Some Techniques of Data Mining in Biology Part 2

Marco Valtorta
mgv@cs.sc.edu

Department of Computer Science and Engineering
University of South Carolina

March 16, 2000

Overview

Basics of Hidden Markov Models

Applications of HMMs to Biological Sequence Analysis and Gene Finding

Preliminaries: Observable Markov Models

See Figure 1 in Rabiner's tutorial.

- N distinct states: $S_1, S_2, \ldots, S_N$.

- Time instants associated with state changes: $t = 1, 2, \ldots$.

- Actual state at time t: q_t.

- Time invariant and first-order Markov chain transition probabilities:
  $a_{ij} = P(q_t = S_j | q_{t-1} = S_i)$, $1 \leq i, j \leq N$

- Initial state probabilities:
  $\pi_i = P(q_1 = S_i)$, $1 \leq i \leq N$

- Probability density function of duration d in state i:
$p_i(d) = P(O|\text{Model}, q_1 = S_i) = (a_{ii})^{d-1}(1 - a_{ii})$,
where $O = < \underbrace{S_i, S_i, S_i, \ldots, S_i}_{d}, S_j \neq S_i >$

- Expected duration in a state:
$\overline{d_i} = \sum_{d=1}^{\infty} d p_i(d) = \sum_{d=1}^{\infty} d(a_{ii})^{d-1}(1 - a_{ii}) = 1/(1 - a_{ii})$

Elements of a Hidden Markov Model

- N, the number of states in the model. Individual states are denoted $S = \{S_1, S_2, \ldots, S_N\}$.

- M, then numer of distinct observation symbols per state, i.e., the discrete alphabet size. The observation symbols correspond to the observable output of the system being modeled. The individual symbols are denoted $V = \{v_1, v_2, \ldots, v_M\}$.

- The state transition probability distribution $A = \{a_{ij}\}$, where

$$a_{ij} = P(q_{t+1} = S_j | q_t = S_i), 1 \leq i, j \leq N \tag{1}$$

- The observation symbol probability in state j, $B = \{b_j(k)\}$, where

$$b_j(k) = P(v_k \text{ at } t | q_t = S_j), 1 \leq j \leq N, 1 \leq k \leq M \tag{2}$$

- The initial state distribution $\pi = \{\pi_i\}$, where

$$\pi_i = P(q_1 = S_i), 1 \leq i \leq N \tag{3}$$

For convenience, we indicate the parameter set of the model by

$$\lambda = (A, B, \pi) \tag{4}$$

The Three Basic Problems for HMMs

Problem 1: Evaluation Given the observation sequence $O = O_1, O_2, \ldots, O_T$ and a model $\lambda = (A, B, \pi)$, efficiently compute $P(O|\lambda)$.

Problem 2: Explanation Given O and λ, find a corresponding (hidden) state sequence $Q = q_1, q_2, \ldots, q_t$ that best explains the observations.

Problem 3: Training Adjust the model parameters λ to maximize $P(O|\lambda)$.

Solution to Problem 1 (Evaluation)

We want to calculate $P(O)$ (given the model λ). The straightforward way is to enumerate all possible state sequences:

$P(O,Q) = P(O|Q)P(Q)$, and therefore
$P(O) = \sum_{\text{all } Q} P(O|Q)P(Q) =$

$$\sum_{q_1,q_2,\ldots,q_T} \pi_{q_1} b_{q_1}(O_1) a_{q_1 q_2} b_{q_2}(O_2) \ldots a_{q_{T-1} q_T} b_{q_T}(O_T) \tag{5}$$

The sum is evaluated over the N^T possible paths through the graph in the space of state transitions. Therefore, $N^T - 1$ additions are needed. Each term in the sum is the product of $2T$ distinct factors, so it requires $2T-1$ multiplications. Overall, $(2T-1)N^T$ multiplications and $N^T - 1$ additions are needed.

For example, for an HMM with 5 states ($N = 5$) and 100 observations ($T = 100$), about $2 \times 100 \times 5^{100} \simeq 10^{72}$ arithmetic operations are needed: too many!

DP to the Rescue: the Forward Procedure

Define the *forward variable*

$$\alpha_t(i) = P(O_1 O_2 \ldots O_t, Q_t = S_i) \qquad (6)$$

as the probability of the partial observation sequence $< O_1 \ldots O_t >$ up to time t and state S_i (given the model λ).

We can compute α_t using only α_{t-1} (and various parameters), as indicated in the data dependency diagram of Figure 4(a) in Rabiner's paper.

So, we can solve for $\alpha_t(i)$ using dynamic programming, as follows:

1. **Initialization**

$$\alpha_1(i) = \pi_i b_i(O_1), q \leq i \leq N$$

2. **Induction**

$$\alpha_{t+1}(j) = (\sum_{i=1}^{N} \alpha_t(i) a_{ij}) b_j(O_{t+1}),$$

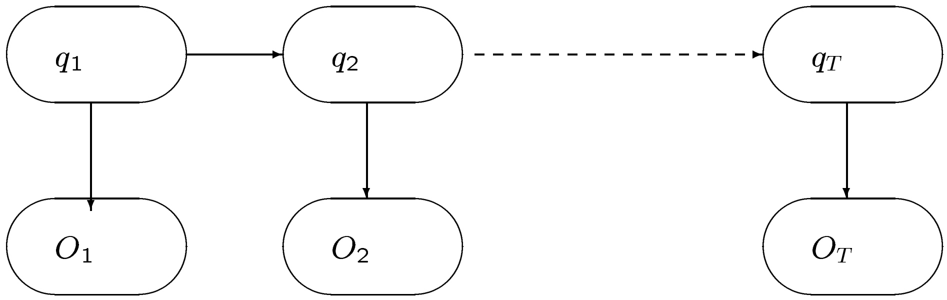
$$1 \leq t \leq T-1, 1 \leq j \leq N$$

3. **Termination**

$$P(O) = \sum_{i=1}^{N} \alpha_T(i)$$

Rabiner (p.262) interprets the computation as follows: since $\alpha_t(i)$ is the probability that $O_1 O_2 \ldots O_t$ are observed and the state at time t is S_i, then the product $\alpha_t(i) a_{ij}$ is the probability that the sequence is observed, the state at time t is S_i, and the state at time $t+1$ is S_j. By summing over all possible N states at time t, we obtain the probability of reaching state S_j at time $t+1$ with the accompanying partial observations up to time t. To account for the observation at time $t+1$, we multiply by $b_j(O_{t+1})$.

HMMs as Bayesian Networks

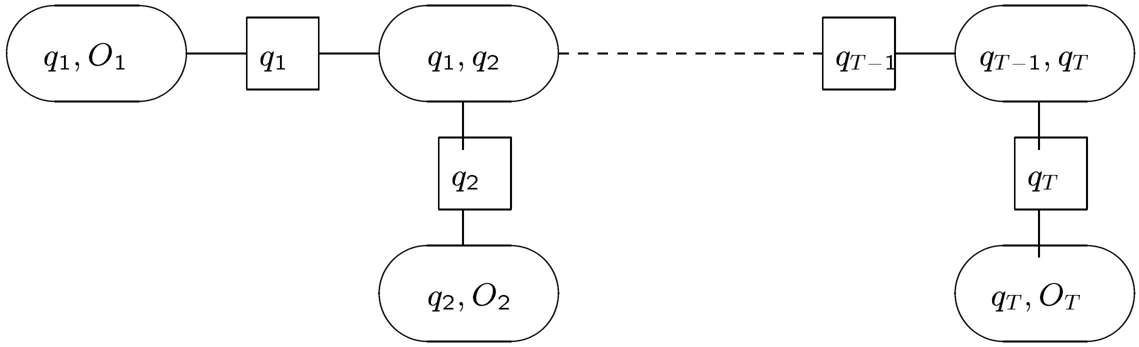


In particular,

$$P(q_1, q_2, \ldots, q_T, O_1, O_2, \ldots, O_T) = P(q_1)P(O_1|q_1)\ldots P(q_T|q_{T-1})P(O_T|q_T).$$

If one marginalizes away the states, one obtains equation (5), when recalling the definition of the parameters $\lambda = (A, B, \pi)$. Note the unfortunate use of lower case letters for the state variables and upper case letters for the observation variables. Recall that the possible values of each state variable are $S_1, S_2, \ldots, S_N$, and the possible values of each observation variable are $v_1, v_2, \ldots, v_M$.

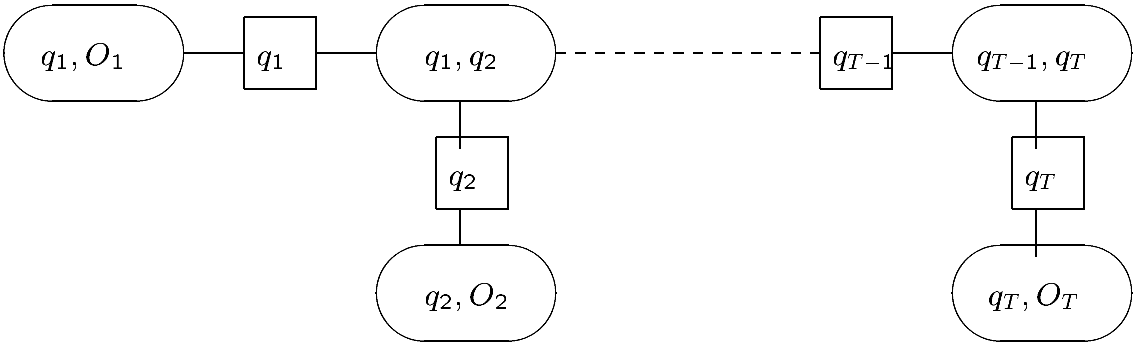
In terms of the Bayesian net, the evaluation problem is the problem of computing the probability of the evidence $O_1 O_2 \ldots O_T$. This, of course, could be done using Hugin.



The probability of the evidence is the sum of the entries in any of the unnormalized potential tables. It seems that Hugin would carry out a lot of unnecessary divisions to do this computation. Comments?

A (fvj): Only a Collect operation is needed. No DistributeEvidence is necessary, and therefore no calibration (with division) is carried out.

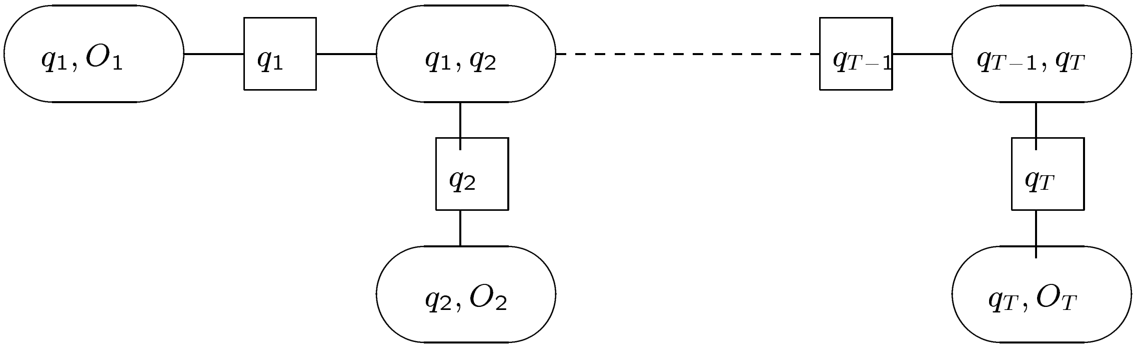
Q: How to exploit sparseness of A? A (Helge): Use factored HMMs.



Suppose that CollectEvidence is carried out from node (q_T, O_T). Consider each calibration operation that takes place. Over the "vertical" links, the evidence variables are marginalized out, so that, for example, q_2 is marginalized out on the leftmost vertical link, resulting in a message

$$\sum_{O_2} P(q_2, O_2),$$

where $P(q_2, O_2) = P(O_2|q_2)P(O_2)$ Since the value of O_2 is observed, the sum is simply $P(o_2|q_2)$, which by definition is $b_{q_2}(o_2)$, where o_2 is the observed value of O_2. Similarly, on the rightmost vertical link, the message is $b_{q_T}(o_T)$.



On the "horizontal" links, the state variables are marginalized out. For example, on the leftmost horizontal link, the variable q_1 is marginalized out, resulting in the message

$$\sum_{q_1} b_{q_1}(o_1) = (def.) = \pi \cdot b(o_1) = (def.) = \alpha_1$$

. On the second leftmost horizontal link, the variable $q2$ is marginalized out of $\alpha_1 \cdot P(q_2|q_1) \cdot b_{q_2}(o_2)$, which is α_2.

It is a simple inductive argument to show that the message passed on the horizontal link labeled by q_i is α_i.

Alternative Solution: the Backward Algorithm

The forward algorithm corresponds to the evaluation of the sum (5) by eliminating (i.e., summing out) in the order $q_1, q_2, \ldots, q_T$.

It is also possible to evaluate the sum (5) by eliminating (i.e., summing out) variables in the order $q_T, q_{T-1}, \ldots, q_1$.

This leads to the backward procedure (with associated backward variable). See formulas (23)-(25) in Rabiner's paper.

Solution to Problem 2 (Explanation)

The problem is that of finding the sequence of states $Q = q_1 q_2 \ldots q_T$ that maximizes $P(Q|O)$, where O is a sequence of T observations, $O = O_1 O_2 \ldots O_T$. Equivalently, since O is given, one can maximize $P(Q, O)$. This is a Most Probable Explanation (MPE) problem. The DP solution of this problem is called the Viterbi algorithm.

We introduce a quantity that represents the highest probability ("best score") along a single path that accounts for the first t observations and ends in state S_i:

$$\delta_t(i) = \max_{q_1, q_2, \cdots, q_{t-1}} P(q_1, q_2, \ldots, q_t = S_i, O_1 O_2 \ldots O_t)$$

The data dependency diagram for δ is the same as that for α (cf. Figure 4(a) in Rabiner's paper). The computation is very similar to that for the forward algorithm, except that sum is now substituted by maximization. We also keep track (using a family of functions $\psi_t(i), 2 \le t \le T, 1 \le i \le N$) of the state that maximizes $\delta_t(i)$ for each time step, to avoid local searches in computing the optimal state sequence once the score of the optimal state sequence has been found. Details follow.

1. **Initialization**

$$\delta_1(i) = \pi_i b_i(O_1)$$

2. **Recursion**

$$\delta_t(j) = \max_{1 \le i \le N} (\delta_{t-1}(i) a_{ij}) b_j(O_t)$$

$$\psi_t(j) = \text{argmax}_{1 \le i \le N} (\delta_{t-1}(i) a_{ij})$$

$$2 \le t \le T, 1 \le j \le N$$

3. **Termination**

$$P^* = \max_{1 \le i \le N} (\delta_T(i))$$

$$q_T^* = \text{argmax}_{1 \le i \le N} (\delta_T(i))$$

4. **Path (state sequence) backtracking**

$$q_t^* = \psi_{t+1}(q_{t+1}^*), t = T-1, T-2, \ldots, 1.$$

Problem 3: Training

Let an observation sequence $O = O_1 O_2 \ldots O_T$ be given. The training problem is to adjust the model parameters $(A, B, \pi)(= \lambda)$ (i.e., the transition probabilities, the observation model, and the initial state distribution) in such a way that $P(O)$ is maximized.

The solution is an instance of the EM algorithm, which in this specific instance is known as the Baum-Welch algorithm. We will derive it directly, without recourse to EM. Please see the paper for the argument that Baum-Welch is an instance of EM.

We first introduce two quantities, $\gamma_t(i)$ and $\xi_t(i,j)$.

$\gamma_t(i)$ is the probability of being in state S_i at time t, given the observation sequence O and the model λ: $\gamma_t(i) = P(q_t = S_i|O)$, which can be expressed in terms of the forwards and backwards variables as follows:

$$\begin{aligned}
\gamma_t(i) &= \\
&= \frac{P(q_t = S_i, O_1 \dots O_t)P(q_t = S_i, O_{t+1} \dots O_T)}{P(O)} \\
&= \frac{\alpha_t(i)\beta_t(i)}{P(O)} \\
&= \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)}
\end{aligned}$$

$\xi_t(i,j)$ is the probability of being in state S_i at time t and state S_j at time $t+1$, given the observation sequence O and the model λ:

$$
\begin{aligned}
\xi_t(i,j) &= P(q_t = S_i, q_{t+1} = S_j | O) \\
&= P(q_t = S_i, O_1 \ldots O_t) \times \\
&\times \; P(q_{t+1} = S_j | q_t = S_i) \times \\
&\times \; \frac{P(q_{t+1} = S_j, O_{t+1} \ldots O_T)}{P(O)}
\end{aligned}
$$

where

$$
P(O) = \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_i(O_{t+1}) \beta_{t+1}(j)
$$

Note that, if sum $\gamma_t(i)$ for all t, we obtain the expected (over time) number of times that S_i is visited or, equivalently, the expected (over time) number of transitions made to state i. Observing that there are no transitions at time T, we conclude that

$\sum_{t=1}^{T-1} \gamma_t(i) =$
expected number of transitions from S_i.

Similarly,

$\sum_{t=1}^{T-1} \xi_t(i,j) =$
expected number of transitions from S_i to S_j.

A set of reestimation formulas for π, A, and B, is:

$\overline{\pi}_i =$ expected frequency in state $S_i$at time 1 $= \gamma_1(i)$

$$\overline{a}_{ij} = \frac{\text{expected number of transitions from state } S_i \text{ to state } S_j}{\text{expected number of transitions from state } S_i} =$$

$$\frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

$$\overline{b}_j(k) = \frac{\text{expected number of times in state } j \text{ and observing symbol } v_k}{\text{expected number of times in state } j} =$$

$$\frac{\sum_{t=1 \text{s.t. } O_t=v_k}^{T} \gamma_t(i)}{\sum_{t=1}^{T} \gamma_t(i)}$$

The Baum-Welch Algorithm

The formulas on the previous transparency are used iteratively. At each iteration, the current value of the parameters, λ, is replaced by the estimated value, $\overline{\lambda}$.

Baum and his colleagues showed that, upon convergence, a local maximum for the maximum likelihood estimate of the parameters is found. Furthermore, it can be shown that the procedure is a realization of the EM algorithm. The algorithm is also known as the forward-backward algorithm.

(Hidden) Markov Models for Biological Sequences

Anders Krogh [1998] describes applications of HMMs to pairwise sequence alignment, finding gene structure, and some related problems. Despite the title of Krogh's paper, most of the HMMs he presents are actually observable Markov models.

A Simple Scoring Model for a Short Sequence

Krogh (p.46) presents this "DNA Motif:"

$$\begin{array}{ccccccccc}
A & C & A & - & - & - & A & T & G \\
T & C & A & A & C & T & A & C & T \\
A & C & A & C & - & - & A & G & C \\
A & G & A & - & - & - & A & T & C \\
A & C & C & G & - & - & A & T & C
\end{array}$$

From these data, the model of Figure 1 (p.47) is constructed. The probability that a particular symbol occurs in a particular position of a sequence is computed from the data, by dividing the occurrence of each symbol at that position by the total number of sequences, so that, for example,

$$P_1(A) = \frac{\text{number of occurrences of } A}{\text{total number of sequences}} = \frac{4}{5} = .8$$

Similarly,

$$P_2(A) = \frac{\text{number of occurrences of } A}{\text{total number of sequences}} = \frac{0}{5} = 0$$

It does not seem trivial to translate these models into BNs without greatly increasing the number of states. There are several differences with respect to HMMs as described by Rabiner:

- Time is replaced by space (location of a nucleotide).

- There is no spatial invariance (which would correspond to time invariance, i.e., $P(q_t = S_j)$ varies with t, where we use t to indicate the position in the sequence.

- Position at t is independent of position $t-1$, i.e., the model is zero-order.

- There is no observation model, because the state is not hidden.

- Gaps (or their complement, "insertions") are allowed.

To deal with insertions, we introduce special states. In Figure 1, we see that an insertion occurs with probability .6. This corresponds to the fact that three of five sequences (namely, sequences 2, 3, and 5) have an insertion. After entering the insertion state (i.e., after inserting one nucleotide), there are two more insertions (from sequence two) and three transitions (because each of the three sequences 2, 3, and 5 must return to the main line in order to finish). This explains the self loop with .4 and the transition with .6 out of the insertion state. The probabilities of each nucleotide within the insertion state are computed in the obvious way; for example, $P(A) = .2$ and $P(C) = .4$, because A occurs once among the five nucleotides in the insertions, while C occurs twice.

The probability of a sequence according to the model is found by multiplying the probability of transitions and states—cf. example in the middle of p.48. The probabilities of various sequences are given in Table 1.

A more common score than the probability is the log-odds score, defined as the logarithm (usually in base 2) of the ratio of the probability of the sequence to be scored divided by the probability of a "null" sequence, which is usually taken to be a random sequence of nucleotides of the same length (L) as the sequence to be scored, so that:

$$\log - \text{odds for sequence } S = \log \frac{P(S)}{0.25^L} = \log P(S) - L \log 0.25$$

For convenience, the probabilities are converted to log-odds, as in the model of Figure 2 (p.49).

Profile HMMs

- Adding null (or silent) states (to main and insert states) results in the generic profile of Figure 3 (p. 50)

- Example: Data of Figure 4 (p.51) result in model of Figure 5 (p.52)

- Adding a pseudocount of one results in the model of Figure 6 (p.53)

Searching a database

So far, we have assumed that we know how to align a sequence S to the profile (i.e., we know where the gaps are), but this is unrealistic, if S that was not used in the construction of a profile.

We need to align S to the profile in the best way. This is analogous to finding the best sequence of states in an HMM, and a modified version of the Viterbi algorithm can be used. (Krogh actually simply states that the Viterbi algorithm is used.)

HMMs for Gene Finding

Some highlights:

- Models for splice sites are first-order Markov models (p.57).

- Models for codons must be at least of order two, because the third codon depends on the previous two; even higher-order models are useful, to model dependencies between neighboring codons. A Markov chain with varying order models is called an inhomogeneous Markov chain (p.58).

- Figure 11 (p.59) shows a Markov model for unspliced genes.

- Figure 12 (p.60) shows a Markov model for spliced genes, which takes into account the three possible reading frames.

References

Krogh, Anders. "An Introduction to Hidden Markov Models for Biological Sequences." Chapter 4 (pp.45–63) in: S.L. Salzberg, D.B. Searls, and S. Kasif (eds.), *Computational Methods in Molecular Biology*. Amsterdam: Elsevier, 1988.

The first ten pages of: Rabiner, Lawrence R. "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition." *Proceedings of the IEEE*, 77, 2 (February 1989), pp.257–286.