

An Odyssey in Deep Reinforcement Learning

Lunjun Zhang
lunjun@cs.toronto.edu

University of Toronto Machine Intelligence Student Team

19 Nov 2018

Overview

① What is Reinforcement Learning

1. Background
2. Formulation

② Algorithms

1. Actor Critic
2. Deep Q Learning
3. Deep Deterministic Policy Gradient

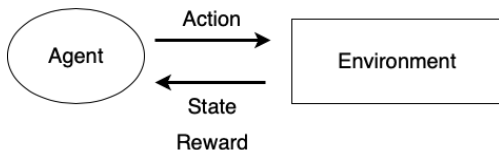
③ References

What is Reinforcement Learning

An agent maximizing the cumulative rewards from an environment

$$\mathbb{E} \left[\sum_t r \right]$$

Learning by interaction



Intuitive Examples

Problems that can be formulated as a RL problem:

- Manage an investment portfolio
 - + reward for earned money
- Win a computer game
 - + reward for increasing score
- Board games (Chess, Go)
 - + reward for winning a game
- Robotic Control
 - + reward for completing a task

Why interesting

An agent maximizing the cumulative rewards from an environment

- sequential decision making
- no supervision, only a reward signal
- current actions affect future states
- delayed feedback

RL is interesting because interesting RL algorithms exist.

Markov Decision Process

Markov Decision Process (MDP) $\langle S, A, P, R \rangle$

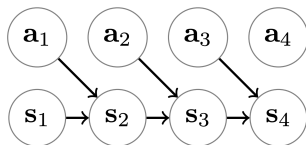
- S : the state space
 - perception: image (raw pixels), sensor readings ...
- A : the action space
 - discrete/continuous action space
- τ : a trajectory
 - a sequence of states and actions
 - $\tau_t = s_1, a_1, s_2, a_2, \dots, s_{t-1}, a_{t-1}, s_t, a_t$
- P : state transition dynamics $p(s_{t+1}|\tau_t)$
- $R: S \times A \rightarrow \mathbb{R}$

Markov Decision Process

Markov Decision Process (MDP) $\langle S, A, P, R \rangle$

- Assume that the environment is fully observable
- Assume that the process is Markov

$$p(s_{t+1}|\tau_t) = p(s_{t+1}|s_t, a_t)$$



Objective

Consider a trajectory τ that has T timesteps.

A distribution over all possible trajectories conditioned on our model θ

$$p_{\theta}(\tau) = p(s_1, a_1, \dots, s_T, a_T \mid \theta)$$

We aim to optimize

$$J = \mathbb{E}_{\tau \sim p_{\theta}(\tau)} \left[\sum_t r(s_t, a_t) \right]$$

Occupancy Measure

$$\pi(a | s, \theta)$$

- the probability of choosing action a in state s under model θ
- note that this is the policy we want to learn!

$$\mu(s | t, \theta)$$

- the probability of visiting state s at timestep t following a trajectory generated by model θ

We define an **occupancy measure**, ρ , as the distribution of state-action pairs that an agent encounters when navigating the environment with its model θ . It follows

$$\rho(s, a | \theta) = \pi(a | s, \theta) \sum_{t=1}^{\infty} \mu(s | t, \theta)$$

Bellman Equation

How do we calculate $\mu(s | T, \theta)$?

For concise notation, we denote $\mu(s | t, \theta)$ by $\mu_t(s | \theta)$.

Initial Condition:

$$\mu_1(s | \theta) = p(s_1 = s)$$

Bellman equation:

$$\mu_{t+1}(s | \theta) = \sum_a \sum_{s'} \mu_t(s' | \theta) \pi(a | s', \theta) p(s | s', a)$$

State-visitation distribution

State-visitation distribution

- the distribution of states that an agent encounters **throughout the trajectory** when navigating the environment with its model θ
- For a trajectory of T timesteps: $\mathbb{P}[s \mid \theta, T]$
- For an infinite horizon: $\mathbb{P}[s \mid \theta]$, or $\mathbb{P}_\theta[s]$

By definition,

$$\mathbb{P}[s \mid \theta, T] = \sum_{t=0}^T \mu(s \mid t, \theta)$$

Finite vs Infinite Horizon Objective

We aim to optimize

$$J = \mathbb{E}_{\tau \sim p_{\theta}(\tau)} \left[\sum_t r(s_t, a_t) \right]$$

Finite horizon:

$$J = \sum_{t=1}^T \mathbb{E}_{(s_t, a_t) \sim p_{\theta}(s_t, a_t)} [r(s_t, a_t)]$$

Infinite horizon:

$$\begin{aligned} J &= \mathbb{E}_{(s, a) \sim \rho_{\theta}(s, a)} [r(s, a)] \\ &= \mathbb{E}_{s \sim \mathbb{P}_{\theta}, a \sim \pi_{\theta}(\cdot | s)} [r(s, a)] \end{aligned}$$

Model-free RL

We will solve this very general problem using a very simple idea:

- Add randomness to your actions
- If the result is better than expected, do more of the same in the future

Policy Gradients:

- Let's just directly learn a policy $\pi(a | s, \theta)$ parameterized by θ
- We optimize J by doing gradient ascent: $\theta \leftarrow \theta + \alpha \nabla_{\theta} J$

Q-learning:

- Let's use $Q(s, a)$ to estimate how good an action a is under state s
- The policy is to take the best action: $a = \arg \max_{a'} Q(s, a')$

Policy Gradient

$$\pi_{\theta}(\tau) = p_{\theta}(s_1, a_1, \dots, s_T, a_T) = p(s_1) \prod_{t=1}^T \pi_{\theta}(a_t | s_t) p(s_{t+1} | s_t, a_t)$$

The goal is find optimal parameters $\theta^* = \arg \max_{\theta} \mathbb{E}_{\tau \sim p_{\theta}(\tau)} \left[\sum_t r(s_t, a_t) \right]$

We do so by gradient ascent:

$$\begin{aligned} \nabla_{\theta} J(\theta) &= \nabla_{\theta} \mathbb{E}_{\tau \sim \pi_{\theta}(\tau)} \left[\sum_t r(s_t, a_t) \right] \\ &= \int \nabla_{\theta} \pi_{\theta}(\tau) r(\tau) d\tau \\ &= \int \pi_{\theta}(\tau) \nabla_{\theta} \log \pi_{\theta}(\tau) r(\tau) d\tau \\ &= \mathbb{E}_{\tau \sim \pi_{\theta}(\tau)} \left[\nabla_{\theta} \log \pi_{\theta}(\tau) r(\tau) \right] \end{aligned}$$

Temporal Structure

$$\begin{aligned}\nabla_{\theta} J(\theta) &= \mathbb{E}_{\tau \sim \pi_{\theta}(\tau)} \left[\nabla_{\theta} \log \pi_{\theta}(\tau) r(\tau) \right] \\ &= \mathbb{E}_{\tau \sim \pi_{\theta}(\tau)} \left[\left(\sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \right) \left(\sum_{t=1}^T r(s_t, a_t) \right) \right]\end{aligned}$$

We would like to add some temporal structure to the expression.

If we repeat the same argument for a single reward at timestep t' :

$$\nabla_{\theta} \mathbb{E}_{\tau \sim \pi_{\theta}(\tau)} [r_{t'}] = \mathbb{E}_{\tau \sim \pi_{\theta}(\tau)} \left[r_{t'} \left(\sum_{t=1}^{t'} \nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \right) \right]$$

Sum this over t' ,

$$\nabla_{\theta} \mathbb{E}_{\tau \sim \pi_{\theta}(\tau)} \left[\sum_{t'=1}^T r_{t'} \right] = \mathbb{E}_{\tau \sim \pi_{\theta}(\tau)} \left[\sum_{t'=1}^T r_{t'} \left(\sum_{t=1}^{t'} \nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \right) \right]$$

Monte Carlo Sampling

$$\begin{aligned}\nabla_{\theta} \mathbb{E}_{\tau \sim \pi_{\theta}(\tau)} \left[\sum_{t'=1}^T r_{t'} \right] &= \mathbb{E}_{\tau \sim \pi_{\theta}(\tau)} \left[\sum_{t'=1}^T r_{t'} \left(\sum_{t=1}^{t'} \nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \right) \right] \\ &= \mathbb{E}_{\tau \sim \pi_{\theta}(\tau)} \left[\left(\sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \right) \left(\sum_{t'=t}^T r(s_{t'}, a_{t'}) \right) \right]\end{aligned}$$

We can interpret this as causality: current action only affects the future.

Using Monte Carlo Sampling,

$$\nabla_{\theta} J(\theta) \approx \frac{1}{N} \sum_{i=1}^N \left(\sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t^i | s_t^i) \left(\sum_{t'=t}^T r(s_{t'}^i, a_{t'}^i) \right) \right)$$

Actor Critic

The idea is to reduce the variance of policy gradients.

Q function: the *expected* reward-to-go by taking action a_t

$$Q^\pi(s_t, a_t) = \sum_{t'=t}^T \mathbb{E}_{\pi_\theta} [r(s_{t'}, a_{t'}) | s_t, a_t]$$

Value Function: how good a state is

$$V^\pi(s_t) = \mathbb{E}_{a_t \sim \pi_\theta(a_t | s_t)} [Q^\pi(s_t, a_t)]$$

Advantage: how good an action is compared to average

$$A^\pi(s_t, a_t) = Q^\pi(s_t, a_t) - V^\pi(s_t)$$

Replace $(\sum_{t'=t}^T r(s_{t'}, a_{t'}))$ by $A^\pi(s_t, a_t)$.

Actor Critic

The trick is to realize

$$\begin{aligned} Q^\pi(s_t, a_t) &= r(s_t, a_t) + \mathbb{E}_{s_{t+1} \sim p(s_{t+1}|s_t, a_t)} [V^\pi(s_{t+1})] \\ &= r(s_t, a_t) + V^\pi(s_{t+1}) \end{aligned}$$

Thus the advantage function becomes

$$A^\pi(s_t, a_t) = r(s_t, a_t) + V^\pi(s_{t+1}) - V^\pi(s_t)$$

Now we only need to fit the Value Function, or the Critic.

Regression $\{s_{i,t}, \sum_{t'=t}^T r(s_{i,t'}, a_{i,t'})\}$ on learning the critic $V^\pi(s_t)$.

Discount Factor

Consider an infinite horizon $T \rightarrow \infty$,

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{\tau \sim \pi_{\theta}(\tau)} \left[\left(\sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \left(\sum_{t'=t}^T r(s_{t'}, a_{t'}) \right) \right) \right]$$

The reward-to-go can approach ∞ !

We introduce a discount factor $\gamma \in (0, 1)$ on future rewards

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{\tau \sim \pi_{\theta}(\tau)} \left[\left(\sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \left(\sum_{t'=t}^T \gamma^{t'-t} r(s_{t'}, a_{t'}) \right) \right) \right]$$

It follows that

$$\hat{V}^{\pi}(s_t) = r(s_t, a_t) + \gamma \hat{V}^{\pi}(s_{t+1})$$

TD Residual

After introducing the discount factor γ ,

$$V^{\pi, \gamma}(s_t) = \mathbb{E}_{s_{t+1:\infty}, a_{t:\infty}} \left[\sum_{l=0}^{\infty} \gamma^l r_{t+l} \right]$$

$$Q^{\pi, \gamma}(s_t, a_t) = \mathbb{E}_{s_{t+1:\infty}, a_{t+1:\infty}} \left[\sum_{l=0}^{\infty} \gamma^l r_{t+l} \right]$$

$$A^{\pi, \gamma}(s_t, a_t) = Q^{\pi, \gamma}(s_t, a_t) - V^{\pi, \gamma}(s_t)$$

It follows that the advantage $A^{\pi, \gamma}(s_t, a_t)$ can be rewritten as

$$A^{\pi, \gamma}(s_t, a_t) = r(s_t, a_t) + \gamma V^{\pi, \gamma}(s_{t+1}) - V^{\pi, \gamma}(s_t)$$

Now we define the Temporal Difference Residual (Sutton & Barto, 1998):

$$\delta_t^V = r(s_t, a_t) + \gamma V(s_{t+1}) - V(s_t)$$

Advantage Estimator

We want an advantage estimator, \hat{A} , to estimate the true advantage function $A^{\pi, \gamma}$, that tells us how good an action is compared to average.

Naive approach: just use TD Residual δ_t^V as \hat{A} !

This works, but it will introduce *bias* since V will NOT be exactly $V^{\pi, \gamma}$.

How do we reduce the bias in the TD Residual?

Baseline

$$\nabla_{\theta} J(\theta) = \mathbb{E}_{\tau \sim \pi_{\theta}(\tau)} \left[\left(\sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \left(\sum_{t'=t}^T r(s_{t'}, a_{t'}) - b(s_{t'}) \right) \right) \right]$$

For an arbitrary function b , the gradient estimator is unbiased.

Proof

$$\begin{aligned} & \mathbb{E}_{\tau \sim \pi_{\theta}(\tau)} \left[\nabla_{\theta} \log \pi_{\theta}(a_t | s_t) b(s_t) \right] \\ &= \mathbb{E}_{s_{0:t}, a_{0:t-1}} \left[\mathbb{E}_{s_{t+1:T}, a_{t:T-1}} \left[\nabla_{\theta} \log \pi_{\theta}(a_t | s_t) b(s_t) \right] \right] \\ &= \mathbb{E}_{s_{0:t}, a_{0:t-1}} \left[b(s_t) \mathbb{E}_{s_{t+1:T}, a_{t:T-1}} \left[\nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \right] \right] \\ &= \mathbb{E}_{s_{0:t}, a_{0:t-1}} \left[b(s_t) \mathbb{E}_{a_t \sim \pi_{\theta}(\cdot | s_t)} \left[\nabla_{\theta} \log \pi_{\theta}(a_t | s_t) \right] \right] \\ &= \mathbb{E}_{s_{0:t}, a_{0:t-1}} \left[b(s_t) \cdot \nabla_{\theta} 1 \right] = \mathbb{E}_{s_{0:t}, a_{0:t-1}} \left[b(s_t) \cdot 0 \right] = 0 \end{aligned}$$

Generalized Advantage Estimation

We proved: $V(s_t)$ does not matter to the bias in the TD Residual:

$$\delta_t^V = r(s_t, a_t) + \gamma V(s_{t+1}) - V(s_t)$$

We want a way to estimate the discounted future return in a less biased way. On the other hand, $\sum_{l=0}^{\infty} \gamma^l r_{t+l}$ has too much variance.

GAE (Schulman et al., 2016):

$$\hat{A}_t^{GAE(\gamma, \lambda)} := \sum_{l=0}^{\infty} (\gamma \lambda)^l \delta_{t+l}^V$$

somewhere in between:

- $GAE(\gamma, 0) : \hat{A}_t = \delta_t$
- $GAE(\gamma, 1) : \hat{A}_t = \sum_{l=0}^{\infty} \gamma^l r_{t+l} - V(s_t)$.

Actor Critic Summary

Five steps

- sample (s_i, a_i) from $\pi_\theta(a | s)$
- fit $V^\pi(s)$ to sampled reward sums
- calculate $\hat{A}(s_i, a_i)$
- $\nabla_\theta J(\theta) \approx \sum_i \nabla_\theta \log \pi_\theta(a_i | s_i) \hat{A}^\pi(s_i, a_i)$
- $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

Q learning

$Q(s,a)$: calculates the expected return by taking action a in state s .

Three steps

- collect (s_i, a, s'_i, r_i) using *some* policy
- $y_i \leftarrow r(s_i, a_i) + \gamma \max_{a'_i} Q_\phi(s'_i, a'_i)$
- $\phi \leftarrow \arg \min_\phi \frac{1}{2} \sum_i \|Q_\phi(s_i, a_i) - y_i\|^2$

Q Learning

Policy:

$$\pi(a_t | s_t) = \begin{cases} 1, & \text{if } a_t = \arg \max_{a_t} Q_\phi(s_t, a_t) \\ 0, & \text{otherwise} \end{cases}$$

The algorithm is *off-policy*: given s and a , transition is independent of π .

Q-iteration:

$$\phi \leftarrow \phi - \alpha \frac{dQ_\phi}{d\phi}(s_i, a_i) (Q_\phi(s_i, a_i) - y_i)$$

Epsilon-greedy

We here assume discrete action space.

$$\pi(a_t | s_t) = \begin{cases} 1, & \text{if } a_t = \arg \max_{a_t} Q_\phi(s_t, a_t) \\ 0, & \text{otherwise} \end{cases}$$

This is a very bad idea at the beginning of training

- the Q value has no idea on what it is doing

We need some *exploration strategies*.

$$\pi(a_t | s_t) = \begin{cases} 1 - \epsilon, & \text{if } a_t = \arg \max_{a_t} Q_\phi(s_t, a_t) \\ \epsilon / (|A| - 1), & \text{otherwise} \end{cases}$$

This is called epsilon greedy strategy.

Target Network

$$\phi \leftarrow \phi - \alpha \frac{dQ_\phi}{d\phi}(s_i, a_i) \left[Q_\phi(s_i, a_i) - (r(s_i, a_i) + \gamma \max_{a'} Q_\phi(s'_i, a'_i)) \right]$$

Problem: this is NOT a regression throughout training.

- ϕ keeps changing; learning is extremely unstable

Solution: make it a regression.

- introduce target network ϕ'
- save target network parameters $\phi' \leftarrow \phi$ at the beginning of each loop
- inner loop regression:

$$\phi \leftarrow \phi - \alpha \frac{dQ_\phi}{d\phi}(s_i, a_i) \left[Q_\phi(s_i, a_i) - (r(s_i, a_i) + \gamma \max_{a'} Q_{\phi'}(s'_i, a'_i)) \right]$$

Replay Buffer

Problem: sequential states are strongly correlated.

Solution: save all transitions into a large *replay buffer*

- then sample from the replay buffer to do Q iteration updates

Q-learning with a replay buffer:

- sample a batch (s_i, a_i, s'_i, r_i) from replay buffer B
- regression:

$$\phi \leftarrow \phi - \alpha \sum_i \frac{dQ_\phi}{d\phi}(s_i, a_i) \left[Q_\phi(s_i, a_i) - (r(s_i, a_i) + \gamma \max_{a'} Q_{\phi'}(s'_i, a'_i)) \right]$$

One more trick, Polyak averaging: $\phi' \leftarrow \tau \phi' + (1 - \tau) \phi$

Statistical Perspective

Experience replay: statistical perspective

- break temporal correlations by mixing more and less recent experience
- stabilize the training of a value function
- sample from a large sliding window replay memory uniformly at random

Neuroscience Perspective

Experience replay: neuroscience perspective

- evidence of experience replay in the **hippocampus** of rodents
 - sequences of prior experience are replayed, either during awake resting or sleep
- in fact, sequences associated with rewards and high TD error appear to be replayed more often
 - Atherton et al., 2015; McNamara et al., 2014
- hippocampus plays important roles in
 - the consolidation of information from short-term memory to long-term memory
 - spatial memory that enables navigation

Overestimation of Q Values

Q values tend to overestimate the return. Why?

$$y_i \leftarrow r(s_i, a_i) + \gamma \max_{a'_i} Q_\phi(s'_i, a'_i)$$

We can rewrite the last term as

$$\max_{a'} Q_\phi(s', a') = Q_\phi(s', \arg \max_{a'} Q_\phi(s', a'))$$

Fact:

$$\mathbb{E}[\max(X_1, X_2)] \geq \max(\mathbb{E}[X_1], \mathbb{E}[X_2])$$

Since Q is not perfect, it *overestimates* the next value!

Double Q Learning

Problem:

- noise in Q values lead to overestimation

Solution:

- de-correlate the noise by using different networks to choose the action and evaluate the value
 - we actually already have two networks (the target network)

$$Q_{\phi_A}(s, a) \leftarrow r + \gamma Q_{\phi_B}(s' \arg \max_{a'_i} Q_{\phi_A}(s'))$$

$$Q_{\phi_B}(s, a) \leftarrow r + \gamma Q_{\phi_A}(s' \arg \max_{a'_i} Q_{\phi_B}(s'))$$

Q Learning with Continuous Actions

How to do Q Learning with continuous actions?

$$\pi(a_t | s_t) = \begin{cases} 1, & \text{if } a_t = \arg \max_{a_t} Q_\phi(s_t, a_t) \\ 0, & \text{otherwise} \end{cases}$$

How do we take the $\arg \max$?

$$y_i \leftarrow r(s_i, a_i) + \gamma \max_{a'_i} Q_\phi(s'_i, a'_i)$$

How do we take the \max ?

An Approximate Maximizer

Train a neural network to take the $\arg \max$!

$$\mu_{\theta}(s) \approx \arg \max_a Q_{\phi}(s, a)$$

Solve:

$$\theta \leftarrow \arg \max_{\theta} Q_{\phi}(s, \mu_{\theta}(s))$$

Using back-propagation:

$$\frac{dQ_{\phi}}{d\theta} = \frac{da}{d\theta} \frac{dQ_{\phi}}{da}$$

Deep Deterministic Policy Gradient

- Lillicrap & Hunt, 2016

$$y_i = r_i + \gamma \arg \max_{\theta} Q_{\phi}(s, \mu_{\theta}(s))$$

$$\phi \leftarrow \phi - \alpha \sum_i \frac{dQ_{\phi}}{d\phi}(s_i, a_i) (Q_{\phi}(s_i, a_i) - y_i)$$

$$\theta \leftarrow \theta + \beta \sum_i \frac{d\mu}{d\theta}(s_i) \frac{dQ_{\phi}}{da}(s_i, a)$$

References

- Atherton, Dupret, and Mellor. [Memory trace replay: the shaping of memory consolidation by neuromodulation](#). Trends in neurosciences, 38(9): 560–570, 2015.
- Levine. [Deep Reinforcement Learning](#), UC Berkeley. 2017.
- Lillicrap, Hunt, Pritzel, Heess, Erez, Tassa, Silver, Wierstra. [Continuous Control with Deep Reinforcement Learning](#). In ICLR, 2016.
- Lin. [Self-improving reactive agents based on reinforcement learning, planning and teaching](#). Machine learning, 8(3-4): 293–321, 1992.

References

- McNamara, Tejero-Cantero, Trouche, Campo-Urriza, Dupret. [Dopaminergic neurons promote hippocampal reactivation and spatial memory persistence](#). Nature neuroscience, 2014.
- Mnih, Kavukcuoglu, Silver, Rusu, Veness, Bellemare, Graves, Riedmiller, Fidjeland, Ostrovski, Petersen, Beattie, Sadik, Antonoglou, King, Kumaran, Wierstra, Legg and Hassabis. [Human-level control through deep reinforcement learning](#). Nature, 518(7540):529–533, 2015.

References

- Silver. [Reinforcement Learning](#), UCL. 2015.
- Silver, Lever, Heess, Degris, Wierstra, Riedmiller. [Deterministic Policy Gradient Algorithms](#). In ICML, 2014.
- Schulman, Moritz, Levine, Jordan, Abbeel. [High-dimensional Continuous Control Using Generalized Advantage Estimation](#). In ICLR, 2016.
- Sutton and Barto. [Introduction to Reinforcement Learning](#). In MIT Press, 1998.