Gaming in 2020: Comparing RL Algorithms

PPO vs Truly PPO

[Kelvin Foster s174210 & Nicolai Weisbjerg s174466]

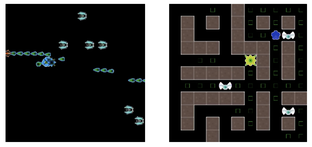
Technical University of Denmark

Why RL for games?

- Decision making in uncertain, complex environments
- Well-defined rewards

Procgen

- OpenAI RL benchmark framework
  - How quickly can a RL agent learn generalizable skills?
- Playing Starpilot and Chaser



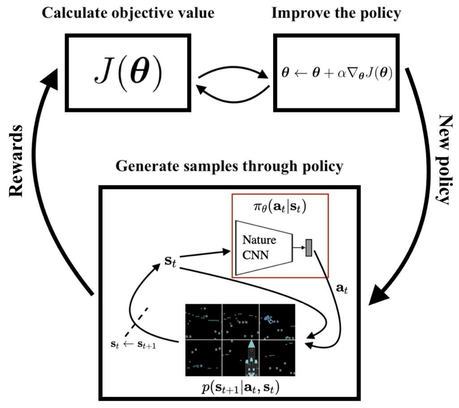
Goal of Study

- Investigate and compare theoretically and empirically:
  - Gold standard: PPO
  - New alternative: Truly PPO
  - The explore/exploit trade-off

Setup

- On-line policy learning approach
- Policy gradient framework with Nature CNN
- Overall objective function:

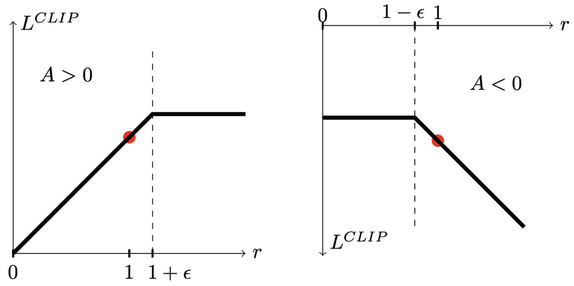
$$J(\boldsymbol{\theta}) = \mathbb{E}_t \left[L_\pi(\boldsymbol{\theta}) - c_L L^{VF}(\boldsymbol{\theta}) + c_H H[\pi_{\boldsymbol{\theta}}](\mathbf{s}_t) \right]$$



PPO: Proximal Policy Optimization

- Seeks to directly determine a policy for the agent
  - Estimating the gradients
- Clipping based on likelihood ratio
- Tries to constrain optimizations of policy on each iteration in order to lower risk of overfitting
- L^{CLIP} is the same as L_π^{PPO} in the figure below

$$L_\pi^{PPO}(\boldsymbol{\theta}) = \hat{\mathrm{E}}_t \left[\min \left(r_t(\pi_{\boldsymbol{\theta}}) \hat{\mathrm{A}}_t, \mathrm{clip}(r_t(\pi_{\boldsymbol{\theta}}), 1-\epsilon, 1+\epsilon) \hat{\mathrm{A}}_t \right) \right]$$

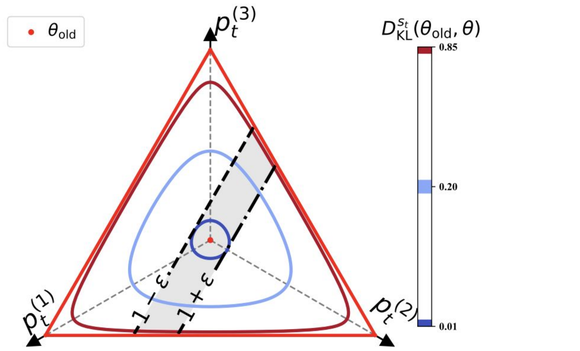


Source: Schulman J., et al.: Proximal Policy Optimization Algorithms (2017)

Truly PPO

- Truly: Trust region and rollback (TR-PPO-RB)
- Trust region: Based directly on KL divergence
- Roll back: Force negative incentives outsides trust region
- More strictly constrained updates of policy than PPO
- Increase in performance stability and sample efficiency

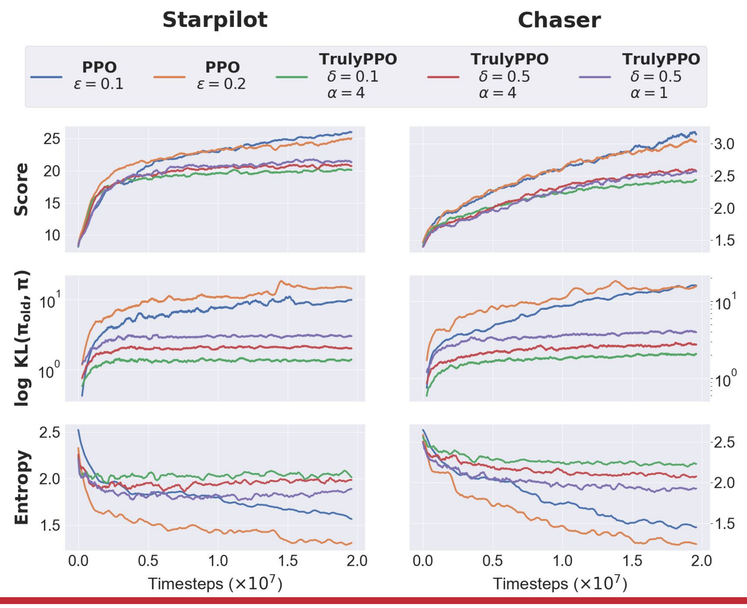
$$L_\pi^{\mathrm{Truly}}(\boldsymbol{\theta}) = r_t(\pi_{\boldsymbol{\theta}})\hat{\mathrm{A}}_t - \begin{cases} \alpha D_{\mathrm{KL}}^{s_t}\left[\pi_{\boldsymbol{\theta}}^{\mathrm{old}}(\mathbf{s}_t), \pi_{\boldsymbol{\theta}}(\mathbf{s}_t)\right] & D_{\mathrm{KL}}^{s_t}\left[\pi_{\boldsymbol{\theta}}^{\mathrm{old}}(\mathbf{s}_t), \pi_{\boldsymbol{\theta}}(\mathbf{s}_t)\right] \geq \delta \text{ and } \\ & r_t(\pi_{\boldsymbol{\theta}})\hat{\mathrm{A}}_t \geq r_t(\pi_{\boldsymbol{\theta}}^{\mathrm{old}})\hat{\mathrm{A}}_t \\ \delta & \text{otherwise} \end{cases}$$



Source: Wang Y., He H., et al.: Truly Proximal Policy Optimization (2019)

Training Results

- Training score: PPO higher than Truly PPO
- KL divergence: Truly PPO ensures constraint (much) better than PPO
- Entropy: Truly PPO ensures higher entropy than PPO
- Graphs based on moving average and mean of 2 seeds for each run
- Score: Mean non-discounted return based on normalized rewards



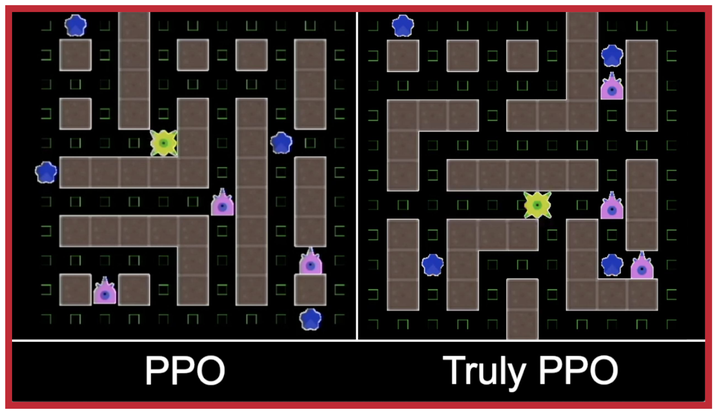
Test Results (Score)

- *Disclaimer*: currently non-normalized rewards and based on longer roll-outs.
- Starpilot: Despite higher training score for PPO, the test score is much closer: generalizability is hard
- Chaser: Truly PPO generalizes better than PPO

	Starpilot	Chaser
PPO $\epsilon = 0.1$	31.93	42.81
PPO $\epsilon = 0.2$	31.67	42.04
Truly PPO $\delta = 0.1$ $\alpha = 4$	28.56	51.36
Truly PPO $\delta = 0.5$ $\alpha = 4$	30.15	46.45
Truly PPO $\delta = 0.5$ $\alpha = 1$	29.69	48.82

Explore/Exploit Trade-off

- Qualitative comparison of algorithms
- Indicates the explore/exploit trade-off



Many Avenues for Future Work!

- Look into generalization as a function of time
  - Normalize test results for direct comparison with training
- Compare PPO and Truly PPO in Actor-Critic framework
- Base graphs on more seeds