CAS CS 440
Lec 31

Policy Learning V: Reinforcement Learning II

1. Last Times
    a. Passive RL
        i. Agent follows whatever policy it is given/calculates
            1. Agent just follows the policy
        ii. Policy Estimation:
            1. Value Iteration
            2. Policy Iteration
            3. Direct Utility Estimation
            4. Adaptive Dynamic Programming
            5. Temporal-Difference Learning
        iii. Requires more time and memory
2. This Time
    a. Active RL
        i. Agent can ignore it's policy
        ii. Why?
        iii. Exploration/Exploitation
            1. Agent can have notion of how good the policy is so that it can explore the world
            2. Once it has more confidence, it can exploit
            3. In passive, the agents have 100% of confidence every time
            4. In active, it can have non 100% of confidence and choose to ignore the policy
        iv. Q Learning → do with neural networks
    b. Neural Networks
        i. How they work
        ii. Function approximators
3. Passive vs Active RL
    a. A passive agent has a policy inside of it
        i. Policy is fixed during an episode agent always obeys policy
        ii. (Can be) slow
        iii. (Can be) bad
    b. An active agent also has a policy inside of it
        i. Agent can ignore the policy
        ii. Agent can modify the policy with new information

4. An Agent ADP Agent
    a. ADP learns transition model Pr[s’|s, a] (by counting)
    b. Want: need to learn Pr[s’|s, a] for every action
        i. Will eventually get there…need to see more episodes
    c. Utilities the agent needs to learn are those of the optimal policy
        i. Can calculate these for the learned model (i.e. the policy it currently has)
    d. Should the agent follow the optimal action recommended by its current policy?
5. Always Following “Optimal” Actions
    a. Let’s pretend we know what the optimal action is (for my current policy)
    b. Should I follow that policy’s recommendation?
        i. Agent is greedy
        ii. If policy is already optimal → action is truly optimal
            1. If the agent did not play a lot of games, it might not know the actual optimal but might know the current most optimal strategy
        iii. If policy is not currently optimal → action is not truly optimal
    c. Agent can learn bad models
        i. Find what works
        ii. Repeat what works
            1. We will never find better solutions if we are greedy
6. Exploration
    a. We want the agent to explore the world
        i. DUE / ADP / TD assume we will eventually see every trajectory possible
        ii. Following “optimal” actions may prevent this from happening
    b. Takeaway:
        i. We want the agent to (sometimes) ignore it’s policy
        ii. Explore the world to see new trajectories
            1. Improve policy with new knowledge
7. Exploration vs Exploitation
    a. Tradeoff:
        i. Exploitation: maximizing reward by following policy
            1. “exploit” already learned knowledge
        ii. Exploration: gather new data to improve the model by ignoring policy
            1. “explore” the trajectory space
    b. The longer the model runs:
        i. The less it should explore, and the more it should exploit
            1. Have to balance explore and exploit
            2. Brand new agent → explore a lot (see all the new things)
            3. As time progresses, it should start to exploit more and explore less since we want the agent to follow optimal paths after building knowledge

        ii. Greedy in the Limit of Infinite Exporation (GLIE):
            1. Must try each action in each state an unbounded number of times
            2. Eventually stop exploring and become greedy
    c. How to explore?
8. How to Explore
    a. Can choose an action at random
        i. Decide to choose a random action with prob 1/t (t = time) so that we choose the random action less often as time passes
        ii. Can take a while to converge
    b. Weight actions
        i. Weights for actions the agent hasn't tried often
        ii. Avoid actions (i.e. low weights) for actions believed to have small utility
        iii. Build this into the Bellman equation (which weights action utilities)
    c.
$$U(s) = R(s) + \gamma \max_a \sum_{s'} \Pr\left[s' \middle| s, a\right] U(s')$$

$$U^{+}(s) = R(s) + \gamma \max_a f\left(\sum_{s'} \Pr\left[s' \middle| s, a\right] U^{+}(s'),\ N(s, a) \right)$$

9. The Exploration Function f(u, n)
    a. Tradeoff between greed and curiosity
        i. Greed = preference for large values of 'u' → utility
        ii. Curiosity = preference for low values of 'n' → try actions we haven't tried often
    b. f should be increasing in 'u' and decreasing in 'n'
    c. One definition:
        i.
$$f(u, n) = \begin{cases} R^{+} & if\ n < N_e \\ u & otherwise \end{cases}$$
            1. R+ >= max (u)
            2. R+ = optimistic estimate of best possible reward in any state
            3. N_e = threshold for "seen this action-state pair enough times"

10. Learning Action-Utility Functions
    a. Active TD agent?
        i. Stop fixing policy
        ii. Passive TD agent learns utilities → need to learn model to choose actions
    b. Why not learn both utilities and model at the same time?
        i. Q-function Q(s,a) = “utility” of choosing action ‘a’ in state ‘s’
        ii. $U(s) = \max_a Q(s,a)$
        iii. Q-function takes the place of learned utilities and transition probs
11. Q-Function Q(s,a)
    a. Also obeys equilibrium constrains (similar to Bellman)

$$Q(s,a) = R(s) + \gamma \sum_{s'} \Pr\left[s' \middle| s, a\right] \max_{a'} Q(s',a')$$

    b. Given a model Pr[s’ | s, a], we can solve this directly
        i. problem: requires a model (avoid)
    c. TD approach requires no model
        i. Just nudge the Q values in the right direction

$$U^{\pi}(s) \leftarrow U^{\pi}(s) + \alpha\left(R(s) + \gamma U^{\pi}(s') - U^{\pi}(s)\right)$$

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left(R(s) + \gamma \max_{a'} Q(s',a') - Q(s,a)\right)$$

    d. Q-Function transitions reinforcement learning into supervised learning (doing supervised learning on the fly)
        i. Alpha is the learning rate (gradient descent)
        ii. The big parentheses is the derivative
        iii. Q(s’, a’) is the ground truth
        iv. Q(s, a) is my prediction