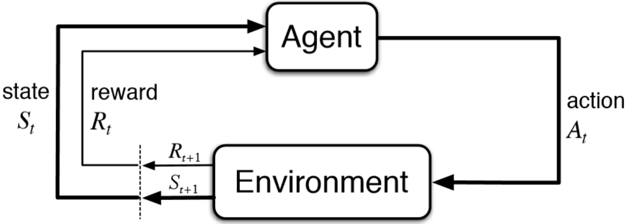
Reinforcement Learning via Policy Optimization

Hanxiao Liu

November 22, 2017

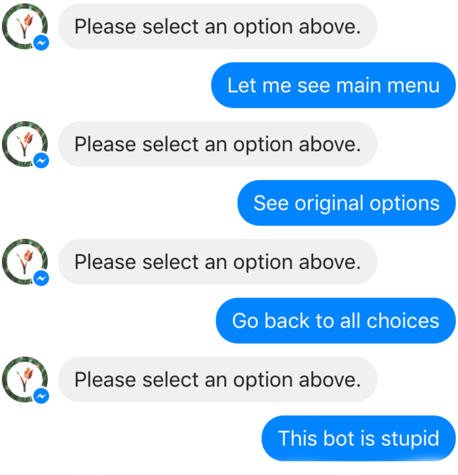
Reinforcement Learning



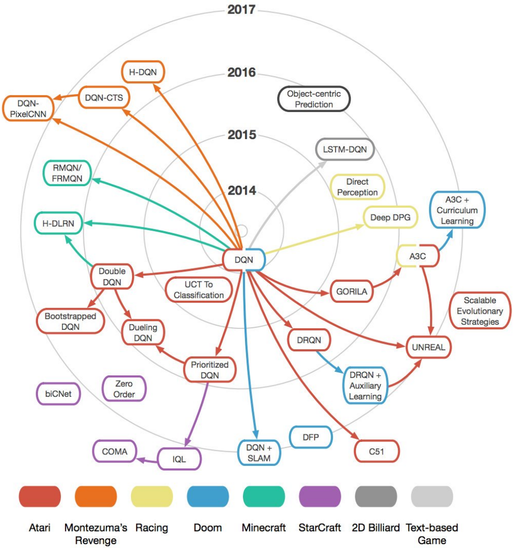
Policy $a \sim \pi(s)$

Example - Mario

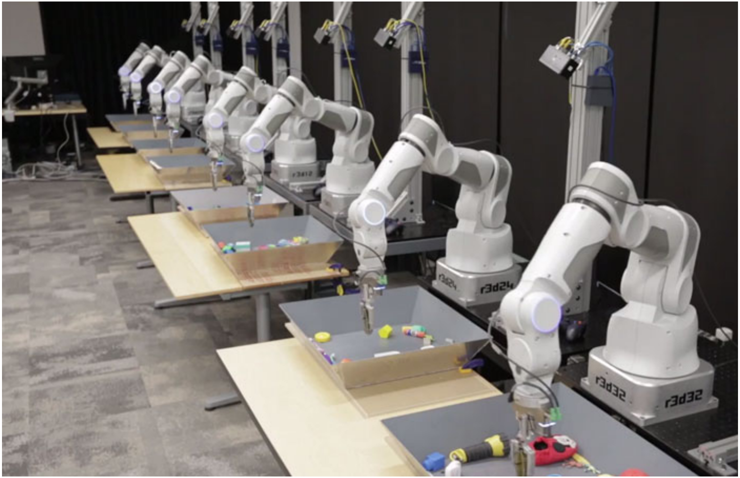
Example - ChatBot



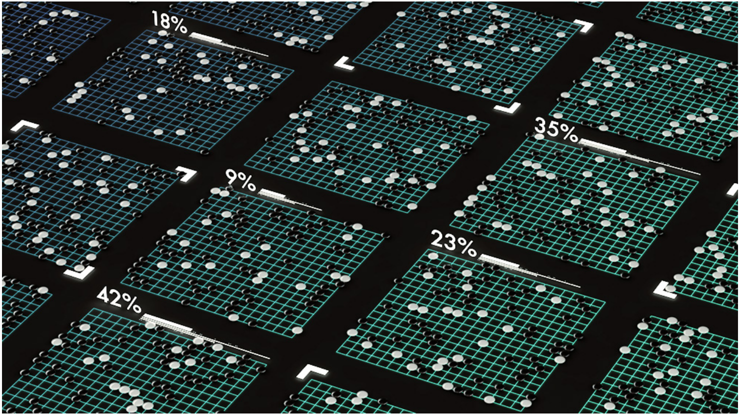
Applications - Video Games



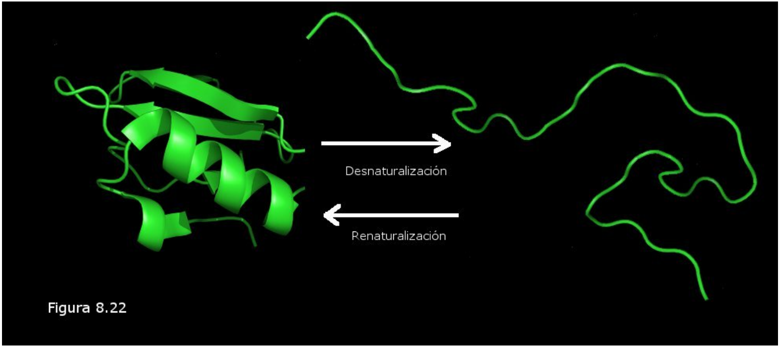
Applications - Robotics



Applications - Combinatorial Problems



Applications - Bioinformatics



Supervised Learning vs RL

Supervised setup

- $s_t \sim P(\cdot)$
- $a_t = \pi(s_t)$
- immediate reward

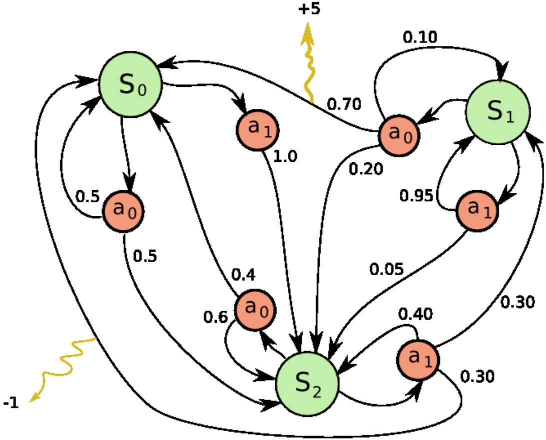
Reinforcement setup

- $s_t \sim P(\cdot | s_{t-1}, a_{t-1})$
- $a_t \sim \pi(s_t)$
- delayed reward

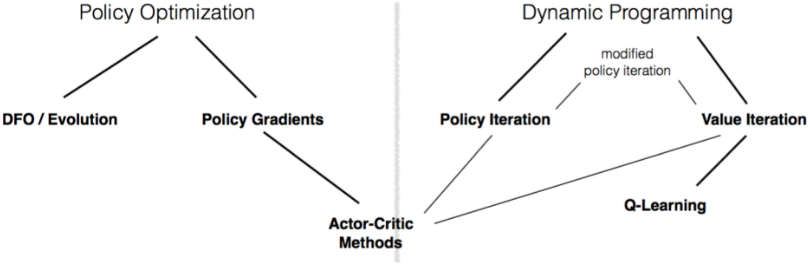
Both aims to maximize the total reward.

Markov Decision Process

What is the underlying assumption?



Landscape


Policy Optimization
Dynamic Programming
modified
policy iteration
DFO / Evolution
Policy Gradients
Policy Iteration
Value Iteration
Q-Learning
Actor-Critic
Methods

Monte Carlo

- Return $R(\tau) = \sum_{t=1}^{T} R(s_t, a_t)$.
- Trajectory $\tau = s_0, a_0, \ldots, s_T, a_T$

Goal: finding a good π such that R is maximized.

Policy evaluation via MC

1. Sample a trajectory τ given π.
2. Record $R(\tau)$

Repeat the above for many times and take the average.

Policy Gradient

Let's parametrize π using a neural network.

- $a \sim \pi_\theta(s)$

Expected return

$$\max_\theta \; U(\theta) = \max_\theta \; \sum_\tau P(\tau; \pi_\theta) R(\tau) \tag{1}$$

Heuristic: Raise the probability of good trajectories.

1. Sample a trajectory τ under π_θ
2. Update θ using gradient $\nabla_\theta \log P(\tau; \pi_\theta) R(\tau)$.

Policy Gradient

The log-derivative trick

$$
\begin{aligned}
\nabla_\theta U(\theta) &= \sum_\tau \nabla_\theta P(\tau;\pi_\theta)R(\tau) && (2)\\
&= \sum_\tau \frac{P(\tau;\pi_\theta)}{P(\tau;\pi_\theta)}\nabla_\theta P(\tau;\pi_\theta)R(\tau) && (3)\\
&= \sum_\tau P(\tau;\pi_\theta)\nabla_\theta \log P(\tau;\pi_\theta)R(\tau) && (4)\\
&= \mathbb{E}_{\tau\sim P(\cdot;\pi_\theta)}\nabla_\theta \log P(\tau;\pi_\theta)R(\tau) && (5)\\
&\approx \nabla_\theta \log P(\tau;\pi_\theta)R(\tau) \quad \tau \sim P(\cdot;\pi_\theta) && (6)
\end{aligned}
$$

Policy Gradient

$$\nabla_\theta U(\theta) \approx \nabla_\theta \log P(\tau; \pi_\theta) R(\tau) \quad \tau \sim P(\cdot; \pi_\theta) \tag{7}$$

- Analogous to SGD (so variance reduction is important), but data distribution here is a moving target (so we may want a trust region).

Policy Gradient

We can subtract any constant from the reward

$$\begin{aligned}
&\nabla_\theta \sum_\tau P(\tau;\pi_\theta)(R(\tau) - b) && (8)\\
=&\nabla_\theta \sum_\tau P(\tau;\pi_\theta)R(\tau) - \nabla_\theta \sum_\tau P(\tau;\pi_\theta)b && (9)\\
=&\nabla_\theta \sum_\tau P(\tau;\pi_\theta)R(\tau) - \nabla_\theta b && (10)\\
=&\nabla_\theta \sum_\tau P(\tau;\pi_\theta)R(\tau) && (11)
\end{aligned}$$

Policy Gradient

The variance is magnified by $R(\tau)$

$$\nabla_\theta U(\theta) \approx \nabla_\theta \log P(\tau; \pi_\theta) R(\tau) \tag{12}$$

More fine-grained baseline?

$$\sum_{t=0} \nabla_\theta \log \pi_\theta(a_t|s_t) \sum_{t'=0} R(s_{t'}, a_{t'}) \tag{13}$$

$$\to \sum_{t=0} \nabla_\theta \log \pi_\theta(a_t|s_t) \underbrace{\sum_{t'=t} R(s_{t'}, a_{t'})}_{R_t} \tag{14}$$

$$\to \sum_{t=0} \nabla_\theta \log \pi_\theta(a_t|s_t) \left(R_t - b_t\right) \tag{15}$$

Policy Gradient

$$\sum_{t=0} \nabla_\theta \log \pi_\theta(a_t|s_t)\,(R_t - b_t) \tag{16}$$

Good choice of b_t?

$$\begin{aligned} b_t^* &= \operatorname{argmin}_b \; \mathbb{E}\,(R_t - b)^2 & (17)\\ &\approx V_\phi(s_t) & (18) \end{aligned}$$

$$\sum_{t=0} \nabla_\theta \log \pi_\theta(a_t|s_t)\,\underbrace{(R_t - V_\phi(s_t))}_{A_t} \tag{19}$$

- Promote actions that lead to positive advantage.

Policy Gradient

$$\sum_{t=0} \nabla_\theta \log \pi_\theta(a_t|s_t) A_t \qquad (20)$$

- Sample a trajectory τ
- For each step in τ
  - $R_t = \sum_{t'=t} r_{t'}$
  - $A_t = R_t - V_\phi(s_t)$
- take a gradient step $\nabla_\phi \sum_{t=0} \|V_\phi(s_t) - R_t\|^2$
- take a gradient step $\nabla_\theta \sum_{t=0} \log \pi_\theta(a_t|s_t) A_t$

TRPO

In PG, our opt objective a moving target defined based on trajectory samples given the current policy

- each sample is only used once

An opt objective that reuses all historical data?

TRPO

Policy gradient

$$\sum_{t=0} \nabla_\theta \log \pi_\theta(a_t|s_t) A_t \tag{21}$$

Objective (on-policy)

$$\mathbb{E}_\pi \left[A(s, a) \right] \tag{22}$$

Objective (off-policy), via importance sampling

$$\mathbb{E}_{\pi_{old}} \left[\frac{\pi(a|s)}{\pi_{old}(a|s)} A_{old}(s, a) \right] \tag{23}$$

TRPO

$$\max_{\pi} \ \mathbb{E}_{\pi_{old}}\left[\frac{\pi(a|s)}{\pi_{old}(a|s)}A_{old}(s,a)\right] \approx \sum_{n=1}^{N}\frac{\pi(a_n|s_n)}{\pi_{old}(a_n|s_n)}A_n \quad (24)$$

$$s.t. \ \mathrm{KL}(\pi_{old}, \pi) \leq \delta \quad (25)$$

- Quadratic approximation is used for the KL.
- Use conjugate gradient for the natural gradient direction $F^{-1}g$.

PPO

$$\max_{\pi} \sum_{n=1}^{N} \frac{\pi(a_n|s_n)}{\pi_{old}(a_n|s_n)} A_n - \beta \mathrm{KL}(\pi_{old}, \pi) \tag{26}$$

Fixed β does not work well

- shrink β when $\mathrm{KL}(\pi_{old}, \pi)$ is small
- increase β otherwise

PPO

Alternative ways to penalize large change in π? Modify

$$\frac{\pi(a_n|s_n)}{\pi_{old}(a_n|s_n)} A_n = r_n(\theta) A_n \tag{27}$$

As

$$\min\left(r_n(\theta)A_n, \mathrm{clip}(1-\epsilon, 1+\epsilon, r_n(\theta))A_n\right) \tag{28}$$

- being pessimistic whenever "a large change in π leads to a better obj."
- same as the original obj otherwise.

Advantage Estimation

Currently

$$\hat{A}_t = r_t + r_{t+1} + r_{t+2} + \cdots - V(s_t) \tag{29}$$

More bias, less variance (ignoring long-term effect)

$$\hat{A}_t = r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots - V(s_t) \tag{30}$$

Bootstrapping

$$\begin{aligned}
\hat{A}_t &= r_t + \gamma r_{t+1} + \gamma^2 r_{t+2} + \cdots - V(s_t) && (31)\\
&= r_t + \gamma(r_{t+1} + \gamma r_{t+2} + \dots) - V(s_t) && (32)\\
&= r_t + \gamma V(s_{t+1}) - V(s_t) && (33)
\end{aligned}$$

We may unroll for more than one steps.

A2C

Advantage Actor-Critic

- Sample a trajectory τ
- For each step in τ
  - $\hat{R}_t = r_t + \gamma V(s_{t+1})$
  - $\hat{A}_t = \hat{R}_t - V(s_t)$
- take a gradient step using $\nabla_{\theta,\phi} \sum_{t=0} \left[-\log \pi_\theta(a_t|s_t)\hat{A}_t + \|V_\phi(s_t) - \hat{R}_t\|^2 \right]$

May use TROP, PPO for policy optimization.

Variance reduction via multiple actors

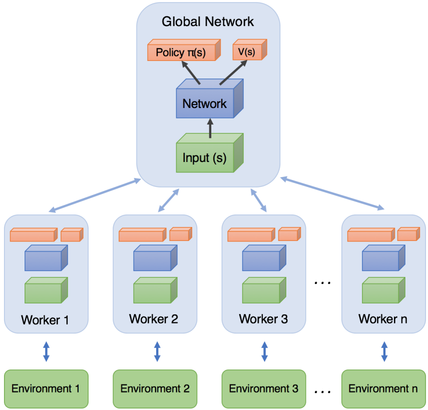


Figure : Asynchronous Advantage Actor-Critic