Q1. Implement Advanced RL algorithm

(a).我選擇實踐的演算法是 Actor-Critic

(b). difference between your implementation and Policy Gradient:

我認為 Actor-Critic 與 Policy Gradient 最主要的差別在於更新 policy 的方式，有無價值函數估計(estimates value function)。

Actor-Critic: 結合了策略優化(policy optimization)和價值函數估計，其中 Actor 負責動作，Critic 負責價值函數估計以此來評估 Actor 的動作，並透過 Critic 的輸出來更新 Actor 的策略(policy)。

Policy Gradient: 直接優化策略參數(policy parameter)的方式來更新 policy，以此來最大化期望的累積獎勵，不需價值函數估計的計算參與其中。

(C). Please describe your implementation explicitly

在全連接層中，我設定兩層全連接層，第一層 input 為 8，output 為 16；第二層 input 為 16，output 為 16； 然後我定義 actor 模型是一個線性層，並輸出 4 個動作的概率分佈，而 critic 模型是另一個線性層，輸出當前狀態的期望獎勵的估計值。values 是一個空矩陣，用於存儲當前 batch 中每個狀態的評價。Optimizer 的部分是選用隨機梯度下降(SGD)，learning rate 為 0.001。def forward 的部分是以狀態作為輸入，通過神經網絡獲取 actor 的輸出和 critic 的輸出。actor 的輸出經過 softmax 函數處理，以獲取動作的概率分佈。critic 的輸出被加到 values 中，以便在訓練時計算損失函數。def learn 用於基於觀察到的獎勵和動作來更新模型的參數。它計算損失函數，包括兩個項目：政策梯度項目，鼓勵模型增加導致高獎勵的動作的概率，以及價值函數項目，鼓勵模型準確預測期望獎勵。損失函數使用隨機梯度下降（SGD）進行優化。最後 def sample 是通過將狀態通過神經網絡以獲取動作的概率分佈，然後使用 PyTorch 中的 Categorical 類從該分佈中抽樣來選擇當前狀態下的動作。它返回選定的動作和基於學習策略的選定動作的對數概率。

Q2.

1. How does the objective function of "PPO-ptx" differ from the “PPO” during RL training as used in the InstructGPT paper?

   PPO-ptx 與 PPO 的 objective function 不同的地方在於，PPO-ptx 的 objective function 多了 KL 懲罰這一項($\gamma E_{x\sim D_{pretrain}}\left[\log\left(\pi_\phi{}^{RL}(x)\right)\right]$)，而 KL 懲罰是透過$\gamma$來控制，假如$\gamma = 0$，那麼就是 PPO 模型。

$$\text{objective}\,(\phi) = E_{(x,y)\sim D_{\pi_\phi^{\text{RL}}}}\left[r_\theta(x,y) - \beta\log\left(\pi_\phi^{\text{RL}}(y\mid x)/\pi^{\text{SFT}}(y\mid x)\right)\right] + \gamma E_{x\sim D_{\text{pretrain}}}\left[\log(\pi_\phi^{\text{RL}}(x))\right]$$

2. what is the potential advantage of using "PPO-ptx" over “PPO” in the InstructGPT paper?

   根據 InstructGPT paper 的結果可知，PPO-ptx 相較於 PPO 的潛在優勢在於 PPO-ptx 可以鼓勵模型生成更完整和有信息性的指導性文本(instructional text)，並對其省略相關信息進行懲罰。這可能會使模型產生更高質量、更有用和更豐富的訊息指導性文本。