Paper Critique

Anirud N, CE21B014

Course: DA7400, Fall 2024, IITM
Paper: Constrained Policy Optimization
Date: 11 September 2024

Make sure your critique addresses the following points:

1. The problem the paper is trying to address
2. Key contributions of the paper
3. Proposed algorithm/framework
4. How the proposed algorithm addressed the described problem

Note: Be concise with your explanations. Unnecessary verbosity will be penalized. Please don't exceed 2 pages.

1 Problem Statement

Reinforcement Learning Algorithms application in real-life environments requires the imposing of constraints to learn along with the reward functions. These constraints can also be used to govern safety during exploration. Policy search algorithms have enhanced high dimensional control capabilities but do not consider the constrained setting. Current deep RL assumes that the agent can explore any behavior during learning. This may not always be possible in realistic scenarios. Methods for high dimensional control are lacking. This paper aims to train policies for high dimensional control while also implementing constrained policy optimization.

2 Key Contributions

- Allows to train neural policies for high dimensional control
- Makes theoretical background and guarantees about policy behavior
- Demonstrate the effectiveness of the algorithm by implementing it on robot locomotion tasks and ensure that safety constraints are imposed on the agent
- This approach bounds the difference in rewards and costs between two different policies.
- This helps to create a better bound for the constraints, and this is used to derive a policy improvement step
- Implements an approximation of the theoretically justified update

3 Proposed Algorithm/Framework

3.1 Constrained Markov Decision Process

The set of feasible policies of a CMDP is:

$$\Pi_C \doteq \{\pi \in \Pi \ : \ \forall i, J_{C_i}(\pi) \leq d_i\},$$

The reinforcement learning problem in a CMDP is

$$\pi^* = \arg\max_{\pi \in \Pi_C} J(\pi).$$

3.2 Constrained policy optimisation

$$\begin{aligned}
\pi_{k+1} = \arg\max_{\pi\in\Pi_\theta}\; & J(\pi) \\
\text{s.t. }\; & J_{C_i}(\pi) \leq d_i \quad i = 1, ..., m \\
& D(\pi, \pi_k) \leq \delta.
\end{aligned} \tag{1}$$

but implementing this is difficult, as it requires constrained functions each and every time to check whether the proposed policy is feasible. This also requires an off-policy evaluation. So this paper implements a surrogate approximation for the same, replacing the objective with surrogates. The paper also involves theoretical proofs to bound the difference between two arbitrary policies in this method.

3.3 Trust Regions

$$\begin{aligned}
\pi_{k+1} = \arg\max_{\pi\in\Pi_\theta} \mathop{}_{\substack{s\sim d^{\pi_k}\\ a\sim\pi}} & [A^{\pi_k}(s, a)] \\
\text{s.t. }\; & \bar{D}_{KL}(\pi||\pi_k) \leq \delta,
\end{aligned} \tag{2}$$

The constrain ensures the trust regions. it is proved to provide monotonic improvements. For a constrained optimisation problem, trust regions are implemented as:

$$\begin{aligned}
\pi_{k+1} = & \arg\max_{\pi\in\Pi_\theta}\; \pi_k \pi A^{\pi_k}(s, a) \\
\text{s.t. }\; & J_{C_i}(\pi_k) + \pi_k \pi A^{\pi_k}_{C_i}(s, a) \leq d_i \quad \forall i \\
& \bar{D}_{KL}(\pi||\pi_k) \leq \delta.
\end{aligned} \tag{3}$$

Note that trust regions are used instead of penalties on policy divergence to enable larger step sizes

3.4 Approximations made for solving the CPO

TO enable implementation in higher dimensional spaces - approximations have been made for faster computation. These approximations made may not always guarantee that the policy to be implement always satisfies the constraint. It gives a faster method to go closer to the policy. So, in order to take this uncertainty into account, i.e., if CPO is an infeasible update of policy, then the algorithm implements a policy recovery update.

$$\theta^* = \theta_k - \sqrt{\frac{2\delta}{b^T H^{-1} b}} H^{-1} b. \tag{4}$$

4 Conclusions and Results

This paper provides the first step towards constrained policy optimization that is guaranteed to both improve and satisfy the safety constraints. This method shows that CPO can train neural networks with thousands of parameters on high-dimensional control tasks. This provides a method to apply RL to real-world scenarios. Comparing CPO to fixed penalty approaches revealed that fixed penalty approaches can be sensitive to the choice of the penalty coefficient. However, CPO automatically pics penalty coefficients to attain the proper trade-off between constraining and reward maximizing. CPO also outperforms PDO in enforcing constraints