

SEQUENCE VERIFICATION AND SEARCH – USE OF HIDDEN MARKOV MODELS

Chitta Baral

Department of Computer Science and Engg.
Arizona State University
Tempe, AZ 85287
chitta@asu.edu

<http://www.public.asu.edu/~cbaral>

April 30, 2003

Motivation

- Given a sequence: Does it belong to a particular family?
- Given a sequence does it contain a specific subsequence; if so where?
 - Finding genes in a genome.
- Example: CpG islands
 - Methylation is suppressed in short stretches of human genome, such as around the promoters (or ‘start’ regions) of many genes. In these regions we see more CpG dinucleotides than elsewhere. Such regions are called CpG islands.
 - Q1: Given a short genomic sequence, does it come from a CpG island?
 - Q2: Given a long sequence, how do we find the CpG islands in it, if there are any?

Markov Chains

- States: The last symbol seen
- Transition probabilities: $a_{st} = P(x_i = t | x_{i-1} = s)$
The probability that the i th symbol will be t , given that the $i-1$ th symbol is s .
- $P(x_1)$: Probability that the first symbol is x_1
With a dummy initial state \mathcal{B} , $P(x_1 = s) = a_{\mathcal{B}s}$ (also written as a_{0s})
- Let x be the sequence $x_1x_2 \dots x_l$.
$$P(x) = P(x_l, x_{l-1}, \dots, x_1) = P(x_l | x_{l-1}, \dots, x_1) P(x_{l-1}, \dots, x_1) = P(x_l | x_{l-1}, \dots, x_1) P(x_{l-1} | x_{l-2}, \dots, x_1) \dots P(x_1)$$
- Key property of Markov chains (called Markov property):
The probability of x_i only depends on x_{i-1} .
That means $P(x_i | x_{i-1}, \dots, x_1) = P(x_i | x_{i-1})$

- Hence, $P(x) = P(x_l|x_{l-1})P(x_{l-1}|x_{l-2}) \dots P(x_1) = P(x_1) \prod_{i=2}^l a_{x_{i-1}x_i}$

Using Markov chains for discrimination: Main idea

- We are given two different Markov chains (or models):
 - one about the kind of strings we are interested in (positive model) or belonging to our class of interest; and
 - the other about strings which are known to be not in our class of interest (negative model)
- Each model consists of the initial probability and transition probabilities.
- When we get a new string we can compute its probability with respect to both models.
- If the probability that it came from the positive model is *significantly* higher than the probability that it came from the negative model, then we can conclude that it belongs to our class of interest.

Using Markov chains for discrimination: learning the models

- Positive model:
 - Consider all sequences that are known to belong to our class of interest (say CpG islands).
 - Let c_{st}^+ be the number of times t followed s in the above mentioned sequences.
 - a_{st}^+ , the probability that t will follow s (in sequences belonging to our class of interest)
 - = the ratio of (i) the number of times t followed s and (ii) the number of times something followed s
 - = $\frac{c_{st}^+}{\sum_{t'} c_{st'}^+}$
 - Maximum likelihood method

- * The above method of estimating probability from data is called *maximum likelihood estimation*.
 - * Probability computed this way maximizes the total probability of all the sequences.
 - * In general, given a model with parameters θ and a set of data D , the maximum likelihood estimate for θ is that value which maximizes $P(D|\theta)$.
 - * Maximum likelihood method may give poor result if we have less data. (In that case Bayesian estimation that can use prior knowledge is better.)
- Negative model:
 - Consider all sequences that are known to not belong to our class of interest (say CpG islands).
 - a_{st}^- , the probability that t will follow s (in sequences not belonging to our class of interest) = $\frac{c_{st}^-}{\sum_{t'} c_{st'}^-}$

- Now given a new sequence x , by comparing $P(x|model+)$ and $P(x|model-)$ we can make a prediction about whether x is in our class of interest or not.
- $P(x|model+) = P(x_l, x_{l-1}, \dots, x_1) = \prod_{i=1}^l a_{x_{i-1}x_i}^+$
- Computing product of small numbers causes overflow. Thus instead of doing the ratio of $P(x|model+)$ and $P(x|model-)$, a better approach is to use their log-odds ratio.
- Also log-odds is easier to visualize than simply odds.
 - e.g., Suppose we are comparing a and b .
 - Case 1: $a = 10 b$. Then $a/b = 10$. $\log(a/b) = 1$.
 - Case 2: $b = 10 a$. Then $a/b = 0.1$ $\log(a/b) = -1$.
 - Case 3: $a = b$. Then $a/b = 1$. $\log(a/b) = 0$.
 - log-odds distinguishes case 2 and case 1 from case 3 better than the simple odds ratio.

- $$S(x) = \log \frac{P(x|model+)}{P(x|model-)} = \log \frac{\prod_{i=1}^l a_{x_{i-1}x_i}^+}{\prod_{i=1}^l a_{x_{i-1}x_i}^-} = \sum_{i=1}^l \log \frac{a_{x_{i-1}x_i}^+}{a_{x_{i-1}x_i}^-}$$

$$= \sum_{i=1}^l \beta_{x_{i-1}x_i}, \text{ where } \beta_{x_{i-1}x_i} = \log \frac{a_{x_{i-1}x_i}^+}{a_{x_{i-1}x_i}^-}$$
- To view the accuracy of using $S(x)$ for discrimination often a histogram of $S(x)$, normalized w.r.t. the length of x is constructed.

Hidden Markov Models (HMM): Motivation

- Suppose we have the Markov model described in the previous slides and we are given a large sequence \mathcal{S} .
- Our goal is to find fragments of that sequence which belongs to the class \mathcal{C} of sequences we are interested in.
- If we knew that all sequences in \mathcal{C} are of fixed size, say 100, then we can consider all fragments of \mathcal{S} of size 100 and using \mathcal{S} to find out which fragments are in \mathcal{S} .
- But, if sequences in \mathcal{C} are not of a fixed size, then a better approach is use Hidden Markov Models.
- Basic Idea
 - Combine Model + and Model - to a single model.
 - Have transitions between states in Model + and Model -.

- For example, if the alphabet is $\{A, T, C, G\}$ then the states in the combined model will be $A^+, T^+, C^+, G^+, A^-, T^-, C^-$ and G^- .
- In the combined model there will be also transitions between the states A^+, T^+, C^+, G^+ , and the states A^-, T^-, C^-, G^- .
- The combined model is called a Hidden Markov Model (HMM) because there is no longer a 1-1 correspondence between states and symbols.
- Given sequence \mathcal{S} its predicted path through the HMM will tell us which part of \mathcal{S} is predicted to belong to \mathcal{C} .

Formal definition of an HMM

- π : The path through an HMM (Note: x is the sequence of symbols)
- π_i : The i th state in the path π .
- $a_{kl} = P(\pi_i = l | \pi_{i-1} = k)$
probability that the i th state is l , given that the $i-1$ th state was k .
- a_{0k} : probability of starting in state k .
- a_{k0} : probability of ending in state k .
- $e_k(b) = P(x_i = b | \pi_i = k)$ (## emission probabilities)
probability that symbol b is seen when in state k .
- Joint probability of an observed sequence x and a state sequence π

$$P(x, \pi) = a_{0\pi_1} \prod_{i=1}^l e_{\pi_i}(x_i) a_{\pi_i \pi_{i+1}}$$
 Not useful in practice as in general we do not know π .

Most probable state path: the Viterbi algorithm

- Given x and the probabilities, we want to find the most probable path π^* , i.e., we want to find π^* for which $P(x, \pi)$ is maximum among all π s.

$$\pi^* = \operatorname{argmax}_{\pi} P(x, \pi)$$
- $v_k(i)$: the probability of the most probable path ending in state k with the i th observation.
- $v_j(i + 1)$ = the probability of the most probable path ending in state j with the $i + 1$ th observation

$$= \max_k (v_k(i) a_{kj} e_j(x_{i+1})) = e_j(x_{i+1}) \max_k (v_k(i) a_{kj})$$
- Initial Conditions: $v_0(0) = 1$. $v_k(0) = 0$, for $k \neq 0$.

- Dynamic programming algorithm: table with states as rows and the sequence x as column (with a 0th column).

v		C	G	C	G
0	$v_0(0) = 1$	$v_0(1) = 0$	$v_0(2) = 0$	$v_0(3) = 0$	$v_4(0) = 0$
$A_+ = 1$	$v_1(0) = 0$				
$C_+ = 2$	$v_2(0) = 0$				
$G_+ = 3$	$v_3(0) = 0$				
$T_+ = 4$	$v_4(0) = 0$				
$A_- = 5$	$v_5(0) = 0$				
$C_- = 6$	$v_6(0) = 0$				
$G_- = 7$	$v_7(0) = 0$				
$T_- = 8$	$v_8(0) = 0$				

- Table filling: $v_j(i) = e_j(x_i) \max_k (v_k(i-1) a_{kj})$
 Keep a pointer about which value lead to the maximum in computing $v_j(i)$.
- Find the maximum value in the rightmost column and trace back,

which gives the most probable path.

- When the algorithm is applied to a long sequence the derived optimal path may switch between the ‘+’ and ‘-’ components of the model, thus giving the precise boundaries of the predicted regions that belongs to our class of interest \mathcal{C} .

Computing the posterior probability: main steps

- Suppose we are given a string x and we need to find if x_i is in a part belonging to our class of interest \mathcal{C} .
- One way is to find the most probable path π^* corresponding to x , and then in that path find the state corresponding to x_i .
- But suppose we have paths π' , π'' and π''' probability 0.32, 0.31, and 0.3 and many others with much smaller probabilities then it does not seem wise to base our decision only on π' .
- Rather we should compute the expected value of x_i being from a + state. This can be computed by:

$$G(i|x) = \sum_k \text{is a + state } P(\pi_i = k|x)$$

- Next step: compute $P(\pi_i = k|x)$ which is equal to $\frac{P(x, \pi_i = k)}{P(x)} =$
 $\frac{P(x_1, \dots, x_i, \pi_i = k)P(x_{i+1} \dots x_l | x_1, \dots, x_i, \pi_i = k)}{P(x)} = \frac{P(x_1, \dots, x_i, \pi_i = k)P(x_{i+1} \dots x_l | \pi_i = k)}{P(x)} = \frac{f_k(i)b_k(i)}{P(x)}$

- Computing $P(x)$ by iterating over $f_k(i)$: the forward algorithm.
 - $f_k(i) = P(x_1, \dots, x_i, \pi_i = k)$, the probability of the observed sequence up to and including x_i together with π_i being k .
 - Recursive equation:

$$f_j(i + 1) = \sum_k f_k(i) a_{kj} e_j(x_{i+1}) = e_j(x_{i+1}) \sum_k f_k(i) a_{kj}$$
 - Initialization: $f_0(0) = 1$, $f_k(0) = 0$, for $k > 0$.
 - $P(x) = \sum_k f_k(l) a_{k0}$, where a_{k0} is the transition probability from state k to the end state.

- Computing $b_k(i)$: the backward algorithm
 - $b_k(i) = P(x_{i+1}, \dots, x_l | \pi_i = k)$, the probability of the observed sequence from x_{i+1} to x_l given that $\pi_i = k$.
 - Recursive equation:

$$b_j(i - 1) = P(x_i, x_{i+1}, \dots, x_l | \pi_{i-1} = j) = \sum_k a_{jk} e_k(x_i) b_k(i)$$
 - Initialization: $b_k(l) = a_{k0}$, for all k .

Parameter estimation for HMMs

- Two parts in learning the model:
 - the design of the structure (what states there are and how they are connected)
 - the assignment of parameter values, the transition and emission probabilities: a_{kl} and $e_k(b)$. (**Our focus**)
- Framework
 - Given training sequences: a set of example sequences (x^1, \dots, x^n)
 - assumption that the training sequences are independent. i.e., $\log P(x^1, \dots, x^n | \theta) = \sum_{j=1}^n \log P(x^j | \theta)$, where θ denotes the entire parameters.

- Estimation when the state sequence is known: maximum likelihood approach
 - example: we are given a set of sequences where the CpG islands are already labelled
 - A_{km} : number of times state s_m follows state s_k .
 - $E_k(b)$: number of times the next input is b when the current state is s_k .
 - $a_{km} = \frac{A_{km}}{\sum_{m'} A_{km'}}$ and $e_k(b) = \frac{E_k(b)}{\sum_{b'} E_k(b')}$
 - Use pseudo counts as the maximum likelihood approach may not be good in presence of insufficient data.
- Estimation when paths are unknown: Baum-Welch algorithm, a special case of *Expectation Maximization* (EM) algorithm
 - Start from some initial guess of the parameters: a_{km} and $e_k(b)$.
 - Compute A_{km} as the *expected* number of times s_m will follow s_k w.r.t. the training sequences.

- * Given a training sequence x , we need to compute $P(\pi_i = k, \pi_{i+1} = m | \theta, x)$ and sum it over the training sequences.
- *
$$P(\pi_i = k, \pi_{i+1} = m | \theta, x) = \frac{P(\pi_i = k, \pi_{i+1} = m, x | \theta)}{P(x | \theta)}$$

$$= \dots = \frac{f_k(i) a_{km} e_m(x_{i+1}) b_m(i+1)}{P(x)}$$
- *
$$A_{kl} = \sum_j \frac{1}{P(x^j)} \sum_i f_k^j(i) a_{km} e_m(x_{i+1}^j) b_m^j(i+1)$$
- Compute $E_k(b)$ as the *expected* number of times the next input will b in a state k , w.r.t. the training sequences.
- *
$$E_k(b) = \sum_j \frac{1}{P(x^j)} \sum_{\{i | x_i^j = b\}} f_k^j(i) b_k^j(i)$$
- Compute $a_{km} = \frac{A_{km}}{\sum_{m'} A_{km'}}$ and $e_k(b) = \frac{E_k(b)}{\sum_{b'} E_k(b')}$
- Iterate the last three steps until a stopping criteria is reached. (Note: Since we are in a continuous space we will never reach a maximum)
 - * a stopping criteria: change in total log likelihood is sufficiently small
 - * another: log likelihood can be normalized by the number of sequences and also by the sequence lengths.

Homework solution

- Show $P(\pi_i = k, \pi_{i+1} = m | \theta, x) = \frac{P(\pi_i=k, \pi_{i+1}=m, x | \theta)}{P(x | \theta)}$
 $= \dots = \frac{f_k(i) a_{km} e_m(x_{i+1}) b_m(i+1)}{P(x)}$
- Since the probability calculations are based on θ , we don't have to mention it explicitly.
- So all we need to show is
$$P(\pi_i = k, \pi_{i+1} = m, x) = f_k(i) a_{km} e_m(x_{i+1}) b_m(i + 1)$$

- LHS = $P(x_1, \dots, x_i, x_{i+1}, \dots, x_l, \pi_i = k, \pi_{i+1} = m)$
- = $P(x_1, \dots, x_i, \pi_i = k)P(x_{i+1}, \dots, x_l, \pi_{i+1} = m | x_1, \dots, x_i, \pi_i = k)$
- **** Using $P(A, B) = P(A)P(B|A)$ ****
- = $f_k(i)P(x_{i+1}, \dots, x_l, \pi_{i+1} = m | \pi_i = k)$
- **** Simple substitution ****
- = $f_k(i)P(\pi_{i+1} = m | \pi_i = k)P(x_{i+1}, \dots, x_l | \pi_{i+1} = m, \pi_i = k)$
- **** Using $P(A, B|C) = P(A|C)P(B|A, C)$ ****
- = $f_k(i)a_{km}P(x_{i+1}, \dots, x_l | \pi_{i+1} = m, \pi_i = k)$
- **** Simple substitution ****
- = $f_k(i)a_{km}P(x_{i+2}, \dots, x_l | \pi_{i+1} = m, \pi_i = k)$
- $P(x_{i+1} | \pi_{i+1} = m, \pi_i = k, x_{i+2}, \dots, x_l)$
- **** Using $P(A, B|C) = P(A|C)P(B|A, C)$ ****
- = $f_k(i)a_{km}P(x_{i+2}, \dots, x_l | \pi_{i+1} = m)P(x_{i+1} | \pi_{i+1} = m)$
- **** Using Markov property ****
- = $f_k(i)a_{km}e_m(x_{i+1})b_m(i+1)$
- **** Simple substitution ****