Week 9 Precept Notes: Proximal policy gradient

ECE 433/COS 435

April 17, 2024

1 Natural Policy Gradient

The motivation for Natural Policy Gradient (NPG) comes from constrained optimization. We want to improve the training stability of the policy by limiting the change you make to the policy at each training epoch. There are two reasons:

(i) We know empirically that smaller policy updates during training are more likely to converge to an optimal solution. We know empirically that smaller policy updates during training are more likely to converge to an optimal solution.

(ii) A too-big step in a policy update can result in falling "off the cliff" (getting a bad policy) and having a long time or even no possibility of recovering.

We consider the following optimization:

$$\theta^{t+1} = \arg\max_\theta \mathbb{E}_{\pi_\theta}\left[\sum_{h\geq 0}\gamma^h r(s_h, a_h)\right] \qquad \text{s.t.} \quad D_{\mathrm{KL}}(\pi_{\theta^t}|\pi_\theta) \leq \varepsilon,$$

where ε is a constant and an additional constraint is imposed so that only a more "conservative" update is allowed. Note that by Lagrangian duality, we can reform the previous constrained optimization into an unconstrained one:

$$\begin{aligned}\theta^{t+1} &= \arg\max_\theta \mathbb{E}_{\pi_\theta}\left[\sum_{h\geq 0}\gamma^h r(s_h, a_h)\right] - \lambda\Big(D_{\mathrm{KL}}(\pi_{\theta^t}|\pi_{\theta'}) - \varepsilon\Big)\\ &\approx \arg\max_\theta \mathbb{E}_{\pi_{\theta^t}}\left[\sum_{h\geq 0}\gamma^h r(s_h, a_h)\right] + \nabla_\theta \mathbb{E}_{\pi_{\theta^t}}\left[\sum_{h\geq 0}\gamma^h r(s_h, a_h)\right]\cdot(\theta - \theta^t)\\ &\quad - \frac{\lambda}{2}(\theta-\theta^t)^\top \nabla_\theta^2 D_{\mathrm{KL}}(\pi_{\theta^t}(s)|\pi_\theta(s))(\theta-\theta^t),\end{aligned}$$

the second line comes from Taylor's expansion. Moreover, by KKT condition, we also have $D_{\mathrm{KL}}(\pi_\theta^{t+1}|\pi_{\theta^t}) = \varepsilon$. Recall that

$$D_{\mathrm{KL}}(\pi_{\theta_t}|\pi_\theta) = \mathbb{E}_{\pi_{\theta^t}}\left[\log\left(\frac{\pi_{\theta_t}(a|s)}{\pi(a|s)}\right)\right] = \int_{\mathcal{A}} \log\left(\frac{\pi_{\theta_t}(a|s)}{\pi_\theta(a|s)}\right)\pi_{\theta^t}(a|s)p_{\theta^t}(s)\mathrm{d}s\mathrm{d}a,$$

it's easy to show that

$$\nabla_\theta D_{\mathrm{KL}}(\pi_{\theta^t}(s)|\pi_\theta(s)) = 0,$$

and

$$F(\theta) := \nabla_\theta^2 D_{\mathrm{KL}}(\pi_{\theta^t}|\pi_\theta) = -\nabla_\theta^2 \mathbb{E}_{\pi_{\theta^t}}[\log(\pi_\theta(a|s))]$$
$$= \mathbb{E}_{\pi_{\theta^t}}[\nabla_\theta \log \pi_{\theta_t}(a|s) \nabla_\theta \log \pi_{\theta_t}(a|s)^\top]$$

The quadratic function above suggests an easy-to-compute update rule given by

$$\theta_{t+1} = \theta_t + \frac{2}{\lambda} F(\theta^t)^{-1} \nabla_\theta \mathbb{E}_{\pi_{\theta_t}}[\sum_{h \geq 0} \gamma^h r(s_h, a_h)] = \theta_t + \alpha g_t.$$

Here $\alpha = \frac{2}{\lambda}$ and $g_t = \nabla_\theta \mathbb{E}_{\pi_{\theta_t}}[\sum_{h \geq 0} \gamma^h r(s_h, a_h)]$. Note that NPG is essentially a weighted version of the policy gradient method, where the Fisher information matrix $F(\theta)$ twists the direction of the policy gradient. In practice, we use the empirical estimate
$\frac{1}{H} \sum_{h=1}^H \nabla_\theta \log \pi_{\theta_t}(a_h|s_h) \nabla_\theta \log \pi_{\theta_t}(a_h|s_h)^\top$ to estimate $F(\theta)$, and $\frac{1}{H} \sum_{h=1}^H \nabla_\theta \log \pi_{\theta_t}(a_h|s_h) \sum_{k \geq h} \gamma^{k-h} r(s_k, a_k)$ for g_t. The only thing we need to finalize is the stepsize α. Although in practice we can determine λ by hyperparameter tuning, we introduce an analytic form here. Note that

$$\varepsilon = D_{\mathrm{KL}}(\pi_\theta^t|\pi_\theta) \approx \frac{1}{2}(\alpha g_t)^\top F(\theta_t)(\alpha g_t),$$

solving this and we have $\alpha = \left(\frac{2\varepsilon}{g_t^\top F(\theta_t) g_t}\right)^{1/2}$.

Algorithm 1 Natural Policy Gradient

```
1: Input: initialize policy parameters $\theta_0$, $\varepsilon$, $t = 0$
2: **repeat**
3:     Collect set of trajectories $\mathcal{D}_t$ on policy $\pi_\theta$
4:     Form sample estimates $\hat{g}_t$, $\hat{F}(\theta_t)$ and $\alpha_t$
5:     Compute Natural Policy Gradient update:
           $$\theta_{t+1} = \theta_t - \alpha_t \hat{F}(\theta_t)^{-1} \hat{g}_t$$
6:     $t = t + 1$
7: **until** convergence
```

2 Trust Region Policy Optimization (TRPO)

In TRPO, an objective function (the "surrogate" objective) is maximized subject to a constraint on the size of the policy update. Specifically, for every iteration t, we consider

$$\theta^{t+1} = \arg\max_\theta \mathbb{E}_{\pi_{\theta^t}} \left[\frac{\pi(a|s)}{\pi_{\theta^t}(a|s)} \hat{A}^{\pi_{\theta^t}}(s, a) \right] \qquad \text{s.t.} \quad D_{\mathrm{KL}}(\pi_{\theta^t}|\pi_\theta) \leq \varepsilon,$$

which is different from NPG as it gives a more "local" update of the policy by using the advantage function to update the policy (instead of a direct policy gradient). This allows it to be incorporated into off-policy methods, as we can update the advantage function $\hat{A}^{\pi_\theta}$ with all samples in a replay buffer, instead of only the trajectories sampled by the current policy. As shown by the previous section, such an objective can be approximately solved by gradient-based methods, after making a linear approximation to the objective and a quadratic approximation to the constraint, or using line search.

Similar to NPG, the theory justifying TRPO actually suggests using a penalty instead of a constraint, i.e., solving the following unconstrained optimization problem:

$$\max_\theta \mathbb{E}_{\pi_{\theta^t}} \left[\frac{\pi_\theta(a|s)}{\pi_{\theta^t}(a|s)} \hat{A}^{\pi_{\theta^t}}(s,a) - \lambda_t \cdot D_{\mathrm{KL}}(\pi_{\theta^t}(s)|\pi_\theta(s)) \right].$$

A notable drawback of TRPO is it is hard to choose a single value of λ_t that performs well across different problems—or even within a single problem, where the characteristics change over the course of learning.

3 Proximal Policy Optimization

3.1 Clipped Surrogate Objective

The idea with Proximal Policy Optimization (PPO) is identical to NPG and TRPO: we want to improve the training stability of the policy by limiting the change you make to the policy at each training epoch: we want to avoid having too large policy updates. So with PPO, we update the policy conservatively. To do so, we need to measure how much the current policy changed compared to the former one using a ratio calculation between the current and former policy. And we clip this ratio in a range $[1-\varepsilon, 1+\varepsilon]$, meaning that we remove the incentive for the current policy to go too far from the old one (hence the proximal policy term).

The main objective we propose is the following:

$$\theta_{t+1} = \arg\max_\theta \mathbb{E}_{\pi_{\theta^t}} \left[\min \left\{ \frac{\pi_\theta(a|s)}{\pi_{\theta_t}(a|s)} \hat{A}^{\pi_{\theta_t}}(s,a), \mathrm{clip}_{[1-\varepsilon,1+\varepsilon]} \left(\frac{\pi_\theta(a|s)}{\pi_{\theta_t}(a|s)} \right) \hat{A}^{\pi_{\theta_t}} \right\} \right]$$

where epsilon is a hyperparameter, say, $\varepsilon = 0.2$. We can estimate the advantage function $\hat{A}^{\pi_{\theta^t}}$ by an additional critic network. The motivation for this objective is as follows:

(i) The first term in the minimization is the objective in TRPO.

(ii) The second term, $\mathrm{clip}(\frac{\pi_\theta(a|s)}{\pi_{\theta_t}(a|s)}, 1-\varepsilon, 1+\varepsilon)\hat{A}^t$, modifies the surrogate objective by clipping the probability ratio, which removes the incentive for moving the policy ratio outside of the

(iii) We take the minimum of the clipped and unclipped objective, so the final objective is a lower bound (i.e., a pessimistic bound) on the unclipped objective. With this scheme, we only ignore the change in probability ratio when it would make the objective improve, and we include it when it makes the objective worse. interval $[1-\varepsilon, 1+\varepsilon]$.

3.2 Adaptive KL Penalty Coefficient

Another approach, which can be used as an alternative to the clipped surrogate objective, or in addition to it, is to use a penalty on KL divergence and to adapt the penalty coefficient so that we achieve some target value of the KL divergence d_{targ} each policy update.

In the simplest instantiation of this algorithm, we perform the following steps in each policy update:

(i) Using several epochs of minibatch SGD, optimize the KL-penalized objective:

$$\max_\theta \mathbb{E}_{\pi_{\theta^t}} \left[\frac{\pi_\theta(a|s)}{\pi_{\theta^t}(a|s)} \hat{A}^{\pi_{\theta^t}}(s,a) - \lambda_t \cdot D_{\mathrm{KL}}(\pi_{\theta^t}(s)|\pi_\theta(s)) \right].$$

(ii) Compute $d = \mathbb{E}[D_{\mathrm{KL}}(\pi_{\theta_t}(\cdot|s), \theta_{t+1}(\cdot|s))]$. If $d < d_{\mathrm{targ}}/1.5$, $\lambda_{t+1} = \lambda_t/2$. If $d > d_{\mathrm{targ}} \times 1.5$, $\lambda_{t+1} = \lambda_t \times 2$. Else, maintain the current λ_t. The updated λ_{t+1} is used for the next policy update. With this scheme, we occasionally see policy updates where the KL divergence is significantly different from d_{targ}, however, these are rare, and λ_t quickly adjusts. The parameters 1.5 and 2 above are chosen heuristically, but the algorithm is not very sensitive to them. The initial value of λ_0 is another hyperparameter but is not important in practice because the algorithm quickly adjusts it.