Computational Biology

Lecture 10
Hidden Markov Model

A Hidden Markov Model HMM is defined as:

- A set of *hidden* states
- [transitional probabilities]
  For each pair of states $i$ and $j$, a transition probability $a_{ij}$.
- $\sum_j a_{ij} = 1$

- An alphabet of symbols $\Sigma$
- [emission probabilities]
  For each state $k$, and symbol $b$
  $e_k(b) = p(x_i = b \mid \pi_i = k)$
  [now we use variable $\pi$ for states and variable $x$ for symbols]
- $\sum_{b \in \Sigma} e_k(b) = 1$ for each state $k$

Markov property:

$p(\pi_n = j \mid x_0 \ldots x_m, \pi_0 \ldots \pi_{m-1}, \pi_m = i) =$
$p(\pi_n = j \mid \pi_m = i)$ $m < n$
if $m = n - 1$, then this is $a_{ij}$
HMM for CpG islands
Questions with HMMs

• **Evaluation**: given $x$, what is the probability $p(x)$ that it was produced by the model?

• **Decoding**: given $x$, what is the most probable path that produces $x$ in the model?

• **Learning**: given $x$, what are the parameters (transitional probabilities and emission probabilities) of the model that maximize $p(x)$. 

Saad Mneimneh
http:// engr.smu.edu/~saad/
Viterbi *decoding* algorithm

- **Initialization**
  \[ \nu_0(0) = 1, \nu_k(0) = 0 \text{ for } k > 0 \]

- **Main iteration**
  for \( i = 1 \ldots n \)
  \[ \nu_i(i) = e_i(x_i) \cdot \max_k (\nu_k(i - 1) \cdot a_{kl}) \]
  \[ \text{ptr}_i(i) = \arg\max_k (\nu_k(i - 1) a_{kl}) \]

- **Termination**
  \[ \rho(x, \pi^*) = \max_k (\nu_k(n) a_{k0}) \]

Time = \( O(k^2n) \)
Space = \( O(kn) \)
Computing $p(x)$

- Before, $p(x) = a_{sx1} \prod_{i=2}^{n} a_{x_{i-1} x_i}$

- Now, $p(x) = \Sigma_{\pi} p(x, \pi)$

- Enumerating all $\pi$ is exponential!

- Use Viterbi, same as before, but change $\text{max}$ to $\Sigma$
Forward evaluation algorithm

- Let $f_j(i) = p(x_1 \ldots x_i, \pi_j = l)$

- Then,
  
  $$f_j(i) = e_j(x_i) \sum_k f_k(i - 1) a_{kl}$$

\[\text{All possible states}\]

$\pi_i = l$ with prob. $e_j(x_i)$
Derivation

\[ f_i(i) \]
\[ = \sum_{\pi_1 \ldots \pi_{i-1}} p(x_1 \ldots x_i, \pi_1 \ldots \pi_{i-1}, \pi_i = l) \]
\[ = \sum_{\pi_1 \ldots \pi_{i-1}} p(x_i, \pi_i = l, x_1 \ldots x_{i-1}, \pi_1 \ldots \pi_{i-1}) \]
\[ = \sum_{\pi_1 \ldots \pi_{i-1}} p(x_i, \pi_i = l | x_1 \ldots x_{i-1}, \pi_1 \ldots \pi_{i-1}) \cdot p(x_1 \ldots x_{i-1}, \pi_1 \ldots \pi_{i-1}) \]
\[ = \sum_{\pi_1 \ldots \pi_{i-1}} p(x_i, \pi_i = l | \pi_{i-1}) \cdot p(x_1 \ldots x_{i-1}, \pi_1 \ldots \pi_{i-1}) \]
\[ = \sum_{\pi_1 \ldots \pi_{i-2}, k} p(x_i, \pi_i = l | \pi_{i-1} = k) \cdot p(x_1 \ldots x_{i-1}, \pi_1 \ldots \pi_{i-2}, \pi_{i-1} = k) \]
\[ = \sum_{\pi_1 \ldots \pi_{i-2}, k} e(x_i) a_{kl} \cdot p(x_1 \ldots x_{i-1}, \pi_1 \ldots \pi_{i-2}, \pi_{i-1} = k) \]
\[ = \sum_k \sum_{\pi_1 \ldots \pi_{i-2}} e(x_i) a_{kl} \cdot p(x_1 \ldots x_{i-1}, \pi_1 \ldots \pi_{i-2}, \pi_{i-1} = k) \]
\[ = \sum_k \sum_{\pi_1 \ldots \pi_{i-2}} p(x_1 \ldots x_{i-1}, \pi_1 \ldots \pi_{i-2}, \pi_{i-1} = k) \cdot e(x_i) a_{kl} \]
\[ = \sum_k f_k(i-1) e(x_i) a_{kl} = e(x_i) \cdot \sum_k f_k(i-1) a_{kl} \]
Forward evaluation algorithm

• Initialization
  \[ f_0(0) = 1, \quad f_k(0) = 0 \text{ for } k > 0 \]

• Main iteration
  for \( i = 1 \ldots n \)
  \[ f_i(i) = e_i(x_i) \cdot \sum_k (f_k(i - 1) \cdot a_{k|i}) \]

• Termination
  \[ p(x) = \sum_k f_k(n) a_{k|0} \]
Problem of small numbers

• In Viterbi and Forward algorithm we multiply probabilities $\rightarrow$ numbers will soon be very small and we loose precision.

• Use log space $\rightarrow$ addition instead of multiplication.
Log space Viterbi

\[ v_j(i) = e_j(x_i) \cdot \max_k (v_k(i - 1) \cdot a_{ki}) \]

Let \( V_j(i) = \log v_j(i) \)

\[
V_j(i) = \log [e_j(x_i) \cdot \max_k (v_k(i - 1) \cdot a_{ki})] \\
= \log e_j(x_i) + \log [\max_k (v_k(i - 1) \cdot a_{ki})] \\
= \log e_j(x_i) + \max_k \log [(v_k(i - 1) \cdot a_{ki})] \\
= \log e_j(x_i) + \max_k (V_k(i - 1) + \log a_{ki})
\]
Log space Forward

\[ f_i(i) = e_i(x_i) \cdot \Sigma_k (f_k(i - 1) \cdot a_{kl}) \]

Let \( F_i(i) = \log f_i(i) \)

\[ F_i(i) = \log [e_i(x_i) \cdot \Sigma_k (f_k(i - 1) \cdot a_{kl})] \]
\[ = \log e_i(x_i) + \log \Sigma_k (f_k(i - 1) \cdot a_{kl}) \]
\[ = \log e_i(x_i) + \Sigma_k \log [(f_k(i - 1) \cdot a_{kl})] \]
\[ = \log e_i(x_i) + \log \Sigma_k e^{(F_k(i - 1) + \log a_{kl})} \]
Back to the most probable path

- The Viterbi algorithm finds it!
- The most probable path might not be the most appropriate basis for judgment.
- We might want, for instance, the most probable state for an observation \( x_i \).
- More generally, we are interested in \( p(\pi_i = k \mid x) \)
Computing $p(\pi_i = k \mid x)$

- $p(\pi_i = k \mid x) = p(x, \pi_i = k)/p(x)$

- I know how to compute $p(x)$: forward alg.

- $p(x, \pi_i = k)$
  
  $= p(x_1 \ldots x_i, \pi_i = k).p(x_{i+1} \ldots x_n \mid x_1 \ldots x_i, \pi_i = k)$
  
  $= p(x_1 \ldots x_i, \pi_i = k).p(x_{i+1} \ldots x_n \mid \pi_i = k)$
  
  $= f_k(i).b_k(i)$
Backward *evaluation* algorithm

- **Initialization**
  \[ b_k(n) = a_{k0} \text{ for all } k \]

- **Main iteration**
  for \( i = n - 1 \ldots 1 \)
  \[ b_k(i) = \sum_j a_{kj} e_j(x_{i+1}) b_j(i+1) \]

- **Termination**
  \[ p(x) = \sum_k a_{0k} e_k(x_1) b_k(1) \]

Time = \( O(k^2 n) \)
Space = \( O(kn) \)
Learning (training the HMM)

• Let $\theta$ be the parameters of the HMM (transition probabilities and emission probabilities, the $a$’s and $e$’s)

• Given independent sequences $x^1, \ldots, x^n$, we would like to find $\theta$ that will maximize:

$$\log p(x^1 \ldots x^n \mid \theta) = \sum_{j=1}^n \log p(x^j \mid \theta)$$

This is called the maximum likelihood parameters.
State sequence is known

• Assume the path for each $x^i$ is known
  – For instance, we have sequences in which CpG islands are already labeled

• Paths are known, let
  – $A_{kl} =$ number of transitions from $k$ to $l$
  – $E_k(b) =$ number of times $b$ emitted in state $k$

• The maximum likelihood parameters are given by:
  – $a_{kl} = A_{kl} / \Sigma_l A_{kl}$
  – $e_k(b) = E_k(b) / \Sigma_b E_k(b)$
Maximum likelihood from counts

• Assume we have a sequence of independent observations \(x_1\ldots x_n\) and that we count \(n_i\) occurrences of outcome \(i, i=1\ldots k\).

• Let \(\theta_i = \text{probability of } i\).

• Then \(\theta^{\text{ML}} = \{\theta_i=n/n, i=1\ldots k\}\) is the maximum likelihood solution for \(\theta\).

• Consider any other \(\theta\). We want to show that
  \[
  p(x \mid \theta^{\text{ML}}) > p(x \mid \theta)
  \]
Proof

\[
\log \frac{p(x | \theta^{ML})}{p(x | \theta)} = \log \frac{\prod_i (\theta_i^{ML})^{n_i}}{\prod_i \theta_i^{n_i}} \\
= \sum_i n_i \log \frac{\theta_i^{ML}}{\theta_i} \\
= n \sum_i \theta_i^{ML} \log \frac{\theta_i^{ML}}{\theta_i} > 0
\]

The last summation is the relative entropy of \(\theta^{ML}\) and \(\theta\) which is always positive and 0 iff \(\theta^{ML} = \theta\) (from information theory)
Some problems

• Maximum likelihood are vulnerable to overfitting if insufficient data.

• For instance, if a state $k$ was never used in the set of training sequences, then
  – $a_{kl} = 0$ for all $l$
  – $e_k(b) = 0$ for all $b$

• To avoid such problem, start with pseudocounts of $r_{kl}$ for $A_{kl}$ and $r_k(b)$ for $E_k(b)$.

• Large pseudocount indicates strong prior belief about the probabilities (will require more data to modify)

• Small pseudocount just to avoid zero probability
Example

Dishonest Casino HMM

\[ r_{0F} = r_{0L} = r_{F0} = r_{L0} = 1; \quad [\text{avoid zero probability}] \]
\[ r_{FL} = r_{LF} = r_{FF} = r_{LL} = 1; \quad [\text{avoid zero probability}] \]

\[ r_F(1) = r_F(2) = \ldots = r_F(6) = 20 \quad [\text{strong belief that fair is fair}] \]

\[ r_L(1) = r_L(2) = \ldots = r_L(6) = 5 \quad [\text{wait and see for loaded}] \]
New species comes in...

- New species with different distribution of CpG islands.
- We do not have labeled genomic sequences for the new species.
- Need to find maximum likelihood $\theta$ of HMM without knowing the paths!
Baum–Welsh algorithm

start at iteration 0 with some \( \theta \), call it \( \theta^0 \)
\( L^0 \leftarrow \log p(x^1 \ldots x^n \mid \theta^0) \)
\( i \leftarrow 0 \)

repeat
  \( i \leftarrow i + 1 \)
  \( A_{kl}^i \leftarrow E \left[ A_{kl} \mid x^1 \ldots x^n, \theta^{i-1} \right] \) (expected value)
  \( E_{k}(b)^i \leftarrow E \left[ E_k(b) \mid x^1 \ldots x^n, \theta^{i-1} \right] \) (expected value)

  calculate \( \theta^i \) using maximum likelihood estimators from counts \( A_{kl}^i \) and \( E_{k}(b)^i \) as before.

  \( L^i \leftarrow \log p(x^1 \ldots x^n \mid \theta^i) \) (new likelihood)

until \( L^i - L^{i-1} < \text{threshold} \)
What is the guarantee?

• Baum–Welsh algorithm is a special case of a general algorithm known as Expectation Maximization (EM)

• EM guarantees that \( p(X | \theta^{i+1}) \geq p(X | \theta^i) \)

• It will therefore converge to a local maximum (not necessarily the maximum)
We need to...

Compute:

- $\mathbb{E} [A_{kl} \mid x^1 \ldots x^n, \theta]$

- $\mathbb{E} [E_k(b) \mid x^1 \ldots x^n, \theta]$
\[ \mathbb{E} \left[ A_{kl} \mid x^1 \ldots x^n, \theta \right] \]

By linearity of expectation:
\[ \mathbb{E} \left[ A_{kl} \mid x^1 \ldots x^n, \theta \right] = \sum_j \mathbb{E} \left[ A_{kl} \mid x^j, \theta \right] \]

By linearity of expectation, again:
\[ \mathbb{E} \left[ A_{kl} \mid x^j, \theta \right] = \sum_i \mathbb{E}[^\# \text{ of } k \rightarrow l \text{ at } x^j_i \mid x^j, \theta] \]
\[ = \sum_i p(k \rightarrow l \text{ at } x^j_i \mid x^j, \theta) \]

\[ p(k \rightarrow l \text{ at } x^j_i \mid x^j, \theta) = p(\pi_i = k, \pi_{i+1} = l \mid x^j, \theta) \]

\[ p(\pi_i = k, \pi_{i+1} = l \mid x^j, \theta) = p(\pi_i = k, \pi_{i+1} = l, x^j_i / \theta) / p(x^j_i / \theta) \]
\[ = f_k^j(i) a_{kl} e_i(x_{i+1}) b_{j(i+1)} / p(x^j_i / \theta) \]
We get:

\[ E[A_{kl} \mid x^1 \ldots x^j, \theta] = \sum_j \frac{1}{p(x^j \mid \theta)} \sum_i f_k^j(i) a_{kl} e_l(x_{i+1}^j) b_l^j(i + 1) \]

\[ E[E_k(b) \mid x^1 \ldots x^j, \theta] = \sum_j \frac{1}{p(x^j \mid \theta)} \sum_{i|x_i^j = b} f_k^j(i) b_k^j(i) \]
Viterbi training

start at iteration 0 with some $\theta$, call it $\theta^0$

$i \leftarrow 0$

repeat

$i \leftarrow i + 1$

$A_{kl}^i \leftarrow$ number of transitions $k \rightarrow l$ on the most probable paths $\pi^1^*, \ldots, \pi^j^*$

$E_k(b)^i \leftarrow$ number of times $k$ emits $b$ on the most probable paths $\pi^1^*, \ldots, \pi^j^*$

calculate $\theta^i$ using maximum likelihood estimators from counts $A_{kl}^i$ and $E_k(b)^i$ as before.

until none of the optimal paths change
What is the guarantee

• It will converge

• It will not necessarily maximize the true likelihood $p(x_1...x^n | \theta)$, but $p(x_1...x^n | \theta, \pi_1^*, ..., \pi_j^*)$

• Usually performs less well than Baum–Welsh

• Practical, don’t have to perform Forward and Backward algorithms, only Viterbi!

• Makes sense if we are using only Viterbi decoding