

Hidden Markov Models



Lecture 6, Thursday April 17, 2003

Review of Last Lecture



Lecture 6, Thursday April 17, 2003

Decoding

GIVEN $x = x_1 x_2 \dots x_N$

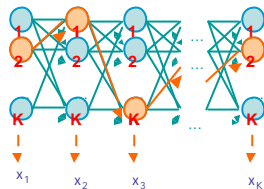
We want to find $\pi = \pi_1, \dots, \pi_N$,
such that $P[x, \pi]$ is maximized

$\pi' = \operatorname{argmax}_{\pi} P[x, \pi]$

We can use dynamic programming!

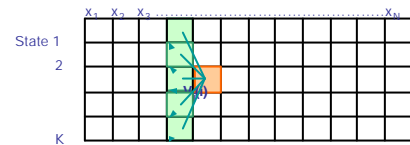
Let $V_k(i) = \max_{(\pi_1, \dots, \pi_{i-1})} P[x_1 \dots x_{i-1}, \pi_1, \dots, \pi_{i-1}, x_i, \pi_i = k]$

= Probability of most likely sequence of states ending at
state $\pi_i = k$



Lecture 6, Thursday April 17, 2003

The Viterbi Algorithm



Similar to "aligning" a set of states to a sequence

Time:
 $O(K^2N)$

Space:
 $O(KN)$

Lecture 6, Thursday April 17, 2003

Evaluation

Compute:

$P(x)$ Probability of x given the model

$P(x_1 \dots x_i)$ Probability of a substring of x given the model

$P(\pi_i = k | x)$ Probability that the i^{th} state is k , given x

Lecture 6, Thursday April 17, 2003

The Forward Algorithm

We can compute $f_k(i)$ for all k, i , using dynamic programming!

Initialization:
 $f_0(0) = 1$
 $f_k(0) = 0$, for all $k > 0$

Iteration:
 $f_i(i) = e_i(x_i) \sum_k f_k(i-1) a_{ki}$

Termination:
 $P(x) = \sum_k f_k(N) a_{0k}$

Where, a_{k0} is the probability that the terminating state is k (usually = a_{0k})

Lecture 6, Thursday April 17, 2003

The Backward Algorithm

We can compute $b_k(i)$ for all k, i , using dynamic programming

Initialization:

$$b_k(N) = a_{k0}, \text{ for all } k$$

Iteration:

$$b_k(i) = \sum_l e_i(x_{i+1}) a_{kl} b_l(i+1)$$

Termination:

$$P(x) = \sum_l a_{0l} e_l(x_1) b_l(1)$$

Lecture 6, Thursday April 17, 2003

Posterior Decoding

We can now calculate

$$P(\pi = k \mid x) = \frac{f_k(i) b_k(i)}{P(x)}$$

Then, we can ask

What is the most likely state at position i of sequence x :

Define π^* by Posterior Decoding:

$$\pi_i^* = \operatorname{argmax}_k P(\pi_i = k \mid x)$$

Lecture 6, Thursday April 17, 2003

Today

- Example: CpG Islands
- Learning

Lecture 6, Thursday April 17, 2003

Implementation Techniques

Viterbi: Sum-of-logs

$$V_i(i) = \log a_i(x_i) + \max_k [V_k(i-1) + \log a_{ki}]$$

Forward/Backward: Scaling by $c(i)$

One way to perform scaling:

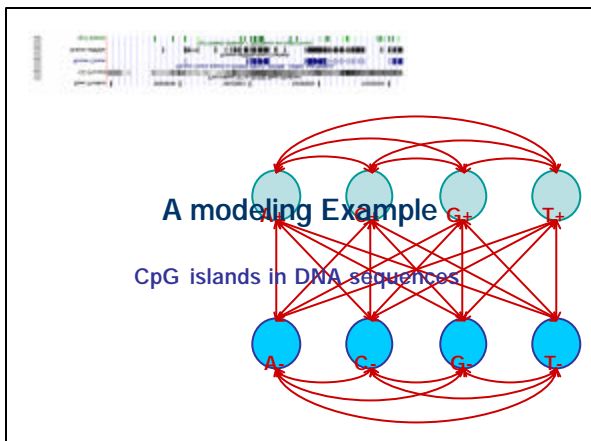
$$f_i(i) = c(i) \times e_i(x_i) \sum_k f_k(i-1) a_{ki}$$

where $c(i) = 1 / (\sum_k f_k(i))$

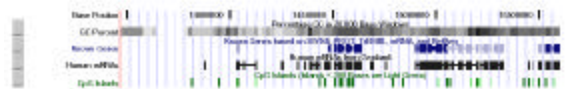
$b_i(i)$: use the same factors $c(i)$

Details in Rabiner's Tutorial on HMMs, 1989

Lecture 6, Thursday April 17, 2003



Example: CpG Islands



CpG nucleotides in the genome are frequently methylated

(Write CpG not to confuse with CG base pair)



Methylation often suppressed around genes, promoters

→ CpG islands

Lecture 6, Thursday April 17, 2003

Example: CpG Islands

The image displays two genomic tracks side-by-side. The top track is labeled 'Human' and the bottom track is labeled 'Mouse'. Both tracks show a genomic region with various annotations. The 'Human' track includes labels for 'Gene Position', 'CG Islands', 'Human CpG Islands', and 'Human CpG Islands (G+C Content > 50%)'. The 'Mouse' track includes labels for 'Gene Position', 'CG Islands', 'Mouse CpG Islands', and 'Mouse CpG Islands (G+C Content > 50%)'. The tracks show a high degree of conservation between the human and mouse genomes, with many CpG islands identified in both species.

In CpG islands,

CG is more frequent

Other pairs (AA, AG, AT...) have different frequencies

Question: Detect CpG islands computationally

Lecture 6, Thursday April 17, 2003

Lecture 6, Thursday April 17, 2003

A model of CpG Islands – (1) Architecture

The diagram illustrates the architecture of a model for CpG Islands. It consists of two rows of nodes, each representing a nucleotide state. The top row, labeled 'CpG Island', contains four light blue circular nodes labeled A+, C+, G+, and T+. The bottom row, labeled 'Not CpG Island', contains four dark blue circular nodes labeled A-, C-, G-, and T-. Every node in the top row is connected to every node in the bottom row by a red arrow, forming a complete bipartite graph. Additionally, there are red curved arrows connecting the nodes within the top row (A+ to C+, C+ to G+, G+ to T+, and A+ to T+) and within the bottom row (A- to C-, C- to G-, G- to T-, and A- to T-), representing transitions between adjacent nucleotides in the sequence.



A model of CpG Islands – (2) Transitions

How do we estimate the parameters of the model?

Emission probabilities: $1/10$

- Transition probabilities within CpG islands

Established from many known (experimentally verified)
CpG islands
(Training Set)

- Transition probabilities within other regions

Established from many known non-CpG islands

	A	C	G	T
A	180	274	426	120
C	171	368	274	188
G	161	332	375	125
T	679	355	384	182

	A	C	G	T
A	280	295	285	210
C	233	290	678	302
G	248	246	298	268
T	177	239	292	202

Lecture 6, Thursday April 17, 2003

Lecture 6, Thursday April 17, 2003

Paranthesis – log likelihoods

A better way to see effects of transitions:

Log likelihoods

$$L(u, v) = \log[P(uv \mid +) / P(uv \mid -)]$$

Given a region $x = x_1 \dots x_N$
 A quick-&-dirty way to decide whether entire x is CpG

	A	C	G	T
A	0.740	+0.419	+0.580	0.803
C	0.913	+0.302	+1.812	0.685
G	0.624	+0.461	+0.331	0.730
T	+1.169	+0.573	+0.393	0.679

$$P(x \text{ is CpG}) > P(x \text{ is not CpG}) \Rightarrow \sum_i L(x_i, x_{i+1}) > 0$$

Lecture 6, Thursday April 17, 2003

Lecture 6, Thursday April 17, 2003

A model of CpG Islands – (2) Transitions

What about transitions between (+) and (-) states?
They affect

- Avg. length of CpG island
- Avg. separation between two CpGIslands

Length distribution of region X:

$$\begin{aligned}P[|x| = 1] &= 1-p \\ P[|x| = 2] &= p(1-p) \\ &\vdots \\ P[|x| = k] &= p^k(1-p)\end{aligned}$$
$$E[|x|] = 1/(1-p)$$

Exponential distribution, with mean $1/(1-p)$

Lecture 6, Thursday April 17, 2003

A model of CpG Islands – (2) Transitions

No reason to favor exiting/entering (+) and (-) regions at a particular nucleotide

To determine transition probabilities between (+) and (-) states

1. Estimate average length of a CpG island: $l_{\text{CPG}} = 1/(1-p) \Rightarrow p = 1 - 1/l_{\text{CPG}}$
2. For each pair of (+) states k, l , let $a_{kl} \leftarrow p \times a_{kl}$
3. For each (+) state k , (-) state l , let $a_{kl} = (1-p)/4$
(better: take frequency of l in the (-) regions into account)
4. Do the same for (-) states

A problem with this model: CpG islands don't have exponential length distribution

This is a defect of HMMs – compensated with ease of analysis & computation

Lecture 6, Thursday April 17, 2003

Lecture 6, Thursday April 17, 2003

Applications of the model

Given a DNA region x ,

The **Viterbi** algorithm predicts locations of CpG islands

Given a nucleotide x_i , (say $x_i = A$)

The **Viterbi** parse tells whether x_i is in a CpG island in the most likely general scenario

The **Forward/Backward** algorithms can calculate

$$P(x_i \text{ is in CpG island}) = P(\pi_i = A^+ | x)$$

Posterior Decoding can assign locally optimal predictions of CpG islands

$$\pi_i^* = \operatorname{argmax}_k P(\pi_i = k | x)$$

Lecture 6, Thursday April 17, 2003

What if a new genome comes?

We just sequenced the porcupine genome

We know CpG islands play the same role in this genome

However, we have no known CpG islands for porcupines

We suspect the frequency and characteristics of CpG islands are quite different in porcupines

How do we adjust the parameters in our model?

- **LEARNING**



Lecture 6, Thursday April 17, 2003

Problem 3: Learning

Re-estimate the parameters of the model based on training data

Two learning scenarios

1. Estimation when the "right answer" is known

Examples:

GIVEN: a genomic region $x = x_1 \dots x_{1,000,000}$ where we have good (experimental) annotations of the CpG islands

GIVEN: the casino player allows us to observe him one evening, as he changes dice and produces 10,000 rolls

2. Estimation when the "right answer" is unknown

Examples:

GIVEN: the porcupine genome: we don't know how frequent are the CpG islands there, neither do we know their composition

GIVEN: 10,000 rolls of the casino player, but we don't see when he changes dice

QUESTION: Update the parameters θ of the model to maximize $P(x|\theta)$

Lecture 6, Thursday April 17, 2003

1. When the right answer is known

Given $x = x_1 \dots x_N$
for which the true $\pi = \pi_1 \dots \pi_N$ is known,

Define:

A_{kl} = # times $k \rightarrow l$ transition occurs in π
 $E_k(b)$ = # times state k in π emits b in x

We can show that the maximum likelihood parameters θ are:

$$a_{kl} = \frac{A_{kl}}{\sum_l A_{kl}} \quad e_k(b) = \frac{E_k(b)}{\sum_c E_k(c)}$$

Lecture 6, Thursday April 17, 2003

1. When the right answer is known

Intuition: When we know the underlying states,
Best estimate is the average frequency of transitions & emissions that occur in the training data

Drawback:

Given little data, there may be **overfitting**:
 $P(x|\theta)$ is maximized, but θ is unreasonable
0 probabilities - VERY BAD

Example:

Given 10 casino rolls, we observe
 $x = 2, 1, 5, 6, 1, 2, 3, 6, 2, 3$
 $\pi = F, F, F, F, F, F, F, F, F, F$

Then:

$$a_{FF} = 1; \quad a_{FL} = 0 \\ e_F(1) = e_F(3) = .2; \\ e_F(2) = .3; e_F(4) = 0; e_F(5) = e_F(6) = .1$$

Lecture 6, Thursday April 17, 2003

Pseudocounts

Solution for small training sets:

Add pseudocounts

$$\begin{aligned} A_{kl} &= \# \text{ times } k \rightarrow l \text{ transition occurs in } \pi & + r_{kl} \\ E_k(b) &= \# \text{ times state } k \text{ in } \pi \text{ emits } b \text{ in } x & + r_k(b) \end{aligned}$$

$r_{kl}, r_k(b)$ are pseudocounts representing our prior belief

Larger pseudocounts \Rightarrow Strong prior belief

Small pseudocounts ($\epsilon < 1$): just to avoid 0 probabilities

Lecture 6, Thursday April 17, 2003

Pseudocounts

Example: dishonest casino

We will observe player for one day, 500 rolls

Reasonable pseudocounts:

$$\begin{aligned} r_{0F} = r_{0L} = r_{F0} = r_{L0} &= 1; \\ r_{FL} = r_{LF} = r_{FF} = r_{LL} &= 1; \\ r_F(1) = r_F(2) = \dots = r_F(6) &= 20 & (\text{strong belief fair is fair}) \\ r_F(1) = r_F(2) = \dots = r_F(6) &= 5 & (\text{wait and see for loaded}) \end{aligned}$$

Above #s pretty arbitrary – assigning priors is an art

Lecture 6, Thursday April 17, 2003

2. When the right answer is unknown

We don't know the true $A_{kl}, E_k(b)$

Idea:

- We estimate our "best guess" on what $A_{kl}, E_k(b)$ are
- We update the parameters of the model, based on our guess
- We repeat

Lecture 6, Thursday April 17, 2003

2. When the right answer is unknown

Starting with our best guess of a model M , parameters θ :

Given $x = x_1 \dots x_N$
for which the true $\pi = \pi_1 \dots \pi_N$ is unknown,

We can get to a provably more likely parameter set θ

Principle: EXPECTATION MAXIMIZATION

1. Estimate $A_{kl}, E_k(b)$ in the training data
2. Update θ according to $A_{kl}, E_k(b)$
3. Repeat 1 & 2, until convergence

Lecture 6, Thursday April 17, 2003

Estimating new parameters

To estimate A_{kl} :

At each position i of sequence x ,

Find probability transition $k \rightarrow l$ is used:

$$P(\pi_i = k, \pi_{i+1} = l \mid x) = [1/P(x)] \times P(\pi_i = k, \pi_{i+1} = l, x_1 \dots x_N) = Q/P(x)$$

$$\begin{aligned} \text{where } Q &= P(x_1 \dots x_N, \pi_i = k, \pi_{i+1} = l, x_{i+1} \dots x_N) = \\ &= P(\pi_{i+1} = l, x_{i+1} \dots x_N \mid \pi_i = k) P(x_1 \dots x_N, \pi_i = k) = \\ &= P(\pi_{i+1} = l, x_{i+1} \dots x_N \mid \pi_i = k) f_k(i) = \\ &= P(x_{i+2} \dots x_N \mid \pi_{i+1} = l) P(x_{i+1} = l \mid \pi_{i+1} = l) P(\pi_{i+1} = l \mid \pi_i = k) f_k(i) = \\ &= b_l(i+1) e_l(x_{i+1}) a_{kl} f_k(i) \end{aligned}$$

$$\text{So: } P(\pi_i = k, \pi_{i+1} = l \mid x, q) = \frac{f_k(i) a_{kl} e_l(x_{i+1}) b_l(i+1)}{P(x \mid q)}$$

Lecture 6, Thursday April 17, 2003

Estimating new parameters

So,

$$A_{kl} = \sum_i P(\pi_i = k, \pi_{i+1} = l \mid x, \theta) = \sum_i \frac{f_k(i) a_{kl} e_l(x_{i+1}) b_l(i+1)}{P(x \mid \theta)}$$

Similarly,

$$E_k(b) = [1/P(x)] \sum_{\{i \mid x_i = b\}} f_k(i) b_k(i)$$

Lecture 6, Thursday April 17, 2003

Estimating new parameters

If we have several training sequences, x^1, \dots, x^M , each of length N ,

$$A_{k,l} = \sum_j \sum_i P(\pi_i = k, \pi_{i+1} = l \mid x, \theta) = \sum_j \sum_i \frac{f_k(l) a_{k,l} e(x_{i+1}) b_{l,i+1}}{P(x \mid \theta)}$$

Similarly,

$$E_k(b) = \sum_j (1/P(x)) \sum_{\{i \mid x_i = b\}} f_k(l) b_k(l)$$

Lecture 6, Thursday April 17, 2003

The Baum-Welch Algorithm

Initialization:

Pick the best-guess for model parameters
(or arbitrary)

Iteration:

Forward

Backward

Calculate $A_{k,l}$, $E_k(b)$

Calculate new model parameters $a_{k,l}$, $e_k(b)$

Calculate new log-likelihood $P(x \mid \theta)$

GUARANTEED TO BE HIGHER BY EXPECTATION-MAXIMIZATION

Until $P(x \mid \theta)$ does not change much

Lecture 6, Thursday April 17, 2003

The Baum-Welch Algorithm – comments

Time Complexity:

iterations $\times O(K^2N)$

- Guaranteed to increase the log likelihood of the model

$$P(\theta \mid x) = P(x, \theta) / P(x) = P(x \mid \theta) / (P(x) P(\theta))$$

- Not guaranteed to find globally best parameters

Converges to local optimum, depending on initial conditions

- Too many parameters / too large model: Overtraining

Lecture 6, Thursday April 17, 2003

Alternative: Viterbi Training

Initialization: Same

Iteration:

Perform Viterbi, to find π^*

Calculate $A_{k,l}$, $E_k(b)$ according to π^* + pseudocounts

Calculate the new parameters $a_{k,l}$, $e_k(b)$

Until convergence

Notes:

- Convergence is guaranteed – Why?
- Does not maximize $P(x \mid \theta)$
- In general, worse performance than Baum-Welch
- Convenient – when interested in Viterbi parsing, no need to implement additional procedures (Forward, Backward)!!

Lecture 6, Thursday April 17, 2003

Exercise – Submit any time – Groups up to 3

- Implement a HMM for the dishonest casino (or any other simple process you feel like)
- Generate training sequences with the model
- Implement Baum-Welch and Viterbi training
- Show a few sets of initial parameters such that
 - Baum-Welch and Viterbi differ significantly, and/or
 - Baum-Welch converges to parameters close to the model, and to unreasonable parameters, depending on initial parameters

- Do not use 0-probability transitions
- Do not use 0s in the initial parameters
- Do use pseudocounts in Viterbi

This exercise will replace the 1-3 lowest problems, depending on thoroughness

Lecture 6, Thursday April 17, 2003