A Markov System

Has $N$ states, called $s_1, s_2 \ldots s_N$

There are discrete timesteps, $t=0, t=1, \ldots$

On the $t$'th timestep the system is in exactly one of the available states. Call it $q_t$

Note: $q_t \in \{s_1, s_2 \ldots s_N\}$

Between each timestep, the next state is chosen randomly.

$N = 3$

$t=0$
$q_0 = s_3$

$t=1$
$q_1 = s_2$

$P(q_{t+1} = s_1 | q_t = s_3) = \frac{1}{3}$

$P(q_{t+1} = s_2 | q_t = s_3) = \frac{2}{3}$

$P(q_{t+1} = s_3 | q_t = s_3) = 0$

Between each timestep, the next state is chosen randomly.

The current state determines the probability distribution for the next state.
A Markov System
Has \(N\) states, called \(s_1, s_2, \ldots, s_N\).
There are discrete timesteps, \(t=0, t=1, \ldots\)
On the \(t\)'th timestep the system is in exactly one of the available states. Call it \(q_t\).
Note: \(q_t \in \{s_1, s_2, \ldots, s_N\}\).
Between each timestep, the next state is chosen randomly.
The current state determines the probability distribution for the next state.

\[
\begin{align*}
N &= 3 \\
t &= 1 \\
q_t &= q_1 = s_2 \\
P(q_{t+1} = s_1 | q_t = s_3) &= \frac{1}{3} \\
P(q_{t+1} = s_2 | q_t = s_3) &= \frac{2}{3} \\
P(q_{t+1} = s_3 | q_t = s_3) &= 0 \\
P(q_{t+1} = s_1 | q_t = s_1) &= 0 \\
P(q_{t+1} = s_2 | q_t = s_1) &= 0 \\
P(q_{t+1} = s_3 | q_t = s_1) &= 1 \\
P(q_{t+1} = s_1 | q_t = s_2) &= \frac{1}{2} \\
P(q_{t+1} = s_2 | q_t = s_2) &= \frac{1}{2} \\
P(q_{t+1} = s_3 | q_t = s_2) &= 0
\end{align*}
\]

Markov Property
\(q_t\) is conditionally independent of \(\{q_{t-1}, q_{t-2}, \ldots, q_1, q_0\}\) given \(q_t\).
In other words:
\[
P(q_{t+1} = s_j | q_t = s_i) = P(q_{t+1} = s_j | q_t = s_i, \text{any earlier history})
\]
Question: what would be the best Bayes Net structure to represent the Joint Distribution of \(\{q_0, q_1, \ldots, q_N\}\) ?

Hidden Markov Models

- Question 1: State Estimation
  What is \(P(q_T = S_i | O_1O_2\ldotsO_T)\)?
  It will turn out that a new cute D.P. trick will get this for us.
- Question 2: Most Probable Path
  Given \(O_1O_2\ldotsO_T\), what is the most probable path that I took?
  And what is that probability?
  Yet another famous D.P. trick, the VITERBI algorithm, gets this.
- Question 3: Learning HMMs:
  Given \(O_1O_2\ldotsO_T\), what is the maximum likelihood HMM that could have produced this string of observations?
  Very very useful. Uses the E.M. Algorithm

Are H.M.M.s Useful?

You bet!!
- Robot planning + sensing when there's uncertainty
- Speech Recognition/Understanding
  Phones \(\rightarrow\) Words, Signal \(\rightarrow\) phones
- Human Genome Project
  Complicated stuff your lecturer knows nothing about.
- Consumer decision modeling
- Economics & Finance.
  Plus at least 5 other things I haven't thought of.

HMM Notation
(from Rabiner's Survey)
The states are labeled \(S_1, S_2, \ldots, S_N\).
For a particular trial, ….
Let \(T\) be the number of observations
\(T\) is also the number of states passed through
\(O = O_1O_2\ldotsO_T\) is the sequence of observations
\(Q = q_1q_2\ldotsq_T\) is the notation for a path of states
\(\lambda = \langle N, M, \{\pi_i\}, \{a_{ij}\}, \{b_{kj}\} \rangle\) is the specification of an HMM

HMM Formal Definition
An HMM, \(\lambda\), is a 5-tuple consisting of
- \(N\) the number of states
- \(M\) the number of possible observations
- \(\{\pi_1, \pi_2, \ldots, \pi_N\}\) The starting state probabilities
  \(P(q_0 = S_i) = \pi_i\)
- \(\{a_{ij}\} \) The state transition probabilities
  \(P(q_{t+1} = S_j | q_t = S_i) = a_{ij}\)
- \(\{b_{kj}\} \) The observation probabilities
  \(P(O_{t+1} = k | q_t = S_i) = b_{kj}\)
Here’s an HMM

Start randomly in state 1 or 2
Choose one of the output symbols in each state at random.

<table>
<thead>
<tr>
<th>State</th>
<th>Symbol X</th>
<th>Symbol Y</th>
<th>Symbol Z</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>1/2</td>
<td>0</td>
<td>1/2</td>
</tr>
<tr>
<td>S2</td>
<td>0</td>
<td>1/2</td>
<td>1/2</td>
</tr>
<tr>
<td>S3</td>
<td>1/2</td>
<td>0</td>
<td>1/2</td>
</tr>
</tbody>
</table>

Copyright © 2001-2003, Andrew W. Moore
Hidden Markov Models: Slide 15
Here’s an HMM

Start randomly in state 1 or 2
Choose one of the output symbols in each state at random.
Let’s generate a sequence of observations:

N = 3
M = 3
π₁ = ½
π₂ = ½
π₃ = 0

a₁₁ = 0
a₁₂ = ⅓
a₁₃ = ⅔
da₁₂ = ⅓
a₂₂ = 0
a₁₃ = ⅔
da₁₃ = ⅓
a₃₂ = ⅓
a₁₃ = ⅓

b₁ (X) = ½
b₂ (X) = 0
b₁ (Y) = ½
b₂ (Y) = ⅓
b₁ (Z) = 0
b₂ (Z) = ½

Start randomly in state 1 or 2
Choose one of the output symbols in each state at random.
Let’s generate a sequence of observations:

State Estimation

Start randomly in state 1 or 2
Choose one of the output symbols in each state at random.
Let’s generate a sequence of observations:

Bayes’ Rule

- O={O₁,O₂,…,Oₜ}
- P(Qₜ|O) = P(O|Qₜ) P(O) / P(Qₜ)

Prob. of a series of observations

What is P(O) = P(O₁, O₂, O₃) = P(O₁ = X ∧ O₂ = X ∧ O₃ = Z)?

Slow, stupid way:

P(O) = \sum_{Q \text{ of length } t} P(O \setminus Q)
= \sum_{Q \text{ of length } t} P(O \setminus Q) / P(Q)

How do we compute P(Q) for an arbitrary path Q?
How do we compute P(O|Q) for an arbitrary path Q?

Prob. of a series of observations

What is P(O) = P(O₁, O₂, O₃) = P(O₁ = X ∧ O₂ = X ∧ O₃ = Z)?

Slow, stupid way:

P(O) = \sum_{Q \text{ of length } t} P(O \setminus Q)
= \sum_{Q \text{ of length } t} P(O \setminus Q) / P(Q)

How do we compute P(Q) for an arbitrary path Q?
How do we compute P(O|Q) for an arbitrary path Q?
Prob. of a series of observations

What is \( P(O) = P(O_1 O_2 O_3) \)?

Slow, stupid way:

\[
P(O) = \sum_{Q \in \text{Paths of length 3}} P(O | Q)
\]

How do we compute \( P(Q) \) for an arbitrary path \( Q \)?

Example in the case \( Q = S_1 S_3 S_3 \):

\[
P(O_1 O_2 O_3 | q_1 q_2 q_3 ) = P(X| S_1) P(X| S_3) P(Z| S_3) = \frac{1}{2} \times \frac{1}{2} \times \frac{1}{2} = \frac{1}{8}
\]

\[
P(O_1 O_2 O_3) = P(O_1 = X \land O_2 = X \land O_3 = Z)
\]

\[
= P(O_1 = X) P(O_2 = X) P(O_3 = Z)
\]

A sequence of 20 observations would need \( 2^{30} \) = 3.5 billion computations and 3.5 billion \( P(O_1 O_2 O_3) \) computations

So let's be smarter...

\[
\alpha_t(i) = \text{easy to define recursively}
\]

\[
\alpha_t(i) = P(O_1 O_2 \ldots O_t \land q_t = S_i | \lambda)
\]

\[
= \sum_{\pi} \prod_{t=1}^{T} P(O_t | q_t = S_i, \lambda) P(q_t = S_i | q_{t-1} = S_j, \lambda)
\]

\[
= \sum_{\pi} a_{t-1}(j) \cdot b_t(q_t) \cdot \pi_t
\]

In our example:

\[
\alpha_2(3) = \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}
\]

\[
\alpha_3(3) = \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{8}
\]

\[
\alpha_3(2) = \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{2}{3} = \frac{1}{3}
\]

\[
\alpha_3(1) = \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{2}{3} = \frac{1}{3}
\]

\[
\text{WE SAW } O_1 O_2 O_3 = X X Z
\]

\[
\alpha_1(1) = \frac{1}{4}
\]

\[
\alpha_2(1) = \frac{1}{12}
\]

\[
\alpha_3(1) = \frac{1}{12}
\]

\[
\alpha_1(2) = \frac{1}{4}
\]

\[
\alpha_2(2) = \frac{1}{12}
\]

\[
\alpha_3(2) = \frac{1}{12}
\]

\[
\alpha_1(3) = \frac{1}{4}
\]

\[
\alpha_2(3) = \frac{1}{12}
\]

\[
\alpha_3(3) = \frac{1}{12}
\]
Easy Question

We can cheaply compute
\[ \alpha_t(i) = P(O_1O_2...O_t \land q_t = S_i) \]
(How) can we cheaply compute\[ P(O_1O_2...O_t) \] ?

(How) can we cheaply compute\[ P(q_t = S_i | O_1O_2...O_t) \]

Most probable path given observations

What’s most probable path given \( O_t \), i.e.
What is \( \text{argmax}_Q P(Q_t | O_1...O_t) \)?

Slow, stupid answer :
\[ \text{argmax}_Q P(O_1...O_t | Q_t) \]
\[ = \text{argmax}_Q \frac{P(O_1...O_t | Q_t)P(Q)}{P(O_1...O_t)} \]
\[ = \text{argmax}_Q P(O_1...O_t | Q_t) \]

Efficient MPP computation

We’re going to compute the following variables:
\[ \delta_t(i) = \max_{q_t} P(q_t, q_{t-1} \land q_t = S_i \land O_t) \]
\[ mpp_t(i) = \text{the path that produced } O_t \text{ and ended in state } S_i \]

Define:
\[ mpp_t(i) = \text{that path} \]
So: \[ \delta_t(i) = \text{Prob}(mpp_t(i)) \]

Easy Question

We can cheaply compute
\[ \alpha_t(i) = P(O_1O_2...O_t \land q_t = S_i) \]
(How) can we cheaply compute\[ P(O_1O_2...O_t) \] ?

(How) can we cheaply compute\[ \sum_{i=1}^{N} \alpha_t(i) \]

The Viterbi Algorithm

\[ \delta_t(i) = q_t q_{t-1}...q_1 \land q_t = S_i \land O_t \]
\[ mpp_t(i) = q_t q_{t-1}...q_1 \land q_t = S_i \land O_t \]
\[ \delta(i) = \text{one choice } P(q_t = S_i \land O_t) \]
\[ = P(q_t = S_i)P(O_t) = \pi(S_i) \]

Now, suppose we have all the \( \delta(i) \)'s and \( mpp(i) \)'s for all \( i \).

HOW TO GET \( \delta_{t+1}(i) \) and \( mpp_{t+1}(i) \)?

The Viterbi Algorithm

The most prob path with last two states \( S_i, S_j \)

is

the most prob path to \( S_i \)
followed by transition \( S_i \rightarrow S_j \)
The Viterbi Algorithm

The most probable path with last two states $S_i, S_j$ is the most probable path to $S_i$, followed by transition $S_i \rightarrow S_j$.

What is the prob of that path?

$$\delta_t(i) x P(S_i \rightarrow S_j \land O_{t+1} | \lambda) = \delta_t(i) a_{ij} b_j (O_{t+1})$$

SO The most probable path to $S_j$ has $S_i^*$ as its penultimate state

$$i^* = \arg\max_i \delta_t(i) a_{ij} b_j (O_{t+1})$$

SO   The most probable path to $S_j$ has $S_i^*$ as its penultimate state

Summary:

$$\delta_{t+1}(j) = \delta_t(i^*) a_{ij} b_j (O_{t+1})$$

What's Viterbi used for?

Classic Example

Speech recognition:

Signal $\rightarrow$ words

HMM $\rightarrow$ observable is signal

$\rightarrow$ Hidden state is part of word formation

What is the most probable word given this signal?

UTTERLY GROSS SIMPLIFICATION

In practice: many levels of inference; not one big jump.

Inferring an HMM

Remember, we've been doing things like

$$P(O_1 O_2 \ldots O_T | \lambda)$$

That "$\lambda$" is the notation for our HMM parameters.

Now We have some observations and we want to estimate $\lambda$ from them.

AS USUAL: We could use

(i) MAX LIKELIHOOD $\lambda = \arg\max \lambda P(O_1 \ldots O_T | \lambda)$

(ii) BAYES

Work out $P(\lambda | O_1 \ldots O_T)$

and then $E(\lambda)$ or $\max \lambda P(\lambda | O_1 \ldots O_T)$.

Max likelihood HMM estimation

Define

$$\gamma(i) = P(q_t = S_i | O_1 O_2 \ldots O_T, \lambda)$$

$$\epsilon(i,j) = P(q_t = S_i \land q_{t+1} = S_j | O_1 O_2 \ldots O_T, \lambda)$$

$\gamma(i)$ and $\epsilon(i,j)$ can be computed efficiently $\forall i,j,t$ (Details in Rabiner paper)

$$\sum_{i=1}^{T-1} \gamma(i) = \text{Expected number of transitions out of state } i \text{ during the path}$$

$$\sum_{t=1}^{T-1} \epsilon(i,j) = \text{Expected number of transitions from state } i \text{ to state j during the path}$$

HMMs are used and useful

But how do you design an HMM?

Occasionally, (e.g. in our robot example) it is reasonable to deduce the HMM from first principles.

But usually, especially in Speech or Genetics, it is better to infer it from large amounts of data. $O_1 O_2 \ldots O_T$ with a big "T".

Observations previously in lecture $O_1 O_2 \ldots O_T$

Observations in the next bit $O_1 O_2 \ldots O_T$
EM for HMMs

If we knew \( \lambda \) we could estimate EXPECTATIONS of quantities such as

- Expected number of times in state \( i \)
- Expected number of transitions \( i \to j \)

If we knew the quantities such as

- Expected number of times in state \( i \)
- Expected number of transitions \( i \to j \)

We could compute the MAX LIKELIHOOD estimate of

\[ \lambda = (\{a_{ij}\}, \{b_{i}(j)\}, \pi) \]

Roll on the EM Algorithm…

EM 4 HMMs

1. Get your observations \( O_1 \ldots O_T \)
2. Guess your first \( \lambda \) estimate \( \lambda(0), k=0 \)
3. \( k = k+1 \)
4. Given \( O_1 \ldots O_T, \lambda(k) \) compute
   \[ \gamma(t) = \begin{cases} 1 & \text{for } S_i(t) \in \text{state } \gamma(t) \\ 0 & \text{otherwise} \end{cases} \]
   \[ \forall 1 \leq t \leq T, \forall 1 \leq i \leq N, \forall 1 \leq j \leq N \]
5. Compute expected freq. of state \( i \), and expected freq. \( i \to j \)
6. Compute new estimates of \( a_{ij}, b_i(k), \pi \) accordingly. Call them \( \lambda(k+1) \)
7. Goto 3, unless converged.

- Also known (for the HMM case) as the BAUM-WELCH algorithm.

Bad News

- There are lots of local minima
- The local minima are usually adequate models of the data.

Notice

- EM does not estimate the number of states. That must be given.
- Often, HMMs are forced to have some links with zero probability. This is done by setting \( a_{ij} = 0 \) in initial estimate \( \lambda(0) \)
- Easy extension of everything seen today: HMMs with real valued outputs

What You Should Know

- What is an HMM?
- Computing (and defining) \( a_{ij} \)
- The Viterbi algorithm
- Outline of the EM algorithm
- To be very happy with the kind of maths and analysis needed for HMMs

*Fairly thorough reading of Rabiner* up to page 266*

[Up to but not including “IV. Types of HMMs”]