

# 6.095/6.895 Recitation 4 - Notes

Pouya Kheradpour

September 30, 2005

For notational consistency, this recitation will attempt to use the same notation as Durbin.

## 1 HMM Basics

A model has the Markov property if the probability of all future events are independent of anything except the current state. For Hidden Markov Models, the future events include the next state and the emission at a state.

Formally, HMMs are specified completely by transition probabilities  $a_{kl} = P(\pi_i = l | \pi_{i-1} = k)$  and emission probabilities  $e_k(b) = P(x_i = b | \pi_i = k)$ . That is to say,  $a_{kl}$  gives the probability of going from state  $k$  to  $l$  and  $e_k(b)$  gives the probability of state  $k$  emitting character  $b$ . The states range from 1 to the number of states and  $a_{0l}$  is used to indicate the probability of starting in state  $l$  and  $a_{k0}$  is used to indicate the probability of ending in state  $k$ .

When modeling a system using an HMM, there are several computational issues we may want to address:

- Given a model, what is the probability of a specific state sequence and emission. This is simply the product of the emission probabilities and transition probabilities.
- **Viterbi algorithm:** Given a sequence of emissions and the model, what is the most likely sequence of states.
- **Forward algorithm:** Given a sequence of emissions and the model, what is the relative probability of a certain state being the final one.
- **Backward algorithm:** Given suffix of a sequence of emissions and the model, what is the relative probability of having a certain state preceding the sequence. Notice, this is not completely analogous to the Forward Algorithm.
- **Baum-Welch algorithm:** Given a state sequence and a number of states, what's the most likely model.

Below, we will discuss each of these algorithms in detail.

## 2 Viterbi algorithm

The Viterbi algorithm directly solves for the probability  $\max_{\pi} P(x, \pi)$ . where  $x$  refers to a emission sequence and  $\pi$  refers to a corresponding state sequence where the sequences have length  $L$ . We can then backtrack our result to find the specific  $\pi$  leading to the max. Notice, that  $P(x, \pi) = P(\pi|x)P(x) \propto P(\pi|x)$ , so the  $\pi$  used in the joint is the most likely state sequence given the emission sequence.

Define  $v_l(i)$  to be  $\max_{\pi_1, \dots, \pi_{i-1}} P(x_1, \dots, x_i, \pi_1, \dots, \pi_{i-1}, \pi_i = l)$ , or the joint probability of having the first  $i$  emissions along with the optimal sequence of states ending with  $l$ .

By the Markov property we have:

$$v_l(i) = e_l(x_i) \max_k (v_k(i-1) a_{kl})$$

which can be shown to be the optimal state by contradiction.

Finally, to obtain our original desired probability, we have

$$\max_{\pi} P(x, \pi) = \max_k (v_k(L) a_{k0})$$

where  $v_0(k) = 1$  when  $k = 0$  and 0 otherwise.

We simply backtrack the values of  $k$  leading to the max to obtain our state sequence. This recurrence will be solved using our friend, dynamic programming. Often we find the  $\log v_l(i)$  for simplify computation (additions are easier than multiplications) and to prevent underflow because the probabilities often become very small.

### 3 Forward algorithm

The forward algorithm is used to find the probability of getting a specific sequence of emissions. Let  $f_k(i) = P(x_1, \dots, x_i, \pi_i = k)$ . Then it follows that  $f_k(i) = e_l(x_i) \sum_k f_k(i-1) a_{kl}$ . We notice that this is essentially the Viterbi recurrence except with the max replaced by a sum. The initialization is the same and the termination is  $P(x) = \sum_k (f_k(L) a_{k0})$ .

### 4 Backward algorithm

The Backward algorithm solves for the probability of a sequence of emissions when the previous state is given. Let  $b_k(i) = P(x_{i+1}, \dots, x_L | \pi_i = k)$ . We can recursively write:  $b_k(i) = \sum_l a_{kl} e_l(x_{i+1}) b_l(i+1)$  with initialization  $b_k(L) = a_{k0}$  and termination  $P(x) = \sum_l a_{0l} e_l(x_1) b_l(1)$ .

Notice, by the definition of the Forward and Backward algorithms, we have  $P(x, \pi_i = k) = f_k(i) b_k(i)$ .

### 5 Baum-Welch algorithm

The Baum-Welch solves what may be the hardest of the HMM problems: learning the model given some output. We must provide the number of states to the model (more states can always fit the data better – we want to use the least number of states that adequately describe the model).

The general description of the algorithm is as follows:

1. Arbitrarily initialize  $a_{kl}$  and  $e_k(b)$ .
2. Calculate  $f_k(i)$  and  $b_k(i)$  for using the forward and backward algorithms.
3. Update *expected* counts for each transition and emission.
4. If there are multiple sequences, repeat the previous two steps for each sequence.
5. Compute new  $a_{kl}$  and  $e_k(b)$  from these counts.
6. Repeat from step 2 until some sort of stopping criteria (small change in model, maximum number of iterations, etc).

This algorithm is an example of expectation maximization and is guaranteed to reach a *local* maximum. By varying the arbitrary initialization step, we can attempt to find better models. Rather than using the expected counts and the forward and backward algorithm, we can use the Viterbi algorithm to obtain the most likely state sequence and obtain counts from it. This gives us Viterbi training, which is typically less used but may be better suited in certain situations.