Pattern Analysis Joint Meeting
Christian Riess

Worksheet 9: Monday June 21 / Friday June 25 2021

Please watch the video prior to the lecture, and think about the questions below. In the joint meeting, you will have 25 minutes time to discuss the questions with your group. Afterwards, we will jointly discuss your solution proposals.

You can print this sheet and use the space below for your notes.

Task 1: Optimization of HMMs

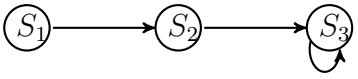
Two questions on the optimization:

(a) HMM training is only locally optimal. What can we do to try to avoid bad local optima?

(b) What is the HMM model selection problem, and how can we address it?

Task 2: "Super-Fast HMM"

Tony files a patent for his variation of the HMM to recognize speech: the "Blazing Fast HMM". It consists of N states. Any state can be starting state, and the transitions are shown here for $N = 3$:



(a) Does the proposed method deserve its name? What is the computational complexity?

(b) There are also some disadvantages of the proposed design. Which ones?

Task 1: Many algorithms only find local optima (k-means, GMMs, also HMMs, also (beyond PA) neural networks)

This means that the quality of the solution depends on the initialization of the parameters

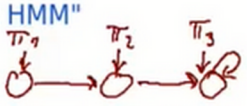
Let's do this → Hence, to increase chances to find a good solution, try different initializations, and either select the best performing model on a held-out validation set, or perform some form of model averaging

What is the HMM model selection problem?

- How many states are used?
- Connection structure of the HMM (left-right HMM, fully connected)

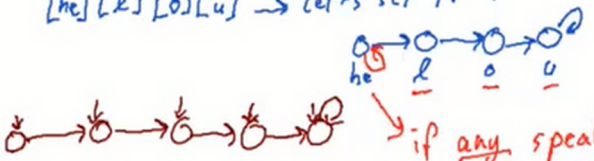
How to optimize these parameters? Cross-validation

Task 2: "Super-Fast HMM"



/hello/

[he] [l] [o] [u] → let's set N=4



if any speaker stretches the "he", then we certainly need a self-loop, otherwise the later states have to deal with the remaining time slices from "he"

a) How fast is it? (for N states)

b) What might be a disadvantage of this design?
⇒ for timing variations in speech, we do need these self-loops.

Q: What is the computational complexity of the forward algorithm?

$O(N^2T)$

↑ # states ↖ speech segment

Reason for N^2: when going from $t \to t+1$, we have to consider all outgoing states at t and all incoming edges at $t+1$

In the super-fast HMM, there is always just one outgoing and incoming state ⇒ the cost is $O(1 \cdot T) = O(T)$

⇒ Analogously, all other costs for state transitions vanish

⇒ This effectively removes the hidden state, the model falls back to a simple first-order Markov model.

Task 3: Small HMMs versus Large HMMs

Tony has a second idea, with the goal of saving on the total number of parameters. Instead of training a single HMM for each word, he could train one large HMM with different paths for different words. That way, he could share representations of identical phonemes in the states. To find out which word was spoken, he could use the Viterbi algorithm to reconstruct the path through the HMM.

What do you think about this proposal? What might be benefits, what might be challenges with it?