs=obs, scaling=False)
h = HMM(mc, [g1, g2, g3])
a_ts, c_ts = mc.forward()
b_ts = mc.backward()
log_prob = h.logprob(c_ts)
print("A_hat =", a_ts, "C_ts =", c_ts, "B_hat =", b_ts)
print("LogProbability = ", log_prob)
```

Figure 1: Forward and backward calls with scaling of the factors.

2.2.1 Finite HMM

$$q = \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad A = \begin{pmatrix} 0,9 & 0,1 & 0 \\ 0 & 0,9 & 0,1 \end{pmatrix} \quad B = \begin{pmatrix} \mathcal{N}(0,\ 1) \\ \mathcal{N}(3,\ 2) \end{pmatrix} \tag{4}$$

$$c_{ts} = \begin{pmatrix} 1 & 0,1625 & 0,8256 & 0,0581 \end{pmatrix} \quad B_{hat} = \begin{pmatrix} 1 & 1,0389 & 0 \\ 8,4154 & 9,3504 & 2,0818 \end{pmatrix} \tag{5}$$

2.2.2 Infinite HMM

$$q = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \quad A = \begin{pmatrix} 0,3 & 0,7 & 0 \\ 0 & 0,5 & 0,5 \\ 0 & 0 & 1 \end{pmatrix} \quad B = \begin{pmatrix} \mathcal{N}(0,\ 1) \\ \mathcal{N}(3,\ 2) \\ \mathcal{N}(6,\ 3) \end{pmatrix} \tag{6}$$

$$\begin{aligned} c_{ts} &= \begin{pmatrix} 0,2419 & 0,1394 & 0,1392 & 0,1295 & 0,0529 & 0,7429 \end{pmatrix} \\ B_{hat} &= \begin{pmatrix} 4,1327 & 7,3471 & 8,1476 & 8,3188 & 0 \\ 3,0307 & 7,1496 & 8,3102 & 9,1991 & 12,7203 \\ 0,8601 & 3,8068 & 4,9843 & 6,5142 & 25,4406 \end{pmatrix} \end{aligned} \tag{7}$$

References

1. STAMP, Mark. A revealing introduction to hidden Markov models. In: Department of Computer Science San Jose State University, 2004, pp. 26–56.
2. LEIJON Arne Eje Henter, Gustav. *Pattern Recognition: Fundamental Theory and Exercise Problems*. Kungliga Tekniska Högskolan, 2015.