Lecture 8

Hidden Markov Models

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(s | 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.

Central problems w/ HMMs

- Evaluation
  - Probability of a particular observation sequence given a model
  - P(O|model)
  - Complicated as states (i.e., coaches) are hidden
  - Useful for sequence classification (next week; see online PDF)

Important problems

- Decoding:
  - Optimal state sequence to produce given observations under a specific model
  - Optimality is used (just like alignment from before)
  - Used for sequence recognition such as gene finding (next week)

Uses of decoding

- Your dorm is hosting a casino night.
- The following sequence of rolls occurs:
  - 1534662666366664666656464646662
- 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

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

Previous example: play calling

- Suppose we simplified the ND offensive playbook into three plays:
  - Run
  - Pass short
  - Long pass

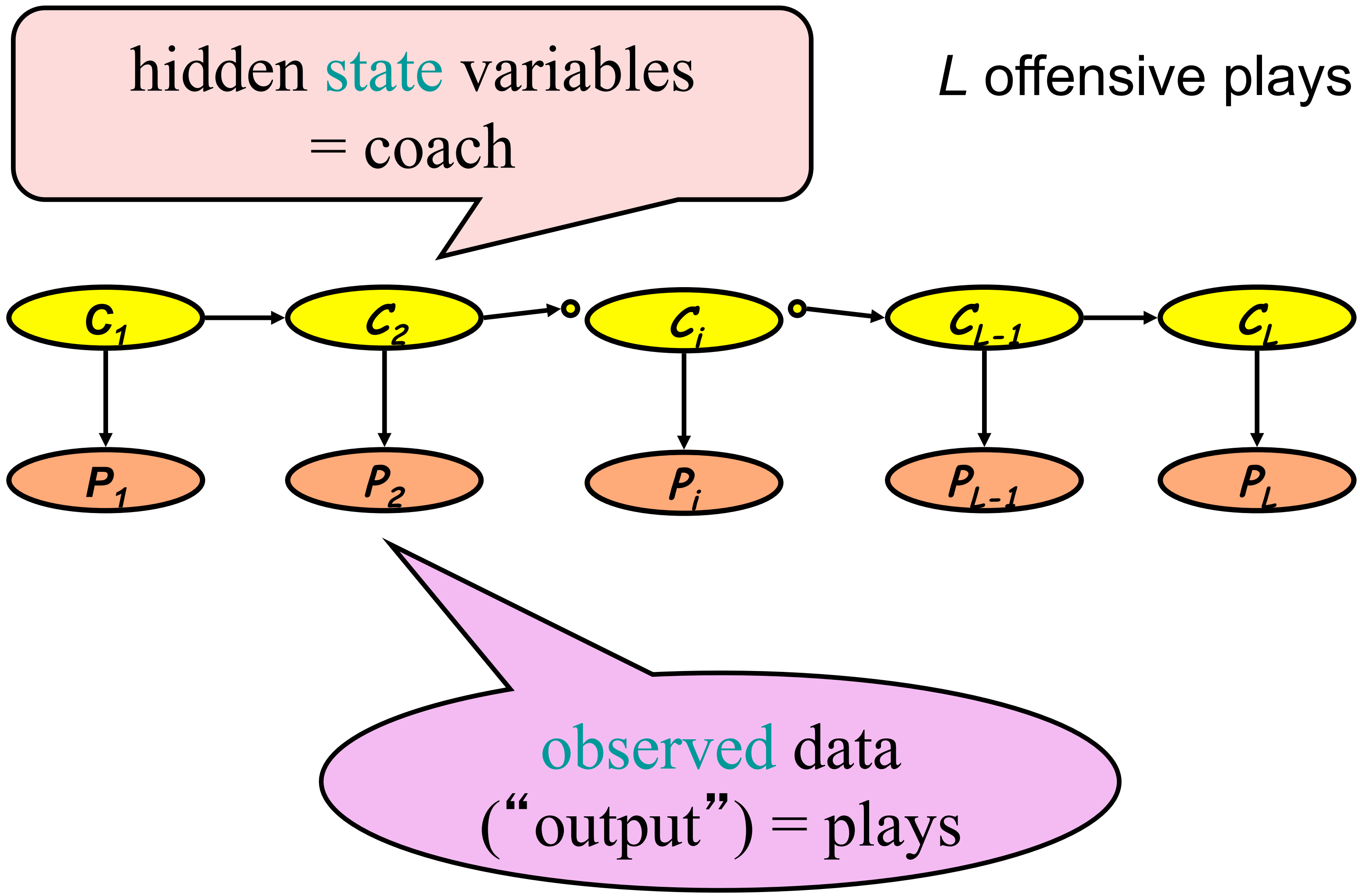
- Further, suppose there are two at most two offensive coaches:
  - Coach Kelly
  - Offensive coordinator Long

In class HMM: Play calling

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

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

ND Football Game



Naïve solution for evaluation

- Details are in handout, but in short we want to compute:

$$P(O \mid \lambda) = \sum_{q} P(O \mid q, \lambda) P(q \mid \lambda)$$

- This sums all over all state paths
- Note: There are N^T state paths, where T is the number of observations

Dynamic programming to the rescue (again)

Forward algorithm:

- Define auxiliary forward variable α:

$$\alpha_t(i) = P(o_1,\ldots,o_t \mid q_t = i, \lambda)$$

$\alpha_t(i)$ is the probability of observing a partial sequence of observables $o_1,\ldots o_t$ such that at time t, state q_t=i

The details

- Recursive algorithm:
  - Initialise:

$$\alpha_1(i) = \pi_i e_i(o_1)$$

  - Calculate:

(Partial obs seq to *t* AND state *i* at *t*) x (transition to *j* at *t+1*) x (sensor)

$$\alpha_{t+1}(j) = [\sum_{i=1}^{N} \alpha_t(i) a_{ij}] e_j(o_{t+1})$$

Sum, as can reach *j* from any preceding state

α incorporates partial obs seq to *t*

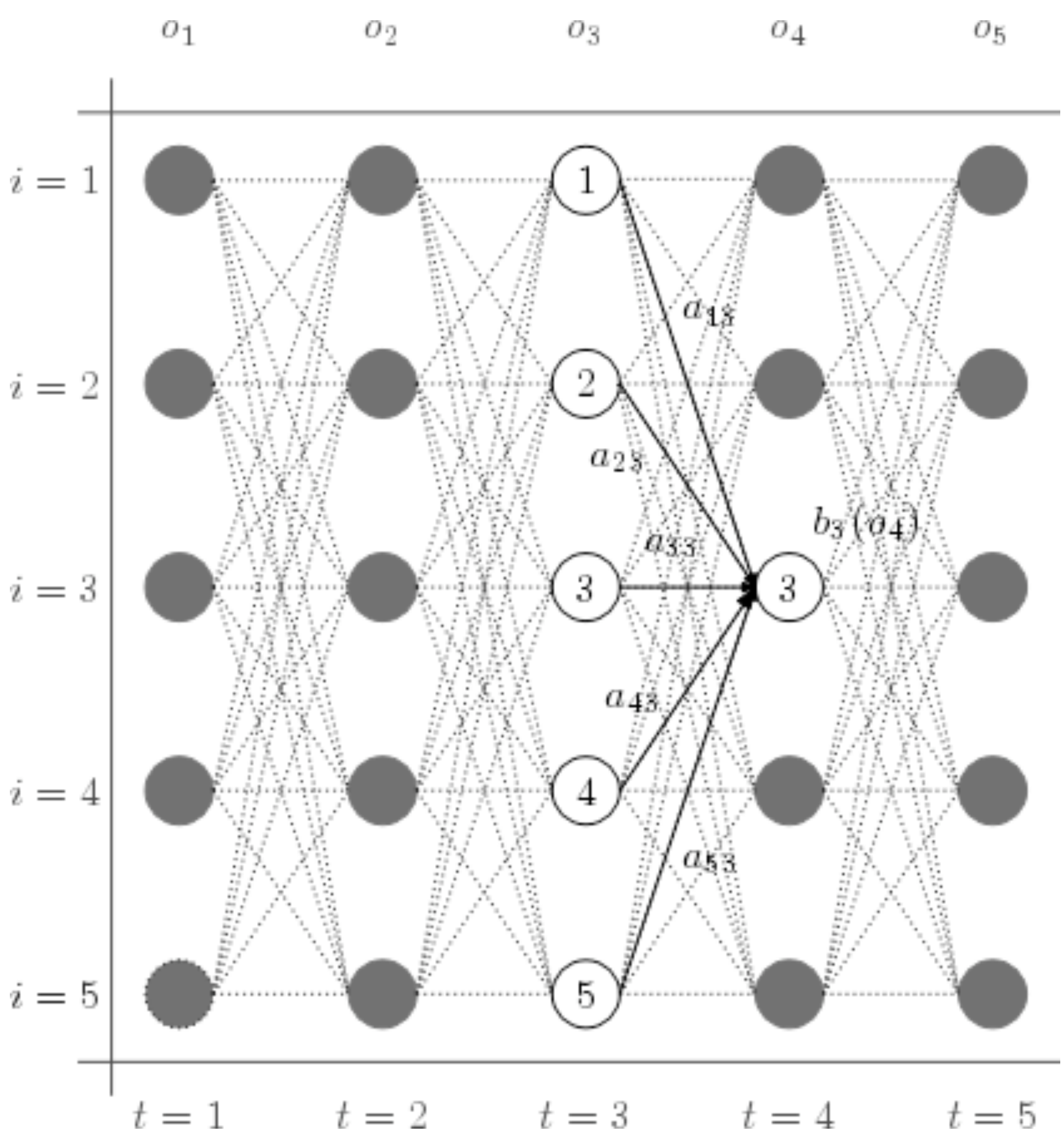
  - Obtain:

$$P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

Sum of different ways of getting obs seq

Complexity is $O(N^2T)$

The Trelis



o_1
o_2
o_3
o_4
o_5
$i = 1$
$i = 2$
$i = 3$
$i = 4$
$i = 5$
1
2
3
4
5
3
a_{13}
a_{23}
a_{33}
a_{43}
a_{53}
$b_3(o_4)$
$t = 1$
$t = 2$
$t = 3$
$t = 4$
$t = 5$

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	

Problem 2: Decoding

- Choose state sequence to maximize probability of an observed sequence
- The Viterbi algorithm is an inductive algorithm that keeps the * best * state sequence for each prefix of observations

Some details

- State sequence to maximise P(O,Q|λ):

$$P(q_1, q_2, \ldots q_T \mid O, \lambda)$$

- Define auxiliary variable δ:

$$\delta_t(i) = \max_q P(q_1, q_2, \ldots, q_t = i, o_1, o_2, \ldots o_t \mid \lambda)$$

$\delta_t(i)$ – the probability of the most probable path ending in state q_t=i

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

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

Algorithm: Viterbi

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

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

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

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

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.

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

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

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

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     LLLLLLFFFFFFFFFFFFLLLLLLLLLLLLLLLLFFFLLLLLLLLLLLLLLFFFFFFFFF
Viterbi LLLLLLFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLFFFFFFFF

Rolls   222555441666566563564324364131513465146353411126414626253356
Die     FFFFFFFFLLLLLLLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLL
Viterbi FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFL

Rolls   366163666466232534413661661163252562462255265252266435353336
Die     LLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF
Viterbi LLLLLLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF

Rolls   233121625364414432335163243633665562466662632666612355245242
Die     FFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLFFFFFFFFFFF
Viterbi FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLFFFFFFFFFFF
```

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