MLRD Supervision 3

Neelu Saraswatibhatla (srns2)

Markov assumption

The first-order Markov assumption is that the state at time t depends only on the state at time $t-1$, i.e. $P(X_t|X_{t-1}, X_{t-2}, \ldots, X_1) \approx P(X_t|X_{t-1})$, where X_t is the state at time t. This means that we don't need to keep lists of state probabilities at loads of different timesteps and calculate probabilities depending on states at many different timesteps, we just calculate probabilities based on the states at the previous timestep. This is known as the Limited Horizon as we can only "see" back 1 step, so we only have a "horizon" of 1 timestep.

The other assumption HMMs use is that of output independence. This is the assumption that the probability of the current output only depends on the current state, and on no previous or future states or outputs, i.e. $P(O_t|X_1, \ldots, X_t, \ldots, X_T, O_1, \ldots, O_t, \ldots, O_T) \approx P(O_t|X_t)$, where X_t and O_t are the state and observation respectively at time t, and T is the final timestep in the Markov chain. This means that we only need to look at the probabilities of states at the current timestep to determine emission probabilities, and don't need to look forward or back and previous or future states or observations.

HMM Artificial data

1.

2.

3.

Smoothing in HMMs

1. Add-one smoothing helps improve accuracy by preventing one probability in a product of probabilities being 0 from making the overall product 0, ignoring all the other probabilities in the product. However this means that the probability is made slightly less accurate. If we are sure that none of the constituent probabilities of a product will be 0 then there is no point in using add-one smoothing and it can be counterproductive.

2. The transition probabilites are the better candidate for smoothing. This is because some transitions will be 0 as proteins with amino acids inside the cell need to go through the membrane before they can go outside and vice versa, so the state transition probability straight from inside to outside or outside to inside is 0. Therefore we may want to smooth the transition probabilities so a transition probability of 0 doesn't make an entire product of probabilities 0.

Viterbi and Forward algorithm

1.

2.

Parts of Speech tagging with HMM

1. In X_a, the order is personal pronoun $\rightarrow$ auxiliary verb $\rightarrow$ verb, so 'can' is an auxiliary verb and 'fish' is a verb, so this sentence can be interpreted as "We are able to fish". In X_b the order is personal pronoun $\rightarrow$ verb $\rightarrow$ noun, so 'can' is a normal verb and not an auxiliary verb, and 'fish' is a noun, so this sentence means "We put fish in cans".

2.

$$P(X_a) = a_{03} \times a_{34} \times a_{41} \times a_{1f} = 0.60 \times 0.40 \times 0.74 \times 0.15 = 0.02664$$
$$P(X_b) = a_{03} \times a_{31} \times a_{12} \times a_{1f} = 0.60 \times 0.40 \times 0.63 \times 0.20 = 0.03024$$
$$P(O) =$$

3.

4.

5.

6.

Viterbi with higher order HMMs

1. If n is the number of states, we need to keep n states at each timestep that our HMM remembers, so for an N order HMM, we need to keep n^N states. Here, we are given that there are N states so for an N order HMM we'd need to keep N^N states.

2. For a first-order HMM, at each timestep the Viterbi algorithm loops through each state and looks at each previous state probability for each one, so for n states we do n^2 lookups, and if there are T timesteps, we do $O(n^2T)$ work. In a second order HMM, we still have t timesteps but at each timestep for each state, we look back to each of the n states, and then for each of those, look back again to each of the n states, getting a total of $O(n^3T)$ cost. This continues, and we add an extra n multiplier to our cost for each increase in order as each increase in order involves looking at each of the n steps again for each of the lookups the previous order had. Therefore for an N order HMM the asymptotic complexity of the Viterbi algorithm would be $O(n^{N+1}T)$. Here we are told that there are N states, so for an N order HMM the Viterbi algorithm's asymptotic complexity would be $O(N^{N+1}T)$.