A3C and PPO

Spring, 2024

Outline

1. Policy Gradient Review
2. Actor-Critic Policy Gradient
3. From ON-Policy to OFF-Policy
4. PPO/TRPO

Table of Contents

Policy Gradient

Different Schools of Reinforcement Learning

1. Value-based RL: solve RL through dynamic programming
   1. Classic RL and control theory
   2. Representative algorithms: Deep Q-learning and its variant
   3. Representative researchers: Richard Sutton (no more than 20 pages on PG out of the 500-page textbook), David Silver, from DeepMind
2. Policy-based RL: solve RL mainly through learning
   1. Machine learning and deep learning
   2. Representative algorithms: PG, and its variants TRPO, PPO, and others
   3. Representative researchers: Pieter Abbeel, Sergey Levine, John Schulman, from OpenAI, Berkelely

Policy Gradient

1. State-of-the-art RL methods are almost all policy-based
   1. **A2C and A3C**: Asynchronous Methods for Deep Reinforcement Learning, ICML'16. Representative high-performance actor-critic algorithm: `https://openai.com/blog/baselines-acktr-a2c/`
   2. **TRPO**: Schulman, L., Moritz, Jordan, Abbeel (2015). Trust region policy optimization: deep RL with natural policy gradient and adaptive step size
   3. **PPO**: Schulman, Wolski, Dhariwal, Radford, Klimov (2017). Proximal policy optimization algorithms: deep RL with importance sampled policy gradient

Policy Gradient

Review – Policy Gradient

baseline

$$\nabla \bar{R}_\theta \approx \frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T_n}\Big(\underbrace{\sum_{t'=t}^{T_n}\gamma^{t'-t}r_{t'}^n}_{G_t^n} - \underline{b}\Big)\nabla log p_\theta(a_t^n|s_t^n)$$

G_t^n : obtained via interaction

Policy Gradient

Review – Policy Gradient

baseline

$$\nabla \bar{R}_\theta \approx \frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T_n}\Big(\underbrace{\sum_{t'=t}^{T_n}\gamma^{t'-t}r_{t'}^n}_{G_t^n} - \underline{b}\Big)\nabla log p_\theta(a_t^n|s_t^n)$$

G_t^n : obtained via interaction

Very unstable

Policy Gradient

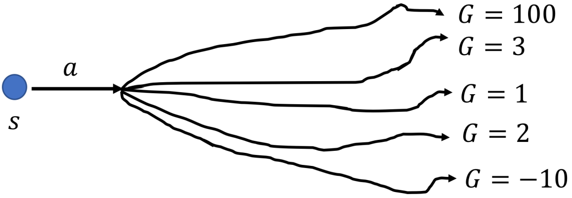
Review – Policy Gradient

baseline

$$\nabla \bar{R}_\theta \approx \frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T_n}\left(\underline{\sum_{t'=t}^{T_n}\gamma^{t'-t}r_{t'}^{n}} - \underline{b}\right)\nabla \log p_\theta(a_t^n|s_t^n)$$

G_t^n : obtained via interaction

Very unstable



Policy Gradient

Review – Policy Gradient

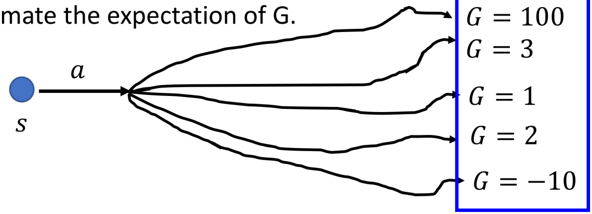
baseline

$$\nabla \bar{R}_\theta \approx \frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T_n}\left(\underline{\sum_{t'=t}^{T_n}\gamma^{t'-t} r_{t'}^n} - b\right)\nabla log p_\theta(a_t^n|s_t^n)$$

G_t^n : obtained via interaction

Very unstable

With sufficient samples,
approximate the expectation of G.



Policy Gradient

Review – Policy Gradient

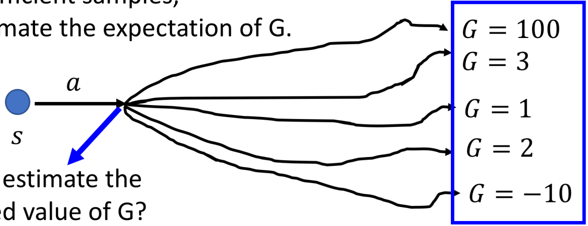
baseline

$$\nabla \bar{R}_\theta \approx \frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T_n}\left(\sum_{t'=t}^{T_n}\gamma^{t'-t}r_{t'}^{n} - b\right)\nabla log p_\theta(a_t^n|s_t^n)$$

G_t^n : obtained via interaction

Very unstable

With sufficient samples,
approximate the expectation of G.



Can we estimate the
expected value of G?

Table of Contents

Actor-Critic

Actor-Critic

$$\nabla \bar{R}_\theta \approx \frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T_n}\Big(\underbrace{\sum_{t'=t}^{T_n}\gamma^{t'-t}r_{t'}^n}_{G_t^n} - \overset{\text{baseline}}{\underline{b}}\Big)\nabla log p_\theta(a_t^n|s_t^n)$$

G_t^n : obtained via interaction

$$E[G_t^n] = Q^{\pi_\theta}(s_t^n, a_t^n)$$

Actor-Critic

Actor-Critic

$V^{\pi_\theta}(s_t^n)$

baseline

$$\nabla \bar{R}_\theta \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} \Big(\underbrace{\sum\nolimits_{t'=t}^{T_n} \gamma^{t'-t} r_{t'}^n}_{G_t^n} - \underline{b} \Big) \nabla log p_\theta (a_t^n | s_t^n)$$

G_t^n : obtained via interaction

$$E[G_t^n] = Q^{\pi_\theta}(s_t^n, a_t^n)$$

Actor-Critic

Actor-Critic

$$\boxed{Q^{\pi_\theta}(s_t^n, a_t^n) - V^{\pi_\theta}(s_t^n)}$$

$$V^{\pi_\theta}(s_t^n)$$

baseline

$$\nabla \bar{R}_\theta \approx \frac{1}{N}\sum_{n=1}^{N}\sum_{t=1}^{T_n}\boxed{\left(\underbrace{\sum_{t'=t}^{T_n}\gamma^{t'-t} r_{t'}^n}_{G_t^n} - b\right)}\nabla log p_\theta(a_t^n|s_t^n)$$

G_t^n : obtained via interaction

$$E[G_t^n] = Q^{\pi_\theta}(s_t^n, a_t^n)$$

Actor-Critic

Advantage Actor-Critic (A2C)

$$\boxed{Q^\pi(s_t^n, a_t^n) - V^\pi(s_t^n)}$$

Estimate two networks? We can only estimate one.

Advantage Function $A^\theta(s_t, a_t)$

How good it is if we take a_t other than other actions at s_t.

Actor-Critic

Advantage Actor-Critic (A2C)

$$\boxed{Q^{\pi}(s_t^n, a_t^n) - V^{\pi}(s_t^n)}$$

Estimate two networks? We can only estimate one.

$$Q^{\pi}(s_t^n, a_t^n) = E[r_t^n + V^{\pi}(s_{t+1}^n)]$$

$$Q^{\pi}(s_t^n, a_t^n) = r_t^n + V^{\pi}(s_{t+1}^n)$$

Actor-Critic

Advantage Actor-Critic (A2C)

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

$$\Downarrow$$

$$r_t^n + V^\pi(s_{t+1}^n) - V^\pi(s_t^n)$$

Estimate two networks? We can only estimate one.

$$Q^\pi(s_t^n, a_t^n) = E[r_t^n + V^\pi(s_{t+1}^n)]$$

$$Q^\pi(s_t^n, a_t^n) = r_t^n + V^\pi(s_{t+1}^n)$$

Actor-Critic

Advantage Actor-Critic (A2C)

$$\boxed{Q^\pi(s_t^n, a_t^n) - V^\pi(s_t^n)}$$

Estimate two networks? We can only estimate one.

$$\Downarrow$$

$$\boxed{r_t^n + V^\pi(s_{t+1}^n) - V^\pi(s_t^n)}$$

Only estimate state value

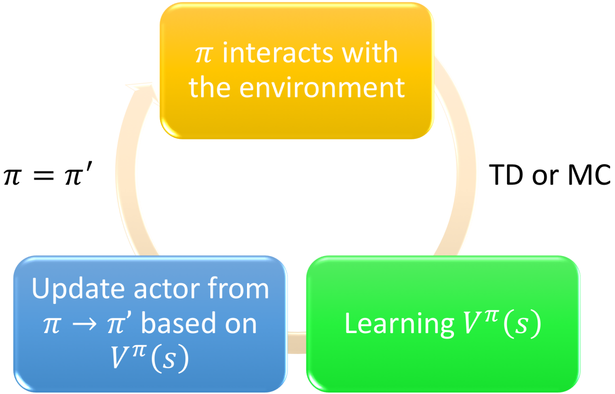
A little bit variance

$$Q^\pi(s_t^n, a_t^n) = E[r_t^n + V^\pi(s_{t+1}^n)]$$

$$Q^\pi(s_t^n, a_t^n) = r_t^n + V^\pi(s_{t+1}^n)$$

Actor-Critic

Advantage Actor-Critic (A2C)

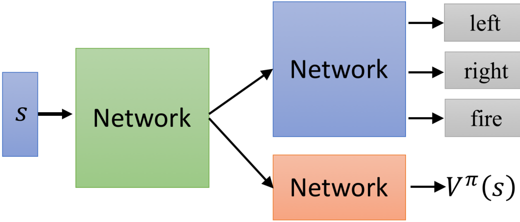


$$\nabla \bar{R}_{\theta} \approx \frac{1}{N} \sum_{n=1}^{N} \sum_{t=1}^{T_n} \left(r_t^n + V^{\pi}(s_{t+1}^n) - V^{\pi}(s_t^n) \right) \nabla log p_{\theta}(a_t^n | s_t^n)$$

Actor-Critic

Advantage Actor-Critic (A2C)

- Tips
  - The parameters of actor $\pi(s)$ and critic $V^\pi(s)$ can be shared



Actor-Critic

- Simple actor-critic algorithm based on action-value critic
- Using linear value function approximate $Q_w(s,a) = \phi(s,a)^T w$

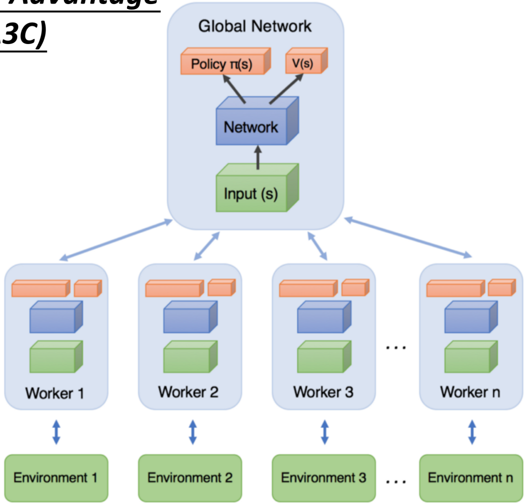
  Critic Updates w by linear TD(0)

  Actor Updates θ by policy gradient

function QAC
  Initialise s, θ
  Sample $a \sim \pi_\theta$
  for each step **do**
    Sample reward $r = \mathcal{R}_s^a$; sample transition $s' \sim \mathcal{P}_{s,\cdot}^a$
    Sample action $a' \sim \pi_\theta(s', a')$
    $\delta = r + \gamma Q_w(s', a') - Q_w(s,a)$
    $\theta = \theta + \alpha \nabla_\theta \log \pi_\theta(s,a) Q_w(s,a)$
    $w \leftarrow w + \beta \delta \phi(s,a)$
    $a \leftarrow a', s \leftarrow s'$
  end for
end function

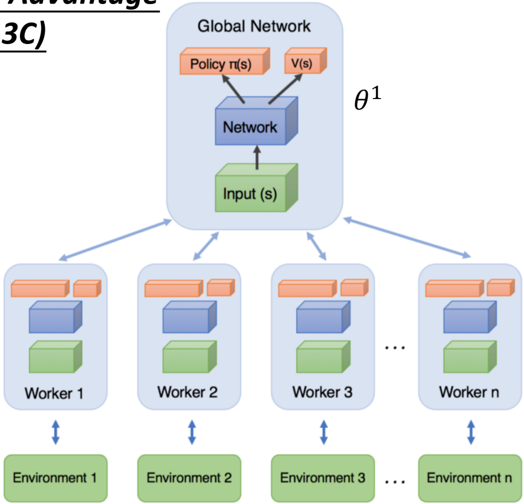
Actor-Critic

Asynchronous Advantage Actor-Critic (A3C)



Actor-Critic

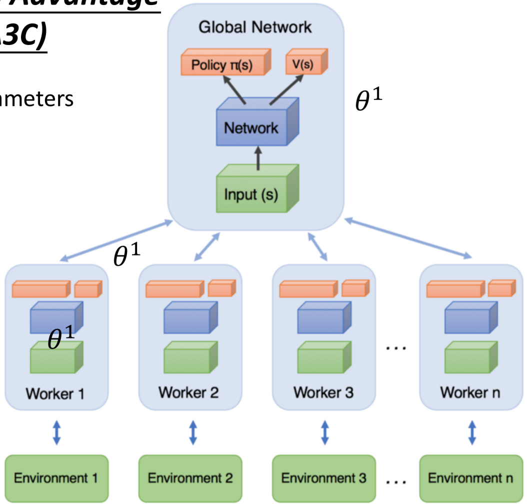
Asynchronous Advantage Actor-Critic (A3C)



Actor-Critic

Asynchronous Advantage Actor-Critic (A3C)

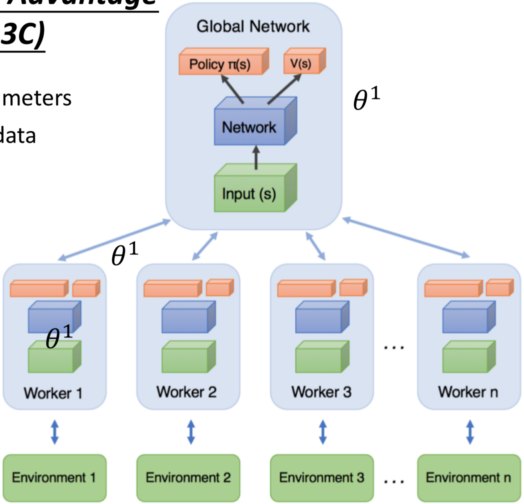
1. Copy global parameters



Actor-Critic

Asynchronous Advantage Actor-Critic (A3C)

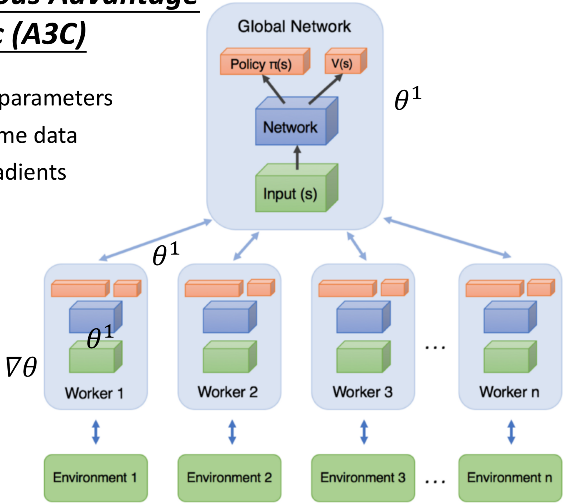
1. Copy global parameters
2. Sampling some data



Actor-Critic

Asynchronous Advantage Actor-Critic (A3C)

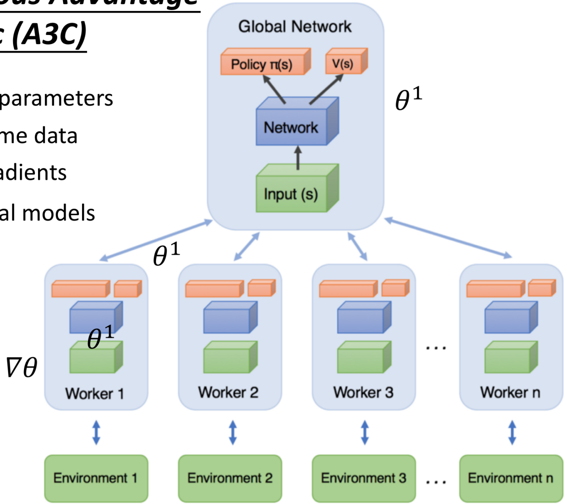
1. Copy global parameters
2. Sampling some data
3. Compute gradients



Actor-Critic

Asynchronous Advantage Actor-Critic (A3C)

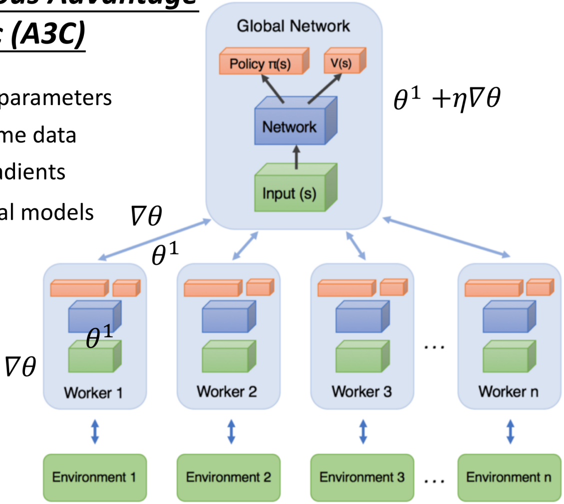
1. Copy global parameters
2. Sampling some data
3. Compute gradients
4. Update global models



Actor-Critic

<u>Asynchronous Advantage Actor-Critic (A3C)</u>

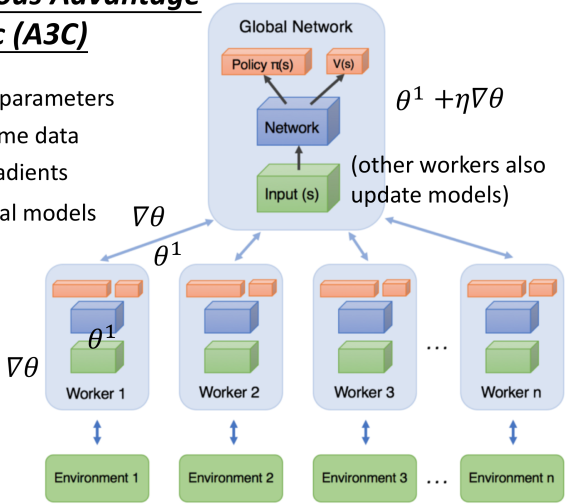
1. Copy global parameters
2. Sampling some data
3. Compute gradients
4. Update global models



Actor-Critic

Asynchronous Advantage Actor-Critic (A3C)

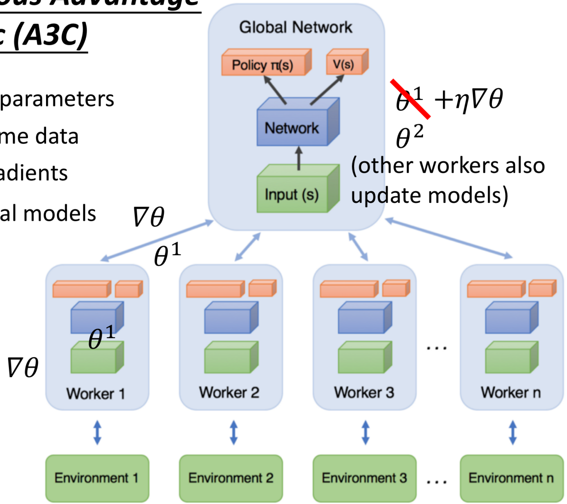
1. Copy global parameters
2. Sampling some data
3. Compute gradients
4. Update global models



Actor-Critic

<u>Asynchronous Advantage Actor-Critic (A3C)</u>

1. Copy global parameters
2. Sampling some data
3. Compute gradients
4. Update global models



Asynchronous Advantage Actor-Critic (A3C)

- Multiple asynchronous actors-learners run in parallel
- Explore different parts of environment, i.e., different exploration policies in different learners
- Stabilize learning, reduce training time, and without large resource requirements
- Add the entropy of policy to objective function
- Use standard non-centered RMSProp update

$$g = \alpha g + (1-\alpha)\Delta\theta^2 \text{ and } \theta \leftarrow \theta - \eta \frac{\Delta\theta}{\sqrt{g+\epsilon}}$$

Asynchronous Methods for Deep Reinforcement Learning, Proceedings of ICML, 2016

A3C Algorithm

Algorithm S3 Asynchronous advantage actor-critic - pseudocode for each actor-learner thread.

// Assume global shared parameter vectors θ and θ_v and global shared counter $T = 0$
// Assume thread-specific parameter vectors θ' and θ'_v
Initialize thread step counter $t \leftarrow 1$
repeat
  Reset gradients: $d\theta \leftarrow 0$ and $d\theta_v \leftarrow 0$.
  Synchronize thread-specific parameters $\theta' = \theta$ and $\theta'_v = \theta_v$
  $t_{start} = t$
  Get state s_t
  repeat
    Perform a_t according to policy $\pi(a_t|s_t;\theta')$
    Receive reward r_t and new state s_{t+1}
    $t \leftarrow t+1$
    $T \leftarrow T+1$
  until terminal s_t **or** $t - t_{start} == t_{max}$
  $R = \begin{cases} 0 & \text{for terminal } s_t \\ V(s_t, \theta'_v) & \text{for non-terminal } s_t \text{// Bootstrap from last state} \end{cases}$
  for $i \in \{t-1, \ldots, t_{start}\}$ **do**
    $R \leftarrow r_i + \gamma R$
    Accumulate gradients wrt θ': $d\theta \leftarrow d\theta + \nabla_{\theta'} \log \pi(a_i|s_i;\theta')(R - V(s_i;\theta'_v))$
    Accumulate gradients wrt θ'_v: $d\theta_v \leftarrow d\theta_v + \partial\left(R - V(s_i;\theta'_v)\right)^2 / \partial\theta'_v$
  end for
  Perform asynchronous update of θ using $d\theta$ and of θ_v using $d\theta_v$.
until $T > T_{max}$

Summary of Policy Gradient Algorithms

- The policy gradient has many equivalent forms

$$
\begin{aligned}
\nabla_\theta J(\theta) &= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\; v_t\right] && \text{REINFORCE} \\
&= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\; Q^w(s,a)\right] && \text{Q Actor-Critic} \\
&= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\; A^w(s,a)\right] && \text{Advantage Actor-Critic} \\
&= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\; \delta\right] && \text{TD Actor-Critic} \\
&= \mathbb{E}_{\pi_\theta}\left[\nabla_\theta \log \pi_\theta(s,a)\; \delta e\right] && \text{TD}(\lambda)\text{ Actor-Critic} \\
G_\theta^{-1}\nabla_\theta J(\theta) &= w && \text{Natural Actor-Critic}
\end{aligned}
$$

- Each leads a stochastic gradient ascent algorithm
- Critic uses policy evaluation (e.g. MC or TD learning) to estimate $Q^\pi(s,a)$, $A^\pi(s,a)$ or $V^\pi(s)$

Table of Contents

Proximal Policy Optimization

1. Policy Gradient→TRPO→ACKTR→PPO
   1. **TRPO**: Trust region policy optimization. Schulman, L., Moritz, Jordan, Abbeel. 2015
   2. **ACKTR**: Scalable trust-region method for deep reinforcement learning using Kronecker-factored approximation. Y. Wu, E. Mansimov, S. Liao, R. Grosse, and J. Ba. 2017
   3. **PPO**: Proximal policy optimization algorithms. Schulman, Wolski, Dhariwal, Radford, Klimov. 2017
2. Q-learning→DDPG→TD3→SAC
   1. **DDPG**: Deterministic Policy Gradient Algorithms, Silver et al. 2014
   2. **TD3**: Addressing Function Approximation Error in Actor-Critic Methods, Fujimoto et al. 2018
   3. **SAC**: Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor, Haarnoja et al. 2018

Proximal Policy Optimization

State-of-the-art RL methods are almost all policy-based

1. **TRPO**: Schulman, L., Moritz, Jordan, Abbeel (2015). Trust region policy optimization
    1. comment: Solid math proofs and guarantee, but hard to follow
2. **ACKTR**: Y. Wu, E. Mansimov, S. Liao, R. Grosse, and J. Ba (2017). Scalable trust-region method for deep reinforcement learning using Kronecker-factored approximation.
    1. comment: numeric optimization-based improvement, scalable to real-problems
3. **PPO**: Schulman, Wolski, Dhariwal, Radford, Klimov (2017). Proximal policy optimization algorithms
    1. comment: Easy to read, elegant design of loss function, easy to implement, widely used

Proximal Policy Optimization

Proximal Policy Optimization (PPO)

default reinforcement learning algorithm at OpenAI

On-Policy vs. Off-Policy

From on-policy to off-policy

Using the experience more than once

On-Policy vs. Off-Policy

On-policy v.s. Off-policy

- On-policy: The agent learned and the agent interacting with the environment is the same.
- Off-policy: The agent learned and the agent interacting with the environment is different.

阿光下棋

佐為下棋、阿光在旁邊看

On-Policy vs. Off-Policy

On-policy → Off-policy

$$\nabla \bar{R}_\theta = E_{\tau \sim p_\theta(\tau)}[R(\tau) \nabla log p_\theta(\tau)]$$

On-Policy vs. Off-Policy

On-policy → Off-policy

$$\nabla \bar{R}_\theta = E_{\tau \sim p_\theta(\tau)}[R(\tau)\nabla log p_\theta(\tau)]$$

- Use π_θ to collect data. When θ is updated, we have to sample training data again.

On-Policy vs. Off-Policy

On-policy → Off-policy

$$\nabla \bar{R}_\theta = E_{\underline{\tau \sim p_\theta(\tau)}}[R(\tau)\nabla log p_\theta(\tau)]$$

- Use π_θ to collect data. When θ is updated, we have to sample training data again.
- Goal: Using the sample from $\pi_{\theta'}$ to train θ. θ' is fixed, so we can re-use the sample data.

On-Policy vs. Off-Policy

On-policy → Off-policy

$$\nabla \bar{R}_\theta = E_{\underline{\tau \sim p_\theta(\tau)}}[R(\tau)\nabla log p_\theta(\tau)]$$

- Use π_θ to collect data. When θ is updated, we have to sample training data again.
- Goal: Using the sample from $\pi_{\theta'}$ to train θ. θ' is fixed, so we can re-use the sample data.

Importance Sampling

$$E_{x \sim p}[f(x)]$$

On-Policy vs. Off-Policy

On-policy → Off-policy

$$\nabla \bar{R}_\theta = E_{\underline{\tau \sim p_\theta(\tau)}}[R(\tau)\nabla log p_\theta(\tau)]$$

- Use π_θ to collect data. When θ is updated, we have to sample training data again.
- Goal: Using the sample from $\pi_{\theta'}$ to train θ. θ' is fixed, so we can re-use the sample data.

Importance Sampling

x^i is sampled from $p(x)$

$$E_{x \sim p}[f(x)] \approx \frac{1}{N}\sum_{i=1}^{N} f(x^i)$$

On-Policy vs. Off-Policy

On-policy → Off-policy

$$\nabla \bar{R}_\theta = E_{\underline{\tau \sim p_\theta(\tau)}}[R(\tau)\nabla log p_\theta(\tau)]$$

- Use π_θ to collect data. When θ is updated, we have to sample training data again.
- Goal: Using the sample from $\pi_{\theta'}$ to train θ. θ' is fixed, so we can re-use the sample data.

Importance Sampling

$$E_{x\sim p}[f(x)] \approx \cancel{\frac{1}{N}\sum_{i=1}^{N} f(x^i)}$$

x^i is sampled from $p(x)$

We only have x^i sampled from $q(x)$

On-Policy vs. Off-Policy

On-policy → Off-policy

$$\nabla \bar{R}_\theta = E_{\underline{\tau \sim p_\theta(\tau)}}[R(\tau)\nabla log p_\theta(\tau)]$$

- Use π_θ to collect data. When θ is updated, we have to sample training data again.
- Goal: Using the sample from $\pi_{\theta'}$ to train θ. θ' is fixed, so we can re-use the sample data.

Importance Sampling

$$E_{x\sim p}[f(x)] \approx \cancel{\frac{1}{N}\sum_{i=1}^{N} f(x^i)}$$

x^i is sampled from $p(x)$

We only have x^i sampled from $q(x)$

$$= \int f(x)p(x)dx \; = \int f(x)\frac{p(x)}{q(x)}q(x)dx \; = E_{x\sim q}[f(x)\frac{p(x)}{q(x)}]$$

On-Policy vs. Off-Policy

On-policy → Off-policy

$$\nabla \bar{R}_\theta = E_{\tau \sim p_\theta(\tau)}[R(\tau)\nabla log p_\theta(\tau)]$$

- Use π_θ to collect data. When θ is updated, we have to sample training data again.
- Goal: Using the sample from $\pi_{\theta'}$ to train θ. θ' is fixed, so we can re-use the sample data.

Importance Sampling

x^i is sampled from $p(x)$

$$E_{x\sim p}[f(x)] \approx \cancel{\frac{1}{N}\sum_{i=1}^{N} f(x^i)}$$

We only have x^i sampled from $q(x)$

$$= \int f(x)p(x)dx = \int f(x)\frac{p(x)}{q(x)}q(x)dx = E_{x\sim q}[f(x)\boxed{\frac{p(x)}{q(x)}}]$$

Importance weight

On-Policy vs. Off-Policy

Issue of Importance Sampling

$$E_{x\sim p}[f(x)] = E_{x\sim q}[f(x)\frac{p(x)}{q(x)}]$$

$$\mathrm{Var}_{x\sim p}[f(x)] \quad \mathrm{Var}_{x\sim q}[f(x)\frac{p(x)}{q(x)}]$$

$$VAR[X] = E[X^2] - (E[X])^2$$

On-Policy vs. Off-Policy

Issue of Importance Sampling

$$E_{x\sim p}[f(x)] = E_{x\sim q}[f(x)\frac{p(x)}{q(x)}]$$

$$Var_{x\sim p}[f(x)] \qquad Var_{x\sim q}[f(x)\frac{p(x)}{q(x)}]$$

$$VAR[X] = E[X^2] - (E[X])^2$$

$$Var_{x\sim p}[f(x)] = E_{x\sim p}[f(x)^2] - \left(E_{x\sim p}[f(x)]\right)^2$$

On-Policy vs. Off-Policy

Issue of Importance Sampling

$$E_{x\sim p}[f(x)] = E_{x\sim q}[f(x)\frac{p(x)}{q(x)}]$$

$$Var_{x\sim p}[f(x)] \qquad Var_{x\sim q}[f(x)\frac{p(x)}{q(x)}]$$

$$VAR[X] = E[X^2] - (E[X])^2$$

$$Var_{x\sim p}[f(x)] = E_{x\sim p}[f(x)^2] - \left(E_{x\sim p}[f(x)]\right)^2$$

$$Var_{x\sim q}[f(x)\frac{p(x)}{q(x)}] = E_{x\sim q}\left[\left(f(x)\frac{p(x)}{q(x)}\right)^2\right] - \left(E_{x\sim q}\left[f(x)\frac{p(x)}{q(x)}\right]\right)^2$$

On-Policy vs. Off-Policy

Issue of Importance Sampling

$$E_{x\sim p}[f(x)] = E_{x\sim q}[f(x)\frac{p(x)}{q(x)}]$$

$$Var_{x\sim p}[f(x)] \qquad Var_{x\sim q}[f(x)\frac{p(x)}{q(x)}]$$

$$\boxed{\begin{aligned} &VAR[X] \\ &= E[X^2] - (E[X])^2 \end{aligned}}$$

$$Var_{x\sim p}[f(x)] = E_{x\sim p}[f(x)^2] - \left(E_{x\sim p}[f(x)]\right)^2$$

$$Var_{x\sim q}[f(x)\frac{p(x)}{q(x)}] = E_{x\sim q}\left[\left(f(x)\frac{p(x)}{q(x)}\right)^2\right] - \left(E_{x\sim q}\left[f(x)\frac{p(x)}{q(x)}\right]\right)^2$$

$$= E_{x\sim p}\left[f(x)^2\frac{p(x)}{q(x)}\right] - \left(E_{x\sim p}[f(x)]\right)^2$$

On-Policy vs. Off-Policy

Issue of Importance Sampling

$$E_{x\sim p}[f(x)] = E_{x\sim q}[f(x)\frac{p(x)}{q(x)}]$$

$$Var_{x\sim p}[f(x)] \qquad Var_{x\sim q}[f(x)\frac{p(x)}{q(x)}]$$

$$VAR[X] = E[X^2] - (E[X])^2$$

$$Var_{x\sim p}[f(x)] = E_{x\sim p}[f(x)^2] - \left(E_{x\sim p}[f(x)]\right)^2$$

$$Var_{x\sim q}[f(x)\frac{p(x)}{q(x)}] = E_{x\sim q}\left[\left(f(x)\frac{p(x)}{q(x)}\right)^2\right] - \left(E_{x\sim q}\left[f(x)\frac{p(x)}{q(x)}\right]\right)^2$$

$$= E_{x\sim p}\left[f(x)^2\boxed{\frac{p(x)}{q(x)}}\right] - \left(E_{x\sim p}[f(x)]\right)^2$$

On-Policy vs. Off-Policy

On-policy → Off-policy

$$\nabla \bar{R}_\theta = E_{\underline{\tau \sim p_\theta(\tau)}}[R(\tau)\nabla log p_\theta(\tau)]$$

- Use π_θ to collect data. When θ is updated, we have to sample training data again.
- Goal: Using the sample from $\pi_{\theta'}$ to train θ. θ' is fixed, so we can re-use the sample data.

Importance Sampling

$$E_{x\sim p}[f(x)] = E_{x\sim q}[f(x)\frac{p(x)}{q(x)}]$$

On-Policy vs. Off-Policy

On-policy → Off-policy

$$\nabla \bar{R}_\theta = E_{\underline{\tau \sim p_\theta(\tau)}}[R(\tau)\nabla log p_\theta(\tau)]$$

- Use π_θ to collect data. When θ is updated, we have to sample training data again.
- Goal: Using the sample from $\pi_{\theta'}$ to train θ. θ' is fixed, so we can re-use the sample data.

$$\nabla \bar{R}_\theta = E_{\underline{\tau \sim p_{\theta'}(\tau)}}\left[\frac{p_\theta(\tau)}{\underline{p_{\theta'}(\tau)}} R(\tau)\nabla log p_\theta(\tau)\right]$$

Importance Sampling

$$E_{x\sim p}[f(x)] = E_{x\sim q}[f(x)\frac{p(x)}{q(x)}]$$

On-Policy vs. Off-Policy

On-policy → Off-policy

$$\nabla \bar{R}_\theta = E_{\tau \sim p_\theta(\tau)}[R(\tau)\nabla log p_\theta(\tau)]$$

- Use π_θ to collect data. When θ is updated, we have to sample training data again.
- Goal: Using the sample from $\pi_{\theta'}$ to train θ. θ' is fixed, so we can re-use the sample data.

$$\nabla \bar{R}_\theta = E_{\tau \sim p_{\theta'}(\tau)}\left[\frac{p_\theta(\tau)}{p_{\theta'}(\tau)} R(\tau)\nabla log p_\theta(\tau)\right]$$

- Sample the data from θ'.
- Use the data to train θ many times.

Importance Sampling

$$E_{x\sim p}[f(x)] = E_{x\sim q}[f(x)\frac{p(x)}{q(x)}]$$

On-Policy vs. Off-Policy

On-policy → Off-policy

Gradient for update

$$\boxed{\nabla f(x) = f(x) \nabla log f(x)}$$

$$= E_{(s_t,a_t)\sim\pi_\theta}[A^\theta(s_t,a_t) \nabla log p_\theta(a_t^n|s_t^n)]$$

On-Policy vs. Off-Policy

On-policy → Off-policy

Gradient for update

$$\nabla f(x) = f(x)\nabla logf(x)$$

$$= E_{(s_t,a_t)\sim\pi_\theta}[A^\theta(s_t,a_t)\nabla logp_\theta(a_t^n|s_t^n)]$$

$$= E_{(s_t,a_t)\sim\pi_{\theta'}}[\frac{P_\theta(s_t,a_t)}{P_{\theta'}(s_t,a_t)}A^\theta(s_t,a_t)\nabla logp_\theta(a_t^n|s_t^n)]$$

On-Policy vs. Off-Policy

On-policy → Off-policy

$$\nabla f(x) = f(x)\nabla log f(x)$$

Gradient for update

$$= E_{(s_t,a_t)\sim\pi_\theta}[A^\theta(s_t,a_t)\nabla log p_\theta(a_t^n|s_t^n)]$$

$$A^{\theta'}(s_t,a_t)$$

This term is from sampled data.

$$= E_{(s_t,a_t)\sim\pi_{\theta'}}[\frac{P_\theta(s_t,a_t)}{P_{\theta'}(s_t,a_t)}\cancel{A^{\boxed{\theta}}(s_t,a_t)}\nabla log p_\theta(a_t^n|s_t^n)]$$

On-Policy vs. Off-Policy

On-policy → Off-policy

Gradient for update

$$\nabla f(x) = f(x)\nabla log f(x)$$

$$= E_{(s_t,a_t)\sim\pi_\theta}[A^\theta(s_t,a_t)\nabla log p_\theta(a_t^n|s_t^n)]$$

$A^{\theta'}(s_t,a_t)$ This term is from sampled data.

$$= E_{(s_t,a_t)\sim\pi_{\theta'}}[\frac{P_\theta(s_t,a_t)}{P_{\theta'}(s_t,a_t)}\cancel{A^{\boxed{\theta}}(s_t,a_t)}\nabla log p_\theta(a_t^n|s_t^n)]$$

$$= E_{(s_t,a_t)\sim\pi_{\theta'}}[\frac{p_\theta(a_t|s_t)}{p_{\theta'}(a_t|s_t)}\frac{p_\theta(s_t)}{p_{\theta'}(s_t)}A^{\theta'}(s_t,a_t)\nabla log p_\theta(a_t^n|s_t^n)]$$

On-Policy vs. Off-Policy

On-policy → Off-policy

Gradient for update

$$\boxed{\nabla f(x) = f(x)\nabla \log f(x)}$$

$$= E_{(s_t,a_t)\sim\pi_\theta}[A^\theta(s_t,a_t)\nabla \log p_\theta(a_t^n|s_t^n)]$$

$$\boxed{A^{\theta'}(s_t,a_t)}$$

This term is from sampled data.

$$= E_{(s_t,a_t)\sim\pi_{\theta'}}[\frac{P_\theta(s_t,a_t)}{P_{\theta'}(s_t,a_t)}\cancel{A^{\boxed{\theta}}(s_t,a_t)}\nabla \log p_\theta(a_t^n|s_t^n)]$$

$$= E_{(s_t,a_t)\sim\pi_{\theta'}}[\frac{p_\theta(a_t|s_t)}{p_{\theta'}(a_t|s_t)}\cancel{\frac{p_\theta(s_t)}{p_{\theta'}(s_t)}}A^{\theta'}(s_t,a_t)\nabla \log p_\theta(a_t^n|s_t^n)]$$

On-Policy vs. Off-Policy

On-policy → Off-policy

$$\nabla f(x) = f(x)\nabla \log f(x)$$

Gradient for update

$$= E_{(s_t,a_t)\sim\pi_\theta}[A^\theta(s_t,a_t)\nabla \log p_\theta(a_t^n|s_t^n)]$$

$$A^{\theta'}(s_t,a_t)$$

This term is from sampled data.

$$= E_{(s_t,a_t)\sim\pi_{\theta'}}[\frac{P_\theta(s_t,a_t)}{P_{\theta'}(s_t,a_t)}\cancel{A^{\boxed{\theta}}(s_t,a_t)}\nabla \log p_\theta(a_t^n|s_t^n)]$$

$$= E_{(s_t,a_t)\sim\pi_{\theta'}}[\frac{p_\theta(a_t|s_t)}{p_{\theta'}(a_t|s_t)}\cancel{\frac{p_\theta(s_t)}{p_{\theta'}(s_t)}}A^{\theta'}(s_t,a_t)\nabla \log p_\theta(a_t^n|s_t^n)]$$

$$J^{\theta'}(\theta) = E_{(s_t,a_t)\sim\pi_{\theta'}}\left[\frac{p_\theta(a_t|s_t)}{p_{\theta'}(a_t|s_t)}A^{\theta'}(s_t,a_t)\right]$$

On-Policy vs. Off-Policy

On-policy → Off-policy

$$\boxed{\nabla f(x) = f(x)\nabla log f(x)}$$

Gradient for update

$$= E_{(s_t,a_t)\sim\pi_\theta}[A^\theta(s_t,a_t)\nabla log p_\theta(a_t^n|s_t^n)]$$

$$\boxed{A^{\theta'}(s_t,a_t)}$$

This term is from sampled data.

$$= E_{(s_t,a_t)\sim\pi_{\theta'}}[\frac{P_\theta(s_t,a_t)}{P_{\theta'}(s_t,a_t)}\cancel{A^{\boxed{\theta}}(s_t,a_t)}\nabla log p_\theta(a_t^n|s_t^n)]$$

$$= E_{(s_t,a_t)\sim\pi_{\theta'}}[\frac{p_\theta(a_t|s_t)}{p_{\theta'}(a_t|s_t)}\cancel{\frac{p_\theta(s_t)}{p_{\theta'}(s_t)}}A^{\theta'}(s_t,a_t)\nabla log p_\theta(a_t^n|s_t^n)]$$

$$J^{\theta'}(\theta) = E_{(s_t,a_t)\sim\pi_{\theta'}}\left[\frac{p_\theta(a_t|s_t)}{p_{\theta'}(a_t|s_t)}A^{\theta'}(s_t,a_t)\right]$$

When to stop?

On-Policy vs. Off-Policy

Add Constraint

Table of Contents

Proximal Policy Optimization

PPO / TRPO

Proximal Policy Optimization (PPO)

$$\nabla f(x) = f(x)\nabla log f(x)$$

$$J_{PPO}^{\theta'}(\theta) = J^{\theta'}(\theta) - \beta KL(\theta, \theta')$$

$$J^{\theta'}(\theta) = E_{(s_t,a_t)\sim\pi_{\theta'}}\left[\frac{p_\theta(a_t|s_t)}{p_{\theta'}(a_t|s_t)} A^{\theta'}(s_t, a_t)\right]$$

Proximal Policy Optimization

PPO / TRPO

Proximal Policy Optimization (PPO)

$$\nabla f(x) = f(x)\nabla log f(x)$$

$$J_{PPO}^{\theta'}(\theta) = J^{\theta'}(\theta) - \beta KL(\theta, \theta')$$

$$J^{\theta'}(\theta) = E_{(s_t,a_t)\sim\pi_{\theta'}}\left[\frac{p_\theta(a_t|s_t)}{p_{\theta'}(a_t|s_t)}A^{\theta'}(s_t, a_t)\right]$$

TRPO (Trust Region Policy Optimization)

$$J_{TRPO}^{\theta'}(\theta) = E_{(s_t,a_t)\sim\pi_{\theta'}}\left[\frac{p_\theta(a_t|s_t)}{p_{\theta'}(a_t|s_t)}A^{\theta'}(s_t, a_t)\right]$$

$$KL(\theta, \theta') < \delta$$

Proximal Policy Optimization

PPO / TRPO

θ cannot be very different from θ'

Constraint on behavior not parameters

Proximal Policy Optimization (PPO)

$$\nabla f(x) = f(x) \nabla log f(x)$$

$$J_{PPO}^{\theta'}(\theta) = J^{\theta'}(\theta) - \beta KL(\theta, \theta')$$

$$J^{\theta'}(\theta) = E_{(s_t,a_t)\sim\pi_{\theta'}}\left[\frac{p_\theta(a_t|s_t)}{p_{\theta'}(a_t|s_t)} A^{\theta'}(s_t, a_t)\right]$$

TRPO (Trust Region Policy Optimization)

$$J_{TRPO}^{\theta'}(\theta) = E_{(s_t,a_t)\sim\pi_{\theta'}}\left[\frac{p_\theta(a_t|s_t)}{p_{\theta'}(a_t|s_t)} A^{\theta'}(s_t, a_t)\right]$$

$$KL(\theta, \theta') < \delta$$

Proximal Policy Optimization

PPO algorithm

$$J^{\theta^k}(\theta) \approx \sum_{(s_t,a_t)} \frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)} A^{\theta^k}(s_t,a_t)$$

- Initial policy parameters θ^0
- In each iteration
  - Using θ^k to interact with the environment to collect $\{s_t, a_t\}$ and compute advantage $A^{\theta^k}(s_t, a_t)$
  - Find θ optimizing $J_{PPO}(\theta)$

$$J_{PPO}^{\theta^k}(\theta) = J^{\theta^k}(\theta) - \beta KL\left(\theta, \theta^k\right)$$

Update parameters several times

Proximal Policy Optimization

PPO algorithm

$$J^{\theta^k}(\theta) \approx \sum_{(s_t,a_t)} \frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)} A^{\theta^k}(s_t,a_t)$$

- Initial policy parameters θ^0
- In each iteration
  - Using θ^k to interact with the environment to collect $\{s_t, a_t\}$ and compute advantage $A^{\theta^k}(s_t, a_t)$
  - Find θ optimizing $J_{PPO}(\theta)$

$$J_{PPO}^{\theta^k}(\theta) = J^{\theta^k}(\theta) - \beta KL\left(\theta, \theta^k\right)$$

Update parameters several times

  - If $KL\left(\theta, \theta^k\right) > KL_{max}$, increase β
  - If $KL\left(\theta, \theta^k\right) < KL_{min}$, decrease β

Proximal Policy Optimization

PPO algorithm

$$J^{\theta^k}(\theta) \approx \sum_{(s_t,a_t)} \frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)} A^{\theta^k}(s_t,a_t)$$

- Initial policy parameters θ^0
- In each iteration
  - Using θ^k to interact with the environment to collect $\{s_t, a_t\}$ and compute advantage $A^{\theta^k}(s_t, a_t)$
  - Find θ optimizing $J_{PPO}(\theta)$

$$J_{PPO}^{\theta^k}(\theta) = J^{\theta^k}(\theta) - \beta KL\left(\theta, \theta^k\right)$$

Update parameters several times

- If $KL\left(\theta, \theta^k\right) > KL_{max}$, increase β
- If $KL\left(\theta, \theta^k\right) < KL_{min}$, decrease β

Adaptive KL Penalty

Proximal Policy Optimization

PPO algorithm

$$J_{PPO}^{\theta^k}(\theta) = J^{\theta^k}(\theta) - \beta KL(\theta, \theta^k)$$

$$J^{\theta^k}(\theta) \approx \sum_{(s_t,a_t)} \frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)} A^{\theta^k}(s_t, a_t)$$

PPO2 algorithm

$$J_{PPO2}^{\theta^k}(\theta) \approx \sum_{(s_t,a_t)} min\left(\frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)} A^{\theta^k}(s_t, a_t), \right.$$
$$\left. clip\left(\frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)}, 1 - \varepsilon, 1 + \varepsilon \right) A^{\theta^k}(s_t, a_t) \right)$$

Proximal Policy Optimization

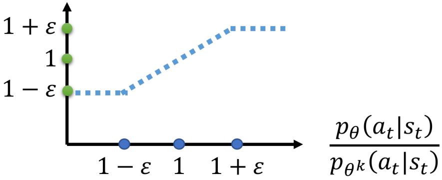
PPO algorithm

$$J_{PPO}^{\theta^k}(\theta) = J^{\theta^k}(\theta) - \beta KL(\theta, \theta^k)$$

$$J^{\theta^k}(\theta) \approx \sum_{(s_t,a_t)} \frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)} A^{\theta^k}(s_t, a_t)$$

PPO2 algorithm

$$J_{PPO2}^{\theta^k}(\theta) \approx \sum_{(s_t,a_t)} clip\left(\frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)}, 1-\varepsilon, 1+\varepsilon\right) A^{\theta^k}(s_t, a_t)$$



Proximal Policy Optimization

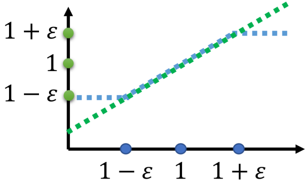
PPO algorithm

$$J_{PPO}^{\theta^k}(\theta) = J^{\theta^k}(\theta) - \beta KL(\theta, \theta^k)$$

$$J^{\theta^k}(\theta) \approx \sum_{(s_t,a_t)} \frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)} A^{\theta^k}(s_t, a_t)$$

PPO2 algorithm

$$J_{PPO2}^{\theta^k}(\theta) \approx \sum_{(s_t,a_t)} min\left(\frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)} A^{\theta^k}(s_t, a_t),\right.$$
$$\left. clip\left(\frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)}, 1-\varepsilon, 1+\varepsilon\right) A^{\theta^k}(s_t, a_t)\right)$$



Proximal Policy Optimization

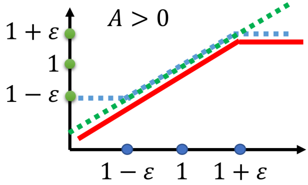
PPO algorithm

$$J_{PPO}^{\theta^k}(\theta) = J^{\theta^k}(\theta) - \beta KL(\theta, \theta^k)$$

$$J^{\theta^k}(\theta) \approx \sum_{(s_t,a_t)} \frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)} A^{\theta^k}(s_t, a_t)$$

PPO2 algorithm

$$J_{PPO2}^{\theta^k}(\theta) \approx \sum_{(s_t,a_t)} min\left(\frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)} A^{\theta^k}(s_t, a_t),\right.$$
$$\left. clip\left(\frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)}, 1-\varepsilon, 1+\varepsilon\right) A^{\theta^k}(s_t, a_t)\right)$$



Proximal Policy Optimization

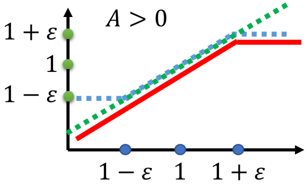
PPO algorithm

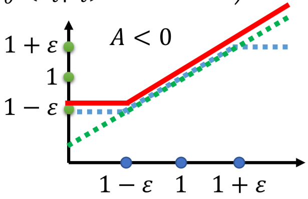
$$J^{\theta^k}_{PPO}(\theta) = J^{\theta^k}(\theta) - \beta KL(\theta, \theta^k)$$

$$J^{\theta^k}(\theta) \approx \sum_{(s_t,a_t)} \frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)} A^{\theta^k}(s_t,a_t)$$

PPO2 algorithm

$$J^{\theta^k}_{PPO2}(\theta) \approx \sum_{(s_t,a_t)} min\left(\frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)} A^{\theta^k}(s_t,a_t), \; clip\left(\frac{p_\theta(a_t|s_t)}{p_{\theta^k}(a_t|s_t)}, 1-\varepsilon, 1+\varepsilon\right) A^{\theta^k}(s_t,a_t)\right)$$





Proximal Policy Optimization

A proximal policy optimization (PPO) algorithm that uses fixed-length trajectory segments is shown below. Each iteration, each of N (parallel) actors collect T timesteps of data. Then we construct the surrogate loss on these NT timesteps of data, and optimize it with minibatch SGD (or usually for better performance, Adam [KB14]), for K epochs.

Algorithm 1 PPO, Actor-Critic Style

> **for** iteration$=1, 2, \ldots$ **do**
>   **for** actor$=1, 2, \ldots, N$ **do**
>   Run policy $\pi_{\theta_{\text{old}}}$ in environment for T timesteps
>   Compute advantage estimates $\hat{A}_1, \ldots, \hat{A}_T$
>   **end for**
>   Optimize surrogate L wrt θ, with K epochs and minibatch size $M \leq NT$
> $\theta_{\text{old}} \leftarrow \theta$
> **end for**

Proximal Policy Optimization

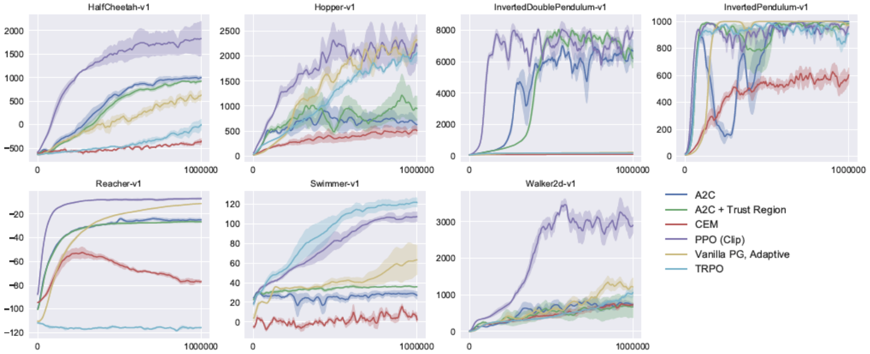


Figure 3: Comparison of several algorithms on several MuJoCo environments, training for one million timesteps.