Hidden Markov Models

Donald M. Pianto*

November 23, 2004

* dpianto@unb.br

Machine Learning Algorithms

- Algorithm receives training data
- Training data provides information about parameters (knowledge)
- Information can be used to analyze test data (use of knowledge)
- Increased amount of training data generally leads to more accurate parameter estimates
- HMM learns unknown probabilistic parameters from training samples and uses them to explain data

CG–Islands

- The least frequent dinucleotide in many genomes is *CG*
- *C* is easily methylated and tends to mutate to *T*
- Methylation often suppressed around genes in *CG–islands*
- We would like to define and locate these islands in a long sequence

"Fair Bet Casino"

- Game in which a dealer flips a coin and a player bets on the outcome (H or T)
- Dealer uses either a fair coin ($P^+(H) = \frac{1}{2}$, $P^+(T) = \frac{1}{2}$) or a biased coin ($P^-(H) = \frac{3}{4}$, $P^-(T) = \frac{1}{4}$)
- Probability of changing coins is 0.1 (doesnt want to get caught)
- Given a sequence of coin tosses ($x = x_1 x_2 x_3 \cdots x_n$ with $x_i \in \{H, T\}$), determine when the fair coin was used and when the biased coin was used ($\pi = \pi_1 \pi_2 \pi_3 \cdots \pi_n$, with $\pi_i \in \{+, -\}$)

Any sequence is a possible *expalanation of any other*

- A long string of heads could be generated by the fair coin
- A long string of tails could be generated by the biased coin
- Lets consider the probability that a given sequence, x, was generated by only the fair coin or only the biased coin
- If x is a sequence of n coin flips with k heads

$$P(x|+) = \prod_{i=1}^{n} P^{+}(x_i) = \frac{1}{2^n}$$
$$P(x|-) = \prod_{i=1}^{n} P^{-}(x_i) = \left(\frac{1}{4}\right)^{n-k} \left(\frac{3}{4}\right)^{k} = \frac{3^k}{4^n}$$

Which coin was more likely used?

- If $P(x|+) > P(x|-)$ then the fair coin was most likely used
- If $P(x|+) < P(x|-)$ then the biased coin was most likely used
- $P(x|+) = \frac{1}{2^n} = \frac{3^k}{4^n} = P(x|-)$ when $\frac{k}{n} = \frac{1}{\log_2(3)}$
- Log odds ratio: $\log_2 \frac{P(x|+)}{P(x|-)} = n - k \log_2 3$
- But, the coin changes! We could use a sliding window to determine if each subsequence was generated by a fair or biased coin, but this may give multiple assignments to the same base pair

A solution, Hidden Markov Models

- Abstract machine with k unknown states
- At each step of the machines operation it:
  1. chooses the next state to move to
  2. emits a symbol from a known alphabet Σ
- Each state has its own probabilities of transitions to other states and emission of symbols

Formal HMM Definition

- $\mathcal{M}(\Sigma, Q, A, E)$
- Q is a set of states each of which emits symbols from Σ
- $A = (a_{kl})$ is a $|Q| \times |Q|$ matrix describing the transition from state k to state l
- $E = (e_k(b))$ is a $|Q| \times |\Sigma|$ matrix describing the probability of emmiting symbol b from state k

The Fair Bet Casino HMM

- $\Sigma = \{0, 1\}$, where $T = 0$ and $H = 1$
- $Q = \{+, -\}$
- $a_{++} = a_{--} = 0.9$ and $a_{+-} = a_{-+} = 0.1$
- $e_+(0) = e_+(1) = \frac{1}{2}$, $e_-(0) = \frac{1}{4}$, and $e_-(1) = \frac{3}{4}$

x	0	1	0	1	1	1	0	1	0	0	1
π	$+$	$+$	$+$	$-$	$-$	$-$	$-$	$-$	$+$	$+$	$+$
$P(x_i\|\pi_i)$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{3}{4}$	$\frac{3}{4}$	$\frac{3}{4}$	$\frac{1}{4}$	$\frac{3}{4}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{2}$
$P(\pi_i \to \pi_{i+1})$	$\frac{9}{10}$	$\frac{9}{10}$	$\frac{1}{10}$	$\frac{9}{10}$	$\frac{9}{10}$	$\frac{9}{10}$	$\frac{9}{10}$	$\frac{1}{10}$	$\frac{9}{10}$	$\frac{9}{10}$	$-$

$$
\begin{aligned}
P(x,\pi) &= P(x|\pi)\cdot P(\pi) \\
&= P(\pi_0 \to \pi_1)\cdot \prod_{i=1}^{n} P(x_i|\pi_i)\cdot P(\pi_i \to \pi_{i+1}) \\
&= a_{\pi_0,\pi_1}\cdot \prod_{i=1}^{n} e_{\pi_i}(x_i)\cdot a_{\pi_i,\pi_{i+1}}
\end{aligned}
$$

The Decoding Problem

- **Input:** Sequence of observations $x = x_1 \cdots x_n$ generated by an HMM, $\mathcal{M}(\Sigma, Q, A, E)$.
- **Output:** A path that maximizes $P(\pi|x)$ over all paths π. Such a path will also maximize $P(x, \pi)$ for fixed x.

Remember: $P(x, \pi) = P(\pi|x)P(x) = P(x|\pi)P(\pi)$
and also: $P(x) = \sum_{\text{all paths } \pi} P(x|\pi)$

Viterbi's Decoding Algorithm

- Generate a directed acyclic graph with $|Q|$ rows and n columns where each vertex (k, i) is connected to all vertices in the next column, $(l, i+1)$, by an edge of weight $e_k(x_i) \cdot a_{k,l}$.
- Each path through the graph corresponds to a sequence π and the product of edge weights for a given path, π, is equal to $P(x, \pi)$.
- Define $s_{k,i}$ as the probability for the most likely path for the prefix $x_1 \cdots x_i$ that ends at state k.
- Then, $s_{l,i+1} = e_l(x_{i+1}) \cdot \max_{k \in Q}\{s_{k,i} \cdot a_{k,l}\}$

- Apply logarithms in order to sum instead of multiply edges.
- Algorithm runs in $O(n|Q|^2)$ time. (The number of edges in the graph.)

Alternative question

- What is the probability, $P(\pi_i = k|x)$ that the HMM was in state k when it emitted x_i?
- $P(\pi_i = k|x) = P(x, \pi_i = k)/P(x)$ where
  $P(x) = \sum_{\text{all paths } \pi} P(x|\pi)$ and
  $P(x, \pi_i = k) = \sum_{\text{all } \pi \text{ with } \pi_i = k} P(x, \pi)$
- Define $f_{k,i}$ as the probability of emitting the prefix $x_1 \cdots x_i$ and reaching the state $\pi_i = k$.
- $f_{k,i} = e_k(x_i) \cdot \sum_{l \in Q} f_{l,i-1} \cdot a_{lk}$
- Same as formula for $s_{k,i}$ with the $\max$ replaced by a sum.
- $f_{k,i} = P(x_1 \cdots x_i, \pi_i = k)$

- Define $b_{k,i}$ as the probability of being at state $\pi_i = k$ and emitting the suffix $x_{i+1} \cdots x_n$.
- $b_{k,i} = \sum_{l \in Q} e_l(x_{i+1}) \cdot b_{l,i+1} \cdot a_{kl}$
- $b_{k,i} = P(x_{i+1} \cdots x_n, \pi_i = k)$
- Given $f_{k,1} = e_k(x_1)/|Q|$ the forward algorithm is defined.
- Given $b_{k,n} = 1/|Q|$ the backward algorithm is defined.
- $P(x, \pi_i = k) = f_{k,i} \cdot b_{k,i}$
- $P(x) = \sum_{k \in Q} P(x, \pi_i = k)$

HMM Parameter Estimation

- **Input:** Q, Σ, and m training sequences $\{x^1, \ldots, x^m\}$.
- **Output:** A and E such that $\prod_{j=1}^{m} P(A, E|x^j)$ is maximized. If all parameter values are assumed equally probable, the values of A and E which maximize $\prod_{j=1}^{m} P(A, E|x^j)$ also maximize $\prod_{j=1}^{m} P(x^j|A, E)$.
- Remember:
  $P(x, A, E) = P(A, E|x)P(x) = P(x|A, E)P(A, E)$

Estimation with know π

Given π^j and x^j we can estimate

- $\hat{a}_{kl} = A_{kl}/(\sum_{q \in Q} A_{kq})$ and
- $\hat{e}_k(b) = E_k(b)/(\sum_{\sigma \in \Sigma} E_k(\sigma))$ where
- A_{kl} is the number of transitions from state k to state l and
- $E_k(b)$ is the number of times b was emitted from state k.

Estimation with unknown π

If the state sequence π is unknown:

1. Guess a sequence $\pi^{(0)}$
2. Set $i = 0$
3. Estimate $\hat{a}$ and $\hat{e}$ based on $\pi = \pi^{(i)}$.
4. Use Viterbi's decoding algorithm to estimate $\pi^{(i+1)}$ based on $\hat{a}$ and $\hat{e}$
5. If $\pi^{(i)}$ and $\pi^{(i+1)}$ are "close enough" to each other then stop ($\hat{a}$ and $\hat{e}$ are the estimates) otherwise increment i and go to step 3.

Profile HMM Alignment

Simplest profile is one that assigns a probability of a given nucleotide at each position of the sequence:

A	.72	.14	0	0	.72	.72	0	0
T	.14	.72	0	0	0	.14	.14	.86
G	.14	.14	.86	.44	0	.14	0	0
C	0	0	.14	.56	.28	0	.86	.14

To align a sequence of length n with a profile of length m as an HMM:

- $\Sigma = \{\mathbf{A}, \mathbf{T}, \mathbf{G}, \mathbf{C}\}$
- $Q = \{M_1, \ldots, M_m, I_0, \ldots, I_m, D_1, \ldots, D_m, \text{Begin}, \text{ End}\}$
- $e_{M_j}(a) = e_j(a)$ from the profile. $e_{I_j}(a) = p(a)$, the frequency of a in all sequences. $e_{D_j}(\emptyset) = 1$.
- In the affine gap penalty model, define a_{MI}, a_{IM}, and a_{II} such that $\log(a_{MI}a_{IM})$ equals the gap initiation penalty and $\log(a_{II})$ equals the gap extension penalty.

Define $v_j^M(i)$ as the log likelihood of the best path for matching $x_1 \cdots x_i$ to a profile HMM ending with x_i emitted by the state M_j. Define $v_j^I(i)$ and $v_j^D(i)$ similarly.

Then we have the following recurrence relation:

$$v_j^M(i) = log\left(\frac{e_{M_j}(x_i)}{p(x_i)}\right) + \max\left|\begin{array}{lcl} v_{j-1}^M(i-1) & + & \log(a_{M_{j-1},M_j}) \\ v_{j-1}^I(i-1) & + & \log(a_{I_{j-1},M_j}) \\ v_{j-1}^D(i-1) & + & \log(a_{D_{j-1},M_j}) \end{array}\right.$$

with similar expressions for I and D states.

Depending on the final value of the log likelihood we can determine a probability of the given sequence belonging to the profiled family.