## PPO(proximal policy optimization)

: TRPO(trust region policy optimization) 의 장점도 갖고 있으면서, 적용하기 더 간단하고, 더 일반적이고, 더 좋은 sample complexity를 가짐

1. Policy Gradient Methods

: estimator of policy gradient를 계산하고, stochastic gradient ascent algorithm 에 접목시킴

: gradient estimator where 파이(세타) is a stochastic policy and At is an estimator of the advantage function at timestep t

$$\hat{g} = \hat{\mathbb{E}}_t \left[ \nabla_\theta \log \pi_\theta(a_t \mid s_t) \hat{A}_t \right]$$

-> expectation indicates the empirical average over a finite batch of samples

$$L^{PG}(\theta) = \hat{\mathbb{E}}_t \left[ \log \pi_\theta(a_t \mid s_t) \hat{A}_t \right].$$

-> the estimator g is obtained by differentiating the above objective

2. Trust Region Methods

: In TRPO, an objective function (surrogate objective) is maximized subject to a constraint on the size of policy update

$$\begin{aligned} &\underset{\theta}{\text{maximize}} \quad \hat{\mathbb{E}}_t \left[ \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} \hat{A}_t \right] \\ &\text{subject to} \quad \hat{\mathbb{E}}_t [\mathrm{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)]] \leq \delta. \end{aligned}$$

-> 세타(old) is the vector of policy parameters before the update

-> using a penalty instead of a constraint for some coefficient 베타

$$\underset{\theta}{\text{maximize}} \, \hat{\mathbb{E}}_t \left[ \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} \hat{A}_t - \beta \, \mathrm{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)] \right]$$

-> a certain surrogate objective forms a lower bound on the performance of the policy 파이

3. Clipped Surrogate Objective

TRPO maximize a surrogate objective

$$L^{CPI}(\theta) = \hat{\mathbb{E}}_t \left[ \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)} \hat{A}_t \right] = \hat{\mathbb{E}}_t \left[ r_t(\theta) \hat{A}_t \right]$$

without a constraint, maximization of L would lead to an ecxessively large policy update

-> to penalize change to the policy that move rt(세타) away from 1

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

where epsilon is a hyperparameter, say, 0.2.

4. Adaptive KL Penalty Coefficient
: as an alternative to the clipped surrogate objective or in addition to it
(일반적으로 clipped surrogate objective보다 안 좋음)

- Using several epochs of minibatch SGD, optimize the KL-penalized objective

$$L^{KLPEN}(\theta) = \hat{\mathbb{E}}_t\left[\frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}\hat{A}_t - \beta\,\text{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)]\right]$$

- Compute $d = \hat{\mathbb{E}}_t[\text{KL}[\pi_{\theta_{\text{old}}}(\cdot \mid s_t), \pi_\theta(\cdot \mid s_t)]]$
  - If $d < d_{\text{targ}}/1.5$, $\beta \leftarrow \beta/2$
  - If $d > d_{\text{targ}} \times 1.5$, $\beta \leftarrow \beta \times 2$

------

정리
1. PPO는 정책 기반 강화 학습 알고리즘으로, 에이전트가 정책을 직접 최적화하는 방법.
- 목표 함수 정의 : 정책 $\pi_\theta(a|s)$를 파라미터 $\theta$에 따라 최적화, 목표는 return 최대
- 그래디언트 추정 : 정책의 파라미터 업데이트를 위해, 정책의 그래디언트 추정
- 업데이트 규칙 : $\theta \rightarrow \theta + \alpha \nabla_\theta J\theta$, alpha는 학습률

2. Trust Region Methods : 정책 업데이트 시 정책의 급격한 변화를 방지
- KL divergence : 새로운 정책과 기존 정책 간 차이를 측정하기 위해 Kullback-Leibler divergence를 사용
- 제약 조건 최적화 : 정책 업데이트 시 KL divergence가 일정 임계값을 넘지 않도록 제약을 가함.
- TRPO(Trust Region Policy Optimiztion) : 대표적인 신뢰 영역 방법, 목적 함수를 최대화하면서도 KL divergence 제약조건을 사용하여 안정적인 학습 보장

3. Clipped Surrogate Objective : TRPO의 복잡성을 줄이면서 비슷한 안정성 제공

- 대리 목표 함수(surrogate) : PPO는 원래 목적 함수를 직접 최적화하는 대신, 대리 목표 함수를 최적화.
- 확률 비율 cliping : 새로운 정책과 기존 정책의 확률 비율을 클립하여 급격한 정책 변화를 방지

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

-> 여기서 $\hat{A}_t$ 는 advantage 추정치, $\epsilon$ 는 클립 한계

4. Adaptive KL Penalty Coefficient : 정책 업데이트 시 KL divergence를 적응적으로 조정

- KL penalty : KL divergence를 직접 제약 조건으로 사용하는 대신, KL divergence를 penalty항으로 추가

$$L^{\text{KL}}(\theta) = L(\theta) - \beta\text{KL}[\pi_{\theta_{\text{old}}}||\pi_\theta]$$

-> $\beta$ 는 패널티 계수

- 적응적 조정 : 학습 과정에서 KL divergence가 너무 작거나 크면, $\beta$ 를 동적으로 조정하여 정책 업데이트의 크기를 조절

■ 정리

**1. 정책 기반 방법 (Policy-based methods)**

PPO는 정책 기반 방법으로, 정책 함수 $\pi_\theta(a|s)$를 직접 최적화합니다. 여기서 $\theta$는 정책의 파라미터, $a$는 행동, $s$는 상태를 의미합니다. 정책 기반 방법은 가치 함수 기반 방법보다 높은 차원의 행동 공간을 다룰 수 있는 장점이 있습니다.

**2. Clipped Surrogate Objective**

PPO는 정책의 큰 변화를 제한하기 위해 클리핑 기법을 사용합니다. 이는 "클립된 대리 목적 함수(Clipped Surrogate Objective)"를 통해 구현됩니다. PPO의 목표는 다음과 같은 목적 함수를 최대화하는 것입니다:

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

여기서:

- $r_t(\theta) = \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)}$ 는 현재 정책과 이전 정책의 비율입니다.
- $\hat{A}_t$는 시간 $t$에서의 어드밴티지 함수(Advantage Function)입니다.
- $\epsilon$은 클리핑 범위를 조절하는 하이퍼파라미터입니다.

이 목적 함수는 정책이 너무 많이 변하지 않도록 제한합니다. 클립핑된 부분은 비율 $r_t(\theta)$가 $[1-\epsilon, 1+\epsilon]$ 범위를 벗어날 경우, 이 범위로 비율을 제한하여 학습 안정성을 높입니다.

**3. 어드밴티지 함수 (Advantage Function)**

어드밴티지 함수 $\hat{A}_t$는 특정 행동이 현재 상태에서 얼마나 좋은지를 나타내는 값으로, 보통 다음과 같이 정의됩니다:

$$\hat{A}_t = Q(s_t, a_t) - V(s_t)$$

여기서 $Q(s_t, a_t)$는 상태-행동 가치 함수, $V(s_t)$는 상태 가치 함수입니다. 일반적으로, 어드밴티지 함수는 GAE (Generalized Advantage Estimation)를 사용하여 계산됩니다.

**4. 상태 가치 함수 (Value Function)**

PPO는 상태 가치 함수 $V(s_t)$를 동시에 학습하여 상태의 가치를 평가합니다. 상태 가치 함수는 다음과 같은 손실 함수로 학습됩니다:

$$L^{VF}(\theta) = \mathbb{E}_t \left[ \left( V_\theta(s_t) - V_t^{target} \right)^2 \right]$$

여기서 $V_t^{target}$는 타겟 가치로, 보통 미래의 할인된 보상을 기반으로 계산됩니다.

**5. 엔트로피 보너스 (Entropy Bonus)**

PPO는 탐색을 촉진하기 위해 정책의 엔트로피를 증가시키는 보너스를 추가합니다. 엔트로피 보너스는 정책의 불확실성을 증가시켜 다양한 행동을 시도하게 만듭니다. 이는 다음과 같은 형태로 목적 함수에 추가됩니다:

$$L^S(\theta) = \mathbb{E}_t[\beta H(\pi_\theta(\cdot|s_t))]$$

여기서 $H(\pi_\theta(\cdot|s_t))$는 상태 $s_t$에서 정책의 엔트로피, $\beta$는 엔트로피 보너스의 가중치를 조절하는 하이퍼파라미터입니다.

## PPO 알고리즘의 단계

1. **환경과 상호작용**: 에이전트는 현재 정책 $\pi_\theta$를 사용하여 환경과 상호작용하며 데이터를 수집합니다. 여기에는 상태, 행동, 보상, 다음 상태 등이 포함됩니다.
2. **어드밴티지 추정**: 수집된 데이터를 바탕으로 어드밴티지 함수 $\hat{A}_t$를 계산합니다. GAE를 사용하여 어드밴티지를 추정하는 경우가 많습니다.
3. **정책 업데이트**: 클립된 대리 목적 함수 $L^{CLIP}(\theta)$를 최대화하도록 정책 $\pi_\theta$를 업데이트합니다. 이는 미니배치 단위로 여러 번 반복됩니다.
4. **가치 함수 업데이트**: 상태 가치 함수 $V(s_t)$를 학습하기 위해 손실 함수 $L^{VF}(\theta)$를 최소화하도록 업데이트합니다.
5. **엔트로피 보너스 적용**: 정책의 엔트로피를 증가시키는 보너스를 목적 함수에 추가하여 업데이트합니다.

## PPO의 장점

- **안정성**: 클리핑 기법을 통해 정책의 급격한 변화를 방지하여 학습의 안정성을 높입니다.
- **단순성**: 기존의 복잡한 신뢰 영역 방법 (Trust Region Methods)보다 구현이 간단하고 효율적입니다.
- **효율성**: 경험을 효율적으로 재사용하여 샘플 효율성을 높입니다.

※ GAE (Generalized Advantage Estimation) : 강화 학습에서 Advantage Function을 계산하는 방법 -> 추정치의 분산을 줄이면서도 편향을 통제할 수 있는 유연한 방법 제공

## GAE의 정의

GAE는 다음과 같은 형태로 정의됩니다:

$$\hat{A}_t^{GAE(\gamma,\lambda)} = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}$$

여기서 $\delta_t$는 Temporal Difference (TD) 에러로, 다음과 같이 정의됩니다:

$$\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

GAE는 $\gamma$ (할인율)과 $\lambda$ (감쇠 인자)라는 두 개의 하이퍼파라미터를 사용합니다. $\gamma$는 미래 보상의 중요도를 결정하고, $\lambda$는 부트스트랩의 길이를 조절합니다. $\lambda$가 1에 가까울수록 GAE는 높은 분산을 가지지만, $\lambda$가 0에 가까울수록 GAE는 낮은 분산과 높은 편향을 가집니다.