18.417 Introduction to Computational Molecular Biology	
Lecture 18: November 9, 2004	Scribe: Chris Peikert
Lecturer: Ross Lippert	Editor: Chris Peikert

Applications of Hidden Markov Models

Review of Notation

Recall our notation for Hidden Markov Models: $T(i,j) = \Pr[i \to j]$, the probability of transitioning from state i to state j. $E(i,x) = \Pr[x \text{ emitted at state } i]$. The starting distribution over states is π. $D_x = \text{diag}(E(:,x))$, that is, D_x is a square matrix with $E(i,x)$ as entry (i,i), and zeros elsewhere.

Using this notation, the probability of a certain sequence of symbols $x_1, \ldots, x_n$ being output from the chain is:

$$\Pr[x_1, \ldots, x_n] = \pi D_{x_1} T \cdots T D_{x_n} 1.$$

This can be split into two important auxiliary quantities that frequently appear:

- $f(k,i) = \Pr[x_1, \ldots, x_k, s_k = i)$, the probability that symbols $x_1, \ldots, x_k$ are output and the Markov chain is at state i at time k. As a vector over all states, $f(k,:) = \pi D_{x_1} T \cdots T D_{x_k}$.
- $b(k,i) = \Pr[x_{k+1}, \ldots, x_n, s_k = i]$, the probability that symbols $x_{k+1}, \ldots, x_n$ are output when the Markov chain is at state i at time k. As a vector over all states, $b(k,:) = T D_{x_{k+1}} \cdots T D_{x_n} 1$.

Thus we get that

$$\Pr[x_1, \ldots, x_n] = \sum_{i \in \text{states}} f(k,i) b(k,i).$$

Using these quantities, we can compute the probability that, given a sequence $x_1, \ldots, x_n$ of output symbols, the Markov chain was in some state i at time k:

$$\Pr[s_k = i] = \frac{f(k,i) b(k,i)}{\sum_j f(k,j) b(k,j)}.$$

However, this isn't very useful because it doesn't capture dependencies between states and output symbols at different times.

Tropical Notation

A semi-ring is a tuple $(S, \mathrm{id}_+, \mathrm{id}_\times, +, \times)$, where S is a set of elements, id_+ and $\mathrm{id}_\times$ are the additive and multiplicative identities (respectively), and $+$ and $\times$ are the addition and muliplication operations. The operations are closed, commutative, associative, and $\times$ distributes over $+$. There are many examples of semi-rings:

Example 1 $(\mathbb{R}^+ \cup \{0\}, 0, 1, +, \times)$*, the non-negative reals under standard addition and multiplication, is a semi-ring.*

Example 2 $(\mathbb{R} \cup \{\infty\}, \infty, 0, \min_T, +)$ *(the "Boltzmann" semi-ring), where*

$$\min_T(a, b) = T \log(\exp(-a/T) + \exp(-b/T)),$$

is a semi-ring useful in statistical mechanics. Note that as $T \to 0^+$, $\min_T$ *"approaches"* min*, in the sense that it becomes a closer and closer approximation of the* min *operation. The semi-ring at this "limit" is known as one of the* tropical semi-rings*.*

Example 3 *Likewise,* $(\mathbb{R} \cup \{-\infty\}, -\infty, 0, \max, +)$*, which arises by taking the Boltzmann semi-ring as* $T \to \infty$*, is the other tropical semi-ring.*

Let's interpret our HMM quantities in the tropical semi-ring from Example 3. To do so, we simply take logarithms of the real quantities, replace $\sum$ by max, and multiplication by $+$.

Recall that

$$\Pr[x_1, \ldots, x_n] = \sum_{i_1, \ldots, i_n} \pi_i E(i_1, x_1) T(i_1, i_2) \cdots = \sum_{\vec{\imath}} f(k, i) b(k, i).$$

We "tropicalize" this expression, viewing it in the appropriate semi-ring:

$$\max_{\vec{\imath}}(\log \pi_i + \log E(i_1, x_1) + \log T(i_1, i_2) + \cdots) = \max_{\vec{\imath}}(\tilde{f}(k, i) + \tilde{b}(k, i))$$

where $\tilde{f}$ and $\tilde{b}$ are defined similarly to f and b, but in the semi-ring. From this, we get a more compact and easier to evaluate expression for the most likely state at time k:

$$\text{state}_k = \arg\max_i(\tilde{f}(k,i) + \tilde{b}(k,i)).$$

It is possible to derive other interesting results via tropicalization:

- The Viterbi algorithm can be viewed as a tropicalization of the likelihood calculation;
- Approximations can be made about random gapped alignment scores;
- Tropical "determinants" of distance matrices provide useful information about trees.

Training an HMM

We want to compute the HMM that was mostly likely to produce a given sequence of symbols.

- Input: the sequence (or, more generally, set of sequences) $x_1, \ldots, x_n$ and the number of states in the HMM.
- Output: the HMM which maximizes $\Pr[x_1, \ldots, x_n]$.

In general, this is a nonlinear optimization problem with constraints: $T \geq 0, E \geq 0, T \cdot 1 = 1, E \cdot 1 = 1$. The objective being optimized, $\Pr[x_1, \ldots, x_n]$, can actually be written as a polynomial in T and E. Unfortunately, the general problem of polynomial optimization is NP-hard.

Expectation Maximization

Expectation Maximization (EM) is a very common tool (often used in machine learning) to solve these kinds of optimization problems. It is an iterative method which walks through the solution space and is guaranteed to increase the likelihood at every step. It follows from general techniques for doing gradient search on constrained polynomials, and always converges to some local (but possibly not global) optimum.

In EM, we start with some initial E and T and iterate, computing more likely values with each iteration. Given the values of E and T at one iteration, the values in the next iteration (called $\hat{E}$ and $\hat{T}$) are given by the equations:

$$\begin{aligned} \hat{E}(i,x) &\propto \sum_{k\,:\,x_k=x} f(k,i)b(k,i) \\ \hat{T}(i,j) &\propto \sum_{k} f(k,i)T(i,j)E(j,x_k)b(k+1,j) \end{aligned}$$

(where f and b are calculated using E and T).

HMMs Applied to Global Alignment

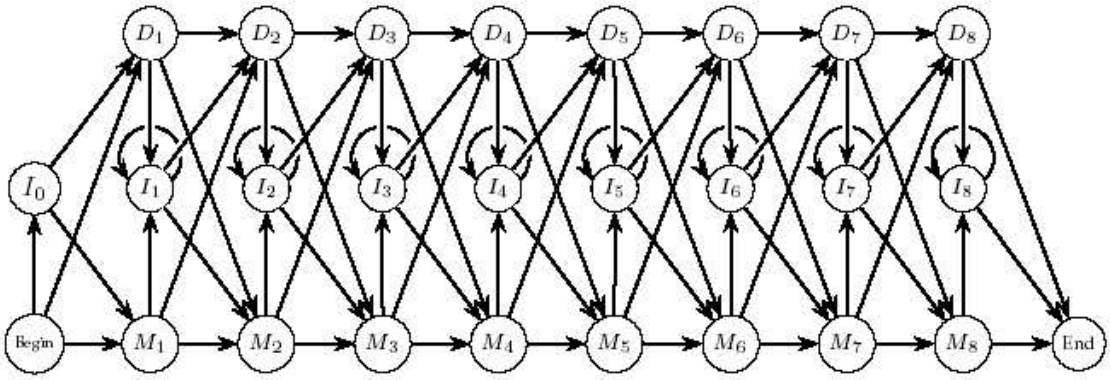

Figure 18.1: An HMM to support alignment

Consider the problem of global alignment with affine gap penalties. We can view this problem in terms of finding an HMM that produces the observed sequences with good likelihood. We limit the HMM to a specific structure (see figure 18.1), which consists of several layers, each containing one of the following three kinds of states:

- Match state: this emits one base (usually with high bias), and links to the insertion state at the same layer and the deletion state at the next layer;
- Insertion state: this emits a random base and links back to itself, as well as to the deletion and matching states of the next layer;
- Deletion state: this emits nothing, and links to the deletion and match states of the next layer.

A path through the HMM corresponds to an alignment in the following way: passing through a match state corresponds to matching bases between the two sequences; cycling through an insertion state corresponds to some number of inserted bases (i.e., gaps in one sequence); passing through deletion states corresponds to some number of deleted bases (i.e., gaps in the other sequence).

Extensions of Alignment via HMMs

Using HMMs we can do alignment to *profiles* (related sequences serving as training data). Multisequence alignment is useful in many ways:

- The Viterbi algorithm produces good *multialignments* (recall that the previous best algorithm for this problem took time and space exponential in the number of sequences).
- Transition probabilities give *position dependent* alignment penalties, which can be much more accurate than in the case of generic affine gap penalties.
- Probabilities within the HMM highlight regions of both high and low conservation of bases across different sequences.
- Pairs of HMMs can provide means for performing clustering: after training two HMMs on different profiles, a new sequence can be clustered by choosing the HMM from which it would be produced with the higher likelihood.