

Hidden Markov Models

Lecture 5, Tuesday April 15, 2003



Review of Last Lecture

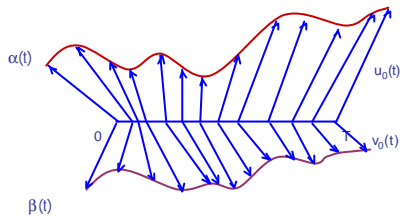
Lecture 2, Thursday April 3, 2003



Time Warping

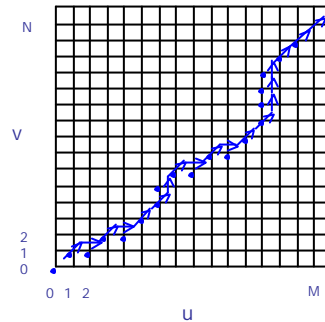
Definition: $\alpha(u)$, $\beta(v)$ are connected by an approximate continuous time warping (u_0, v_0) , if:

u_0, v_0 are strictly increasing functions on $[0, T]$, and $\alpha(u_0(t)) \approx \beta(v_0(t))$ for $0 \leq t \leq T$



Lecture 5, Tuesday April 15, 2003

Time Warping



Define possible steps:

$(\Delta u, \Delta v)$ is the possible difference of u and v

between steps $h-1$ and h

$$(\Delta u, \Delta v) = \begin{cases} (1, 0) \\ (1, 1) \\ (0, 1) \end{cases}$$

Lecture 5, Tuesday April 15, 2003

Definition of a hidden Markov model

Definition: A hidden Markov model (HMM)

- Alphabet $\Sigma = \{b_1, b_2, \dots, b_M\}$
- Set of states $Q = \{1, \dots, K\}$
- Transition probabilities between any two states

a_{ij} = transition prob from state i to state j

$a_{i1} + \dots + a_{iK} = 1$, for all states $i = 1 \dots K$

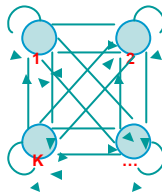
- Start probabilities a_{0i}

$a_{01} + \dots + a_{0K} = 1$

- Emission probabilities within each state

$e_i(b) = P(x_i = b \mid \pi_i = i)$

$e_i(b_1) + \dots + e_i(b_M) = 1$, for all states $i = 1 \dots K$



Lecture 5, Tuesday April 15, 2003

The three main questions on HMMs

1. Evaluation

GIVEN a HMM M , and a sequence x ,
FIND $\text{Prob}[x \mid M]$

2. Decoding

GIVEN a HMM M , and a sequence x ,
FIND the sequence π of states that maximizes $P[x, \pi \mid M]$

3. Learning

GIVEN a HMM M , with unspecified transition/emission probs., and a sequence x ,

FIND parameters $\theta = (e_i(\cdot), a_{ij})$ that maximize $P[x \mid \theta]$

Lecture 5, Tuesday April 15, 2003

Today

- Decoding
- Evaluation

Lecture 5, Tuesday April 15, 2003

Problem 1: Decoding

Find the best parse of a sequence

Decoding

GIVEN $x = x_1 x_2 \dots x_N$

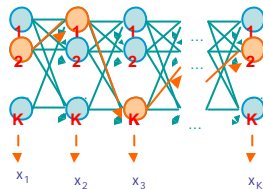
We want to find $\pi = \pi_1, \dots, \pi_N$, such that $P[x, \pi]$ is maximized

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

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 5, Tuesday April 15, 2003

Decoding – main idea

Given that for all states k , and for a fixed position i ,

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

What is $V_k(i+1)$?

From definition,

$$\begin{aligned} V_k(i+1) &= \max_{(\pi_1, \dots, \pi_i)} P[x_1 \dots x_i, \pi_1, \dots, \pi_i, x_{i+1}, \pi_{i+1} = k] \\ &= \max_{(\pi_1, \dots, \pi_i)} P(x_{i+1}, \pi_{i+1} = k \mid x_1 \dots x_i, \pi_1, \dots, \pi_i) P[x_1 \dots x_i, \pi_1, \dots, \pi_i] \\ &= \max_{(\pi_1, \dots, \pi_i)} P(x_{i+1}, \pi_{i+1} = k \mid \pi_i) P[x_1 \dots x_i, \pi_1, \dots, \pi_i, \pi_i] \\ &= \max_k P(x_{i+1}, \pi_{i+1} = k \mid \pi_i = k) \max_{(\pi_1, \dots, \pi_{i-1})} P[x_1 \dots x_{i-1}, \pi_1, \dots, \pi_{i-1}, x_i, \pi_i = k] \\ &= e_i(x_{i+1}) \max_k a_{kl} V_k(i) \end{aligned}$$

Lecture 5, Tuesday April 15, 2003

The Viterbi Algorithm

Input: $x = x_1 \dots x_N$

Initialization:

$$\begin{aligned} V_0(i) &= 1 \\ V_k(i) &= 0, \text{ for all } k > 0 \end{aligned} \quad (0 \text{ is the imaginary first position})$$

Iteration:

$$\begin{aligned} V_j(i) &= e_j(x_i) \times \max_k a_{kj} V_k(i-1) \\ \text{Ptr}_j(i) &= \operatorname{argmax}_k a_{kj} V_k(i-1) \end{aligned}$$

Termination:

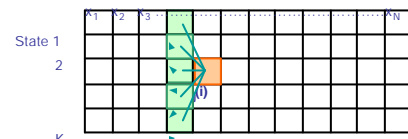
$$P(x, \pi^*) = \max_k V_k(N)$$

Traceback:

$$\begin{aligned} \pi_N^* &= \operatorname{argmax}_k V_k(N) \\ \pi_{i-1}^* &= \text{Ptr}_{\pi_i^*}(i) \end{aligned}$$

Lecture 5, Tuesday April 15, 2003

The Viterbi Algorithm



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

Time:

$$O(K^2 N)$$

Space:

$$O(KN)$$

Lecture 5, Tuesday April 15, 2003

Viterbi Algorithm – a practical detail

Underflows are a significant problem

$$P[x_1, \dots, x_i, \pi_1, \dots, \pi_i] = a_{0\pi_1} a_{\pi_1\pi_2} \dots a_{\pi_i} e_{\pi_1}(x_1) \dots e_{\pi_i}(x_i)$$

These numbers become extremely small – underflow

Solution: Take the logs of all values

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

Lecture 5, Tuesday April 15, 2003

Example

Let x be a sequence with a portion of $\sim 1/6$ 6's, followed by a portion of $\sim 1/2$ 6's...

$x = 123456123456\dots 123456626364656\dots 1626364656$

Then, it is not hard to show that optimal parse is (exercise):

FFF.....FFF LLL.....LLL

6 nucleotides "123456" parsed as F, contribute $.95^6 \times (1/6)^6 = 1.6 \times 10^{-5}$
 parsed as L, contribute $.95^6 \times (1/2)^1 \times (1/10)^5 = 0.4 \times 10^{-5}$

"162636" parsed as F, contribute $.95^6 \times (1/6)^6 = 1.6 \times 10^{-5}$
 parsed as L, contribute $.95^6 \times (1/2)^3 \times (1/10)^3 = 9.0 \times 10^{-5}$

Lecture 5, Tuesday April 15, 2003

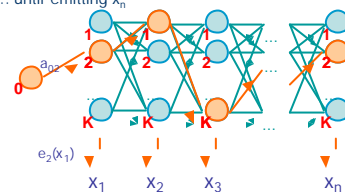
Problem 2: Evaluation

Find the likelihood a sequence is generated by the model

Generating a sequence by the model

Given a HMM, we can generate a sequence of length n as follows:

1. Start at state π_1 according to prob $a_{0\pi_1}$
2. Emit letter x_1 according to prob $e_{\pi_1}(x_1)$
3. Go to state π_2 according to prob $a_{\pi_1\pi_2}$
4. ... until emitting x_n



Lecture 5, Tuesday April 15, 2003

A couple of questions

Given a sequence x ,

- What is the probability that x was generated by the model?
- Given a position i , what is the most likely state that emitted x_i ?

Example: the dishonest casino

Say $x = 12341623162616364616234161221341$

Most likely path: $\pi = FF\dots FL$

However: marked letters more likely to be L than unmarked letters

Lecture 5, Tuesday April 15, 2003

Evaluation

We will develop algorithms that allow us to compute:

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

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

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

A more refined measure of which states x may be in

Lecture 5, Tuesday April 15, 2003

The Forward Algorithm

We want to calculate

$P(x)$ = probability of x , given the HMM

Sum over all possible ways of generating x :

$$P(x) = \sum_{\pi} P(x, \pi) = \sum_{\pi} P(x | \pi) P(\pi)$$

To avoid summing over an exponential number of paths π , define

$$f_k(i) = P(x_1 \dots x_i, \pi_i = k) \quad (\text{the forward probability})$$

Lecture 5, Tuesday April 15, 2003

The Forward Algorithm – derivation

Define the forward probability:

$$f_i(i) = P(x_1 \dots x_i, \pi_i = i)$$

$$= \sum_{\pi_1 \dots \pi_{i-1}} P(x_1 \dots x_{i-1}, \pi_1, \dots, \pi_{i-1}, \pi_i = i) e_i(x_i)$$

$$= \sum_k \sum_{\pi_1 \dots \pi_{i-2}} P(x_1 \dots x_{i-1}, \pi_1, \dots, \pi_{i-2}, \pi_{i-1} = k) a_{ki} e_i(x_i)$$

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

Lecture 5, Tuesday April 15, 2003

The Forward Algorithm

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

Initialization:

$$f_0(0) = 1$$

$$f_k(0) = 0, \text{ for all } k > 0$$

Iteration:

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

Termination:

$$P(x) = \sum_k f_k(N) a_{k0}$$

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

Lecture 5, Tuesday April 15, 2003

Relation between Forward and Viterbi

VITERBI

Initialization:

$$V_0(0) = 1$$

$$V_k(0) = 0, \text{ for all } k > 0$$

Iteration:

$$V_i(i) = e_i(x_i) \max_j V_k(i-1) a_{kj}$$

Termination:

$$P(x, \pi^*) = \max_k V_k(N)$$

FORWARD

Initialization:

$$f_0(0) = 1$$

$$f_k(0) = 0, \text{ for all } k > 0$$

Iteration:

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

Termination:

$$P(x) = \sum_k f_k(N) a_{k0}$$

Lecture 5, Tuesday April 15, 2003

Motivation for the Backward Algorithm

We want to compute

$$P(\pi_i = k | x),$$

the probability distribution on the i^{th} position, given x

We start by computing

$$\begin{aligned} P(\pi_i = k, x) &= P(x_1 \dots x_i, \pi_i = k, x_{i+1} \dots x_N) \\ &= P(x_1 \dots x_i, \pi_i = k) P(x_{i+1} \dots x_N | x_1 \dots x_i, \pi_i = k) \\ &= P(x_1 \dots x_i, \pi_i = k) P(x_{i+1} \dots x_N | \pi_i = k) \end{aligned}$$

Forward, $f_k(i)$

Backward, $b_k(i)$

Lecture 5, Tuesday April 15, 2003

The Backward Algorithm – derivation

Define the backward probability:

$$b_k(i) = P(x_{i+1} \dots x_N | \pi_i = k)$$

$$= \sum_{\pi_{i+1} \dots \pi_N} P(x_{i+1}, x_{i+2}, \dots, x_N, \pi_{i+1}, \dots, \pi_N | \pi_i = k)$$

$$= \sum_l \sum_{\pi_{i+2} \dots \pi_N} P(x_{i+1}, x_{i+2}, \dots, x_N, \pi_{i+1} = l, \pi_{i+2}, \dots, \pi_N | \pi_i = k)$$

$$= \sum_l e_l(x_{i+1}) a_{kl} \sum_{\pi_{i+2} \dots \pi_N} P(x_{i+2}, \dots, x_N, \pi_{i+2}, \dots, \pi_N | \pi_{i+1} = l)$$

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

Lecture 5, Tuesday April 15, 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:

$$P(x) = \sum_l a_{0l} e_1(x_1) b_l(1)$$

Lecture 5, Tuesday April 15, 2003

Computational Complexity

What is the running time, and space required, for Forward, and Backward?

Time: $O(K^2 N)$

Space: $O(KN)$

Useful implementation technique to avoid underflows

Viterbi: sum of logs

Forward/Backward: rescaling at each position by multiplying by a constant

Lecture 5, Tuesday April 15, 2003

Posterior Decoding

We can now calculate

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

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 5, Tuesday April 15, 2003

Posterior Decoding

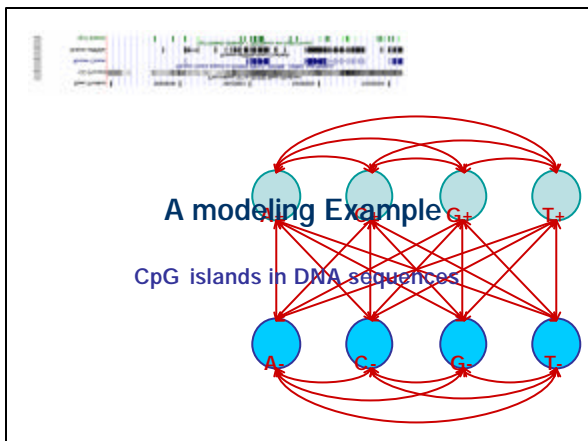
For each state,

Posterior Decoding gives us a curve of

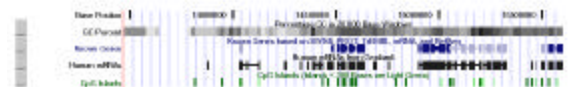
likelihood of state for each position

That is sometimes more informative than Viterbi path π^*

Lecture 5, Tuesday April 15, 2003



Example: CpG Islands



CpG nucleotides in the genome are frequently methylated

(Write CpG not to confuse with CG base pair)

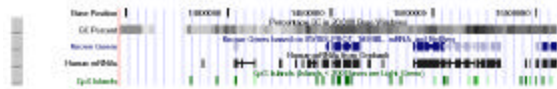


Methylation often suppressed around genes, promoters

→ CpG islands

Lecture 5, Tuesday April 15, 2003

Example: CpG Islands



In CpG islands,

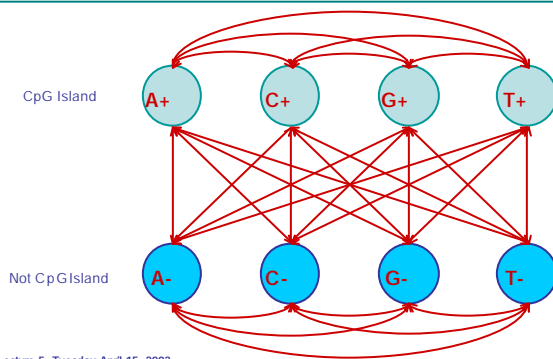
CG is more frequent

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

Question: Detect CpG islands computationally

Lecture 5, Tuesday April 15, 2003

A model of CpG Islands – (1) Architecture



Lecture 5, Tuesday April 15, 2003