CSE 527 Notes - Lecture 16

William Pentney

December 6, 2001

1 Viterbi algorithm, continued

Given a collection of states, a probability associated with each symbol for each state, and probabilities from moving between states, the Viterbi algorithm will calculate $V_i(k)$, or the probability that the most likely path ended at state k after emitting the first i symbols. Typically, this calculation is performed in logarithmic space, as the probabilities of following any given path can rapidly grow extremely small.

2 The Forward Algorithm

Say we wish to find the probability that a string x was emitted. This probability will be equal to $\sum_\pi P(x, \pi)$, or the probability of the string being emitted for each possible state path π. Summing over all possible choices of path is difficult, but it involves a fair amount of redundant calculation, and a more efficient dynamic programming algorithm, the Forward Algorithm, exists to exploit this.

2.1 The Algorithm

Define $f_k(i) = P(x_1, x_2, ...x_i, \pi_i = k)$, or the probability of emitting symbols x_1 through x_i in that order, ending in state π_i. Then define $e_l(x_i + 1)$ to be the probability of emitting symbol $x + 1$ from state l, and $a_{k,l}$ to be the probability of moving from state k to state l. Define $f_l(i + 1) = e_l(x_i + 1) \sum_k f_k(i) a_{k,l}$.

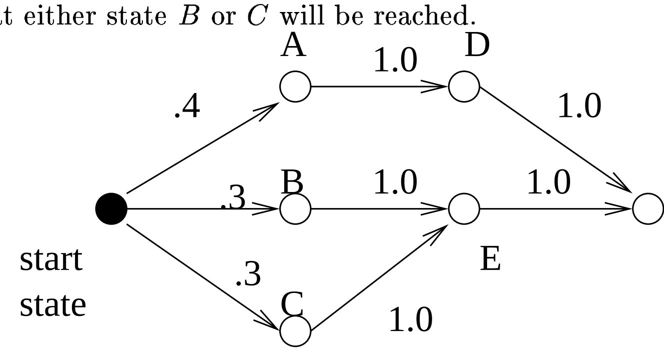
We begin with $f(s) = 1$, where s is the starting state, and $f(i) = 0$ for states $i \neq s$. (This assumes a known start state, of course; if the start state is unknown, the probabilities of the respective start states should be used as appropriate.) Then for a string $x = x_1, x_2, ..., x_n$, we can calculate $f_k(n)$ for each state k, and thus calculate the probability that x was emitted.

2.2 Running time

Given Q possible states, and L many symbols emitted, the algorithm will take $O(Q^2L)$ time, since there will be Q values to sum for each of Q stages at each stage, of which there are L many such stages.

3 Backward Algorithm

Say we want to calculate $P(\pi_i = k|x)$, or the probability that our state after i symbols is k given the observation of x. Consider the diagram below, for instance, which shows a series of states, each with an associated probability of movement from one state to another after the emission of a symbol. Assume all symbols are equally likely to be emitted. After the first symbol is emitted, the most likely state will be state A. However, after the second symbol is emitted, the most likely state will be state E, since states B and C will both always move to E after a symbol is emitted, and it is more likely that either state B or C will be reached.



To apply Bayes' rule, we calculate
$P(x, \pi_i = k) = P(x_1, ..., x_i, \pi_i = k)P(x_{i+1}, ..x_L|x_1, ..., x_i, \pi_i = k)$.
The multiplicand can be calculated by the Forward Algorithm.
$P(x, \pi_i = k) = f_k(i)P(x_{i+1}, ...x_L, \pi_i = k)$
The multiplier will be calculated using the Backward Algorithm.

3.1 The Algoritihm

Let $b_k(i) = \sum_l a_{k,l}e_l(x_{i+1})b_e(i+1)$, and let $b_k(L) = a_k$ for all k. Then $b_k(i+1)$ will give use the desired result above.

3.2 Running time

Like the Forward Algorithm, the Backward Algorithm operates in polynomial time, specifically in $O(Q^2L)$ time when given Q stages and L symbols.

4 Probability of Being Within a Set of States

We may wish to calculate the probability of being within a set of states at the ith symbol when x is emitted - for instance, if we are attempting to find whether we fall within a CpG island. Then we calculate $\sum_k g(k)P(\pi_i = k|x)$, where each k is a state in the set and $g(k)$ is the probability of being in state k.

5 Parameter Estimation

So far, we have assumed knowledge of the parameters involved so far - the probability of moving from one state to another, or of emitting a particular symbol, for instance. We shall now consider the problem of estimating these values giving training data, assuming a fixed architecture (i.e., the possible transitions of state are known).

Say we are given n training sequences $x^1, x^2, ...x^n$, each of some length. If we have $\pi_1, ...\pi_n$ state transition sequences corresponding to the outputs, the movement is straightforward - we simply calculate the number of times we have moved from one state to another, so $a_{k,l} = P$(count of $k \rightarrow l$ transitions)/(count of transitions from k total).

Normally, however, we don't have the state transition sequences, so we use algorithms to estimate them based on training data. One method of doing so is Viterbi training, in which we estimate π^*, the most likely path for a sequence, based on the training data.

6 The Baum-Welch Algorithm

The Baum-Welch Algorithm solves a special case of the problem solved by EM algorithms. It attempts to calculate $P(\pi_i = k, \pi_i + 1 = l|x, \theta)$, where θ represents estimates for the parameters. The algorithm calculates $w(x, \pi_i = k) = (f_k(i)a_{k,l}e_l(x_{i+1})b_l(i + 1))/P(x))$. The expected count of $k \rightarrow l$ transitions would be $\sum_{x^j}(1/P(x^j))$ times the numerator of the w value calculated above summed over all i, for all training sequences x^j.