Today’s Lecture: HMMs

• Definitions

• Examples

• Probability calculations
  – WDAG
  – Viterbi algorithm
HMMs: Formal Definition

• Alphabet $\mathcal{B} = \{b\}$ of *observed symbols*

• Set $\mathcal{S} = \{k\}$ of *hidden states* (usually $k = 0, 1, 2 \ldots, m$; 0 is reserved for “begin” state, and sometimes also an “end” state)

• (Markov chain property): prob of state occurring at given position depends only on immediately preceding state, and is given by

  transition probabilities ($a_{kl}$): $a_{kl} = \text{Prob}(\text{next state is } l | \text{curr state is } k)$

  $\sum_l a_{kl} = 1$, for each $k$.

  – Usually, many transition probabilities are set to 0.

  – Model *topology* is the # of states, and *allowed* (i.e. $a_{kl} \neq 0$) transitions.

Sometimes omit begin state, in which case need *initiation probabilities* ($p_k$) for sequence starting in a given state
observed symbols

A → G → C → A → T

0 → π₁ → π₂ → π₃

unobserved states

\[ e_{\pi_1}(A) \quad e_{\pi_2}(G) \quad e_{\pi_3}(C) \quad \ldots \quad e_{\pi_i}(A) \quad \ldots \quad e_{\pi_n}(T) \]

\[ a_{\pi_0 \pi_1} \quad a_{\pi_{i-1} \pi_i} \quad a_{\pi_{i+1} \pi_i} \quad a_{\pi_{i+1} \pi_i+1} \]
• Prob that symbol occurs at given sequence position depends only on hidden state at that position, and is given by

emission probabilities:

\[ e_k(b) = \text{Prob(observed symbol is } b \mid \text{ curr state is } k) \]

(begin and end states do not emit symbols)

• Note that
  – there are no direct dependencies between observed symbols in the sequence, however
  – there are indirect dependencies implied by state dependencies
Where do the parameters come from?

• Can either
  – *define* parameter values *a priori*, or
  – *estimate* them from training data (observed sequences of the type to be modelled).

• Usually one does a mixture of both –
  – model topology is defined (some transitions set to 0), but
  – remaining parameters estimated
Hidden Markov Model

observed symbols

A → G → C → A → T

unobserved states

0 → \( \pi_1 \) → \( \pi_2 \) → \( \pi_3 \) → \( \pi_n \) → 0

\( e_{\pi_1}(A) \) → \( e_{\pi_2}(G) \) → \( e_{\pi_3}(C) \) → ... → \( e_{\pi_n}(T) \)

\( a_{\pi_0 \pi_1} \) → \( a_{\pi_1 \pi_2} \) → \( a_{\pi_2 \pi_3} \) → \( a_{\pi_3 \pi_4} \) → ... → \( a_{\pi_i \pi_{i+1}} \)
HMM Examples

• Site models:
  – “states” correspond to positions (columns in the tables). state $i$ transitions only to state $i+1$:
    • $a_{i,i+1} = 1$ for all $i$;
    • all other $a_{ij}$ are 0
  – emission probabilities are position-specific frequencies: values in frequency table columns
Topology for Site HMM: ‘allowed’ transitions (transits with non-zero prob – all are 1)
HMM for *C. elegans* 3’ Splice Sites

<table>
<thead>
<tr>
<th>Exon</th>
<th>Intron</th>
<th>3’ ss</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>3276</td>
<td>3516</td>
</tr>
<tr>
<td>C</td>
<td>970</td>
<td>648</td>
</tr>
<tr>
<td>G</td>
<td>593</td>
<td>575</td>
</tr>
<tr>
<td>T</td>
<td>3353</td>
<td>3453</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>0.400</th>
<th>0.429</th>
<th>0.282</th>
<th>0.058</th>
<th>0.008</th>
<th>0.092</th>
<th>0.029</th>
<th>1.000</th>
<th>0.000</th>
<th>0.410</th>
<th>0.293</th>
<th>0.307</th>
</tr>
</thead>
<tbody>
<tr>
<td>C</td>
<td>0.118</td>
<td>0.079</td>
<td>0.081</td>
<td>0.029</td>
<td>0.016</td>
<td>0.135</td>
<td>0.834</td>
<td>0.000</td>
<td>0.000</td>
<td>0.156</td>
<td>0.187</td>
<td>0.225</td>
</tr>
<tr>
<td>G</td>
<td>0.072</td>
<td>0.070</td>
<td>0.063</td>
<td>0.018</td>
<td>0.005</td>
<td>0.073</td>
<td>0.001</td>
<td>0.000</td>
<td>1.000</td>
<td>0.310</td>
<td>0.159</td>
<td>0.191</td>
</tr>
<tr>
<td>T</td>
<td>0.409</td>
<td>0.422</td>
<td>0.574</td>
<td>0.896</td>
<td>0.971</td>
<td>0.700</td>
<td>0.135</td>
<td>0.000</td>
<td>0.000</td>
<td>0.124</td>
<td>0.361</td>
<td>0.276</td>
</tr>
</tbody>
</table>

**CONSENSUS**

| W | W | W | T | T | T | t | c | C | A | G | r | w | w |

0 → 1 → 2 → 3 → 4 → 5 → 6 → 7 → 8 → 9 → 10 → 11 → 12

‘hidden’ states
– Can expand model to allow omission of nuc at some positions by including other (downstream) transitions (or via “silent states”)
– Can allow insertions by including additional states.
– transition probabilities no longer necessarily 1 or 0
Insertions & Deletions in Site Model

insertion state

other transitions correspond to deletions
Examples (cont’d) – 1-state HMMs

• single state, emitting residues with specified freqs:
  = ‘background’ model
Examples (cont’d) – 2-state HMMs

- if $a_{11}$ and $a_{22}$ are small (close to 0), and
  $a_{12}$ and $a_{21}$ are large (close to 1),
then get (nearly) periodic model with period 2; e.g.
  – dinucleotide repeat in DNA, or
  – (some) beta strands in proteins.

- if $a_{11}$ and $a_{22}$ large, and
  $a_{12}$ and $a_{21}$ small,
then get models of alternating regions of different compositions (specified by emission probabilities), e.g.
  – higher vs. lower G+C content regions (RNA genes in thermophilic bacteria); or
  – hydrophobic vs. hydrophilic regions of proteins (e.g. transmembrane domains).
A A T G C C T G G A T A A

G+C-rich state

A+T-rich state
2-state HMMs

• Can find most probable state decomposition (‘Viterbi path’) consistent with observed sequence

• Advantages over linked-list dynamic programming method (lecture 3) for finding high-scoring segments:
  – That method assumes you know appropriate parameters to find targeted regions; HMM method can estimate parameters.
  – HMM (easily) finds multiple segments
  – HMM can attach probabilities to alternative decompositions
  – HMM generalization to > 2 types of segments is easy – just allow more states!

• Disadvantage:
  – Markov assumption on state transitions implies geometric distribution for lengths of regions -- may not be appropriate
Hidden Markov Model

observed symbols

A G C

unobserved states

T

A

T

0 → π₁ → π₂ → π₃ → ... → πᵢ → πᵢ₊₁ → πₙ → 0

e_{π₁}(A) e_{π₂}(G) e_{π₃}(C) ... e_{πᵢ}(A) ... e_{πₙ}(T)

a_{0,π₁} a_{π₁,π₂} a_{π₂,π₃} a_{π₃,π₄} ... a_{πᵢ,πᵢ₊₁}
HMM Probabilities of Sequences

• Prob of sequence of states $\pi_1 \pi_2 \pi_3 \ldots \pi_n$ is
  $$a_{\pi_1} a_{\pi_1 \pi_2} a_{\pi_2 \pi_3} a_{\pi_3 \pi_4} \ldots a_{\pi_{n-1} \pi_n}.$$

• Prob of seq of observed symbols $b_1 b_2 b_3 \ldots b_n$, conditional on state sequence is
  $$e_{\pi_1}(b_1)e_{\pi_2}(b_2)e_{\pi_3}(b_3)\ldots e_{\pi_n}(b_n)$$

• Joint probability $= a_{\pi_1} \prod_{i=1}^{n} a_{\pi_i \pi_{i+1}} e_{\pi_i}(b_i)$
  (define $a_{\pi_n \pi_{n+1}}$ to be 1)

• (Unconditional) prob of observed sequence
  = sum (of joint probs) over all possible state paths
  – not practical to compute directly, by ‘brute force’! We will use dynamic programming.
Computing HMM Probabilities

• WDAG structure for sequence HMMs:
  – for $i^{th}$ position in seq ($i = 1, \ldots, n$), have 2 nodes for each state:
    • total # nodes = $2ns + 1$, where $n =$ seq length, $s =$ # states
  – Pair of nodes for a given state at $i^{th}$ position is connected by an emission edge
    • Weight is the emission prob for $i^{th}$ observed residue.
    • Can omit node pair if emission prob = 0.
  – Have transition edges connecting (right-hand) state nodes at position $i$ with (left-hand) state nodes at position $i+1$
    • Weights are transition probs
    • Can omit edges with transition prob = 0.
WDAG for 3-state HMM, length $n$ sequence

weights are emission probabilities $e_k(b_i)$ for $i^{th}$ residue $b_i$

weights are transition probabilities $a_{kl}$

$b_{i-1}$
position $i-1$

$b_i$
position $i$

$b_{i+1}$
position $i+1$
Beginning of Graph
• **Paths** through graph from begin node to end node correspond to *sequences of states*

• **Product weight** along path
  \[= \text{joint probability of state sequence} \& \text{observed symbol sequence}\]

• Sum of (product) path weights, over all paths,
  \[= \text{probability of observed sequence}\]

• Sum of (product) path weights over
  – all paths going through a particular node, or
  – all paths that include a particular edge,
  \[= \text{posterior probability of that edge or node}\]

• **Highest-weight path** = *highest probability state sequence*
Path Weights

\[ e_i(b_{i-1}) \]

\[ e_2(b_i) \]

\[ e_3(b_{i+1}) \]

position \( i-1 \)  
position \( i \)  
position \( i+1 \)
• By general results on WDAGs, can use dynamic programming to find highest weight path:
  = “Viterbi algorithm” to find highest probability path (most probable “parse”)
  – in this case can use log probabilities & sum weights
  – (N.B. paths are constrained to begin at the begin node!)
The Viterbi path is the most probable parse!