CS 188
Summer 2019

Section Handout 6

Q1. Pacman with Feature-Based Q-Learning

We would like to use a Q-learning agent for Pacman, but the size of the state space for a large grid is too massive to hold in memory. To solve this, we will switch to feature-based representation of Pacman's state.

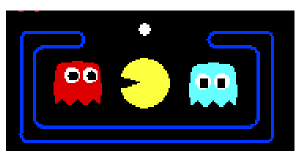
(a) We will have two features, F_g and F_p, defined as follows:

$$F_g(s,a) = A(s) + B(s,a) + C(s,a)$$
$$F_p(s,a) = D(s) + 2E(s,a)$$

where

$$\begin{aligned}
A(s) &= \text{number of ghosts within 1 step of state } s \\
B(s,a) &= \text{number of ghosts Pacman touches after taking action } a \text{ from state } s \\
C(s,a) &= \text{number of ghosts within 1 step of the state Pacman ends up in after taking action } a \\
D(s) &= \text{number of food pellets within 1 step of state } s \\
E(s,a) &= \text{number of food pellets eaten after taking action } a \text{ from state } s
\end{aligned}$$

For this pacman board, the ghosts will always be stationary, and the action space is $\{left, right, up, down, stay\}$.



calculate the features for the actions $\in \{left, right, up, stay\}$

(b) After a few episodes of Q-learning, the weights are $w_g = -10$ and $w_p = 100$. Calculate the Q value for each action $\in \{left, right, up, stay\}$ from the current state shown in the figure.

(c) We observe a transition that starts from the state above, s, takes action *up*, ends in state s' (the state with the food pellet above) and receives a reward $R(s, a, s') = 250$. The available actions from state s' are *down* and *stay*. Assuming a discount of $\gamma = 0.5$, calculate the new estimate of the Q value for s based on this episode.

(d) With this new estimate and a learning rate (α) of 0.5, update the weights for each feature.

2 Feature-Based Q-Learning

1. When using features to represent the Q-function is it guaranteed that the feature-based Q-learning finds the same optimal $Q*$ as would be found when using a tabular representation for the Q-function?

Q3. MDPs and RL

Recall that in approximate Q-learning, the Q-value is a weighted sum of features: $Q(s,a) = \sum_i w_i f_i(s,a)$. To derive a weight update equation, we first defined the loss function $L_2 = \frac{1}{2}(y - \sum_k w_k f_k(x))^2$ and found $dL_2/dw_m = -(y - \sum_k w_k f_k(x)) f_m(x)$. Our label y in this set up is $r + \gamma \max_a Q(s', a')$. Putting this all together, we derived the gradient descent update rule for w_m as $w_m \leftarrow w_m + \alpha \left(r + \gamma \max_a Q(s', a') - Q(s,a)\right) f_m(s,a)$.

In the following question, you will derive the gradient descent update rule for w_m using a different loss function:

$$L_1 = \left| y - \sum_k w_k f_k(x) \right|$$

(a) Find dL_1/dw_m. Ignore the non-differentiable point.

(b) Write the gradient descent update rule for w_m, using the L_1 loss function.

4 Probability

Use the probability table to calculate the following values:

X_1	X_2	X_3	$P(X_1, X_2, X_3)$
0	0	0	0.05
1	0	0	0.1
0	1	0	0.4
1	1	0	0.1
0	0	1	0.1
1	0	1	0.05
0	1	1	0.2
1	1	1	0.0

1. $P(X_1 = 1, X_2 = 0)$

2. $P(X_3 = 0)$

3. $P(X_2 = 1 | X_3 = 1)$

4. $P(X_1 = 0 | X_2 = 1, X_3 = 1)$

5. $P(X_1 = 0, X_2 = 1 | X_3 = 1)$