Homework 2

4771 Reinforcement Learning (Spring 2026)

Submission deadline: 11:59pm, February 28

The latex template is here.

1 PPO

We continue to implement contextual bandits algorithms in the same problem in Homework 1. You will present the results in the same way as in Homework 1. In this problem, you will implement Proximal Policy Optimization (PPO) (Algorithm 1). Similarly to Homework 1, please try to reach a best "overall score" of 0.55 in order to get full score. The starter code is here. The starter code keeps the same structure as Homework 1, and you can see a lot of similarities between the value-based and policy-based algorithms.

Algorithm 1 Proximal Policy Optimization for Contextual Bandits

Default hyperparameters: $N = 16$, $M = 10$, and $\alpha = 0.1$.
Randomly initialize a policy network π_θ that takes contexts as input and outputs an action distribution.
Let θ_1 be the initial weights for the policy network.
If using adaptive baseline (see Eq. (2)), additionally initialize a baseline network b_ϕ.
for $t = 1, \dots, T$ **do**
  for $n = 1, \dots, N$ **do**
    Receive context $x_{t,n}$.
    Sample action $a_{t,n} \sim \pi_{\theta_t}(\cdot|x_{t,n})$
    Receive reward $r_{t,n}$.
  $\theta \leftarrow \theta_t$
  for $m = 1, \dots, M$ **do**

$$\theta \leftarrow \theta + \lambda \nabla_\theta \left\{ \frac{1}{N} \sum_{n=1}^{N} \left[\frac{\pi_\theta(a_{t,n}|x_{t,n})}{\pi_{\theta_t}(a_{t,n}|x_{t,n})} (r_{t,n} - b_{t,n}) - \alpha \left(\frac{\pi_\theta(a_{t,n}|x_{t,n})}{\pi_{\theta_t}(a_{t,n}|x_{t,n})} - 1 - \ln \frac{\pi_\theta(a_{t,n}|x_{t,n})}{\pi_{\theta_t}(a_{t,n}|x_{t,n})} \right) \right] \right\}, \tag{1}$$

    where

$$b_{t,n} = \begin{cases} b & \text{for static baseline} \\ b_\phi(x_{t,n}) + b & \text{for adaptive baseline} \end{cases} \tag{2}$$

    If using adaptive baseline, update

$$\phi \leftarrow \phi - \lambda' \nabla_\phi \left[\frac{1}{N} \sum_{n=1}^{N} \left(b_\phi(x_{t,n}) - r_{t,n} \right)^2 \right]. \tag{3}$$

  $\theta_{t+1} \leftarrow \theta$

The `TODO` is to code up the operations within the m-for-loop (Line 11). Some implementation details are following:

- In Eq. (1), the gradient should only be taken with respect to θ. The gradient should NOT be taken over the θ_t in the denominator. Also, if using adaptive baseline, $b_{t,n}$ will involve $b_\phi(x_{t,n})$; the gradient should NOT also be taken over the ϕ there. To this end, we need to call the `tensor.detach()` function to prevent PyTorch to take gradient over them. This has been done in the starter code (search for "detach").
- If using adaptive baseline, you need to simultaneously maximize the objective in Eq. (1) and minimize the objective in Eq. (3). One easy way to do this is combining them as a single loss:

$$\text{Loss} = -\frac{1}{N}\sum_{n=1}^{N}\left[\frac{\pi_\theta(a_{t,n}|x_{t,n})}{\pi_{\theta_t}(a_{t,n}|x_{t,n})}(r_{t,n}-b_{t,n}) - \alpha\left(\frac{\pi_\theta(a_{t,n}|x_{t,n})}{\pi_{\theta_t}(a_{t,n}|x_{t,n})} - 1 - \ln\frac{\pi_\theta(a_{t,n}|x_{t,n})}{\pi_{\theta_t}(a_{t,n}|x_{t,n})}\right)\right] + \tau\cdot\frac{1}{N}\sum_{n=1}^{N}(b_\phi(x_{t,n})-r_{t,n})^2$$

  where τ specifies the relative importance between the two objective, and is a hyperparameter you may tune. You may start with $\tau = 1$. Notice the minus sign in the first term above—this is because in Eq. (1) we would like to *maximize* that term, but the default in PyTorch is loss minimization. We flip the sign to make it a *loss*.

In general, one may perform *mini-batch* gradient descent in Eq. (1). That means in each iteration $m = 1, 2, \ldots, M$, we only use B out of the N samples to perform the update for some $B < N$. This is the version we presented on Page 41 of this slide, and is also the more standard PPO for larger-scale problems. In this homework, we do it without mini-batching for simplicity.

You are free to change the default hyperparameters. In the tables below, you may also change the values of hyperparameters or add additional ones if you feel that the given values cannot reflect the trend.

We provide more intuition about the KL estimator $\left(\frac{\pi_\theta(a_{t,n}|x_{t,n})}{\pi_{\theta_t}(a_{t,n}|x_{t,n})} - 1 - \ln\frac{\pi_\theta(a_{t,n}|x_{t,n})}{\pi_{\theta_t}(a_{t,n}|x_{t,n})}\right)$ in Appendix A.

1.1 The effect of Baseline

For this part, you could set $\alpha = 0.1$ in Eq. (1) or any other fixed value. It will be tuned in Section 1.2.

Static Baseline

(a) (5%) Implement Algorithm 1 with static baseline (so Eq. (3) can be omitted and use the first option in Eq. (2)) and, for different values of b, record in the table below the average reward in Phase 1, Phase 2, and over the entire horizon.

b	Phase 1	Phase 2	Overall
2			
1.5			
1			
0.5			
0			
-0.5			

(b) (5%) Paste the running average reward plot over time generated by the code for your experiments in (a).

(c) (5%) How does the baseline affect the average reward in Phase 1 and Phase 2, respectively?

Adaptive Baseline

(d) (5%) Implement Algorithm 1 with adaptive baseline and, for different values of the extra baseline b in Eq. (2), record in the table below the average reward in Phase 1, Phase 2, and over the entire horizon.

b	Phase 1	Phase 2	Overall
0.3			
0.15			
0			
-0.15			
-0.3			

(e) (5%) Paste the running average reward plot over time generated by the code for your experiments in (d).

(f) (5%) What are the potential advantages or disadvantages of using adaptive baseline compared to static baselines?

1.2 The Effect of KL Regularization

(g) (5%) Use your best performed baseline setting discovered in Section 1.1 with different values of α in Eq. (1). Record in the table below the average reward in Phase 1, Phase 2, and over the entire horizon.

α	Phase 1	Phase 2	Overall
0			
0.1			
0.2			
0.5			
1			

(h) (5%) Paste the running average reward plot over time generated by the code for your experiments in (g).

(i) (5%) What is the effect of the KL regularization?

2 Survey

(5%) Leave any feedback for the course or the assignments.

A Some Intuition for the KL Estimator

In Eq. (1), we sample $a \sim \pi'$ and use

$$\frac{\pi(a)}{\pi'(a)} - 1 - \ln \frac{\pi(a)}{\pi'(a)} \tag{4}$$

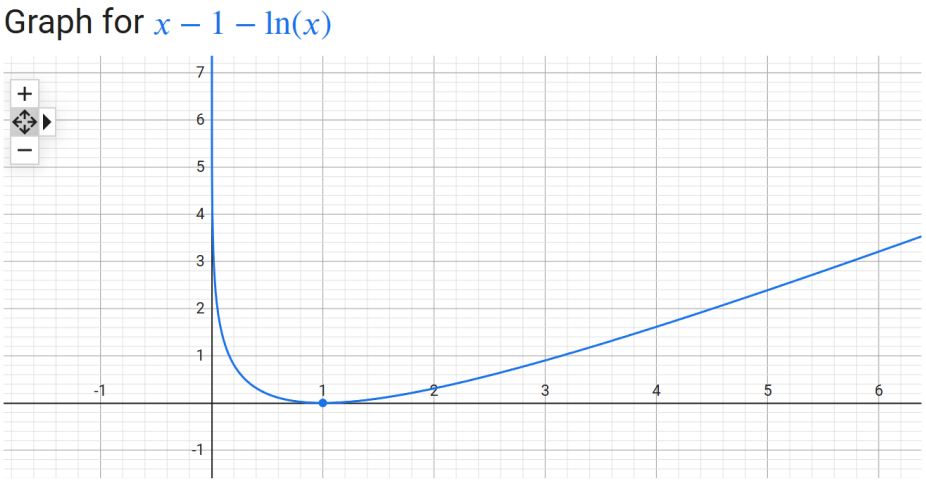
as a distance measure between two distributions π and π'.

Theoretical Analysis The expectation of this quantity is

$$\begin{aligned}
&\mathbb{E}_{a\sim\pi'}\left[\frac{\pi(a)}{\pi'(a)} - 1 - \ln \frac{\pi(a)}{\pi'(a)}\right] \\
&= \sum_a \pi'(a) \left(\frac{\pi(a)}{\pi'(a)} - 1 - \ln \frac{\pi(a)}{\pi'(a)}\right) \\
&= \sum_a \pi(a) - \sum_a \pi'(a) + \sum_a \pi'(a) \ln \frac{\pi'(a)}{\pi(a)} \\
&= \sum_a \pi'(a) \ln \frac{\pi'(a)}{\pi(a)} && \text{(because } \pi \text{ and } \pi' \text{ are distributions, we have } \textstyle\sum_a \pi(a) = \sum_a \pi'(a) = 1\text{)} \\
&= \mathrm{KL}(\pi', \pi). && \text{(by the definition of KL divergence)}
\end{aligned}$$

As mentioned in the class, KL is a distance measure between distributions.

The shape of Eq. (4) We plot the curve $y = x - 1 - \ln(x)$:



Here, x is the ratio $\frac{\pi(a)}{\pi'(a)}$ in Eq. (4). We see that Eq. (4) is non-negative, and is only zero when $\frac{\pi(a)}{\pi'(a)} = 1$. In other words, if $\pi(a) \neq \pi'(a)$ for some a, then Eq. (4) is positive. Therefore, it serves as a distance measure between π and π'.