Reinforcement Learning for Robotic Locomotions

Bo Liu, Huanzhong Xu, Songze Li
{bliuxix, xuhuanvc, songzeli}@stanford.edu

Introduction & Motivation

In an environment with high cost evaluation, policy optimization could be a good choice since the possible policy space can be significantly smaller than the state action space. In this project, we investigate variants of Trust Region Policy Optimization algorithm in environments related with human motor control.

Data & Features

Our data comes from Mujuco, a physics engine from openAI gym
We experiment on 3 specific environment:

- *Swimmer*: 10-dimensional state space. Linear Reward.
- *Hopper*: 12-dimensional state space. Same reward as *Swimmer*, with a positive bonus for being in non-terminal state.
- *Walker*: 18-dimensional state space. To separate walker from Hopper, we added a penalty for strong impacts of the feet against the ground

Method & Model

- **The base for all models: *Policy Gradient Method***
  Policy Gradient Method estimates the policy by approximating the derivative of the expected future rewards. The estimator is of the form:
  $$\hat{g} = \hat{\mathbb{E}}_t\left[\nabla_\theta \log \pi_\theta(a_t \mid s_t)\hat{A}_t\right]$$
- ***True Region Policy Optimization(TRPO)***
  The theory of TRPO is based on an important identity (K & L(2002)) that describes the relationship between policies:
  $$\eta(\tilde{\pi}) = \eta(\pi) + \sum_s \rho_{\tilde{\pi}}(s) \sum_a \tilde{\pi}(a|s) A_\pi(s,a)$$
  The η function here denote the expected reward of policy π. However, since $\rho_{\tilde{\pi}}(s)$ ard to estimate, we use the state visitation density of the old policy instead.
  $$L_\pi(\tilde{\pi}) = \eta(\pi) + \sum_s \rho_\pi(s) \sum_a \tilde{\pi}(a|s) A_\pi(s,a)$$
  If policy π is parameterized by θ where π_θ is differentiable, then
  $$\nabla_\theta L_{\pi_{\theta_0}}(\pi_\theta)\big|_{\theta=\theta_0} = \nabla_\theta \eta(\pi_\theta)\big|_{\theta=\theta_0}$$
  so a sufficiently small step in π that improves L_π will also improve η. But how to quantify a "small step" is the key part of the problem. TRPO suggests using KL divergence, with the lower bound:
  $$\eta(\tilde{\pi}) \geq L_\pi(\tilde{\pi}) - C D_{\mathrm{KL}}^{\max}(\pi, \tilde{\pi})$$
  where $D_{\mathrm{KL}}^{\max}(\pi,\tilde{\pi}) = \max_s D_{\mathrm{KL}}(\pi(\cdot|s) \parallel \tilde{\pi}(\cdot|s))$
  So now our final theoretical objective function is
  $$\underset{\theta}{\text{maximize}}\,[L_{\theta_{\text{old}}}(\theta) - C D_{\mathrm{KL}}^{\max}(\theta_{\text{old}}, \theta)]$$
  with some penalty coefficient *C*.

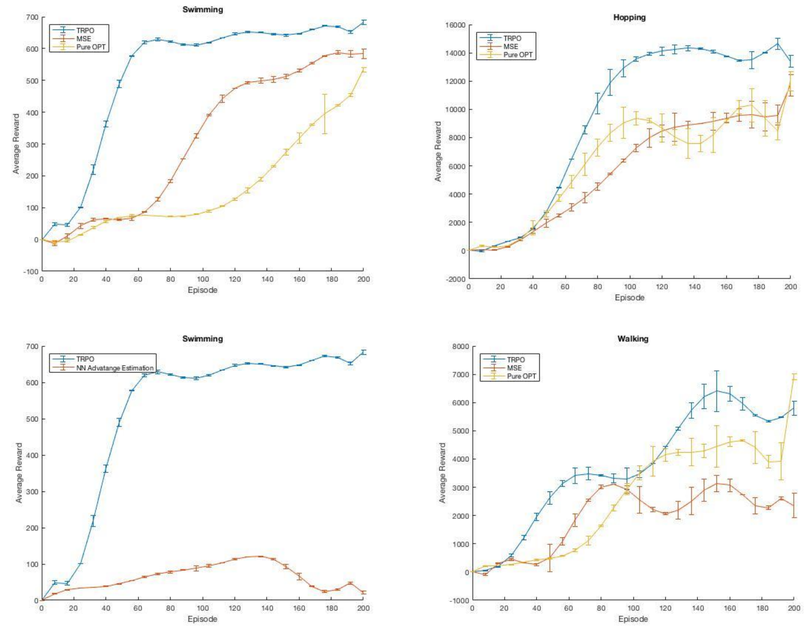
Instead of including KL divergence in optimization objective directly, TRPO suggests using a "trust region", so the problem becomes:

$$\underset{\theta}{\text{maximize}}\, L_{\theta_{\text{old}}}(\theta)$$
$$\text{subject to } \overline{D}_{\mathrm{KL}}^{\rho_{\theta_{\text{old}}}}(\theta_{\text{old}}, \theta) \leq \delta$$

- ***Monte-Carlo Simulation***
  To approximate $L_\theta(\theta)$, we may replace $\sum_s \rho_{\theta_{\text{old}}}(s)[\dots]$ by $\frac{1}{1-\gamma}\mathbb{E}_{s\sim\rho_{\theta_{\text{old}}}}[\dots]$, and replace sum over actions via sampling:
  $$\underset{\theta}{\text{maximize}}\, \mathbb{E}_{s\sim\rho_{\theta_{\text{old}}}, a\sim q}\left[\frac{\pi_\theta(a|s)}{q(a|s)} Q_{\theta_{\text{old}}}(s,a)\right]$$
  $$\text{subject to } \mathbb{E}_{s\sim\rho_{\theta_{\text{old}}}}\left[D_{\mathrm{KL}}(\pi_{\theta_{\text{old}}}(\cdot|s) \parallel \pi_\theta(\cdot|s))\right] \leq \delta$$
- ***Model for TRPO***
  For TRPO, we used neural networks for both policy model and value model. The policy's action distribution is normal, so the neural net output is mean and standard deviation of action. The value network will take state observation as input and create a single value output.
- ***Modifications on constraints***
  Since TRPO is a constraint optimization problem, our first thought is replacing the KL constraint by some other constraints that also measure policy similarity. A natural thought would be using **MSE loss** on θ. We noticed later that this in fact corresponds to the standard policy gradient update. We have also tried to directly optimize the objective without any constraint.
- ***Neural Net Advantage Estimation and Direct Improvement***
  Back to the original equation:
  $$\eta(\tilde{\pi}) = \eta(\pi) + \sum_s \rho_{\tilde{\pi}}(s) \sum_a \tilde{\pi}(a|s) A_\pi(s,a)$$
  We actually want to make the sum on right hand side positive. Intuitively, we would like to increase the policy's probability of actions that have higher advantage value. Follow this, we use a 3-layer neural net to estimate advantage for state observation and mean action. Then we calculate the difference with real advantage. If the difference is negative, we maximize the corresponding log-probability of action by doing a line search along the gradient.

Result and Analysis

We implemented the TRPO algorithm with estimated KL divergence loss, MSE loss and no constraint. We also implemented our variant algorithm using neural net advantage estimation. We evaluated the models on 3 different Mujuco environments and the results are shown below. All plots are rewards versus number of training epochs.



Discussion

As we can see, TRPO with KL divergence constraint outperforms its variants with different constraints. However, our advantage estimation method is not working properly. We spent a lot of time debugging. One thing we notice is that the algorithm tends to have large MSE loss for advantage estimation after some consecutive epochs of improvement. Therefore, we think this gives enough evidence that as long as the advantage estimation is accurate, our algorithm should work. We conjecture the unexpected drop might result from the overfitting of our advantge neural net. So whenever it sees a new observation and action that differ from previous experience, it cannot generalize well and thus updating the policy in a wrong direction. In fact, by carefully tuning l2-regularization constant and adding dropout, the model indeed improves. We will include the figure for improved algorithm in final report.

The TRPO algorithm is actually more complicated than we have expected. We've spent much time learning the theory, including mathematics like conjugate gradient methods.

Future Work

We may continue our investigation of measurements to replace KL divergence constraint. In addition, we would also like to do more error analysis on our advantage estimation method.

References

1. J. Schulman, S. Levine, P. Moritz, M. I. Jordan, and P. Abbeel. "Trust region policy optimization". In: CoRR, abs/1502.05477 (2015).
2. Kakade, Sham and Langford, John. Approximately optimal approximate reinforcement learning. In ICML, volume 2, pp. 267–274, 2002.