Lecture 8

Intro to Hidden Markov Models

Intro: why?

- HMMs are a general probabilistic framework for designing algorithms to detect sequence features.
- Mostly easy to propose for many applications.
- Machine learning in practice: parameters can be learned from real data!

Basics

- Stochastic (probabistic)
- States, state transitions, emission
- Generative: output generated based on state and associated emission probabilities.
- "memoryless"

Example: play calling

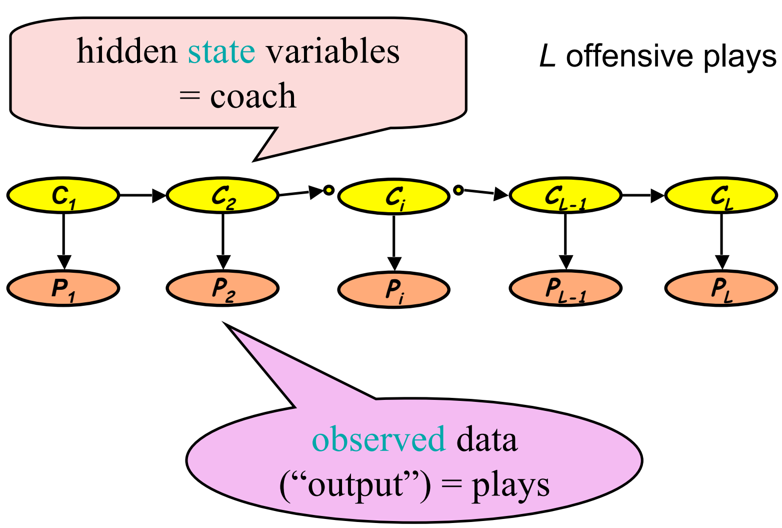
- Suppose we simplify the ND offensive playbook into three plays:
  - Run
  - Pass short
  - Long pass
- Further, lets suppose there are two at most two offensive coaches:
  - Coach Weiss
  - Offensive coordinator Haywood

Play calling

- Coach Weiss:
  - P(run) = 0.1
  - P(short pass) = 0.1
  - P(long pass) = 0.8

- Offensive coordinator Haywood:
  - P(run) = 0.8:
  - P(short pass) = 0.15
  - P(long pass) = 0.05

ND Football Game



Overview

- Structure
  - Number of states Q_1 .. Q_N
  - M output symbols
- Parameters:
  - Transition probability matrix a_{ij}
  - Emission probabilities $b_i(a)$, which is the probability state i emits character a
  - Initial distribution vector π_i

Notation (Rabiner)

- Let T be the number of observations
- Note T is also the number of states visited
- Sequence of visited states:
  - $Q = q_1 q_2 q_3 q_4 \ldots q_T$
- Sequence of emitted symbols:
  - $O = O_1 O_2 O_3 O_4 \ldots O_T$

Model = $\lambda = \left\langle N, M, \{\pi_i\}, \{a_{ij}\}, \{b_i(j)\} \right\rangle$

Big questions for today

- Evaluation
  - How likely is a sequence given a model?
  - More formally, given a model M and a sequence s, find Pr(*x* | *M*).

- Decoding (or inference)
  - Given a sequence and a model, try and figure out which states were visited.
  - More formally, given a model M and an observation sequence s, find a state sequence t such that Pr (*s,t* | *M*) is maximal.

Uses of evaluation

- Given a model of ND coaching, how likely is a series of play calls?

- Given a model of a fair coin, how likely is a given game based on flipping?

- Given a model of a fair casino, how likely is a certain series of rolls

Uses of decoding

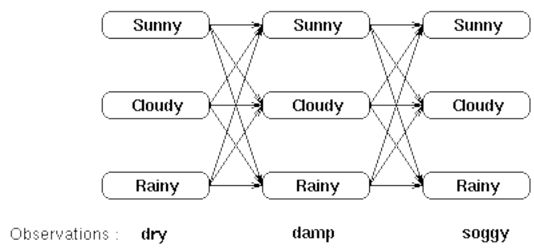
- Your dorm is hosting a casino night.
- The following sequence of rolls occurs:
  - 153466266636666466665646464646662
- Should the dice be checked?
  - By eye, a likely state sequence has a many loaded states where 6 is more likely

Solutions

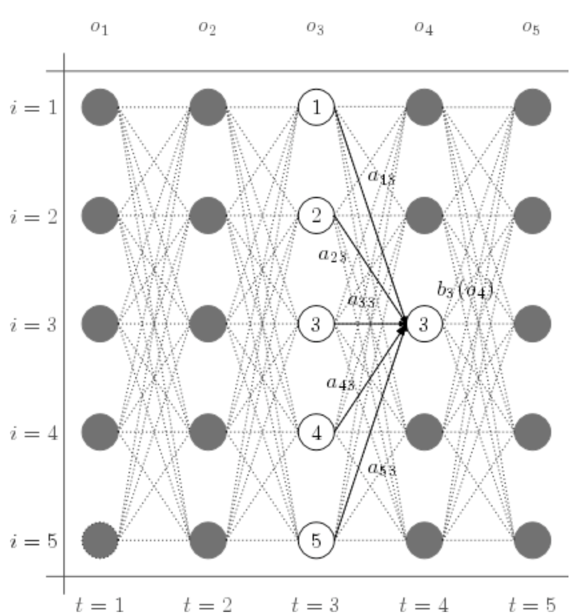
Problem	**Algorithm**	**Complexity**
Evaluation	Forward/ Backward	$O(TN^2)$
Decoding	Viterbi	$O(TN^2)$
Learning	Baum-Welch (EM)	$O(TN^2)$

T is # timesteps (or observations) *N* = # states

Forward algorithm



The Trelis



Initialization

- Consider the case in class where we have a start state where 0 characters were observed.
- $F_0(0) = 1$ given we always start here
- $F_k(0) = 0$ for all non-silent states

The occasionally dishonest casino – Forward algorithm

Emissions:	1	2	3	4	5	6
State F	1/6	1/6	1/6	1/6	1/6	1/6
State L	0.1	0.1	0.1	0.1	0.1	0.5

Transitons:	1	2	0
State 0	0.50	0.50	0.00
State 1	0.94	0.05	0.01
State 2	0.89	0.10	0.01

Algorithm: Forward algorithm

Initialisation ($i = 0$): $f_0(0) = 1,\ f_k(0) = 0$ for $k > 0$.

Recursion ($i = 1 \ldots L$): $f_l(i) = e_l(x_i) \sum_k f_k(i-1) a_{kl}$.

Termination: $P(x) = \sum_k f_k(L) a_{k0}$.

x_i	1	2	6	6	6	5	end
$e_1(x_i)$	0.1667	0.1667	0.1667	0.1677	0.1677	0.1677	
$e_2(x_i)$	0.1000	0.1000	0.5000	0.5000	0.5000	0.1000	

The occasionally dishonest casino – Viterbi algorithm

```
Emissions: 1   2   3   4   5   6
--------------------------------
State F  1/6 1/6 1/6 1/6 1/6 1/6
State L  0.1 0.1 0.1 0.1 0.1 0.5
```

```
Transitons: 1    2    0
-----------------------
State 0  0.50 0.50 0.00
State 1  0.94 0.05 0.01
State 2  0.89 0.10 0.01
```

Algorithm: Viterbi

Initialisation ($i = 0$): $v_0(0) = 1,\ v_k(0) = 0$ for $k > 0$.

Recursion ($i = 1 \dots L$): $v_l(i) = e_l(x_i)\max_k(v_k(i-1)a_{kl})$;
$\mathrm{ptr}_i(l) = \mathrm{argmax}_k(v_k(i-1)a_{kl})$.

Termination: $P(x,\pi^*) = \max_k(v_k(L)a_{k0})$;
$\pi_L^* = \mathrm{argmax}_k(v_k(L)a_{k0})$.

Traceback ($i = L \dots 1$): $\pi_{i-1}^* = \mathrm{ptr}_i(\pi_i^*)$.

x_i	1	2	6	6	6	5	end
$e_1(x_i)$	0.1667	0.1667	0.1667	0.1677	0.1677	0.1677	
$e_2(x_i)$	0.1000	0.1000	0.5000	0.5000	0.5000	0.1000	

Observed sequence, hidden path and Viterbi path

```
Rolls    315116246446644245311321631164152133625144543631656626566666
Die      FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLL
Viterbi  FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLL

Rolls    651166453132651245636664631636663162326455236266666625151631
Die      LLLLLLFFFFFFFFFFFFLLLLLLLLLLLLLLLLFFFLLLLLLLLLLLLLFFFFFFFFFF
Viterbi  LLLLLLFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLFFFFFFF

Rolls    222555441666566563564324364131513465146353411126414626253356
Die      FFFFFFFFLLLLLLLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLL
Viterbi  FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFL

Rolls    366163666466232534413661661163252562462255265252266435353336
Die      LLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Viterbi  LLLLLLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

Rolls    233121625364414432335163243633665562466662632666612355245242
Die      FFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLFFFFFFFFFFF
Viterbi  FFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLFFFFFFFFFFF
```

Figure 3.5 *The numbers show 300 rolls of a die as described in the example. Below is shown which die was actually used for that roll (F for fair and L for loaded). Under that the prediction by the Viterbi algorithm is shown.*

From Durbin

The Forward Algotithm

Algorithm: Forward algorithm

Initialisation ($i = 0$): $f_0(0) = 1,\ f_k(0) = 0$ for $k > 0$.

Recursion ($i = 1 \dots L$): $f_l(i) = e_l(x_i) \sum_k f_k(i-1) a_{kl}$.

Termination: $P(x) = \sum_k f_k(L) a_{k0}$.

Algorithm: Viterbi

Initialisation ($i = 0$): $v_0(0) = 1,\ v_k(0) = 0$ for $k > 0$.

Recursion ($i = 1 \dots L$): $v_l(i) = e_l(x_i) \max_k (v_k(i-1) a_{kl})$;
$\mathrm{ptr}_i(l) = \mathrm{argmax}_k (v_k(i-1) a_{kl})$.

Termination: $P(x, \pi^*) = \max_k (v_k(L) a_{k0})$;
$\pi^*_L = \mathrm{argmax}_k (v_k(L) a_{k0})$.

Traceback ($i = L \dots 1$): $\pi^*_{i-1} = \mathrm{ptr}_i(\pi^*_i)$.

The structure of the Forward algorithm is essentially the same as that of the Viterbi algorithm, except that a maximization operation is replaced by summation.

108 *5 Profile HMMs for sequence families*

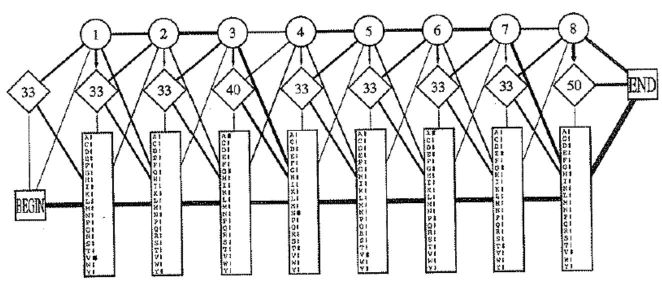
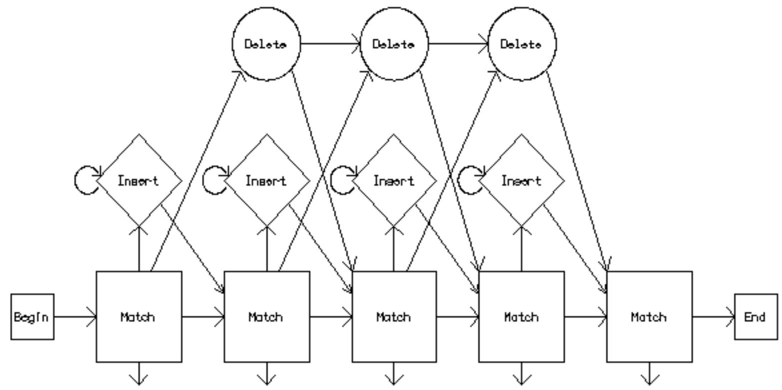


Figure 5.4 *A hidden Markov model derived from the small alignment shown in Figure 5.3 using Laplace's rule. Emission probabilities are shown as bars opposite the different amino acids for each match state, and transition probabilities are indicated by the thickness of the lines. The* I → I *transition probabilities times 100 are shown in the insert states. (Figure generated automatically using the* SAM *package.)*



http://genome.jouy.inra.fr/doc/genome/suite-logicielle/gcg-11.1/html/figure/hmmessay_2.gif