Pattern and Speech Recognition Tutorial
Exercise 11

Exercise 1 (3 points)

Consider a HMM with 3 states ($Q = \{1, 2, 3\}$) and 2 output symbols ($\Sigma = \{1, 2\}$), with the transition matrix:

$$A = \begin{pmatrix} 0.5 & 0 & 0 \\ 0.3 & 0.6 & 0 \\ 0.2 & 0.4 & 1 \end{pmatrix}$$

where $A_{i,j} = P(\pi_{t+1} = i \mid \pi_t = j)$. The emission matrix is (first row corresponds to 1, second row to 2):

$$B = \begin{pmatrix} 0.7 & 0.4 & 0.8 \\ 0.3 & 0.6 & 0.2 \end{pmatrix}$$

where $B_{i,j} = P(o_t = i \mid \pi_t = j)$. The initial probabilities are 0.9, 0.1, 0 for the three states respectively.
Given that the observed sequence is $o_1 = 1$, $o_2 = 2$, $o_3 = 1$, compute $P(o_1, o_2, o_3 | \pi_1 = i)$ for all states $i \in Q$ (by hand). Use the backward algorithm.

Exercise 2 (3 points)

Implement a function `forward(hmm,`$[o_0, \ldots, o_n]$`)` which uses the forward algorithm to determine the likelihood of emitting a sequence $o_0, \ldots, o_n \in \Sigma^{n+1}$ for a given HMM `hmm` using the forward algorithm. How can you use this function to verify your results from exercise 1? Explain.

Exercise 3 (4 points)

1. Write a function to randomly initialize the transition matrix and the emission matrix of a HMM.

2. Implement a function `viterbi_training(hmm,`$[o_0, \ldots, o_n]$`)` which uses the Viterbi training algorithm (slide 52) to iteratively update the transition and emission matrices based on the Viterbi path π^*. You function is supposed to stop when the Viterbi path no longer changes. You can reuse code from assignment sheet 10.

3. Consider a HMM with $Q = \{1, 2\}$, $\Sigma = \{a, b\}$, a uniform initial distribution, and $\mathbf{o} = aabaabaaabbababbba$. Apply Viterbi training with different initial parameters (use exercise 3.1). Experiment with different pseudocounts and report some of your results. For each iteration record the Viterbi path π^*, its corresponding probability, and the probability of emitting $\mathbf{o}$ (i.e. the result of the forward algorithm).

Submission architecture

You have to generate a **single ZIP file** respecting the following architecture:

```
tutorial1_<matriculation_nb1>_<matriculation_nb2>_<matriculation_nb3>
|
+--- source
|      |
|      +------- file 1
|      +------- file 2
|      +------- ...
+--- rapport.pdf
+--- README.txt
```

where

- **source** contains the source code of your project,
- **rapport.pdf** is the report where you present your solution with **the explanations (!)** and the plots,
- **README** which contains group member informations (name, matriculation numbers and emails) and a **clear** explanation about how to compile and run your source code

The ZIP filename has to be :

`tutorial11_<matriculation_nb1>_<matriculation_nb2>_<matriculation_nb3>.zip`

You have to choose between the following languages **python** or **matlab**. Other languages won't be accepted.

Some hints

We advice you to follow the following guidelines in order to avoid problems :

- Avoid building complex systems. The exercises are simple enough.
- Do not include any executables in your submission, as this will cause the e-mail server to reject it.

Grading

Send your assignment to the tutor who is responsible of your group:

- Gerrit Gromann `gerritgr@gmail.com`
- Sbastien Le Maguer `slemaguer@coli.uni-saarland.de`
- Kata Naszdi `b.naszadi@gmail.com`

The email subject should start with `[PSR TUTORIAL 11]`