CSE 573: Artificial Intelligence
Autumn 2010

Lecture 11: Hidden Markov Models II
11/4/2010

Luke Zettlemoyer

Many slides over the course adapted from either Dan Klein, Stuart Russell or Andrew Moore
Outline

- Probabilistic sequence models (and inference)
  - (Review) Markov Chains
  - Hidden Markov Models
  - Particle Filters
  - Most Probable Explanations
  - Dynamic Bayesian networks
Let’s say we have two distributions:
- Prior distribution over ghost location: \( P(G) \)
  - Let’s say this is uniform
- Sensor reading model: \( P(R | G) \)
  - Given: we know what our sensors do
  - \( R = \) reading color measured at \((1,1)\)
  - E.g. \( P(R = \text{yellow} | G=(1,1)) = 0.1 \)

We can calculate the posterior distribution \( P(G|r) \) over ghost locations given a reading using Bayes’ rule:

\[
P(g|r) \propto P(r|g)P(g)
\]
Recap: Markov Models

- A Markov model is:
  - a MDP with no actions (and no rewards)
  - a chain-structured Bayesian Network (BN)

- A Markov model includes:
  - Random variables $X_t$ for all time steps $t$ (the state)
  - Parameters: called transition probabilities or dynamics, specify how the state evolves over time (also, initial probs)

$$P(X_1) \quad \text{and} \quad P(X_t | X_{t-1})$$
Recap: Markov Models

A Markov model defines
- a joint probability distribution:

\[
P(X_1, \ldots, X_n) = P(X_1) \prod_{t=2}^{N} P(X_t|X_{t-1})
\]

One common inference problem:
- Compute marginals \( P(X_t) \) for all time steps \( t \)
Recap: Mini-Forward Algorithm

- Question: What’s $P(X)$ on some day $t$?
  - We don’t need to enumerate every sequence!

- Forward simulation

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

\[
P(x_1) = \text{known}
\]
Recap: 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
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
  - POMDPs without actions (or rewards).
  - As a Bayes’ net:
An HMM is defined by:

- **Initial distribution:** $P(X_1)$
- **Transitions:** $P(X_t|X_{t-1})$
- **Emissions:** $P(E|X)$
Hidden Markov Models

- Defines a joint probability distribution:

\[
P(X_1, \ldots, X_n, E_1, \ldots, E_n) = P(X_{1:n}, E_{1:n}) = \\
P(X_1) P(E_1 | X_1) \prod_{t=2}^{N} P(X_t | X_{t-1}) P(E_t | X_t)
\]
- \( P(X_1) = \text{uniform} \)
- \( P(X'|X) = \text{usually move clockwise, but sometimes move in a random direction or stay in place} \)
- \( P(E|X) = \text{same sensor model as before: red means close, green means far away.} \)

\[
\begin{array}{ccc}
1/6 & 1/6 & 1/2 \\
0 & 1/6 & 0 \\
0 & 0 & 0 \\
\end{array}
\]

\[
\begin{array}{ccc}
1/9 & 1/9 & 1/9 \\
1/9 & 1/9 & 1/9 \\
1/9 & 1/9 & 1/9 \\
\end{array}
\]

\[
\begin{array}{cc}
X_1 \rightarrow X_2 \\
E_1 \rightarrow E_1 \\
X_2 \rightarrow X_3 \\
E_1 \rightarrow E_3 \\
X_3 \rightarrow X_4 \\
E_3 \rightarrow E_4 \\
X_4 \rightarrow \ldots \\
E_4 \rightarrow \ldots \\
\end{array}
\]
HMM Computations

- Given
  - joint $P(X_{1:n}, E_{1:n})$
  - evidence $E_{1:n} = e_{1:n}$

- Inference problems include:
  - Filtering, find $P(X_t | e_{1:t})$ for all $t$
  - Smoothing, find $P(X_t | e_{1:n})$ for all $t$
  - Most probable explanation, find
    \[ x^{*}_{1:n} = \arg\max_{x_{1:n}} P(x_{1:n} | e_{1:n}) \]
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.
Example: Robot Localization
Example: Robot Localization

\[ t=2 \]
Example: Robot Localization

Prob

0

1

t=3
Example: Robot Localization

t=4

Prob

0 1

$t=4$
Example: Robot Localization

\[ t=5 \]
Inference Recap: Simple Cases

\[ P(X_1|e_1) \]

\[ P(x_1|e_1) = \frac{P(x_1, e_1)}{P(e_1)} \]
\[ \propto_{X_1} P(x_1, e_1) \]
\[ = P(x_1)P(e_1|x_1) \]

\[ P(x_2) = \sum_{x_1} P(x_1, x_2) \]
\[ = \sum_{x_1} P(x_1)P(x_2|x_1) \]
Online Belief Updates

- Every time step, we start with current $P(X | \text{evidence})$
- We update for time:

$$P(x_t|e_{1:t-1}) = \sum_{x_{t-1}} P(x_{t-1}|e_{1:t-1}) \cdot P(x_t|x_{t-1})$$

- We update for evidence:

$$P(x_t|e_{1:t}) \propto \sum_x P(x_t|e_{1:t-1}) \cdot P(e_t|x_t)$$
Passage of Time

- Assume we have current belief $P(X \mid \text{evidence to date})$

$$B(X_t) = P(X_t \mid e_{1:t})$$

- Then, after one time step passes:

$$P(X_{t+1} \mid e_{1:t}) = \sum_{x_t} P(X_{t+1} \mid x_t) P(x_t \mid e_{1:t})$$

- Or, compactly:

$$B'(X') = \sum_x P(X' \mid x) B(x)$$

- Basic idea: beliefs get “pushed” through the transitions
  - With the “B” notation, we have to be careful about what time step the belief is about, and what evidence it includes
Example: Passage of Time

- As time passes, uncertainty “accumulates”

$$B'(X') = \sum_x P(X'|x) B(x)$$

Transition model: ghosts usually go clockwise
Assume we have current belief $P(X \mid \text{previous evidence})$:

$$B'(X_{t+1}) = P(X_{t+1} \mid e_{1:t})$$

Then:

$$P(X_{t+1} \mid e_{1:t+1}) \propto P(e_{t+1} \mid X_{t+1})P(X_{t+1} \mid e_{1:t})$$

Or:

$$B(X_{t+1}) \propto P(e \mid X)B'(X_{t+1})$$

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”

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

- We to know:  
  \[ B_t(X) = P(X_t|e_{1:t}) \]
- We can derive the following updates:
  \[
  P(x_t|e_{1:t}) \propto_X P(x_t, e_{1:t}) = \sum_{x_{t-1}} P(x_{t-1}, x_t, e_{1:t})
  = \sum_{x_{t-1}} P(x_{t-1}, e_{1:t-1})P(x_t|x_{t-1})P(e_t|x_t)
  = P(e_t|x_t) \sum_{x_{t-1}} P(x_t|x_{t-1})P(x_{t-1}, e_{1:t-1})
  \]
- To get \( B_t(X) \), compute each entry and normalize
Example: Run the Filter

- An HMM is defined by:
  - Initial distribution: $P(X_1)$
  - Transitions: $P(X_t|X_{t-1})$
  - Emissions: $P(E|X)$
Example HMM
Example Pac-man
Summary: Filtering

- Filtering is the inference process of finding a distribution over $X_T$ given $e_1$ through $e_T$: $P(X_T | e_{1:t})$
- We first compute $P(X_1 | e_1)$:

$$P(x_1|e_1) \propto P(x_1) \cdot P(e_1|x_1)$$

- For each $t$ from 2 to $T$, we have $P(X_{t-1} | e_{1:t-1})$
- **Elapse time:** compute $P(X_t | e_{1:t-1})$

$$P(x_t|e_{1:t-1}) = \sum_{x_{t-1}} P(x_{t-1}|e_{1:t-1}) \cdot P(x_t|x_{t-1})$$

- **Observe:** compute $P(X_t | e_{1:t-1}, e_t) = P(X_t | e_{1:t})$

$$P(x_t|e_{1:t}) \propto P(x_t|e_{1:t-1}) \cdot P(e_t|x_t)$$
Recap: Reasoning Over Time

- **Stationary Markov models**

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

- **Hidden Markov models**

  \[ P(E|X) \]

<table>
<thead>
<tr>
<th>X</th>
<th>E</th>
<th>P</th>
</tr>
</thead>
<tbody>
<tr>
<td>rain</td>
<td>umbrella</td>
<td>0.9</td>
</tr>
<tr>
<td>rain</td>
<td>no umbrella</td>
<td>0.1</td>
</tr>
<tr>
<td>sun</td>
<td>umbrella</td>
<td>0.2</td>
</tr>
<tr>
<td>sun</td>
<td>no umbrella</td>
<td>0.8</td>
</tr>
</tbody>
</table>
Recap: Filtering

Elapse time: compute $P( X_t | e_{1:t-1} )$

$$P(x_t|e_{1:t-1}) = \sum_{x_{t-1}} P(x_{t-1}|e_{1:t-1}) \cdot P(x_t|x_{t-1})$$

Observe: compute $P( X_t | e_{1:t} )$

$$P(x_t|e_{1:t}) \propto P(x_t|e_{1:t-1}) \cdot P(e_t|x_t)$$

Belief: $<P(\text{rain}), P(\text{sun})>$

- $P(X_1) <0.5, 0.5>$ Prior on $X_1$
- $P(X_1 | E_1 = \text{umbrella}) <0.82, 0.18>$ Observe
- $P(X_2 | E_1 = \text{umbrella}) <0.63, 0.37>$ Elapse time
- $P(X_2 | E_1 = \text{umb}, E_2 = \text{umb}) <0.88, 0.12>$ Observe
Particle Filtering

- Sometimes $|X|$ is too big to use exact inference
  - $|X|$ may be too big to even store $B(X)$
  - E.g. $X$ is continuous
  - $|X|^2$ may be too big to do updates

- Solution: approximate inference
  - Track samples of $X$, not all values
  - Samples are called particles
  - Time per step is linear in the number of samples
  - But: number needed may be large
  - In memory: list of particles, not states

- This is how robot localization works in practice
Our representation of $P(X)$ is now a list of $N$ particles (samples)

- Generally, $N << |X|$
- Storing map from $X$ to counts would defeat the point

$P(x)$ approximated by number of particles with value $x$

- So, many $x$ will have $P(x) = 0!$
- More particles, more accuracy

For now, all particles have a weight of 1
Particle Filtering: Elapse Time

- Each particle is moved by sampling its next position from the transition model,

\[ x' = \text{sample}(P(X'|x)) \]

- This is like prior sampling – samples’ frequencies reflect the transition probs
- Here, most samples move clockwise, but some move in another direction or stay in place

- This captures the passage of time
  - If we have enough samples, close to the exact values before and after (consistent)
Particle Filtering: Observe

- **Slightly trickier:**
  - We don’t sample the observation, we fix it
  - We weight our samples based on the evidence

\[
w(x) = P(e|x) \]

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

- Note that, as before, the weights/probabilities don’t sum to one, since most have been downweighted (in fact they sum to an approximation of \( P(e) \))
Particle Filtering: Resample

- Rather than tracking weighted samples, we resample.

- N times, we choose from our weighted sample distribution (i.e. draw with replacement).

- This is equivalent to renormalizing the distribution.

- Now the update is complete for this time step, continue with the next one.

Old Particles:

- (3,3) w=0.1
- (2,1) w=0.9
- (2,1) w=0.9
- (3,1) w=0.4
- (3,2) w=0.3
- (2,2) w=0.4
- (1,1) w=0.4
- (3,1) w=0.4
- (2,1) w=0.9
- (3,2) w=0.3

New Particles:

- (2,1) w=1
- (2,1) w=1
- (2,1) w=1
- (3,2) w=1
- (2,2) w=1
- (2,1) w=1
- (2,1) w=1
- (1,1) w=1
- (3,1) w=1
- (2,1) w=1
- (1,1) w=1
Summary: Particle Filtering

At each time step $t$, we have a set of $N$ particles / samples

- Three step procedure, to move to time $t+1$:
  1. **Sample transitions:** for each each particle $x$, sample next state
     \[ x' = \text{sample}(P(X'|x)) \]
  2. **Reweight:** for each particle, compute its weight
     \[ w(x) = P(e|x) \]
  3. **Resample:** normalize the weights, and sample $N$ new particles from the resulting distribution over states
In robot localization:
- We know the map, but not the robot’s position
- Observations may be vectors of range finder readings
- State space and readings are typically continuous (works basically like a very fine grid) and so we cannot store $B(X)$
- Particle filtering is a main technique
Robot Localization
Which Algorithm?

Exact filter, uniform initial beliefs
Which Algorithm?

Particle filter, uniform initial beliefs, 300 particles
Which Algorithm?

Particle filter, uniform initial beliefs, 25 particles
**P4: Ghostbusters**

- **Plot:** Pacman's grandfather, Grandpac, learned to hunt ghosts for sport.

- He was blinded by his power, but could hear the ghosts’ banging and clanging.

- **Transition Model:** All ghosts move randomly, but are sometimes biased.

- **Emission Model:** Pacman knows a “noisy” distance to each ghost.

### Noisy distance prob

<table>
<thead>
<tr>
<th>True distance = 8</th>
</tr>
</thead>
<tbody>
<tr>
<td>15</td>
</tr>
<tr>
<td>14</td>
</tr>
<tr>
<td>13</td>
</tr>
<tr>
<td>12</td>
</tr>
<tr>
<td>11</td>
</tr>
<tr>
<td>10</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>1</td>
</tr>
</tbody>
</table>
Best Explanation Queries

- Query: most likely seq:

\[
\text{arg max}_{x_{1:t}} P(x_{1:t} | e_{1:t})
\]
Viterbi Algorithm

\[ x^*_{1:T} = \arg \max_{x_{1:T}} P(x_{1:T} | e_{1:T}) = \arg \max_{x_{1:T}} P(x_{1:T}, e_{1:T}) \]

\[ m_t[x_t] = \max_{x_{1:t-1}} P(x_{1:t-1}, x_t, e_{1:t}) \]

\[ = \max_{x_{1:t-1}} P(x_{1:t-1}, e_{1:t-1}) P(x_t | x_{t-1}) P(e_t | x_t) \]

\[ = P(e_t | x_t) \max_{x_{t-1}} P(x_t | x_{t-1}) \max_{x_{1:t-2}} P(x_{1:t-1}, e_{1:t-1}) \]

\[ = P(e_t | x_t) P(x_t | x_{t-1}) m_{t-1}[x_{t-1}] \]
Example
Dynamic Bayes Nets (DBNs)

- We want to track multiple variables over time, using multiple sources of evidence
- Idea: Repeat a fixed Bayes net structure at each time
- Variables from time $t$ can condition on those from $t-1$
- Discrete valued dynamic Bayes nets are also HMMs
DBN Particle Filters

- A particle is a complete sample for a time step
- **Initialize**: Generate prior samples for the t=1 Bayes net
  - Example particle: $G_1^a = (3,3) \; G_1^b = (5,3)$
- **Elapse time**: Sample a successor for each particle
  - Example successor: $G_2^a = (2,3) \; G_2^b = (6,3)$
- **Observe**: Weight each entire sample by the likelihood of the evidence conditioned on the sample
  - Likelihood: $P(E_1^a | G_1^a) \times P(E_1^b | G_1^b)$
- **Resample**: Select prior samples (tuples of values) in proportion to their likelihood