

HMM Lecture Notes

Tuesday, October 31st

Rose Hoberman and Dannie Durand

1 Overview

The last two lectures on HMMs deal with the modeling and discovery functions of HMMs. We are given observed sequences, O^1, O^2, \dots, O^k , and wish to construct an HMM with parameters, λ , to model these sequences.

If the sequences are *labeled*, the main problem is to design the topology. Once the states and connectivity have been chosen, the transition and emission probabilities can be estimated easily using MLE. If the sequences are *unlabeled*, then it is necessary both to design the topology and to learn the motif and the model parameters. The *Baum Welch* algorithm will learn the parameters from the data and implicitly, also discovers the motif. To determine the motif explicitly, we use the Viterbi algorithm on the new HMM to label the states of each input sequence.

In the current lecture, we discuss the Baum Welch algorithm and introduce topology modeling. In the next lecture we discuss topology in more detail, including the widely used Profile HMM model.

2 Notation

1. N states ($S_1..S_N$)
2. M symbols in alphabet, Σ
3. parameters, λ :
 1. initial distribution of states $\pi(i)$
 2. transition probabilities $a_{ij} = P(q_t = S_i | q_{t-1} = S_j)$. Note that $\sum_{i=1}^N a_{ij} = 1, \forall j$
 3. emission probabilities $e_i(a)$ probability state i emits a
4. Sequence of symbols: $O = O_1, O_2, \dots, O_T$
5. Sequence of states: $Q = q_1, q_2, \dots, q_T$

3 Baum Welch

In general, if we have labeled data (that is, we know the state sequence), we can obtain the parameters using Maximum Likelihood Estimation. The Baum-Welch algorithm is used to estimate the model parameters when the state path is unknown. Given sequences O^1, O^2, \dots , we wish to determine $\lambda = \{a_{ij}, e_i(\cdot), \pi_i\}$. We generally want to choose parameters that will maximize the likelihood of our data.

However, finding a global maximum is intractable. We would have to enumerate over all parameter sets, λ_k , and then calculate

$$\text{Score}(\lambda_k) = \sum_d P(O^d | \lambda_k) = \sum_d \sum_Q P(O^d | \lambda_k, Q)$$

for each λ_k . Instead, people settle for heuristics which are guaranteed to find at least a local maximum. Since these are heuristics, evaluation is usually done empirically by withholding some of the training data for testing, but we will not discuss this further.

The algorithm (BW) used for selecting the parameter values belongs to a family of algorithms called Expectation Maximization (EM) algorithms. They all work by guessing initial parameter values, then estimating the likelihood of the data under the current parameters. These likelihoods can then be used to re-estimate the parameters, iteratively until a local maximum is reached.

Setup The alphabet and the number of states, N , are fixed. We are given the observed sequences (denoted $O^d = O_1^d, O_2^d, \dots$).

The intuition behind the algorithm is as follows

1. Choose some initial values for $\lambda(\pi, a_{ij}, e_i(\cdot))$.
2. Determine “probable paths” $Q^d = q_1^d, q_2^d, \dots$
3. Count the expected number of transitions, A_{ij} , from state i to state j , given the current estimate of λ .
4. Count, $E_i(\sigma)$, the expected number of times character σ is emitted from state i .
5. Re-estimate $\lambda(\pi, a_{ij}, e_i(\cdot))$ from A_{ij} and $E_i(\sigma)$,
6. if not converged, go to step 2

For a given sequence, O^d , probability of transiting from state i to j at time t is

$$P(q_t^d = i, q_{t+1}^d = j | O^d, \lambda) = \frac{P(q_t^d = i, q_{t+1}^d = j, O^d)}{P(O^d)} = \frac{\alpha_t(i) a_{ij} e_j(O_{t+1}^d) \beta_{t+1}(j)}{P(O^d)}$$

The term $\alpha_t(i)$ is the probability that the model has emitted symbols $O_1^d \dots O_t^d$ and is in state S_i at time t . This probability can be obtained using the Forward algorithm. Similarly, the Backward algorithm yields $\beta_{t+1}(j)$, the probability of emitting the rest of the sequence if we are in state j at time $t+1$. The remaining two terms, a_{ij} and $e_j(O_{t+1}^d)$ give the probability of making the transition from i to j and emitting the $t+1$ st character.

From this we can estimate

$$A_{ij} = \sum_d \frac{1}{P(O^d)} \sum_t \alpha(t, i) a_{ij} e_i(O_{t+1}^d) \beta(t+1, i) \quad (1)$$

The probability of O^d can be estimated using current parameter values using the Forward algorithm.

Similarly,

$$E_i(\sigma) = \sum_d \frac{1}{P(O^d)} \sum_{\{t|O_t^d=\sigma\}} \alpha(t, i) \beta(t, i). \quad (2)$$

From A_{ij} and $E_i(\sigma)$ we re-estimate the parameters.

Stated formally:

Algorithm: Baum Welch

Input:

A set of observed sequences, O^1, O^2, \dots

Initialization:

Select arbitrary model parameters, $\lambda' = a_{ij}, e_i()$.
 score = $\sum_d P(O^d | \lambda')$.

Repeat

{

$\lambda = \lambda', S = S'$

For each sequence, O^d ,

{

/* Calculate “probable paths” $Q^d = q_1^d, q_2^d, \dots$ */

Calculate $\alpha(t, i)$ for O^d using the Forward algorithm.

Calculate $\beta(t, i)$ for O^d using the Backward algorithm.

Calculate the contribution of O^d to A using (1).

Calculate the contribution of O^d to E using (2).

}

$a_{ij} = \frac{A_{ij}}{\sum_l A_{il}}$

$e_i(\sigma) = \frac{E_i(\sigma)}{\sum_\tau E_i(\tau)}$

score = $\sum_d P(O^d | a_{ij}, e_i())$.

}

Until (the change in score is less than some predefined threshold.)

The estimation of “probable paths” $Q^d = q_1^d, q_2^d, \dots$ in the inner loop is done efficiently using dynamic programming. This was not covered in class. This is done using the Forward and Backward algorithms. You are not responsible for the details of how the Forward and Backward algorithms

are used in Baum Welch, but you should understand the basic idea of the algorithm and are responsible for knowing when Baum Welch should be applied.

Convergence It can be proven that if current estimate is replaced by these new estimates then the likelihood of the data will not decrease (i.e. will increase unless already at a local maxima/critical point). See Durbin, Section 11.6 for discussion of avoiding local maxima and other typical pitfalls with this algorithm.

4 Topology

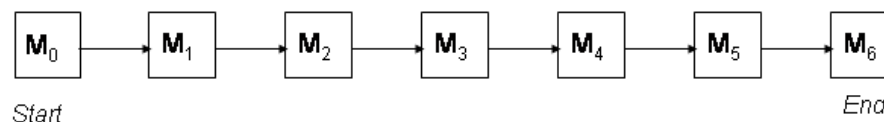
- Characteristics: number of nodes, alphabet, which edges to consider. We could just choose a fully connected graph, but this has too many parameters to estimate.
- Instead we can exploit **domain knowledge**. Choose a topology that limits the number of states and edges while still being expressive enough to represent the relationships they believe to exist.
- The choice of topology can impose a probability distribution on the length of the sequences that the HMM recognizes. For example, a simple self loop with probability p results in an exponentially decaying (geometric) distribution $P(l \text{ residues}) = (1 - p)p^{l-1}$. There are topologies that assume other length distributions (see Durbin, 3.4 for more on this subject).

A basic topology:

Suppose we wish to construct an HMM for the WEIRD motif, based on the following alignment which has no gaps and no positional dependencies:

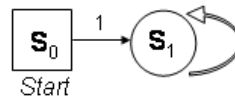
```
WEIRD
WEIRD
WEIRE
WEIQH
```

We can recognize the WEIRD motif using an HMM with this topology:



where the transitions probabilities are $a_{i,j} = 1$ if $j = i + 1$ and zero, otherwise. The emission probabilities are $e_j(\alpha) = F[\alpha, j]$, where $F[\alpha, j]$ is the same frequency matrix that we derived for the PSSM example, using pseudocounts. The Start and End states (M_0 and M_6) are silent. The above model is our alternate hypothesis, H_A .

To score a new sequence, we also need a background model (the null hypothesis, H_0):



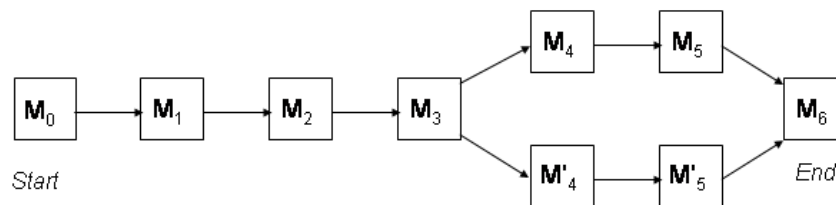
In this model, all transition probabilities are equal to one. The emission probabilities are $e_j(\alpha) = p(\alpha)$, where $p(\alpha)$ is the background frequency of residue α . We can then score a new sequence, O , by calculating $\log \frac{P(O|H_A)}{P(O|H_0)}$. We obtain a score equivalent to $\sum_{i=1}^5 S[o_i, i]$, the score we would have obtained with the PSSM for the WEIRD motif.

Positional dependencies:

Now suppose that our motif has a positional dependency like this one, in which we see either RD or QH in the last two positions, but never QD or RH.

WEIRD
 WEIRD
 WEIQH
 WEIRD
 WEIQH
 WEIQH

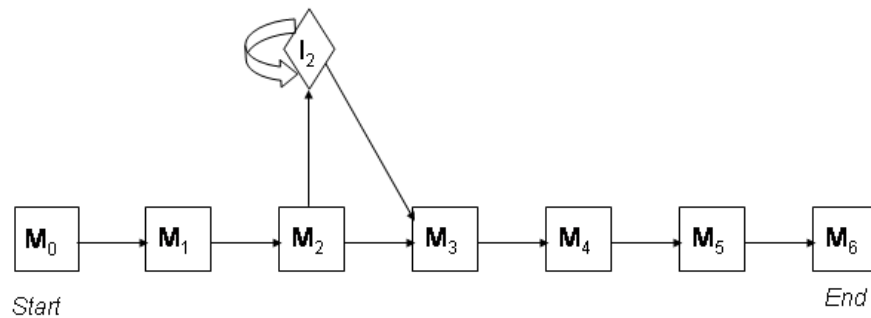
A PSSM for this motif, however, would give the sequences WEIRD and WEIRH equally good scores. So would the basic HMM above. We can construct an HMM to model this pairwise dependency like this:



where the emission probabilities are $e_{M_4}(R) = 1$, $e_{M_5}(D) = 1$, $e_{M'_4}(Q) = 1$ and $e_{M'_5}(H) = 1$.

Insertions:

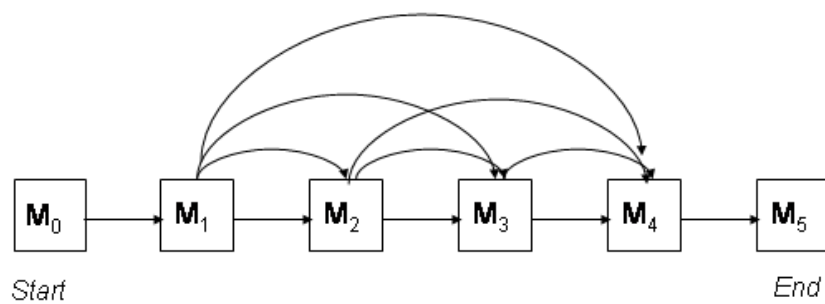
We can modify the basic HMM to recognize query sequences with insertions such as $O = \text{WECIRD}$:



where the emission probabilities for the insertion states are the background frequencies.

Deletions:

Suppose our query sequences has a deletion, e.g., $O = \text{WERD}$. One approach to capturing such deletions would be to add edges allowing us to jump over any set of match states:



The disadvantage to this approach is that to infer the transitions, we would need a very large set of training data, one in which all deletions of all possible sizes were represented. Instead, we can model long deletions as sequences of short ones, as seen below