**Part 1: Policy-Based and Value-Based Algorithms**

**Chapter 2: REINFORCE:**

* Reinforce: during learning, actions that resulted in good outcome should become more probable and the one with bad outcomes should be less probable. If learning is successful, over the course of many iterations action probabilities produced by the policy shift to distribution that results in good performance in environment.
* Action probabilities are changed by following the policy gradient, there fore reinforce is known as policy gradient algo
* Algo need 3 components:

+ A parameterized policy

+ An objective to be maximized

+ A method for updating the policy parameter

**1/ Policy:**

* Policy pi is a function mapping states to action probabilities, which is used to sample an action a.
* In reinforce, an agent learns a policy and uses this to act in an environment.
* A good policy maximizes the cumulative discounted rewards.
* We can represent policy with DNN.

**2/ Objective function:**

* We defined the objective that is maximized by the agent in Reinforce algo.
* An objective can be understood as the agent’s goal such as winning a game or getting the highest score possible.

**3/ Policy Gradient:**

* Policy and Objective are important to derive policy gradient algorithms.

+ Policy provides a way for an agent to act

+ objective provides a target to maximize

* The policy gradient algorithms solves: Text

  Description automatically generated

+ To maximize the objective, we perform gradient ascent on the policy param theta

Text

Description automatically generated

* 2.5 means that the expected sum of the gradients of the log prob of the action a\_t multiplied y the corresponding returns.
* The policy gradient is the mechanism by which action probabilities produced by the policy are changed. If the return Rt > 0, the the probability of the action is increased. Conversely, the prob of action is decreased. Over many update, the policy will learn to produce action which result in high Rt

**4/ Monte Carlo Sampling:**

* The REINFORCE algorithm numerically estmates the policy gradient using Monte Carlo sampling.
* Monte Carlo sampling refers to any method that usese random sampling to generate data used to approximate a function.
* In essence, it is just “approximation with random sampling.”
* Monte Carlo Sampling can be used to estimate Pi
* Monte Carlo can be used to numerically estimate the policy gradient

**5/ Reinforce Algorithms + On-policy algorithm:**

Graphical user interface, text

Description automatically generated

* First initialize the learning rate alpha and construct a policy network pi with randomly initialized weights
* Then, iterate for multiple episode:

+ Use the policy betwork Pi to generate a trajectory for an episode.

* For each time step t in the trajectory, compute the return R
* Use R to estimate the policy gradient.
* Sum the policy graidents for all time
* The trajectory is discarded after each param update – it can’t be reused since the reinforce is an on-policy algorithm – meaning the parameter update equation depends on the current policy

a/ Improve Reinforce:

* REINFORCE algorithm estimate policy gradient using Monte Carlo sapling with a single trajectory and cause high variance. Why>

+ Actions have some randomness because they are sampled from a prob distribution

+ Starting state may vary per episode

+ The Env transition function maybe stochastic