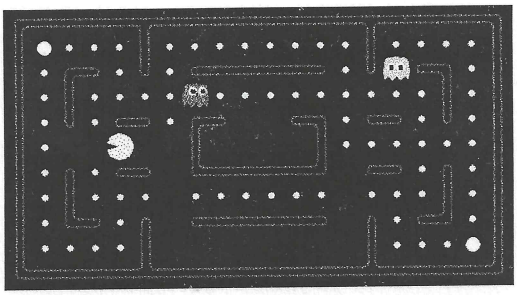
CSE 5525 Artificial Intelligence II

Quiz #4: RL with Feature-based Representations

Wei Xu, Ohio State University

Your Name: _____ OSU Username: _____

We would like to use a Q-learning agent for Pacman, but the state size 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. Here's a Pacman board:

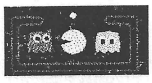


Questions:

1) What features would you extract from a Pacman board to judge the expected outcome of the game?

Lecture slides & project #2

2) Say our two minimal features are the number of ghosts within 1 step of Pacman (F_g) and the number of food pellets within 1 step of Pacman (F_p). What are the values for these two feature for this tiny pacman board below?



$F_g = 2$. $F_p = 1$

3) With Q Learning, we train off of a few episodes, so our weights begin to take on values. Right now $w_g = 100$ and $w_p = -10$. Calculate the Q value for the state above.

Q value will not depend on what action is taken, because the features we extract here do not depend on the action, but the state.

$Q(s,a) = w_g f_g + w_p f_p = 100 \times 2 + (-10) \times 1 = 190$

4) We receive an episode, so now we need to update our values. An episode consists of a start state s, an action a, an end state s', and a reward $R(s, a, s')$. The start state of the episode is the state above (where you already calculated the feature values and the expected Q value). The next state has feature values $F_g = 0$ and $F_p = 2$ and the reward is 50. Assuming a discount of 0.5, calculate the new estimate of the Q value for s based on this episode.

$Q_{new}(s,a) = R(s,a,s') + \gamma \cdot \max_{a'} Q(s', a')$

recall, in this case, Q value do not depend on the action, only the state.

$= 50 + 0.5 \times (100 \times 0 + (-10) \times 2)$

$= 40$

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

$w_g = w_g + \alpha \times (Q_{new}(s,a) - Q(s,a)) \times f_g(s,a) = 100 + 0.5 \times (40 - 190) \times 2 = -50$

$w_p = w_p + \alpha \times (Q_{new}(s,a) - Q(s,a)) \times f_p(s,a) = -10 + 0.5 \times (40 - 190) \times 1 = -85$

6) Good job on updating the weights. Now let's think about this entire process one step back. What values do we learn in this process (assuming features are defined)? When we have completed learning, how do we tell if Pacman does a good job?

We learn the feature weights.

To evaluate, we run the game multiple times and look at the win/lose outcomes and the rewards.

7) In some sense, we can think about this entire process, on a meta level, as an input we control that produces an output that we would like to maximize. If you have a magical function $F(input)$ that maps an input to an output you would like to maximize, what techniques (from math, CS, etc) can we use to search for the best inputs? Keep in mind that the magical function is a black box.

many different ways, such as:

① random search (changing values indiscriminately)

② hill climbing (going in the direction of maximum positive change)

8) Now say we can calculate the derivative of the magical function $F'(input)$, giving us a gradient or slope. What techniques can we use now?

Gradient descent, and many other techniques from optimization developed for such problems,

e.g. conjugate gradient

Quasi-newton

L-BFGS

etc.