Advanced Actor-Critic Methods (PPO, TRPO)

Alina Vereshchaka

CSE4/510 Reinforcement Learning
Fall 2019

avereshc@buffalo.edu

November 7, 2019

*Slides are adopted from Deep Reinforcement Learning by Sergey Levine & Policy Gradients Algorithms by Lilian Weng
Table of Contents

1 Recap: Actor-Critic

2 Importance Sampling

3 Trust region policy optimization (TRPO)

4 Proximal Policy Optimization (PPO)
Value Based and Policy-Based RL

- **Value Based**
  - Learn Value Function
  - Implicit policy

- **Policy Based**
  - No Value Function
  - Learn Policy

- **Actor-Critic**
  - Learn Value Function
  - Learn Policy
Actor-Critic
The policy gradient has many equivalent forms:

\[ \nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta} [\nabla_\theta \log \pi_\theta(s, a) G_t] \]

\[ = \mathbb{E}_{\pi_\theta} [\nabla_\theta \log \pi_\theta(s, a) Q_w(s, a)] \]

\[ = \mathbb{E}_{\pi_\theta} [\nabla_\theta \log \pi_\theta(s, a) A_w(s, a)] \]

\[ = \mathbb{E}_{\pi_\theta} [\nabla_\theta \log \pi_\theta(s, a) \delta] \]

- REINFORCE
- Q Actor-Critic
- Advantage Actor-Critic (A2C)
- TD Actor-Critic

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 \( V_\pi(s) \).
DDPG does soft updates ("conservative policy iteration") on the parameters of both actor and critic

\[ \theta' \leftarrow \tau \theta + (1 - \tau)\theta' \]

In this way, the target network values are constrained to change slowly, different from the design in DQN that the target network stays frozen for some period of time.
Deep Deterministic Policy Gradient (DDPG)

**Algorithm 1** DDPG algorithm

Randomly initialize critic network $Q(s, a|\theta^Q)$ and actor $\mu(s|\theta^\mu)$ with weights $\theta^Q$ and $\theta^\mu$.

Initialize target network $Q'$ and $\mu'$ with weights $\theta^{Q'} \leftarrow \theta^Q$, $\theta^{\mu'} \leftarrow \theta^\mu$.

Initialize replay buffer $R$.

for episode = 1, M do

Initialize a random process $N$ for action exploration.

Receive initial observation state $s_1$.

for $t = 1, T$ do

Select action $a_t = \mu(s_t|\theta^\mu) + N_i$ according to the current policy and exploration noise.

Execute action $a_t$ and observe reward $r_t$ and observe new state $s_{t+1}$.

Store transition $(s_t, a_t, r_t, s_{t+1})$ in $R$.

Sample a random minibatch of $N$ transitions $(s_i, a_i, r_i, s_{i+1})$ from $R$.

Set $y_i = r_i + \gamma Q'(s_{i+1}, \mu'(s_{i+1}|\theta^{\mu'})|\theta^{Q'})$.

Update critic by minimizing the loss: $L = \frac{1}{N} \sum_i (y_i - Q(s_i, a_i|\theta^Q))^2$.

Update the actor policy using the sampled policy gradient:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N} \sum_i \nabla_a Q(s, a|\theta^Q)|_{s=s_i, a=\mu(s_i)} \nabla_{\theta^\mu} \mu(s|\theta^\mu)|_{s_i}$$

Update the target networks:

$$\theta^{Q'} \leftarrow \tau \theta^Q + (1 - \tau) \theta^{Q'}$$

$$\theta^{\mu'} \leftarrow \tau \theta^\mu + (1 - \tau) \theta^{\mu'}$$

end for

end for
Deep Deterministic Policy Gradient (DDPG)

- DDPG is an algorithm which concurrently learns a Q-function and a policy
- DDPG is an off-policy algorithm
- DDPG can only be used for environments with continuous action spaces
- DDPG can be thought of as being deep Q-learning for continuous action spaces
Table of Contents

1 Recap: Actor-Critic

2 Importance Sampling

3 Trust region policy optimization (TRPO)

4 Proximal Policy Optimization (PPO)
Problems in Policy Gradient

\[ \pi_{\theta_1}(a_t|s_t) \]

OLD STATE VISITATION

\[ \pi_{\theta_{old}}(a_t|s_t) \]

NEW POLICY'S STATE VISITATION

BAD POLICY UPDATE

NEW POLICY'S STATE VISITATION

GOOD POLICY UPDATE
On-policy Sampling

\[ \theta^* = \arg \max_{\theta} J(\theta) \]

\[ J(\theta) = E_{\tau \sim \pi_\theta(\tau)}[r(\tau)] \]

\[ \nabla_\theta J(\theta) = E_{\tau \sim \pi_\theta(\tau)}[\nabla_\theta \log \pi_\theta(\tau) r(\tau)] \]

- Neural networks change only a little bit with each gradient step
- On-policy learning can be extremely inefficient!

REINFORCE algorithm:
1. sample \( \{\tau^i\} \) from \( \pi_\theta(a_t|s_t) \) (run it on the robot)
2. \( \nabla_\theta J(\theta) \approx \sum_i \left( \sum_t \nabla_\theta \log \pi_\theta(a_t^i|s_t^i) \right) \left( \sum_t r(s_t^i, a_t^i) \right) \)
3. \( \theta \leftarrow \theta + \alpha \nabla_\theta J(\theta) \)

this is trouble...
can’t just skip this!
On-policy Sampling
$\theta^* = \arg \max_{\theta} J(\theta)$

$J(\theta) = E_{\tau \sim \pi_\theta(\tau)}[r(\tau)]$

what if we don’t have samples from $\pi_\theta(\tau)$?
(we have samples from some $\bar{\pi}(\tau)$ instead)

$J(\theta) = E_{\tau \sim \bar{\pi}(\tau)} \left[ \frac{\pi_\theta(\tau)}{\bar{\pi}(\tau)} r(\tau) \right]$

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

$\frac{\pi_\theta(\tau)}{\bar{\pi}(\tau)} = \frac{p(s_1) \prod_{t=1}^{T} \pi_\theta(a_t|s_t)p(s_{t+1}|s_t, a_t)}{p(s_1) \prod_{t=1}^{T} \bar{\pi}(a_t|s_t)p(s_{t+1}|s_t, a_t)} = \frac{\prod_{t=1}^{T} \pi_\theta(a_t|s_t)}{\prod_{t=1}^{T} \bar{\pi}(a_t|s_t)}$

importance sampling

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

$= \int \frac{q(x)}{q(x)} p(x)f(x)dx$

$= \int q(x) \frac{p(x)}{q(x)} f(x)dx$

$= E_{x \sim q(x)} \left[ \frac{p(x)}{q(x)} f(x) \right]$
Off-Policy Sampling

On-policy

Sample collection with $\theta^k$

Update from $\theta^k$ to $\theta^{k+1}$

Off-policy

Update from $\theta^{k+1}$ to $\theta^{k+2}$

Off-policy

Update from $\theta^{k+2}$ to $\theta^{k+3}$

Sample collection with $\theta^{k+3}$

Off-policy

Update from $\theta^{k+3}$ to $\theta^{k+4}$

On-policy
1 Recap: Actor-Critic

2 Importance Sampling

3 Trust region policy optimization (TRPO)

4 Proximal Policy Optimization (PPO)
To improve training stability, we should avoid parameter updates that change the policy too much at one step.
To improve training stability, we should avoid parameter updates that change the policy too much at one step.

Trust region policy optimization (TRPO) (Schulman, et al., 2015) carries out this idea by enforcing a KL divergence constraint on the size of policy update at each iteration.
Trust region policy optimization (TRPO)

If off policy, the objective function measures the total advantage over the state visitation distribution and actions, while the rollout is following a different behavior policy $\beta(a|s)$:

$$J(\theta) = \sum_{s \in S} \rho^\pi \sum_{a \in A} (\pi(a|s)\hat{A}_{\theta_{old}}(s,a))$$

where $\theta_{old}$ is the policy parameters before the update and thus known to us; $\beta(a|s)$ is the behavior policy for collecting trajectories.
Trust region policy optimization (TRPO)

If off policy, the objective function measures the total advantage over the state visitation distribution and actions, while the rollout is following a different behavior policy $\beta(a|s)$:

$$J(\theta) = \sum_{s \in S} \rho_{\pi_{\text{old}}} \sum_{a \in A} (\pi_{\theta}(a|s) \hat{A}_{\theta_{\text{old}}}(s, a))$$

$$= \sum_{s \in S} \rho_{\pi_{\text{old}}} \sum_{a \in A} \left( \beta(a|s) \frac{\pi_{\theta}(a|s)}{\beta(a|s)} \hat{A}_{\theta_{\text{old}}}(s, a) \right) \quad ; \text{Importance sampling}$$
Trust region policy optimization (TRPO)

If off policy, the objective function measures the total advantage over the state visitation distribution and actions, while the rollout is following a different behavior policy $\beta(a|s)$:

$$J(\theta) = \sum_{s \in S} \rho_{\pi_{\theta_{\text{old}}}} \sum_{a \in A} (\pi_{\theta}(a|s) \hat{A}_{\theta_{\text{old}}}(s, a))$$

$$= \sum_{s \in S} \rho_{\pi_{\theta_{\text{old}}}} \sum_{a \in A} (\beta(a|s) \frac{\pi_{\theta}(a|s)}{\beta(a|s)} \hat{A}_{\theta_{\text{old}}}(s, a)) \quad ; \text{Importance sampling}$$

$$= \mathbb{E}_{s \sim \rho_{\pi_{\theta_{\text{old}}}}, a \sim \beta} \left[ \frac{\pi_{\theta}(a|s)}{\beta(a|s)} \hat{A}_{\theta_{\text{old}}}(s, a) \right]$$

where $\theta_{\text{old}}$ is the policy parameters before the update and thus known to us; $\beta(a|s)$ is the behavior policy for collecting trajectories.
The Kullback-Leibler (KL) Divergence score, or KL divergence score, quantifies how much one probability distribution differs from another probability distribution.

The KL divergence between two distributions $Q$ and $P$ is defined as:

$$ D_{KL}(P || Q) = \sum_{x \in X} p(x) \log \frac{p(x)}{q(x)} $$

Intuition: when the probability for an event from $P$ is large, but the probability for the same event in $Q$ is small $\rightarrow$ there is a large divergence. If the probability from $P$ is small and the probability from $Q$ is large $\rightarrow$ a large divergence, but not as large as the first case.
The Kullback-Leibler (KL) Divergence score, or KL divergence score, quantifies how much one probability distribution differs from another probability distribution.

The KL divergence between two distributions $Q$ and $P$:

$$KL(P∥Q)$$

where the $∥$ indicates “divergence” or $P$'s divergence from $Q$.
The Kullback-Leibler (KL) Divergence score, or KL divergence score, quantifies how much one probability distribution differs from another probability distribution.

The KL divergence between two distributions $Q$ and $P$:

$$KL(P || Q)$$

where the $||$ indicates “divergence” or $P$'s divergence from $Q$.

Given two probability distributions $p(x)$ and $q(x)$ over a discrete random variable $X$, the relative entropy given by $D(p||q)$ is defined as follows:

$$D_{KL}(p || q) = \sum_{x \in X} p(x) \log \frac{p(x)}{q(x)}$$
The Kullback-Leibler (KL) Divergence score, or KL divergence score, quantifies how much one probability distribution differs from another probability distribution.

\[
KL(P \| Q)
\]

where the \( \| \) indicates “divergence” or P’s divergence from Q.

Given two probability distributions \( p(x) \) and \( q(x) \) over a discrete random variable \( X \), the relative entropy given by \( D(p \| q) \) is defined as follows:

\[
D_{KL}(p \| q) = \sum_{x \in X} p(x) \log \frac{p(x)}{q(x)}
\]

Intuition: when the probability for an event from P is large, but the probability for the same event in Q is small \( \rightarrow \) there is a large divergence. If the probability from P is small and the probability from Q is large \( \rightarrow \) a large divergence, but not as large as the first case.
KL-Divergence

Measure the distance of two distributions

\[ D_{KL}(P \| Q) = \sum_x P(x) \log \frac{P(x)}{Q(x)} \]

KL divergence of two policies

\[ D_{KL}(\pi_1 \| \pi_2)[s] = \sum_{a \in A} \pi_1(a|s) \log \frac{\pi_1(a|s)}{\pi_2(a|s)} \]
Trust Region Policy Optimization (TRPO)

Objective function:

\[ J(\theta) = \mathbb{E}_{s \sim \rho, a \sim \pi_{\theta \text{old}}}[\frac{\pi_{\theta}(a|s)}{\pi_{\theta \text{old}}(a|s)} \hat{A}_{\theta \text{old}}(s, a)] \]

TRPO aims to maximize \( J(\theta) \) subject to, trust region constraint which enforces the distance between old and new policies measured by KL-divergence to be small enough, within a parameter \( \delta \):

\[ \mathbb{E}_{s \sim \rho} \pi_{\theta \text{old}}[D_{KL}(\pi_{\theta \text{old}}(.|s)||\pi_{\theta}(.|s))] \leq \delta \]

In this way, the old and new policies would not diverge too much when this hard constraint is met. While still, TRPO can guarantee a monotonic improvement over policy iteration.
Gradient descent is fast and simple in optimizing an objective function. It picks the steepest direction and then move forward by a step size, but may not work in reinforcement learning.

Trust region determines the maximum step size that we want to explore. Then, locate the optimal point within the trust region and resume the search from there.
Table of Contents

1 Recap: Actor-Critic

2 Importance Sampling

3 Trust region policy optimization (TRPO)

4 Proximal Policy Optimization (PPO)
Given that TRPO is relatively complicated, proximal policy optimization (PPO) simplifies it by using a clipped surrogate objective while retaining similar performance.
Proximal Policy Optimization (PPO)

- Given that TRPO is relatively complicated, proximal policy optimization (PPO) simplifies it by using a clipped surrogate objective while retaining similar performance.

- Let’s denote the probability ratio between old and new policies as:

\[
r(\theta) = \frac{\pi_{\theta}(a|s)}{\pi_{\theta_{old}}(a|s)}
\]
Proximal Policy Optimization (PPO)

- Given that TRPO is relatively complicated, proximal policy optimization (PPO) simplifies it by using a clipped surrogate objective while retaining similar performance.

- Let’s denote the probability ratio between old and new policies as:

  \[ r(\theta) = \frac{\pi_\theta(a|s)}{\pi_{\theta_{\text{old}}}(a|s)} \]

- Then, the objective function of TRPO (on policy) becomes:

  \[ J_{\text{TRPO}}(\theta) = \mathbb{E}[r(\theta)\hat{A}_{\theta_{\text{old}}}(s, a)] \]
Proximal Policy Optimization (PPO)

- Given that TRPO is relatively complicated, proximal policy optimization (PPO) simplifies it by using a clipped surrogate objective while retaining similar performance.

- Let’s denote the probability ratio between old and new policies as:

  \[ r(\theta) = \frac{\pi_\theta(a|s)}{\pi_{\theta_{old}}(a|s)} \]

- Then, the objective function of TRPO (on policy) becomes:

  \[ J^{TRPO}(\theta) = \mathbb{E}[r(\theta)\hat{A}_{\theta_{old}}(s, a)] \]

- Without a limitation on the distance between \( \theta_{old} \) and \( \theta \), it would lead to instability with extremely large parameter updates and big policy ratios.
Proximal Policy Optimization (PPO)

- Given that TRPO is relatively complicated, proximal policy optimization (PPO) simplifies it by using a clipped surrogate objective while retaining similar performance.

- Let’s denote the probability ratio between old and new policies as:

  \[ r(\theta) = \frac{\pi_\theta(a|s)}{\pi_{\theta_{old}}(a|s)} \]

- Then, the objective function of TRPO (on policy) becomes:

  \[ J_{TRPO}(\theta) = \mathbb{E}[r(\theta)\hat{A}_{\theta_{old}}(s, a)] \]

- Without a limitation on the distance between \( \theta_{old} \) and \( \theta \), it would lead to instability with extremely large parameter updates and big policy ratios.

- PPO force \( r(\theta) \) to stay within a small interval around 1 \( \to [1 - \epsilon, 1 + \epsilon] \), where \( \epsilon \) is a hyperparameter.
Proximal Policy Optimization (PPO)

\[ J^{\text{CLIP}}(\theta) = \hat{E}_t \left[ \min \left( r(\theta)\hat{A}_{\theta_{\text{old}}}(s, a), \text{clip}(r(\theta), 1 - \epsilon, 1 + \epsilon)\hat{A}_{\theta_{\text{old}}}(s, a) \right) \right] \]

- \( \theta \) is the policy parameter
- \( \hat{E}_t \) is the empirical expectation over timesteps
- \( r_t \) is the ratio of the probability under the new and old policies
- \( \hat{A}_t \) is the estimated advantage at time \( t \)
- \( \epsilon \) is a hyperparameter, usually 0.1 or 0.2

- Function \( \text{clip}(r(\theta), 1 - \epsilon, 1 + \epsilon) \) clips the ratio within \([1 - \epsilon, 1 + \epsilon]\)

- The objective function of PPO takes the minimum one between the original value and the clipped version

- Thus we lose the motivation for increasing the policy update to extremes for better rewards
Proximal Policy Optimization (PPO)

$$J^{\text{CLIP}}(\theta) = \hat{E}_t \left[ \min \left( r(\theta)\hat{A}_{\theta_{\text{old}}}(s, a), \text{clip}(r(\theta), 1 - \epsilon, 1 + \epsilon)\hat{A}_{\theta_{\text{old}}}(s, a) \right) \right]$$

\(\theta\) is the policy parameter
\(\hat{E}_t\) is the empirical expectation over timesteps
\(r_t\) is the ratio of the probability under the new and old policies
\(\hat{A}_t\) is the estimated advantage at time \(t\)
\(\epsilon\) is a hyperparameter, usually 0.1 or 0.2

- Function \text{clip}(r(\theta), 1 - \epsilon, 1 + \epsilon) clips the ratio within \([1 - \epsilon, 1 + \epsilon]\)
- The objective function of PPO takes the minimum one between the original value and the clipped version
- Thus we lose the motivation for increasing the policy update to extremes for better rewards
Proximal Policy Optimization (PPO)

\[
J_{\text{CLIP}}(\theta) = \mathbb{E}_t \left[ \min \left( r(\theta) \hat{A}_{\text{old}}(s, a), \text{clip}(r(\theta), 1 - \epsilon, 1 + \epsilon) \hat{A}_{\text{old}}(s, a) \right) \right] 
\]

If \( \hat{A}_t > 0 \)
Proximal Policy Optimization (PPO)

\[ J^{\text{CLIP}}(\theta) = \hat{E}_t \left[ \min \left( r(\theta) \hat{A}_{\theta_{\text{old}}}(s, a), \text{clip}(r(\theta), 1 - \epsilon, 1 + \epsilon) \hat{A}_{\theta_{\text{old}}}(s, a) \right) \right] \]

If \( \hat{A}_t > 0 \)

- The action is better than the average of all the actions in that state
Proximal Policy Optimization (PPO)

\[ J_{\text{CLIP}}(\theta) = \hat{E}_t \left[ \min \left( r(\theta)\hat{A}_{\theta_{\text{old}}}(s, a), \text{clip}(r(\theta), 1 - \epsilon, 1 + \epsilon)\hat{A}_{\theta_{\text{old}}}(s, a) \right) \right] \]

If \( \hat{A}_t > 0 \)
- The action is better than the average of all the actions in that state
- Therefore, the action should be encouraged by increasing \( r_t \), so that this action has a higher chance to be adopted
Proximal Policy Optimization (PPO)

\[ J^{\text{CLIP}}(\theta) = \tilde{E}_t \left[ \min \left( r(\theta)\hat{A}_{\theta_{\text{old}}}(s, a), \text{clip}(r(\theta), 1 - \epsilon, 1 + \epsilon)\hat{A}_{\theta_{\text{old}}}(s, a) \right) \right] \]

If \( \hat{A}_t > 0 \)

- The action is better than the average of all the actions in that state
- Therefore, the action should be encouraged by increasing \( r_t \), so that this action has a higher chance to be adopted
- Increasing \( r_t \) implies increasing the new policy \( \pi_\theta(a_t|s_t) \)
Proximal Policy Optimization (PPO)

$$J^{\text{CLIP}}(\theta) = \hat{E}_t \left[ \min \left( r(\theta)\hat{A}_{\theta_{\text{old}}} (s, a), \text{clip}(r(\theta), 1 - \epsilon, 1 + \epsilon)\hat{A}_{\theta_{\text{old}}} (s, a) \right) \right]$$

If $\hat{A}_t > 0$

- The action is better than the average of all the actions in that state
- Therefore, the action should be encouraged by increasing $r_t$, so that this action has a higher chance to be adopted
- Increasing $r_t$ implies increasing the new policy $\pi_\theta(a_t|s_t)$
- Because of the clip, $r_t$ will only grows to as much as $1 + \epsilon$
Loss Function in PPO

\[ L_{\text{CLIP}} \]

For \( A > 0 \):

- The function starts at 0.
- It increases linearly to 1.
- It remains flat at 1 for \( 1 + \epsilon \).

For \( A < 0 \):

- The function starts at 0.
- It remains flat at 1 for \( 1 - \epsilon \).
- It decreases linearly to 0 for \( r \).
Loss Function in PPO

\[ r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \]

\[ L^{CPIP}(\theta) = \hat{E}_t \left[ \min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon)\hat{A}_t) \right] \]

Ratio

Advantage Function

\[ \hat{A}_t = Q(s_t, a_t) - V(s_t) \]

Normal Policy Gradient Objective

Clipped version of Normal Policy Gradient Objective
Algorithm 5 PPO with Clipped Objective

Input: initial policy parameters $\theta_0$, clipping threshold $\epsilon$

for $k = 0, 1, 2, \ldots$ do

Collect set of partial trajectories $D_k$ on policy $\pi_k = \pi(\theta_k)$

Estimate advantages $\hat{A}^\pi_k$ using any advantage estimation algorithm

Compute policy update

$$\theta_{k+1} = \arg \max_{\theta} \mathcal{L}_{\theta_k}^{CLIP} (\theta)$$

by taking $K$ steps of minibatch SGD (via Adam), where

$$\mathcal{L}_{\theta_k}^{CLIP} (\theta) = \mathbb{E}_{\tau \sim \pi_k} \left[ \sum_{t=0}^{T} \min(r_t(\theta)\hat{A}^\pi_k, \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon)\hat{A}^\pi_k) \right]$$

end for

$$A\pi(s, a) = Q\pi(s, a) - V\pi(s), \text{ where}$$

$$a_t \sim \pi(a_t|s_t), s_{t+1} \sim P(s_{t+1}|s_t, a_t), \text{ for } t \geq 0$$
- OpenAI Blog Article
<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Model</th>
<th>Policy</th>
<th>Action Space</th>
<th>Observation Space</th>
<th>Operator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q-Learning</td>
<td>Model-Free</td>
<td>Off-policy</td>
<td>Discrete</td>
<td>Discrete</td>
<td>Q-value</td>
</tr>
<tr>
<td>SARSA</td>
<td>Model-Free</td>
<td>On-policy</td>
<td>Discrete</td>
<td>Discrete</td>
<td>Q-value</td>
</tr>
<tr>
<td>DQN</td>
<td>Model-Free</td>
<td>Off-policy</td>
<td>Discrete</td>
<td>Continous</td>
<td>Q-value</td>
</tr>
<tr>
<td>DDPG</td>
<td>Model-Free</td>
<td>Off-policy</td>
<td>Continous</td>
<td>Continous</td>
<td>Q-value</td>
</tr>
<tr>
<td>TRPO</td>
<td>Model-Free</td>
<td>Off-policy</td>
<td>Continous</td>
<td>Continous</td>
<td>Advantage</td>
</tr>
<tr>
<td>PPO</td>
<td>Model-Free</td>
<td>Off-policy</td>
<td>Continous</td>
<td>Continous</td>
<td>Advantage</td>
</tr>
</tbody>
</table>