PROBABILITY

Conditional probability: $P(x|y) = \frac{P(x,y)}{P(y)} = \frac{P(y|x)P(x)}{P(y)}$ **Chain rule:** $P(X_1,\ldots,X_n) = \prod_{i=1}^{n} P(x_i|\text{parents}(X_i))$ $X \perp\!\!\!\perp Y|Z$ if $P(x,y|z) = P(x|z)P(y|z)$or$P(x|y,z) = P(x|z)$or$P(y|x,z) = P(y|z)$

BAYES NETS

Inference by enumeration: chain rule: $P(X_1,\ldots,X_n) = \prod_{i=1}^{n} P(x_i|\text{parents}(X_i))$. sum out the hidden variables.

$$P(B|j,m) \propto P(B,j,m) = \sum_e \sum_a P(B,j,m,e,a)$$

$$= \sum_e \sum_a P(b)P(e)P(a|b,e)P(j|a)P(m|a)$$

Variable elimination: do a calculation once and save it for later use

$$P(B,j,m) = \underbrace{P(B)}_{f_1(B)} \sum_e \underbrace{P(e)}_{f_2(E)} \sum_a \underbrace{P(a|B,e)}_{f_3(A,B,E)} \underbrace{P(j|a)}_{f_4(A)} \underbrace{P(m|a)}_{f_5(A)}$$

Sum out A from the product of f_3, f_4, f_5 to make a new 2×2 factor $f_6(B,E)$:

$$f_6(B,E) = \sum_a f_3(A,B,E) \times f_4(A) \times f_5(A) =$$

$$f_3(a,B,E)\times f_4(a)\times f_5(a)+f_3(\neg a,B,E)\times f_4(\neg a)\times f_5(\neg a)$$

Now we have

$$P(B,j,m) = f_1(B) \times \sum_e f_2(E) \times f_6(B,E)$$

(remember that if you have an entry $P(A,B) \times P(B,C) \Rightarrow P(A,B,C) = P(A,B) \cdot P(B,C)$)

Likelihood weighting avoids the inefficiency of rejection sampling by generating only events that are consistent with the evidence e. We fix the values for the evidence var **E** and sample only the nonevidence variables, guaranteeing that each event we generate is consistent with the evidence. Before tallying the counts in the distribution for the query var, we have to weight event by the **likelihood** that the event accords to the evidence, measured by the product of the conditional probabilities for each evidence variable given its parents. Consider the query P(Rain|Cloudy=true, WetGrass=true) with the ordering Cloudy, Sprinkler, Rain, WetGrass (though any topological ordering will work). First, set weight w to 1.0 then generate an event: **1.** Cloudy is an evidence var with the value true so we take $w \leftarrow w \cdot P$(Cloudy=true) = .5 **2.** Sprinkler is **not** an evidence var, so sample from the distribution so far: P(Sprinkler|Cloudy=true). Say we get "false" from this sample. **3.** Rain is not an evidence var, so we sample from P(Rain|Cloudy=true). say we get "true" from this sample. **4.**WetGrass **is** an evidence var with the value true, so we adjust the weight again: $w \leftarrow w \cdot P$(WetGrass=true|Sprinkler=false ,Rain=true) Now we have a weighted sample, the event [true,false,true,true] with a weight w and we tally this in our sample distribution under Rain = true.

Gibbs sampling: Look at the query P(Rain|Sprinkler=true, WetGrass=true). Essentially, what we do is set evidence then set al other variables to random values (by prior sampling or uniformly sampling) then choosing a non-evidence variable and sample this variable conditioned on nothing else changing (we generate samples where each sample is different from the previous one by only a single variable). Each state visited during this process is a sample that contributes to the estimate for the query variable Rain. If the process visits 20 states where Rain is true and 60 where Rain is false, then P(Rain=true) = .25 and P(Rain=false) = .75.

Active triples: (*active* means it carries information, or dependence) **1.** $A \rightarrow B \rightarrow C$ **2.** $A \leftarrow B \rightarrow C$ **3.** $A \rightarrow \underline{B} \leftarrow C$ **4.** $A \rightarrow B \leftarrow C : B \rightarrow \underline{D}$

Inactive triples 1. $A \rightarrow \underline{B} \rightarrow C$ **2.** $A \leftarrow \underline{B} \rightarrow C$ **3.** $A \rightarrow B \leftarrow C$

HIDDEN MARKOV MODELS

Defined by: initial distribution $P(X_1)$, Transitions $P(X|X_{-1})$ and emissions $P(E|X)$. Two important independence properties: markov hidden process, future depends on past via the present. Current observation independent of all else given current state.

- Passage of Time: Have current belief $P(X|\text{evidence to date})$: $B(X_t) = P(X_t|e_{1:t})$

  After one time step passes: $P(X_{t+1}|e_{1:t}) = \sum_{x_t} P(X_{t+1}|x_t)P(x_t|e_{1:t})$, or $B'(X_{t+1}) = \sum_{x_t} P(X'|x)B(x_t)$ Beliefs get "pushed" through the transitions

- Observation: Have current Belief $P(X|$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 $B(X_{t+1}) \propto P(e|X)B'(X_{t+1})$ Beliefs are reweighted by likelihood of all evidence. But unlike passage of time, we have to renormalize.

- Forward algorithm: Given evidence at each time and want to know $B_t(X) = P(X_t|e_{1:t})$, Can derive the following updates: $P(x_t|e_{1:t}) \propto P(e_t|x_t)\sum_{x_{t-1}} P(x_t|x_{t-1})P(x_{t-1}, e_{1:t-1})$. This is variable elimination in order $X_1, X_2, \ldots$

- Backward algorithm: $P(e_{t+1:N}|x_t) = \sum_{x_{t+1}} P(e_{t+1:N}|x_{t+1})P(x_{t+1}|x_t)$

- Online Belief Updates: for every time step, we start with current $P(X|$evidence). We update for time: $P(x_t|e_{1:t-1}) = \sum_{x_{t-1}} P(x_{t-1}|e_{1:t-1}) \cdot P(x_t|x_{t-1})$. Then we update for evidence: $P(x_t|e_{1:t}) \propto_X P(x_t|e_{1:t-1}) \cdot P(e_t|x_t)$. The forward algorithm does both at once (and doesn't normalize)

- Particle filtering (the other inference method for HMMs). Filtering is an approximate solution. Sometimes X is too big to use exact inference. Track samples of X, not all values (samples are called particles)

  Representation: representation of $P(X)$ is now a list of N particles (samples). Generally, $N << |X|$. $P(X)$ is approximated by number of particles with value x.

  Elapse Time: Each particle is moved by sampling its next position from the transition model: $x' =$sample$(P(X'|x))$. This is like prior sampling – samples' frequencies reflect the transition probabilities. This captures the passage of time.

  Observe: don't sample the observation, we fix it. This is similar to likelihood weighting, so we downweight our samples based on the evidence:

  $w(x) = P(e|x) \quad B(X) \propto P(e|X)B'(X)$

  Resample: Rather than tracking weighted samples, we resample N times. We choose from our weighted sample distribution (draw w/ replacement). This is equivalent to renormalizing the distribution. Now the update is complete for this time step - continue w/ the next one

Dynamic Bayes Nets (DBNs)

- We want to track multiple variables over time, using multiple sources of evidence. 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

- Exact inference in DBNs: variable elimination applies to dynamic Bayes nets. Procedure: "unroll" the network for T time steps, then eliminate variables until $P(X_t|e_{1:T})$ is computed. Online belief updates: Eliminate all variables from the previous time step; store factors for current time only

- DBN Particle Filters: a particle is a complete sample for a time step. **Initialize**: Generate prior samples for the $t = 1$ Bayes net. **Elapse time**: sample a successor for each particle **Observe**: weight each entire sample by the likelihood of the evidence condition on the sample

MACHINE LEARNING

Binary Perceptron Update: Start with zero weights. For each training instance, classify with current weights:

$$y = \begin{cases} +1 & \text{if } w \cdot f(x) \geq 0 \\ -1 & \text{if } w \cdot f(x) < 0 \end{cases}$$

Learning multiclass perceptron: start with zero weights, pick up training instances one by one, classify with current weights:

$$y = \text{argmax}_y w_y \cdot f(x) = \text{argmax}_y \sum_i w_{y,i} \cdot f_i(x)$$

If correct, no change! If wrong, lower score of wrong answer, raise score of right answer: $w_y = w_y - f(x) \quad w_{y^*} = w_{y^*} + f(x)$
Fixing the perceptron: adjust the weight update to mitigate these effects. MIRA (Margin Infused Relaxed Algorithm) chooses an update size that fixes the current mistake. $\min_w \frac{1}{2}||w_y - w'_y||^2$. We want $w_{y^*} \cdot f(x) \geq w_y \cdot f(x) + 1$ (the +1 helps to generalize)

$$w_y = w'_y - \tau f(x)$$

$$w_{y^*} = w'_{y^*} + \tau f(x)$$

. We want to minimize τ but not $\tau = 0$ or we can't update anything, so we need the minimum where equality holds. w_y is weight for true label, w'_y is weight for guessed label