Starcraft2 RL model

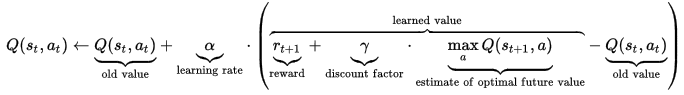
Terminology:

Reinforcement learning:

Adjust the model according to environment.

Q-learning:

Model free, off-policy. Has form (*s,a,r,s’*).



SARSA (State-Action-Reward-State-Action):

Model free, on-policy. Has form (st, αt, rt+1, st+1, αt+1).



Least-Squares Policy Iteration:

model-free, approximate policy iteration algorithm. Great success in continues or discrete states and discrete actions.

Environment: Abstracted low resolution grids of features

Rewards are based on the score from the starcraft engine

Several simplified mini-games are provided in addition to full game maps

Large datasets based on game play recorded from human players and can be used as benchmark for testing the model of interesting aspects of perception, memory and attention, sequence prediction and so on.

2 different structures: 1st ternary (1/0/-1) received at the end of the game, 2nd blizzard score.

Pysc2 has already provided many functional coding environment, all I need to do is to write the policy and reward function.

Available function:

Policy function:



Where π means policy, *a* means action, *s* means state. L means the length of action space depend on *a*0. This function could be used for policy to study the possibility of a sequence actions.

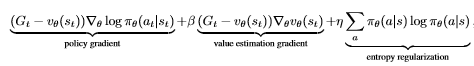
An alternative function can be Markov Decision Processes. (*S <state space>, A<action space>, P<transition model>, R<reward function>, y<discount factor>*)

Reward function:



This function could be used as a goal, where *y* is a discount factor, *rt* is the reward. The variables would be changed to maximise this goal.

Learning function:



Based on previous function, this function called Asynchronous Advantage Actor Critic (A3C) can be used to learning the variables.

State value function:



It is used for estimating future rewards. It is used in learning function.

Action value function:



It calculates the expected value for the current state and a particular action.

Which can derive optimal policy (Bellman equation):

