CS532, Winter 2010
Hidden Markov Models

Dr. Alan Fern, afern@eecs.oregonstate.edu

March 8, 2010

1 Hidden Markov Models

The world is dynamic and evolves over time. An intelligent agent in such a world needs to be able to make observations about the world using its sensors and use those observations to make inferences about the state of affairs. Apart from artificial intelligence, there are many applications in engineering and science where noisy observations of a system or process are available and we would like to infer something about the underlying state of the system. This general problem of making inferences about the "hidden" state of a system from noisy observations is one of the primary focusses of the area of probabilistic temporal reasoning. To formalize this type of reasoning process we will model temporal processes using the notions of states and observations over time:

- X_t is a set of state variables at time t of the process of interest. For example, $X_t = \{ \text{hungry}_t, \text{wet}_t, \text{tired}_t \}$ might be state variables describing a baby.
- E_t is a set of evidence variables, or observations, at time t. For example,

$$E_t = \{ \text{cryloud}_t, \text{crysoft}_t, \text{laugh}_t, \text{waveArm}_t, \text{squirming}_t \} .$$

- As time pases the process state evolves and each state produces a corresponding observation in terms of the evidence variables. This gives rise to a sequence of observations $E_1, E_2, \cdots, E_T$ and states $X_1, X_2, \cdots, X_T$.

We will assume the ability to directly observe the evidence variables, but the state variables are often hidden though those are the variables of primary interest. The goal then is to make inferences about the state variables given the observation sequence. In order to make such inferences, it is common to assume that:

- the process follows a probabilistic model,
- the evolution of process states is not affected by the evidence,
- the following first-order Markov condition holds:

$$\forall\, t,\ \mathbf{P}(X_t|X_0, \cdots, X_{t-1}) = \mathbf{P}(X_t|X_{t-1})$$

- the process is stationary, like a proposal distribution of MCMC:

$$\mathbf{P}(X_t|X_{t-1}) = \mathbf{P}(X_{t-1}|X_t)$$

Under these assumptions we can encode the state evolution using just a prior on X_0, $\mathbf{P}(X_0)$, which gives the probability of the intial state being equal to X_0 and $\mathbf{P}(X_t|X_{t-1})$ which gives the probability of the state being X_t when the previous state was X_{t-1}.

It is also common to assume that evidence E_t at time t only depends on X_t:

$$\mathbf{P}(E_t|E_{0:t-1}, X_{0:t}) = \mathbf{P}(E_t|X_t)$$

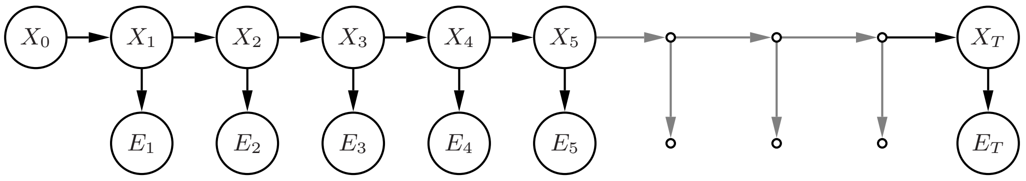
Also, assume that the **observation model** is a stationary distribution. That is,

$$\forall\, t, v\ \mathbf{P}(E_t|X_t) = \mathbf{P}(E_v|X_v)$$

From the above we will specify a first-order Markov process using the following three components:

Initial State Prior:	$\mathbf{P}(X_0)$
Transition Prob.:	$\mathbf{P}(X_t|X_{t-1})$
Observation Prob.:	$\mathbf{P}(E_t|X_t)$

Given this model we draw a graphical representation of the first-order Markov process as follows:



Under this model the joint probability of a pair of observation and state sequences is given by the following:

$$\mathbf{P}(X_0, \cdots, X_T, E_1, \cdots, E_T) = \mathbf{P}(X_0) \prod_{t=1}^{T} \mathbf{P}(X_t|X_{t-1})\mathbf{P}(E_t|X_t)$$

A **Hidden Markov Model (HMM)** is a first-order Markov process where X_t and E_t contain a single discrete variable each. HMMs are widely used in speech recognition, computer vision, grammatical inference, bioinformatics, communications, etc. Note that even when X_t and E_t contain multiple variables, we can think of them as forming a single joint variable with many values. Thus, HMMs are really quite general from a representational perspective.

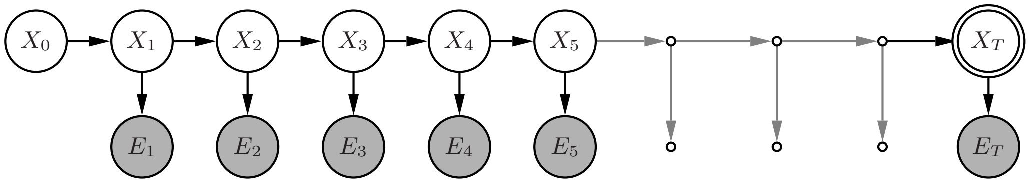
2 Inference Problems for HMMs

We are typically interested in three types of temporal inference problems.

2.1 Filtering or Monitoring

Given evidence up to time t, what is the distribution over X_t:

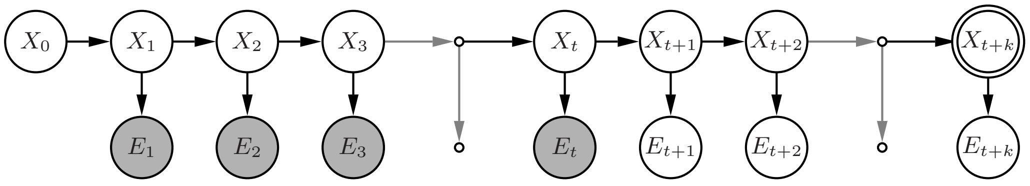
$$\mathbf{P}(X_t|e_{1:t})$$



Ideally, we would like to have an incremental method to compute $\mathbf{P}(X_{t+1}|E_{t+1} = e_{t+1})$ from $\mathbf{P}(X_t|E_t = e_t)$ since often filtering will be done on streaming data and must be fast.

2.2 Prediction

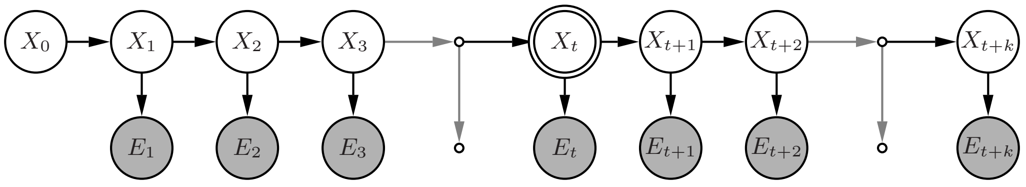
$$\mathbf{P}(X_{t+k}|e_{1:t})$$



This inference problem is similar to what a meteorologist must do each day. Predict the future weather from past and current observations. The accuracy of the predictions will typically degrade as the value of k increases.

2.3 Smoothing

$$\mathbf{P}(X_t|e_{1:t+k})$$



Here we want to get the distribution of X_t given past and future observations. This is similar to filtering, but allows for the use of "future observations". This inference scenario arises when one is storing evidence sequences and wants to computing the most accurate distribution over X_t possible. By reasoning about both the time before and after t higher accuracy can be achieved than in pure filtering where evidence is not consider after time t.

2.4 Most Likely Sequence

$$\arg\max_{X_{0:t}} \mathbf{P}(X_{0:t}|e_{0:t})$$

Here we want to compute the most likely state sequence for a given observation sequence.

3 Filtering with HMMs

We want an online algorithm that – given $\mathbf{P}(X_t|e_{1:t})$ can efficiently compute $\mathbf{P}(X_{t+1}|e_{1:t+1})$ upon observing e_{t+1}. Note that for $t = 0$ the distribution is given by the initial state distribution of the model.

$$\begin{aligned}
\mathbf{P}(X_{t+1}|e_{1:t+1}) &= \mathbf{P}(X_{t+1}|e_{1:t}, e_{t+1}) \\
&= \mathbf{P}(e_{t+1}|X_{t+1}, e_{1:t})\mathbf{P}(X_{t+1}|e_{1:t})/\mathbf{P}(e_{t+1}|e_{1:t}) \\
&= \alpha\mathbf{P}(e_{t+1}|X_{t+1}, e_{1:t})\mathbf{P}(X_{t+1}|e_{1:t}) \\
&= \alpha\mathbf{P}(e_{t+1}|X_{t+1})\mathbf{P}(X_{t+1}|e_{1:t})
\end{aligned}$$

$$\begin{aligned}
\mathbf{P}(X_{t+1}|e_{1:t}) &= \sum_{X_t} \mathbf{P}(X_{t+1}, X_t|e_{1:t}) \\
&= \sum_{X_t} \mathbf{P}(X_{t+1}|X_t)\mathbf{P}(X_t|e_{1:t}) \\
\mathbf{P}(X_{t+1}|e_{1:t+1}) &= \alpha\mathbf{P}(e_{t+1}|X_{t+1}) \sum_{X_t} \mathbf{P}(X_{t+1}|X_t)\mathbf{P}(X_t|e_{1:t})
\end{aligned}$$

For those familiar with Bayes net inference, this algorithm is just a special case of belief propagation.

What is the complexity of each filtering step? We must compute a summation over values of X_t for each value of X_{t+1}. So, time complexity is O(k^2), where k is the number of states. Space complexity is k, since we just need to store $\mathbf{P}(X_{t+1}|e_{1:t+1})$.

The book shows an example calculation on page 543.

4 Prediction with HMMs

We can get a recursive form for prediction, allowing us to easily compute the prediction for $t+k+1$ from the prediction for $t+k$.

$$\mathbf{P}(X_{t+k+1}|e_{1:t}) = \sum_{X_{t+k}} \mathbf{P}(X_{t+k+1}|X_{t+k})\mathbf{P}(X_{t+k}|e_{1:t})$$

The base case of $k = 0$ corresponds exactly to the filtering problem $\mathbf{P}(X_t|e_{1:t})$, which we already described.

5 Smoothing with HMMs

We want to compute:

$$\mathbf{P}(X_k|e_{1:t})$$

for each $1 \leq k \leq t$. Note that $\mathbf{P}(X_k|e_{1:t})$ is just a marginal distribution (or belief) at X_k.

This inference problem can be solved with a general technique known as belief propagation for general graphical models. We will derive this algorithm for the special case of HMMs. The resulting algorithm is known as the **forward-backward algorithm**.

$$\begin{aligned}
\mathbf{P}(X_k|e_{1:t}) &= \mathbf{P}(X_k|e_{1:k}, e_{k+1:t}) \\
&= \mathbf{P}(X_k|e_{1:k})\mathbf{P}(e_{k+1:t}|X_k, e_{1:k})/\mathbf{P}(e_{k+1:t}|e_{1:k}) \\
&= \alpha\mathbf{P}(X_k|e_{1:k})\mathbf{P}(e_{k+1:t}|X_k, e_{1:k}) \\
&= \alpha\mathbf{P}(X_k|e_{1:k})\mathbf{P}(e_{k+1:t}|X_k)
\end{aligned}$$

The above shows that we need to compute two factors for the smoothing problem. We can compute the first factor $\mathbf{P}(X_k|e_{1:k})$ for each k by running a filtering pass of the sequence as already described. For the second factor we can derive:

$$\begin{aligned}
\mathbf{P}(e_{k+1:t}|X_k) &= \sum_{X_{k+1}} \mathbf{P}(e_{k+1:t}, X_{k+1}|X_k) \\
&= \sum_{X_{k+1}} \mathbf{P}(e_{k+1:t}|X_{k+1}, X_k)\mathbf{P}(X_{k+1}|X_k) \\
&= \sum_{X_{k+1}} \mathbf{P}(e_{k+1:t}|X_{k+1})\mathbf{P}(X_{k+1}|X_k) \\
&= \sum_{X_{k+1}} \mathbf{P}(e_{k+1}, e_{k+2:t}|X_{k+1})\mathbf{P}(X_{k+1}|X_k) \\
&= \sum_{X_{k+1}} \mathbf{P}(e_{k+1}|X_{k+1})\mathbf{P}(e_{k+2:t}|X_{k+1})\mathbf{P}(X_{k+1}|X_k)
\end{aligned}$$

The second term $\mathbf{P}(e_{k+2:t}|X_{k+1})$ can then be computed recursively.

In other words, $\mathbf{P}(e_{k+1:t}|X_k)$ is computed for each k in a backward pass over the sequence initialized with

$$\mathbf{P}(e_{t+1:t}|X_t) = 1$$

for all values of X_t. The book provides an example on page 545.

5.1 Most Likely Sequence

We want to compute

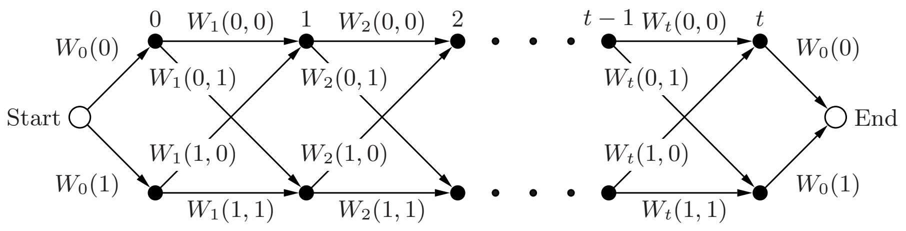
$$X^*_{1:t} = \arg\max_{X_{1:t}} \mathbf{P}(X_{1:t}|e_{1:t})$$

This is *not* the same as computing $\arg\max \mathbf{P}(X_t|e_{1:t})$ for each i.[1]

We will use the **Viterbi algorithm** for this problem. The Viterbi algorithm is simple the max-product algorithm applied using variable elimination order $X_1, \cdots, X_n$. See the book for detailed equations.

$$\begin{aligned}
\arg\max_{X_{1:t}} \mathbf{P}(X_{1:t}|e_{1:t}) &= \arg\max_{X_{1:t}} \log \mathbf{P}(X_{1:t}|e_{1:e}) \\
&= \arg\max_{X_{1:t}} \log \mathbf{P}(X_0)\mathbf{P}(X_1|X_0)\mathbf{P}(e_1|X_1)\cdots\mathbf{P}(x_t|X_{t-1})\mathbf{P}(X_t|e_t) \\
&= \arg\max_{X_{1:t}} \log \mathbf{P}(X_0) + \log \mathbf{P}(X_1|X_0)\mathbf{P}(e_1|X_1) + \cdots + \log \mathbf{P}(x_t|X_{t-1})\mathbf{P}(X_t|e_t) \\
&= \arg\max_{X_{1:t}} W_0(X_0) + W_1(X_0, X_1) + \cdots + W_t(X_{t-1}, X_t)
\end{aligned}$$

The Viterbi algorithm is often visualized using a **Viterbi trellis**:



The problem can thus be viewed as finding shortest path in trellis from start to finish.

[1]You will show this in the homework.