Structured Priors for Policy Optimisation

Sihui Wang, Bill Byrne, Tom Gunter

June 19, 2017

UNIVERSITY OF CAMBRIDGE
DEPARTMENT OF ENGINEERING

Motivation

Why policy optimization?

- Greater efficiency and better convergence guarantees
- Able to learn stochastic policies
- Easy to use, value function might complicated

Why structured priors?

- We hope to achieve better sample efficiency by explicitly structuring policy search as a hierarchical problem with options.
- Hierarchically structured policies are suitable for many problems of interest, e.g. dialogue, robotics

Gradient-based Optimization

Policy function $\pi(a|s;\theta)$ is a mapping from $S \times A \rightarrow [0,1]$.

1. Objective function, expected return of π : We can estimate η from sampled data

$$\eta(\pi) = \eta(\pi_{old}) + \mathbb{E}_{s\sim\pi,a\sim\pi}[A_{\pi_{old}}(s,a)]$$
$$= \eta(\pi_{old}) + \mathbb{E}_{s\sim\pi,a\sim\pi_{old}}\Big[\frac{\pi(a|s)}{\pi_{old}(a|s)}A_{\pi_{old}}(s,a)\Big]$$

However the state depends on the new policy parameter which makes it difficult to optimize.

2. Local approximation, $L(\pi)$:

$$L_{\pi_{old}}(\pi) = \eta(\pi_{old}) + \mathbb{E}_{s,a\sim\pi_{old}}\Big[\frac{\pi(a|s)}{\pi_{old}(a|s)}A_{\pi_{old}}(s,a)\Big]$$

L matches η to first order

$$\nabla_\theta L_{\pi_{\theta_{old}}}(\pi_\theta)|_{\theta=\theta_{old}} = \nabla_\theta \eta(\pi_\theta)|_{\theta=\theta_{old}}$$

3. Monotonic improvement by updating surrogate function M[1]:

$$\eta(\pi) \geq L_{\pi_{old}}(\pi) - CD_{KL}^{max}(\pi_{old},\pi) = M_{\pi_{old}}(\pi)$$

where $C = \frac{2\epsilon\gamma}{(1-\gamma)^2}$, $\epsilon = \max_s |\mathbb{E}_{a\sim\pi(a|s)}[A_{\pi_{old}}(s,a)]|$

Minorization-Maximazation algorithm:

$$\eta(\pi) - \eta(\pi_{old}) \geq M_{\pi_{old}}(\pi) - M_{\pi_{old}}(\pi_{old})$$

4. Reinforcement learning problem to optimization problem:

$$\max_\theta [L_{\theta_{old}}(\theta) - CD_{KL}^{max}(\theta_{old},\theta)]$$

Practical Approximation: TRPO

- Use a hard constrain on KL divergence to allow large update

$$\max_\theta L_{\theta_{old}}(\theta) \quad \text{subject to } D_{KL}^{max}(\theta_{old},\theta) \leq \delta$$

- Impractical due to large number of constraint, use average KL instead.

$$\max_\theta \mathbb{E}_{s,a\sim\pi_{old}}\Big[\frac{\pi(a|s)}{\pi_{old}(a|s)}A_{\pi_{old}}(s,a)\Big] \quad \textit{subject to } \mathbb{E}_{s\sim\pi_{old}}[D_{KL}(\pi_{\theta_{old}}(\cdot|s)||\pi_\theta(\cdot|s))] \leq \delta \tag{1}$$

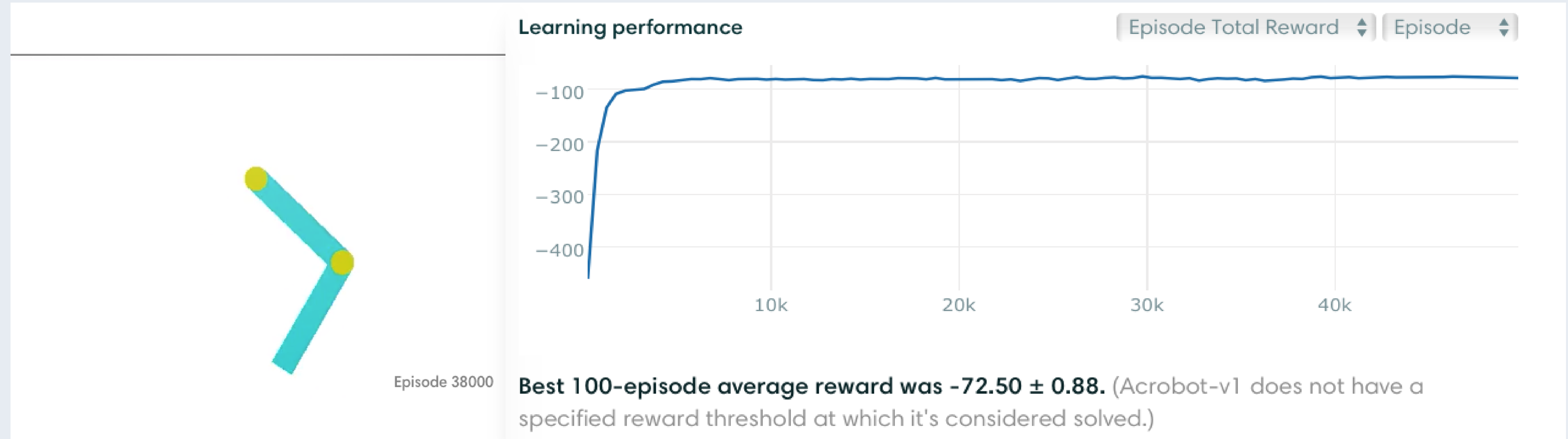
Solving the new approximated problem

1. Linear approximation to $L_{\pi_{old}}$ and quadratic approximation to average KL divergence. Search direction $= A^{-1}g$, where A is the Hessian matrix of average KL and g is the gradient of L.
2. Use conjugate gradient method to calculate search direction
3. Perform line search to ensure objective improves

Results from OpenAI Gym

1. Continuous state space and discrete action: $\pi(\cdot|s) = softmax(w^Ts+b)$
CartPole-v1: Balance a pole on a cart [State dim: 4, No. of actions: 2]

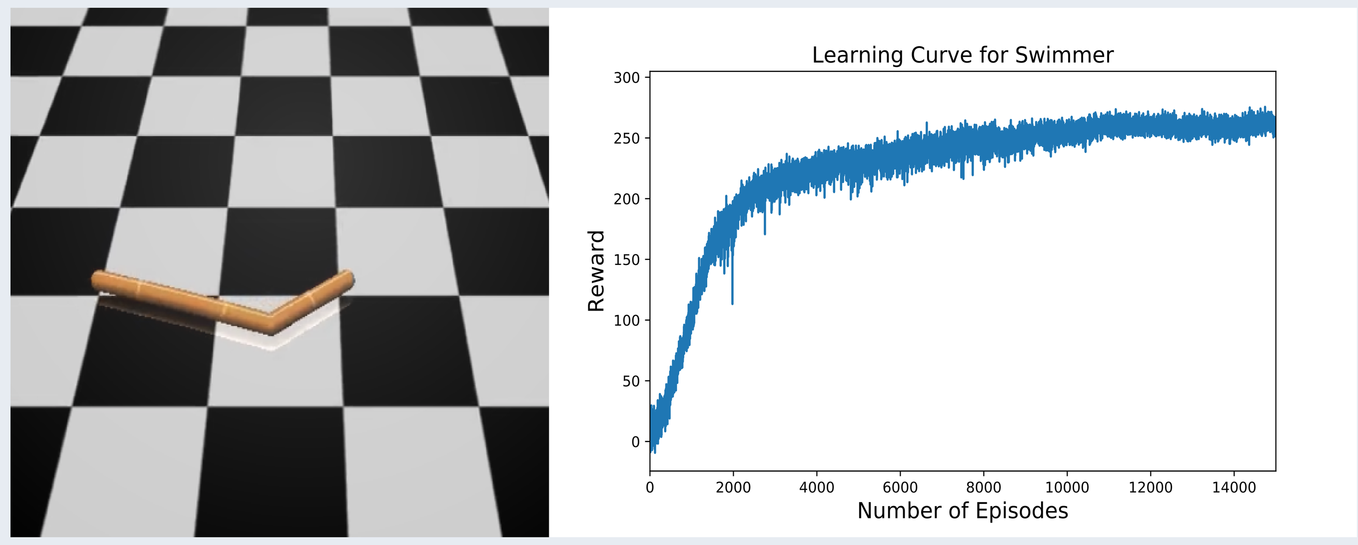
Acrobot-v1: Swing up a two link robot. [State dim: 6, No. of actions: 3]



2. Continuous state space and continuous action space:

$$\pi(a|s) \sim N(mean = NeuralNet(s;\{\mathbf{W},\mathbf{b}\}), std = exp(w_{std}))$$

Swimmer: swim forward as fast as possible. [State dim: 8, action dim: 2]
Half-cheetah: Make a 2D cheetah robot run. [State dim: 17, action dim: 6]



Structured TRPO

The idea of hierarchical policy is introduced in [2]. A hierarchical policy $\pi(a|s)$ consists of a set of several sub-policies and a gating network $\pi(o|s)$.

$$\pi(a|s) = \sum_o \pi(o|s)\pi(a|s,o) \tag{2}$$

Algorithm 1 Vanilla Hierarchical TRPO

1: Initialize both gating network and sub-policy parameters
2: **for** $i = 1 : N$ **do**
3: Reset the environment and sample $o \sim \pi(o|s)$
4: **for** $j = 1 : M$ **do**
5: Follow $a \sim \pi(a|s,o)$
6: If done: reset environment and start a new episode
7: **end for**
8: Using TRPO to update parameters of a particular sub-policy

$$\max_\theta \mathbb{E}_{s,a\sim\pi_{old}}\Big[\frac{\pi(a|o,s)}{\pi_{old}(a|o,s)}A_{\pi_{old}}(s,a)\Big] \tag{3}$$

subject to $\overline{D_{KL}}(\pi_{\theta_{old}}(\cdot|o,s)||\pi_\theta(\cdot|o,s)) \leq \delta_{sub-policy}$

9: **end for**
10: Using TRPO to update gating network parameters

$$\max_\phi \mathbb{E}_{s,a\sim\pi_{old}}\Big[\frac{\pi(o|s)}{\pi_{old}(o|s)}A_{\pi_{old}}(s,a)\Big] \tag{4}$$

subject to $\overline{D_{KL}}(\pi_{\phi_{old}}(\cdot|s)||\pi_\phi(\cdot|s)) \leq \delta_{option}$

11: Go back to step 2

Future Work

- Learn to use different option within a single episode.
- Reduce the variance in policy gradient method.
- Consider using boosting methods to include multiple sub-policies.

References

[1] Schulman, John, Levine, Sergey, Moritz, Philipp, Jordan, Michael I, and Abbeel, Pieter. *Trust region policy optimization. arXiv preprint arXiv:1502.05477*, (2015)

[2] Daniel, Christian, Neumann, Gerhard, Kroemer, Oliver, Peters, Jan. *Hierarchical Relative Entropy Policy Search. Journal of Machine Learning Research*, pp.1-50 (2016)