Lecture 9

Intro to Hidden Markov Models
(decoding, basic learning)

Assumptions

- Markov assumption
  - States depend on previous states
- Stationary assumption
  - Transition probabilities are independent of time (“memoryless”)
- Output independence
  - Observations are independent of previous observations

Review

- Structure
  - Number of states Q_1 .. Q_N
  - M output symbols

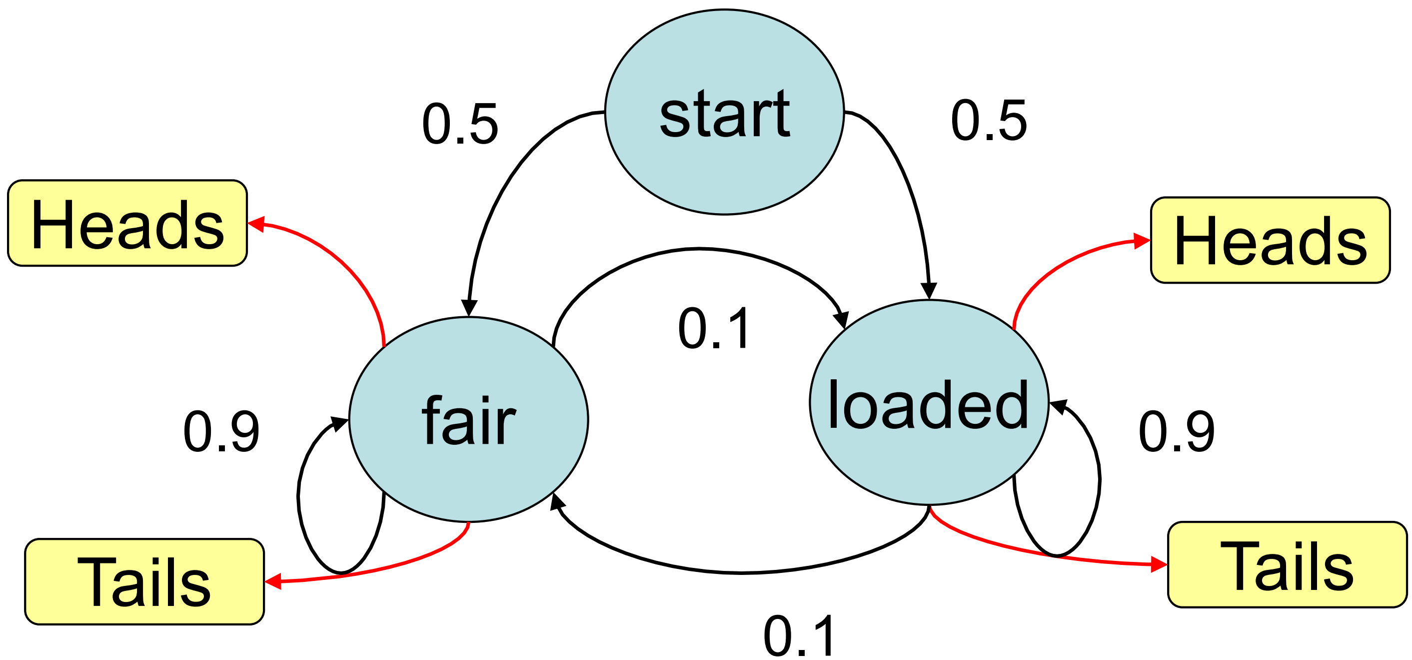
- Parameters:
  - Transition probability matrix a_{ij}
  - Emission probabilities $b_i(a)$, which is the probability state i emits character a
  - Initial distribution vector π_i

Cases

Example	**Observations**	**Hidden state**
Football	Plays	Coach
Text	Words	Shakespeare / monkey
Casino	Rolled numbers	Fair/loaded
DNA	ACGT	Coding/not

In class (re)review

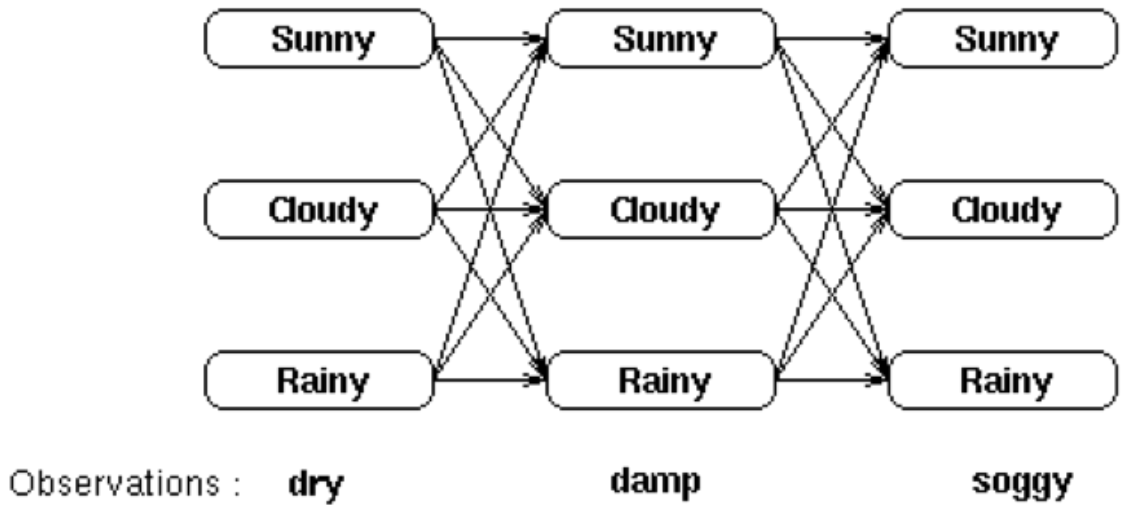
- Suppose in instead of a dishonest casino we used fair and loaded coins.
- Just like before the player shifts between fair and loaded states.
- How could we model this?


start
0.5
0.5
Heads
Heads
0.1
fair
loaded
0.9
0.9
Tails
Tails
0.1

Basic problems

- Evaluation
  - What is the probability that the observations were generated by a given model?
- Decoding
  - Given a model and a sequence of observations, what is the most likely state observations?
- Learning:
  - Given a model and a sequence of observations, how should we modify the model parameters to maximize p{observe|model)

Forward algorithm



Decoding

- Text: Shakespeare or Monkey?
- Case 1:
  - Fehwufhweuromeojulietpoisonjigjreijge
- Case 2:
  - mmmmbananammmmmmmbananammm

Observed sequence, hidden path and Viterbi path

```
Rolls   315116246446644245311321631164152133625144543631656626566666
Die     FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLLLLL
Viterbi FFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLLLLLLLLLLL

Rolls   651166453132651245636664631636663162326455236266666625151631
Die     LLLLLLFFFFFFFFFFFFLLLLLLLLLLLLLLLLFFFLLLLLLLLLLLLLLLFFFFFFFFF
Viterbi LLLLLLFFFFFFFFFFFFLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLLFFFFFFFF

Rolls   222555441666566563564324364131513465146353411126414626253356
Die     FFFFFFFFLLLLLLLLLLLLFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFLL
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

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.

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

Learning

- If state path is known and there are no hidden states, this is easy and involves:
  - Counting how often each parameter is used
  - Normalizing to get probabilities
  - Then treating it just like Markov chain models

- Harder without knowing state paths
  - Idea: estimate counts by considering every path weighted by its probability

Parameter estimation for HMMs

- We generally need to estimate transition and emission probabilities a_{ij} and $e_k(b)$.
- We have in hand a set of training examples, that correspond to output from the HMM.
- Two potential strategies:
  - Estimation when state sequence is known
  - Estimation when paths are unknown

Estimation when state sequence is known

- Easier than estimation when paths unknown
- Maximum likelihood estimators are:

$$a_{kl} = \frac{A_{kl}}{\sum_{l'} A_{kl'}} \qquad e_k(b) = \frac{E_k(b)}{\sum_{b'} E_k(b')}$$

- A_{kl} = number of transitions k to l in training data + r_{kl}
- $E_k(b)$ = number of emissions of b from k in training data + $r_k(b)$

Potential problems

- Maximum likelihood estimators are prone to overfitting
  - For example, states never encountered
- For this reason, we introduce rkl and rk(b), which reflect prior biases
- Can be interpreted as parameters of a Bayesian Dirichlet prior.

Estimation when paths are unknown

- More complex than when paths are known
- Because we can’t use maximum likelihood estimators, we will use an iterative algorithm
  - Baum-Welch

Baum-Welch Algorithm

- *Aka* the Forward-Backward algorithm
- Also an example of an expectation maximization (EM) algorithm
- Idea: hidden state path is the best that explains a training sequence

Overview

- More formally, Baum-Welch calculates Akl and Ek(b) as the expected number of times each transition or emission is used.

- This will use the same Forward and Backward probabilities as posterior decoding.
  - Topic of discussion maybe next week

Drawbacks

- ML estimators
  - Vulnerable to overfitting if not enough data
  - Estimations can be undefined if never used in training set (so use of pseudocounts)
- Baum-Welch
  - Many local maximums instead of global maximum can be found, depending on starting values of parameters
  - This problem will be worse for large HMMs

Example from Durbin

1: 1/6	1: 1/10	1: 0.19	1: 0.07
2: 1/6	2: 1/10	2: 0.19	2: 0.10
3: 1/6	3: 1/10	3: 0.23	3: 0.10
4: 1/6	4: 1/10	4: 0.08	4: 0.17
5: 1/6	5: 1/10	5: 0.23	5: 0.05
6: 1/6	6: 1/2	6: 0.06	6: 0.52

Note transition probabilities are different from real ones
Partly a result of local minima, but its never possible to
Estimate parameters exactly

Other methods

- Durbin also discusses an alternative method called Viterbi training based on the Viterbi algorithm.
- Does not maximize the true likelihood as a function of model parameters, but rather finds the model from the most probable paths.
- For this reason it generally does worse than Baum-Welch, but it is widely used.