Notes for CSE 527, Lecture 12 Nov. 6, 2006
Field Cady

Hidden Markov Models Again

Motivation: a base pair can either be in a CpG island or not. The probability of G following a C depends on whether they are in a CpG island or not.

Hidden Markov Model: set of states and transition probabilities between states. Each state emits a symbol. In this case, state={nucleotide (A C G or T), whether or not it's in a CpG island(+ or -) } and symbol emitted=nucleotide.

pi used to denote sequence of states, pi_i denotes individual state. x denotes sequence of emissions, x_i an individual letter.

$a_{lkl}=P(pi_i = l \mid pi_{i-1} = k)$	transition probabilities
$e_k(b)=P(k \text{ emitted} \mid \text{in state } k)$	emission probabilities

Viterbi Algorithm

given transition and emission probabilities and have observed sequence x of symbols, it finds the sequence of states pi^8 that maximizes P(x,pi). It's a dynamic programming algorithm (somewhat in the spirit of Smith-Waterman).

Definition: $V_l(i)$=probability of the most probable path (pi_1, ..., pi_l) that emits (x_1, ..., x_l) and ends in state l.

Recursion: $V_l(i+1)=e_l(x_{i+1})*max_k(V_k(i)*a_{kl})$

Make a table that for each symbol (nucleotide) has a list of the states of the system. Build from the start, filling the table according to the recursion step, and pick the best probability at the end. Of course, in reality use log-likelihood.

NOT A PERFECT ALGORITHM. For example, one state may be the most likely to occupy the ith position, but the single most probable path doesn't include it, and so it's ignored. We can account for this with...

Posterior Decoding

Forward Algorithm:
$f_k(i)=P(x_1,...,x_{i-1} \mid pi_i=k)$
Backward Algorithm:
For a given state/time, we want the total probability of all paths from it, w/ given

emissions, conditioned on state.
$b_k(i)=P(x_{i+1},...,x_n \mid pi_i=k)$

Now, to find the most likely single state for the ith position,
$P(pi_i=k \mid x) = P(x|pi_i=k)/P(x) = f_k*b_k/P(x)$

Choose $pi_i=k$ that maximizes maximize $P(pi_i=k \mid x)$. This is different from Viterbi! The path chosen this way might not even be allowed.

Back to bio: we use an algorithm to reconstruct the sequence of states, and wherever some of the states are CpG island, we label that as an island. On sample data, Posterior decoding had more false positives than Viterbi, but found same true positives. Performance improved when they incorporated some post-processing; throwing out short islands, joining nearby ones.

Training

Basic:
Given data, take natural estimations for transition and emission probabilities. May want to incorporate a pseudo-count if data insufficient.

Viterbi training:
Estimate params (transition and emission probabilities) theta. Find Viterbi path for each sequence. Use these sequences of states to estimate new params. Find best paths with these params, etc. Very often used.

Baum-Welch training:
Similar to Viterbi, but more rigorous.
$P(pi_i=k,pi_{i+1}=l \mid x,theta) = f(i|theta)a_{kl}e_l(x_{i+1})b_l(i+1|theta)/P(x|theta)$
estimated number of k->l transitions
$A_{kl}=sum_{\text{training data}}\ sum_i P(pi_i=k,pi_{i+1}=l \mid x^j.theta)$
Then a_{kl} estimated as A_{kl}/sum_iA_{ki}. This gives a new estimate of the parameters (similar formula for estimates of emission parameters), and then feed these back in. BW is formally an EM algorithm, & has usual EM guarantee of finding a local (but perhaps not global) max. Viterbi training is EM-like, but is not rigorously EM.

Final motivating example

Proteins in a certain family can be modeled as HMMs also. However, you can have big holes in the middle of one sequence relative to another. The way to model this is to have some “non-emitting” states, which give no symbol (when you're aligning a potential protein against a known one, these states correspond to gaps in the alignment).