CS 188
Spring 2019

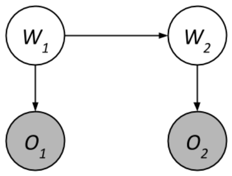
Section Handout 9 Solutions

CS188 Spring 2019 Section 9: HMMs and Particle Filtering

1 HMMs

State variables W_t and observation (evidence) variables (O_t), which are supposed to be shaded below. Transition model $P(W_{t+1}|W_t)$. Sensor model $P(O_t|W_t)$. The joint distribution of the HMM can be factorized as

$$P(W_1, ..., W_T, O_1, ...O_T) = P(W_1) \prod_{t=1}^{T-1} P(W_{t+1}|W_t) \prod_{t=1}^{T} P(O_t|W_t) \quad (1)$$



Define the following belief distribution

- $B(W_t) = P(W_t|O_1, ..., O_t)$: Belief about state W_t given all the observations up until (and including) timestep t.
- $B'(W_t) = P(W_t|O_1, ..., O_{t-1})$: Belief about state W_t given all the observations up until (but not including) timestep t.

2 Forward Algorithm

- *Prediction update*: $B'(W_{t+1}) = \sum_{w_t} P(W_{t+1}|w_t)B(w_t)$
- *Observation update*: $B(W_{t+1}) \propto P(O_{t+1}|W_{t+1})B'(W_{t+1})$

3 Particle Filtering

The Hidden Markov Model analog to Bayes' net sampling is called **particle filtering**, and involves simulating the motion of a set of particles through a state graph to approximate the probability (belief) distribution of the random variable in question.

Instead of storing a full probability table mapping each state to its belief probability, we'll instead store a list of n particles, where each particle is in one of the d possible states in the domain of our time-dependent random variable.

Once we've sampled an initial list of particles, the simulation takes on a similar form to the forward algorithm, with a time elapse update followed by an observation update at each timestep:

- *Prediction update* - Update the value of each particle according to the transition model. For a particle in state W_t, sample the updated value from the probability distribution given by $Pr(W_{t+1}|w_t)$. Note the similarity of the prediction update to prior sampling with Bayes' nets, since the frequency of particles in any given state reflects the transition probabilities.

- *Observation update* - During the observation update for particle filtering, we use the sensor model $Pr(O_t|W_t)$ to weight each particle according to the probability dictated by the observed evidence and the particle's state. Specifically, for a particle in state w_t with sensor reading o_t, assign a weight of $Pr(o_t|w_t)$. The algorithm for the observation update is as follows:

  1. Calculate the weights of all particles as described above.
  2. Calculate the total weight for each state.
  3. If the sum of all weights across all states is 0, reinitialize all particles.
  4. Else, normalize the distribution of total weights over states and resample your list of particles from this distribution.

  Note the similarity of the observation update to likelihood weighting, where we again downweight samples based on our evidence.

4 Viterbi Algorithm

Question: *What is the most likely sequence of hidden states the system followed given the observed evidence variables so far?* In other words, we would like to solve for $\arg\max_{w_{1:T}} P(w_{1:T}|o_{1:T}) = \arg\max_{w_{1:T}} P(w_{1:T}, o_{1:T})$.

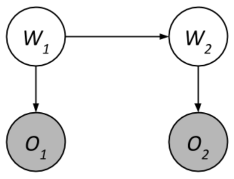
Define $m_t[w_t] = \max_{w_{1:t-1}} P(w_{1:t}, o_{1:t})$, or the maximum probability of a path starting at any w_0 and the evidence seen so far to a given w_t at time t. This is the same as the highest weight path through the trellis of possible state values from step 1 to t. Also note that

$$\begin{aligned} m_t[w_t] &= \max_{w_{1:t-1}} P(o_t|w_t)P(w_t|w_{t-1})P(w_{1:t-1}, o_{1:t-1}) & (2)\\ &= P(o_t|w_t)\max_{w_{t-1}} P(w_t|w_{t-1}) \max_{w_{1:t-2}} P(w_{1:t-1}, o_{1:t-1}) & (3)\\ &= P(o_t|w_t)\max_{w_{t-1}} P(w_t|w_{t-1}) m_{t-1}[w_{t-1}]. & (4) \end{aligned}$$

This suggests that we can compute m_t for all t recursively via dynamic programming. This makes it possible to determine the last state w_N for the most likely path, but we still need a way to backtrack to reconstruct the entire path. Let's define $a_t[w_t] = P(o_t|w_t)\arg\max_{w_{t-1}} P(w_t|w_{t-1})m_{t-1}[w_{t-1}] = \arg\max_{w_{t-1}} P(w_t|w_{t-1})m_{t-1}[w_{t-1}]$ to keep track of the last transition along the best path to w_t.

5 HMMs

Consider the following Hidden Markov Model. O_1 and O_2 are supposed to be shaded.



W_1	$P(W_1)$
0	0.3
1	0.7

W_t	W_{t+1}	$P(W_{t+1}\|W_t)$
0	0	0.4
0	1	0.6
1	0	0.8
1	1	0.2

W_t	O_t	$P(O_t\|W_t)$
0	a	0.9
0	b	0.1
1	a	0.5
1	b	0.5

Suppose that we observe $O_1 = a$ and $O_2 = b$.
Using the forward algorithm, compute the probability distribution $P(W_2|O_1 = a, O_2 = b)$ one step at a time.

(a) Compute $P(W_1, O_1 = a)$.

$P(W_1, O_1 = a) = P(W_1)P(O_1 = a|W_1)$
$P(W_1 = 0, O_1 = a) = (0.3)(0.9) = 0.27$
$P(W_1 = 1, O_1 = a) = (0.7)(0.5) = 0.35$

(b) Using the previous calculation, compute $P(W_2, O_1 = a)$.

$P(W_2, O_1 = a) = \sum_{w_1} P(w_1, O_1 = a)P(W_2|w_1)$
$P(W_2 = 0, O_1 = a) = (0.27)(0.4) + (0.35)(0.8) = 0.388$
$P(W_2 = 1, O_1 = a) = (0.27)(0.6) + (0.35)(0.2) = 0.232$

(c) Using the previous calculation, compute $P(W_2, O_1 = a, O_2 = b)$.

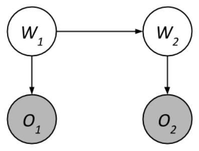
$P(W_2, O_1 = a, O_2 = b) = P(W_2, O_1 = a)P(O_2 = b|W_2)$
$P(W_2 = 0, O_1 = a, O_2 = b) = (0.388)(0.1) = 0.0388$
$P(W_2 = 1, O_1 = a, O_2 = b) = (0.232)(0.5) = 0.116$

(d) Finally, compute $P(W_2|O_1 = a, O_2 = b)$.

Renormalizing the distribution above, we have
$P(W_2 = 0|O_1 = a, O_2 = b) = 0.0388/(0.0388 + 0.116) \approx 0.25$
$P(W_2 = 1|O_1 = a, O_2 = b) = 0.116/(0.0388 + 0.116) \approx 0.75$

6 Particle Filtering

Let's use Particle Filtering to estimate the distribution of $P(W_2|O_1 = a, O_2 = b)$. Here's the HMM again. O_1 and O_2 are supposed to be shaded.



W_1	$P(W_1)$
0	0.3
1	0.7

W_t	W_{t+1}	$P(W_{t+1}\|W_t)$
0	0	0.4
0	1	0.6
1	0	0.8
1	1	0.2

W_t	O_t	$P(O_t\|W_t)$
0	a	0.9
0	b	0.1
1	a	0.5
1	b	0.5

We start with two particles representing our distribution for W_1.
$P_1 : W_1 = 0$
$P_2 : W_1 = 1$
Use the following random numbers to run particle filtering:

$$[0.22, 0.05, 0.33, 0.20, 0.84, 0.54, 0.79, 0.66, 0.14, 0.96]$$

(a) **Observe**: Compute the weight of the two particles after evidence $O_1 = a$.

$w(P_1) = P(O_t = a|W_t = 0) = 0.9$
$w(P_2) = P(O_t = a|W_t = 1) = 0.5$

(b) **Resample**: Using the random numbers, resample P_1 and P_2 based on the weights.

We now sample from the weighted distribution we found above. After normalizing the weights, we find that P_1 maps to range [0, 0.643), and P_2 maps to range [0.643, 1). Using the first two random samples, we find:
$P_1 = sample(weights, 0.22) = 0$
$P_2 = sample(weights, 0.05) = 0$

(c) **Predict**: Sample P_1 and P_2 from applying the time update.

$P_1 = sample(P(W_{t+1}|W_t = 0), 0.33) = 0$
$P_2 = sample(P(W_{t+1}|W_t = 0), 0.20) = 0$

(d) **Update**: Compute the weight of the two particles after evidence $O_2 = b$.

$w(P_1) = P(O_t = b|W_t = 0) = 0.1$
$w(P_2) = P(O_t = b|W_t = 0) = 0.1$

(e) **Resample**: Using the random numbers, resample P_1 and P_2 based on the weights.

Because both of our particles have $X = 0$, resampling will still leave us with two particles with $X = 0$.
$P_1 = 0$
$P_2 = 0$

(f) What is our estimated distribution for $P(W_2|O_1 = a, O_2 = b)$?

$P(W_2 = 0|O_1 = a, O_2 = b) = 2/2 = 1$
$P(W_2 = 1|O_1 = a, O_2 = b) = 0/2 = 0$

Q7. HMMs (Optional)

Now we will consider a Hidden Markov Model, and look at properties of the Viterbi Algorithm. The Viterbi algorithm finds the most probable sequence of hidden states $X_{1:T}$ given a sequence of observations $y_{1:T}$. Recall that for the canonical HMM structure, the Viterbi algorithm performs the following update at each time step:

$$m_t[x_t] = P(y_t|x_t) \max_{x_{t-1}}[P(x_t|x_{t-1})m_{t-1}[x_{t-1}]]$$

Assume we have an HMM where:

- The hidden variable X can take on n values
- The (observed) emission variable Y can take on m values
- Our sequence has T steps

(a) What is the run time of the Viterbi algorithm?

○ $O(Tmn)$ ○ $O(Tmn^2)$ [✓ $O(Tn^2)$]
○ $O(Tn)$ ○ $O(mn)$ ○ $O(mn^2)$
○ $O(Tn^2 + Tmn)$

Ignoring the storage of the emission probabilities, $P(Y_t|X_t)$, and the transition probabilities, $P(X_t|X_{t-1})$, what are the storage requirements of the Viterbi algorithm?

○ $O(T)$ ○ $O(m)$ ○ $O(n)$
[✓ $O(Tn)$] ○ $O(Tm)$ ○ $O(mn)$
○ $O(T+n)$ ○ $O(T+m)$ ○ $O(m+n)$
○ $O(Tmn)$

Now, assume that most of the transitions in our nMM have probability zero. In particular, suppose that for any given hidden state value, there are only K possible next state values for which the transition probability is non-zero. To exploit this sparsity, we change the Viterbi Algorithm to only consider the non-zero transition edges during each max computation inside each update. You can think of this as the Viterbi algorithm ignoring edges that correspond to zero probability transitions in the transition lattice diagram.

(b) What is the run time of this modified algorithm?

○ $O(Tmn)$ ○ $O(Tmn^2)$ ○ $O(Tn^2)$
○ $O(Tn)$ ○ $O(mn)$ ○ $O(mn^2)$
○ $O(Tn^2 + Tmn)$
○ $O(TmK)$ ○ $O(TmnK)$ [✓ $O(TnK)$]
○ $O(TK)$ ○ $O(mK)$ ○ $O(mnK)$
○ $O(TK + TmK)$ ○ $O(TnK + TmK)$

Ignoring the storage of the emission probabilities, $P(Y_t|X_t)$, and the transition probabilities, $P(X_t|X_{t-1})$, what are the storage requirements of this modified Viterbi algorithm?

○ $O(T)$ ○ $O(m)$ ○ $O(n)$
[✓ $O(Tn)$] ○ $O(Tm)$ ○ $O(mn)$
○ $O(T+n)$ ○ $O(T+m)$ ○ $O(m+n)$
○ $O(Tmn)$
○ $O(K)$ ○ $O(TK)$ ○ $O(mK)$
○ $O(nK)$ ○ $O(T+K)$ ○ $O(m+K)$
○ $O(n+K)$ ○ $O(TmK)$