Implementation Matters in Deep RL: A Case Study on PPO and TRPO¹²

 $^{^1\,\}mbox{{\sc ''}Implementation}$ Matters in Deep RL: A Case Study on PPO and TRPO" by Engstrom et al., ICLR 2020

²Summary: http://tiny.cc/zh0lnz

► Performance of many RL algorithms depends a lot on tricks used in implementation

- ► Performance of many RL algorithms depends a lot on tricks used in implementation
- ▶ Authors dived into PPO and found out that it is not an exception

- ► Performance of many RL algorithms depends a lot on tricks used in implementation
- ▶ Authors dived into PPO and found out that it is not an exception
- Key observation: tricks used in PPO give more boost than PPO itself

- ► Performance of many RL algorithms depends a lot on tricks used in implementation
- ▶ Authors dived into PPO and found out that it is not an exception
- Key observation: tricks used in PPO give more boost than PPO itself
- ▶ PPO+tricks works a bit better than TRPO+tricks

A common part between TRPO and PPO is the off-policy policy gradient (with importance sampling):

$$J_{\mathsf{PG}}(\theta) = \underset{(s_t, a_t) \sim \pi}{\mathbb{E}} \left[\frac{\pi_{\theta} \left(a_t | s_t \right)}{\pi \left(a_t | s_t \right)} \hat{A}_{\pi} \left(s_t, a_t \right) \right] \tag{1}$$

A common part between TRPO and PPO is the off-policy policy gradient (with importance sampling):

$$J_{\mathsf{PG}}(\theta) = \underset{(s_t, a_t) \sim \pi}{\mathbb{E}} \left[\frac{\pi_{\theta} \left(a_t | s_t \right)}{\pi \left(a_t | s_t \right)} \hat{A}_{\pi} \left(s_t, a_t \right) \right] \tag{1}$$

TRPO differs from PG by constraining the optimization step:

$$\max_{\theta} J_{PG}(\theta)$$
s.t. $D_{KL}(\pi_{\theta}(\cdot|s)||\pi(\cdot|s)) \leq \delta$ (2)

A common part between TRPO and PPO is the off-policy policy gradient (with importance sampling):

$$J_{\mathsf{PG}}(\theta) = \underset{(s_t, a_t) \sim \pi}{\mathbb{E}} \left[\frac{\pi_{\theta} \left(a_t | s_t \right)}{\pi \left(a_t | s_t \right)} \hat{A}_{\pi} \left(s_t, a_t \right) \right] \tag{1}$$

TRPO differs from PG by constraining the optimization step:

$$\max_{\theta} J_{PG}(\theta)$$
s.t. $D_{KL}(\pi_{\theta}(\cdot|s)||\pi(\cdot|s)) \le \delta$ (2)

PPO differs from PG by clipping the ratio inside the objective:

$$J_{\text{PPO}} = \underset{(s_t, a_t) \sim \pi}{\mathbb{E}} \left[\min \left(\text{clip} \left(\rho_t, 1 - \varepsilon, 1 + \varepsilon \right) \hat{A}_{\pi} \left(s_t, a_t \right), \rho_t \hat{A}_{\pi} \left(s_t, a_t \right) \right) \right]$$
(3)

where

$$\rho_t = \frac{\pi_\theta \left(a_t | s_t \right)}{\pi \left(a_t | s_t \right)}. \tag{4}$$

Authors looked at the PPO implementation by OpenAI and found a lot of tricks in it.

Authors looked at the PPO implementation by OpenAl and found a lot of tricks in it. The main ones were:

Authors looked at the PPO implementation by OpenAI and found a lot of tricks in it. The main ones were:

1. Value function clipping (during value function fitting)

Authors looked at the PPO implementation by OpenAI and found a lot of tricks in it. The main ones were:

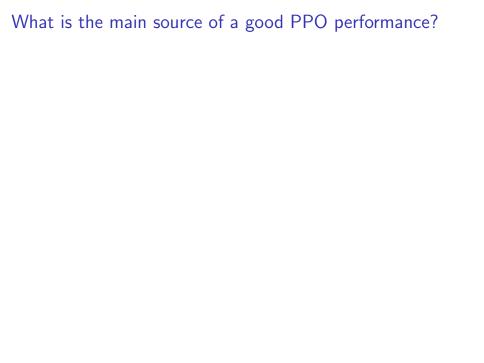
- 1. Value function clipping (during value function fitting)
- 2. Reward scaling

Authors looked at the PPO implementation by OpenAI and found a lot of tricks in it. The main ones were:

- 1. Value function clipping (during value function fitting)
- 2. Reward scaling
- 3. Orthogonal init + layer scaling

Authors looked at the PPO implementation by OpenAI and found a lot of tricks in it. The main ones were:

- 1. Value function clipping (during value function fitting)
- 2. Reward scaling
- 3. Orthogonal init + layer scaling
- 4. Learning rate annealing



What is the main source of a good PPO performance?

Authors ran several experiments for PPO and TRPO with and without tricks and obtained the following results

| WALKER2D-V2 | HOPPER-V2 | HUMANOID-V2 |
|-------------|------------------------------|--|
| 2867 | 2371 | 831 |
| 2735 | 2142 | 674 |
| 2791 | 2043 | 586 |
| 3292 | 2513 | 806 |
| 3050 | 2466 | 1030 |
| | 2867 2735 2791 3292 | 2867 2371 2735 2142 2791 2043 3292 2513 |

What is the main source of a good PPO performance?

Authors ran several experiments for PPO and TRPO with and without tricks and obtained the following results

| | WALKER2D-V2 | HOPPER-V2 | HUMANOID-V2 |
|-----------------|-------------|-----------|-------------|
| PG +tricks | 2867 | 2371 | 831 |
| PG +PPO | 2735 | 2142 | 674 |
| PG +TRPO | 2791 | 2043 | 586 |
| PG +PPO +tricks | 3292 | 2513 | 806 |
| PG +TRPO+tricks | 3050 | 2466 | 1030 |

So,

- tricks improve the performance better than PPO or TRPO
- original paper should have used tricks for TRPO as well