Hidden Markov Models

George Konidaris
gdk@cs.brown.edu

Spring 2017
Recall: Bayesian Network

Flu

Allergy

Nose

Headache
Recall: BN

Flu | P
---|---
True | 0.6
False | 0.4

Allergy | P
---------|---
True | 0.2
False | 0.8

Sinus

Flu | Allergy | P
---|---|---
True True True | 0.9
True True False | 0.1
True False True | 0.6
True False False | 0.4
False False True | 0.2
False False False | 0.8
False False True | 0.4
False False False | 0.6

Nose

Flu | Sinus | P
---|---|---
True True | 0.8
False True | 0.2
True False | 0.3
False False | 0.7

Headache

Flu | Sinus | P
---|---|---
True True | 0.6
False True | 0.4
True False | 0.5
False False | 0.5

joint: 32 (31) entries
Inference

Given A compute $P(B \mid A)$. 

- Flu
- Allergy
- Nose
- Headache
Bayesian Networks (so far) contain no notion of time.

However, in many applications:
- Target tracking
- Patient monitoring
- Speech recognition
- Gesture recognition

... how a signal changes over time is critical.
States

In probability theory, we talked about *atomic events*:

- All possible outcomes.
- Mutually exclusive.

In time series, we have *state*:

- System is in a *state* at time $t$.
- Describes system completely.
- Over time, transition from *state to state*.
Example

The weather today can be:

- Hot
- Cold
- Chilly
- Freezing

The weather has four states.

At each point in time, the system is in one (and only one) state.
Example

State at time $t$

State transition

Freezing

Chilly

Chilly

Hot

t=1
t=2
t=3

... 

t=n
The Markov Assumption

We are probabilistic modelers, so we’d like to model:

\[ P(S_t | S_{t-1}, S_{t-2}, \ldots, S_0) \]

A state has the Markov property when we can write this as:

\[ P(S_t | S_{t-1}) \]

Special kind of independence assumption:

- Future independent of past given present.
Markov Assumption

Model that has it is a *Markov model*.

Sequence of states thus generated is a *Markov chain*.

Can describe transition probabilities with matrix:
- $P(S_i | S_j)$
- Steady state probabilities.
- Convergence rates.
State Machines

P(S0 | S1) = 0.8
P(S0 | S2) = 0.5
P(S1 | S0) = 0.4
P(S1 | S2) = 0.5
P(S2 | S0) = 0.6
P(S2 | S1) = 0.2

Time implicit

<table>
<thead>
<tr>
<th></th>
<th>S0</th>
<th>S1</th>
<th>S2</th>
</tr>
</thead>
<tbody>
<tr>
<td>S0</td>
<td>0.0</td>
<td>0.8</td>
<td>0.5</td>
</tr>
<tr>
<td>S1</td>
<td>0.4</td>
<td>0.0</td>
<td>0.5</td>
</tr>
<tr>
<td>S2</td>
<td>0.6</td>
<td>0.2</td>
<td>0.0</td>
</tr>
</tbody>
</table>

states not state vars!
State Machines

Assumptions:
- Markov assumption.
- Transition probabilities don’t change with time.
- Event space doesn’t change with time.
- Time moves in discrete increments.
Hidden State

State machines are cool but:
• Often state is not observed directly.
• State is latent, or hidden.

Instead you see an observation, which contains information about the hidden state.

State: forehand
<table>
<thead>
<tr>
<th>State</th>
<th>Observation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Word</td>
<td>Phone</td>
</tr>
<tr>
<td>Chemical State</td>
<td>Color, Smell etc.</td>
</tr>
<tr>
<td>Flu?</td>
<td>Runny Nose</td>
</tr>
<tr>
<td>Cardiac Arrest?</td>
<td>Pulse</td>
</tr>
</tbody>
</table>
Hidden Markov Models

Must store:
- $P(O_t | S_t)$
- $P(S_{t+1} | S_t)$
HMMs

**Monitoring/Filtering**
- \( P(S_t \mid E_0 \ldots E_t) \)
- E.g., estimate patient disease state.

**Prediction**
- \( P(S_t \mid E_0 \ldots E_k), k < t \)
- Given first two phonemes, what word?

**Smoothing**
- \( P(S_t \mid E_0 \ldots E_k), k > t \)
- What happened back there?

**Most likely state sequence.** (a path, not a distribution)
Example: Robot Localization

observations:
walls each side?

states:
position
Example: Robot Localization

We start off not knowing where the robot is.
Robot sense: obstacles up and down. Updates distribution.
Example: Robot Localization

Robot moves right: updates distribution.
Example: Robot Localization

Obstacles up and down, updates distribution.
What Happened

This is an instance of robot tracking.

Could also:

• Predict (where will the robot be in 3 steps?)
• Smooth (where was the robot?)
• Most likely path (what was the robot’s path?)

All of these are questions about the HMM’s state at various times.
Most Likely Path

How to compute? (assume we have CPTs)
Most Likely State

(Assume all start states equally likely)

Consider a path of length 1.

- \( P(S_0) = P(S_0 \mid O_0) \).
- We can just write this down.

Now, what about a path of length 2?

- \( P(S_1) = P(S_0) P(S_1 \mid S_0) P(S_1 \mid O_1) \)
- \[= P(S_0 \mid O_0) P(S_1 \mid S_0) P(S_1 \mid O_1) \]

See the recursion?

\[ P(S_n) = P(S_{n-1}) P(S_n \mid S_{n-1}) P(S_n \mid O_n) \]

transition model  
observation model
Viterbi Algorithm

Most likely path $S_0 \ldots S_n$:

$V_{i,k}$: probability of path at ending in state $S^k$, including observations up to $O_i$ ($t=i$).

For each state $S^k$:

$V_{0,k} = P(O_0 | S^k)$

For $i = 1 \ldots n$,

For each $k$:

$V_{i,k} = \max_x P(O_i | S^k) \cdot P(S^k | S^x) \cdot V_{i-1,x}$

Recursive, but values repeat, so save them (dynamic programming).
Common Form

Very common form:

• Noisy observations of true state
Viterbi

“The algorithm has found universal application in decoding the convolutional codes used in both CDMA and GSM digital cellular, dial-up modems, satellite, deep-space communications, and 802.11 wireless LANs.” (wikipedia)

(photograph credit: MIT)