CPSC 422

Practice Midterm Exam Solutions

March 2011

The exam covers up to and including Q-learning and SARSA in Reinforcement learning.

1. (a) A transduction is a function from a percept trace into a command trace. It specifies the action of an agent as a function of what is observed. An agent can only base its action at any time in its previous and current observations (not its future observations), so it can only implement causal transductions which are transductions that only depend on percepts up to a certain time.

   (b) An agent does not have access to its history; it can only access what it has remembered. A belief state of an agent encodes what an agent has remembered.

   (c) To implement a causal transduction, an agent needs access to decide what to do based only on what it has remembered and its percepts (this is the command function). It also needs to remember information so it can be sued for future actions. All other functions are just there to make programming easier, and are not strictly needed.

2. $q[34,7] = q[34,7] + \alpha * (3 + \gamma * max_a q[65,a] - q[34,7])$

3. The initial values are not as good estimates as newer values, and so we may not want to weight them as much. It is simpler to ignore the counts (and so keep α fixed). With a fixed α an agent is able to adapt when the environment changes.

4. It gets stuck in non-optimal policies because it does not explore enough to find the best action from each state. To explore, it can pick random actions occasionally. You could also set the initial values high, so that unexplored regions look good.

5. With no discounting the sum of the rewards is often infinite. Discounting means that more recent rewards are more valuable than rewards far in the future.

6. In standard value iteration all of the values are updated from the previous values in a sweep through the values. In asynchronous value iteration, the values are updated from the current value and can be done in any order (you don't need to sweep through all of the values). It often works better because the latest values are always used and it can concentrate on updating values where they make the most difference (as it doesn't need to sweep through all of the values each time).

7. Q-learning is like asynchronous value iteration in that it updates the Q values, but it uses experience rather than using a model. Thus the average is the average over its experience rather than computing the expected value using a model.

8. (a) $Q(s1, \mathit{right}) = 0$. $Q(s2, \mathit{right}) = -10$. $Q(s3, \mathit{right}) = 0$. $Q(s4, \mathit{right}) = 10$. Each action gets its immediate rewards, as the future value is 0.

   (b) $Q(s1, \mathit{right}) = 0$. $Q(s2, \mathit{right}) = -5$. $Q(s3, \mathit{right}) = 4.5$. $Q(s4, \mathit{right}) = 10$. (Assuming that $s5$ does not have a positive q-value). State $s3$ has no immediate reward, so its average value is $0.5 * 0.9 * 10$. $s2$ gets its average reward, which is -5. Note that $s1$ works differently to $s4$, as it uses the maximum Q-value of $s2$ which is zero.

   (c) In SARSA, $Q(s1, \mathit{right})$ would be -4.5, because it uses $Q(s2, \mathit{right})$, which is -10, rather than $\max_a Q(s2, a)$, which is 0.

9. EM algorithm is used for learning probabilities when the value for some variable is not observed in the data. (E.g., the class variable may not be observed). In the E-step, the data is filled in based on the probabilistic model (we get the expected number in the data). In the M-step the probabilities are updated based on the augmented data (we get the maximum likelihood probabilities).

10. This gives not very good estimates if m is small or if $n = 0$ or $n = m$. Just because you have not observed something does not mean that it should have probability zero (which means it is impossible).

11. Particle filtering would be most useful, as it can operate at the scale of a factory, and can handle multiple sensors. It requires a model of the dynamics of the robot, and a sensor model, both of which require a map to determine where it is.

Variable elimination is not suitable, as there would be too many states to have an explicit representation of the distribution over states.

In rejection sampling, all of the hypotheses would be quickly rejected, leaving us with no samples to estimate probabilities.

12. (a) is true, and the others are false. It can always do better on the test set. One the training set, the empirical proportion can keep the error the same or improve it. However, adding 1 can make the extra split worse.

    Overfitting can occur if the split in t_1 improves the prediction on the training set but makes the prediction on the test set worse.

13. It is possible that a naive Bayesian classifier can have a worse error on the training data than just predicting a point probability of the class (ignoring all of the input features)? If so, when would this occur. If not, explain why.

    Yes, it can have a worse error if the independence of the classifier is not appropriate. That is if the other features are not independent given the class.