Proximal Policy Optimization with Dynamic Clipping

Student: Rishikesh Vaishnav
University of California, San Diego

Mentor: Sicun Gao, Ph.D
University of California, San Diego

Abstract

Proximal Policy Optimization (PPO), a policy gradient algorithm drawing closely from the theory supporting Trust Region Policy Optimization (TRPO), has emerged as one of the most effective tools in reinforcement learning (RL) problems. PPO makes use of a loss function with a clipped importance sampling ratio using a single parameter ϵ. Although PPO shows promising empirical performance, it is vulnerable to problem-specific imbalances in its handling of positive and negative advantages. We investigate one such imbalance, addressing a discrepancy in expected penalty contributions of positive and negative estimators. By precisely calculating this discrepancy and minimizing it before each model update, we empirically demonstrate that eliminating this discrepancy can improve the overall performance of PPO.

1 Introduction

Note: This paper assumes knowledge of the common terms and concepts in reinforcement learning (see [TODO cite sutton and barto]).

Among current reinforcement learning (RL) algorithms, Policy Gradient methods have seen significant success at a wide range of tasks. These methods seek to learn performant policy distributions directly, rather than learning a value function to indirectly guide the policy [TODO cite paper covering policy gradient]. Using a policy $\pi_\theta(a)$ parameterized by θ, these algorithms have the general form:

Algorithm 1 Generic Policy Gradient

```
Initialize θ arbitrarily
while True do                                  ▷ loop forever
    θ_old ← θ
    rollout ← (s, a, r) from multiple
        episodes following π_θ
    Set θ to maximize the loss function
        L(rollout, θ, θ_old)
end while
```

Within this framework, the choice of a loss function is the key to an effective algorithm, and current research in reinforcement learning focuses particularly on finding new ways to represent this loss. A recent innovation in policy gradient methods is Trust Region Policy Optimization (TRPO) [TODO cite TRPO], which approximates the updates perfomed by an agent that uses the following loss function to guarantee monotonic improvement in the policy's performance:

$$L_{\theta_{old}}(\theta) - CD_{KL}^{max}(\theta, \theta_{old}) \tag{1}$$

where C is a constant,

$$L_{\theta_{old}}(\theta) = \mathbb{E}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} A_t \right] \tag{2}$$

and A_t is the advantage at time t. In TRPO implementations, this loss function results in relatively small step sizes, so in practice the penalty term $CD_{KL}^{max}(\theta, \theta_{old})$ is removed from equation (1), and we instead maximize (2) with a constraint on the KL divergence.

Implementations of TRPO generally have significant computational complexity relative to other RL algorithms, largely due to the need to calculate KL divergences and perform constraint optimization. To address this, a new algorithm based on the theory behind TRPO, Proximal Policy Optimization (PPO) [TODO cite PPO], was developed.

Rather than maximizing (2) subject to a constraint, PPO maximizes the following "clipped" loss function:

$$L^{CLIP}(\theta) = \mathbb{E}_t \left[\min \left(r_t A_t, \text{clip}(r_t, 1-\epsilon, 1+\epsilon) A_t \right) \right] \tag{3}$$

where $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$.

This loss function can be maximized using standard optimization techniques and does not require constraint optimization, significantly simplifying implementation relative to TRPO. This loss function can be thought of as a heuristic approximation to the loss function from (1), where the minimization substitutes the penalty term.

PPO performs well in practice, in many cases outperforming TRPO. However, its simple equation does not allow for precise control over what penalties are actually introduced, and, as we will show, can exhibit problem-specific imbalances in the way positive and negative advantages are handled.

2 Expected Penalty Contributions

(3) can be split into positive and negative advantage cases as follows:

$$L^{CLIP}(\theta) = \mathbb{E}_t \left[\begin{cases} \min\left(r_t, \text{clip}(r_t, 1+\epsilon)\right) A_t & A_t > 0 \\ \max\left(r_t, \text{clip}(r_t, 1-\epsilon)\right) A_t & A_t < 0 \end{cases} \right] \quad (4)$$

Now, let

$$r^+_{t,CLIP} = \min\left(r_t, \text{clip}(r_t, 1+\epsilon)\right)$$
$$r^-_{t,CLIP} = \max\left(r_t, \text{clip}(r_t, 1-\epsilon)\right)$$

It follows mathematically that $\mathbb{E}_t\left[r_t\right] = 1$, because ratios are sampled according to the old policy distribution. Therefore, it must be the case that $\mathbb{E}_t\left[r^+_{t,CLIP}\right] < 1$ and $\mathbb{E}_t\left[r^-_{t,CLIP}\right] > 1$. As the new policy changes relative to the old policy, clipping becomes more frequent, generally causing these expectations to become smaller and larger, respectively. Assuming independence between ratios and advantages, the effect of this is to make positive advantages less positive and make negative advatages more negative. This is a reflection of how penalization occurs when a new policy is learned.

We can define the "expected penalty contributions" $1 - \mathbb{E}_t[r^+_{t,CLIP}]$ and $\mathbb{E}_t[r^-_{t,CLIP}] - 1$.

In practice, the assumption of independence between ratios and advantages is incorrect, because if these were independent, we would expect to see a monotonically decreasing loss at each model update. In reality, however, an optimizer will specifically work around this by assigning ratios to advantages such that the loss tends to increase at each model update. Therefore, these expected ratios do not suggest the actual proportional contributions of positive and negative advantages to the overall loss.

However, it seems likely that these expectations have a significant influence on how positive and negative advatages are considered in calculating the overall loss. Addressing these expectations and controlling their relative values will affect the loss that is calculated, and could have an influence on overall performance.

Notes

[1] Remember to use endnotes, not footnotes!

References

[1] https://arxiv.org/abs/1707.06347

[2] https://arxiv.org/abs/1506.02438

[3] https://arxiv.org/abs/1502.05477