CS 188: Artificial Intelligence
Fall 2009

Lecture 19: Hidden Markov Models
11/3/2009

Dan Klein – UC Berkeley

Announcements

- Written 3 is up!
  - Due on 11/12 (i.e. under two weeks)

- Project 4 up very soon!
  - Due on 11/19 (i.e. a little over two weeks)

- Contest!

- Mid-Semester Evals: We’ll go over Thursday

Markov Models

- A Markov model is a chain-structured BN
  - Each node is identically distributed (stationarity)
  - Value of X at a given time is called the state
  - As a BN:

\[
P(X_1) \quad P(X|X_{-1})
\]

- Parameters: called transition probabilities or dynamics, specify how the state evolves over time (also, initial probs)

Example: Markov Chain

- Weather:
  - States: X = {rain, sun}
  - Transitions:

\[
P(X_2 = \text{sun}) = P(X_2 = \text{sun}|X_1 = \text{sun})P(X_1 = \text{sun}) + P(X_2 = \text{sun}|X_1 = \text{rain})P(X_1 = \text{rain})
\]

\[
0.9 \cdot 1.0 + 0.1 \cdot 0.0 = 0.9
\]

Mini-Forward Algorithm

- Question: What’s P(X) on some day t?
  - An instance of variable elimination!

\[
P(X_t) = \sum_{x_{t-1}} P(x_t|x_{t-1})P(x_{t-1})
\]

\[
P(X_1) = \text{known}
\]

Example

- From initial observation of sun

\[
\begin{bmatrix}
1.0 \\
0.0
\end{bmatrix}
\] \[\begin{bmatrix}
0.9 \\
0.1
\end{bmatrix}
\] \[\begin{bmatrix}
0.82 \\
0.18
\end{bmatrix}
\] \[\begin{bmatrix}
0.5 \\
0.5
\end{bmatrix}
\]

P(X_1) \quad P(X_2) \quad P(X_3) \quad P(X_4)

- From initial observation of rain

\[
\begin{bmatrix}
1.0 \\
0.0
\end{bmatrix}
\] \[\begin{bmatrix}
1.0 \\
0.1
\end{bmatrix}
\] \[\begin{bmatrix}
0.18 \\
0.82
\end{bmatrix}
\] \[\begin{bmatrix}
0.5 \\
0.5
\end{bmatrix}
\]

P(X_1) \quad P(X_2) \quad P(X_3) \quad P(X_4)
Stationary Distributions

- If we simulate the chain long enough:
  - What happens?
  - Uncertainty accumulates
  - Eventually, we have no idea what the state is!

- Stationary distributions:
  - For most chains, the distribution we end up in is independent of the initial distribution
  - Called the stationary distribution of the chain
  - Usually, can only predict a short time out

Web Link Analysis

- PageRank over a web graph
  - Each web page is a state
  - Initial distribution: uniform over pages
  - Transitions:
    - With prob. c, uniform jump to a random page (dotted lines, not all shown)
    - With prob. 1-c, follow a random outlink (solid lines)

- Stationary distribution
  - Will spend more time on highly reachable pages
  - E.g. many ways to get to the Acrobat Reader download page
  - Somewhat robust to link spam
  - Google 1.0 returned the set of pages containing all your keywords in decreasing rank, now all search engines use link analysis along with many other factors (rank actually getting less important over time)

Hidden Markov Models

- Markov chains not so useful for most agents
  - Eventually you don’t know anything anymore
  - Need observations to update your beliefs

- Hidden Markov models (HMMs)
  - Underlying Markov chain over states S
  - You observe outputs (effects) at each time step
  - As a Bayes’ net:

Example

- An HMM is defined by:
  - Initial distribution: $P(X_1)$
  - Transitions: $P(X|X_{-1})$
  - Emissions: $P(E|X)$

Ghostbusters HMM

- $P(X_i) = \text{uniform}$
- $P(X|X) = \text{usually move clockwise, but sometimes move in a random direction or stay in place}$
- $P(R|X) = \text{same sensor model as before; red means close, green means far away.}$

Conditional Independence

- HMMs have two important independence properties:
  - Markov hidden process, future depends on past via the present
  - Current observation independent of all else given current state

- Quiz: does this mean that observations are independent given no evidence?
  - [No, correlated by the hidden state]
Real HMM Examples

- Speech recognition HMMs:
  - Observations are acoustic signals (continuous valued)
  - States are specific positions in specific words (so, tens of thousands)

- Machine translation HMMs:
  - Observations are words (tens of thousands)
  - States are translation options

- Robot tracking:
  - Observations are range readings (continuous)
  - States are positions on a map (continuous)

Filtering / Monitoring

- Filtering, or monitoring, is the task of tracking the distribution $B(X)$ (the belief state) over time

- We start with $B(X)$ in an initial setting, usually uniform

- As time passes, or we get observations, we update $B(X)$

- The Kalman filter was invented in the 60’s and first implemented as a method of trajectory estimation for the Apollo program

Example: Robot Localization

- Sensor model: never more than 1 mistake
- Motion model: may not execute action with small prob.
Inference Recap: Simple Cases

Example: Passage of Time

- As time passes, uncertainty “accumulates”

\[ B'(X^t) = \sum_{X} P(X^t|z)B(z) \]

<table>
<thead>
<tr>
<th>T = 1</th>
<th>T = 2</th>
<th>T = 5</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="https://example.com/image1.png" alt="Image 1" /></td>
<td><img src="https://example.com/image2.png" alt="Image 2" /></td>
<td><img src="https://example.com/image3.png" alt="Image 3" /></td>
</tr>
</tbody>
</table>

Observation

- Assume we have current belief \( P(X | \text{previous evidence}) \):

\[ B(X_{t+1}) = P(X_{t+1}|e_{1:t}) \]

- Then:

\[ P(X_{t+1}|e_{1:t+1}) \propto P(e_{t+1}|X_{t+1})P(X_{t+1}|e_{1:t}) \]

- Or:

\[ B(X_{t+1}) \propto P(e|X)B'(X_{t+1}) \]

Basic idea: beliefs reweighted by likelihood of evidence

Unlike passage of time, we have to renormalize
Example: Observation

- As we get observations, beliefs get reweighted, uncertainty "decreases"

\[ B(X) \propto P(e|X)B'(X) \]

Before observation

After observation

Example HMM

The Forward Algorithm

- We are given evidence at each time and want to know
  \[ B_t(X) = P(X_t|e_{1:t}) \]

- We can derive the following updates
  \[
  P(x_t|e_{1:t}) \propto \sum_{x_{t-1}} P(x_{t-1}, x_t, e_{1:t})
  \]

Online Belief Updates

- Every time step, we start with current \( P(X|\text{evidence}) \)
- We update for time:
  \[
  P(x_t|e_{1:t-1}) = \sum_{x_{t-1}} P(x_{t-1}|e_{1:t-1}) \cdot P(x_t|e_{t-1})
  \]

- We update for evidence:
  \[
  P(x_t|e_{1:t}) \propto P(x_t|e_{1:t-1}) \cdot P(e_t|x_t)
  \]

- The forward algorithm does both at once (and doesn’t normalize)
- Problem: space is \(|X|\) and time is \(|X|^2\) per time step