PPO 논문 결과 재현 및 논문에서 커버되지 않은 파라미터 이용한 재현 실험

- 수업명 : 강화학습의 기초
- 팀명 : 獨GoDying
- 팀장(팀원) : 문정인 (1人)
- 학번 : A71027
- 제출일: 2025-12-07
- GitHub Repository
  : https://github.com/a71027/Reinforcement-Learning-Project-PPO-CartPole/tree/main

목차

1. 프로젝트 주제 선정배경 및 재현 논문
2. 논문 재현 프로젝트 목표
3. 재현 논문 요약(1/2) – 연구배경/해결방안
4. 재현 논문 요약(2/2) – 실험결과/논문기여
5. 논문 재현 환경 및 State/Action/Reward 정의
6. 논문 재현 실험 설계(Baseline) & 구현(1/2) - 알고리즘 구성요소 및 하이퍼 파라미터
7. 논문 재현 실험 설계(Baseline) & 구현(2/2) - 실험 구성요소 및 결과 그래프
8. 논문 개선 실험 설계 & 구현(1/2) - 실험 구성요소 및 결과 구현
9. 논문 개선 실험 설계 & 구현(2/2) - 결과 그래프(λ비교, 논문 미포함 파라미터)
10. 결론

프로젝트 주제 선정 배경 및 재현 논문

프로젝트 주제 선정 배경

- 가장 권장되는 현업 데이터는 스팸문자 처리 Data로 강화학습 도다 Classification에 적합하여 제외
- 강화학습 논문 결과 재현 및 논문에서 커버되지 않는 파리미터 이용 재현 실험 선정

프로젝트 재현 논문

- 재현 실험 강화학습 논문 : Proximal Policy Optimization Algorithms (2017년, OpenAI 연구진)
- 논문 출처 : https://arxiv.org/pdf/1707.06347 (Cornell University 운영, 전자논문 저장소)
- 논문저자 : John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, Oleg Klimov

재현 논문 선정 사유

- 강화학습의 기초 수업자료로 제공받았으나, 학습 진도상 교수님께서 강의하시지 못한 수업(PPO 알고리즘)에 대해 스스로 학습/재현 하고자 했음

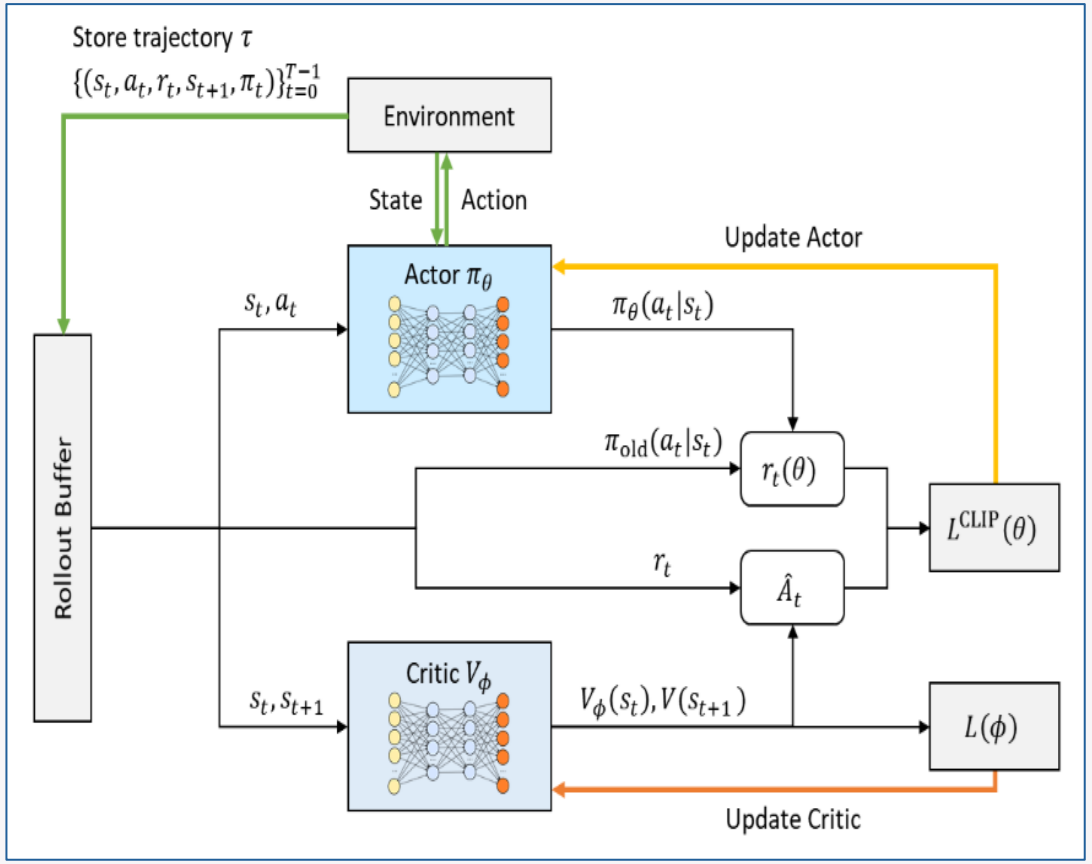
재현 프로젝트 목표

① 2017년 OpenAI 제안한 Proximal Policy Optimization 알고리즘 논문에서 제시된 기본 설정 기반 핵심개념 재현

- 논문에서 사용된 기본 설정(clip ε=0.2, λ=0.95, γ=0.99)통한 PPO 학습 특성/수렴 경향을 CartPole-V1 환경에서 재현
  → 논문 기본설정으로 Baseline 실험

② 논문에서 다루지 않은 중요 파라미터(GAE λ) 값의 변화가 학습 안정성과 수렴 성능에 미치는 영향 분석

- clip ε = 0.2 baseline + GAE λ 변화 실험
- λ ∈ {0.80, 0.90, 0.95, 0.99}로 변경하며, PPO의 GAE (Generalized Advantage Estimation) 파라미터 변화가 학습 안정성, Reward 수렴 속도, Variance에 미치는 영향 비교·분석



[Proximal Policy Optimization 알고리즘 구조]

재현 논문 요약(1/2) – 연구배경 / 해결방안

연구 배경(기존 방법 한계)

- Vanilla Policy Gradient
  - 정책이 한번에 크게 이동하면 학습 불안정
- TRPO (Trust Region Policy Optimization)
  - 안정적 Policy 업데이트可 →구현 복잡, 계산량 높음

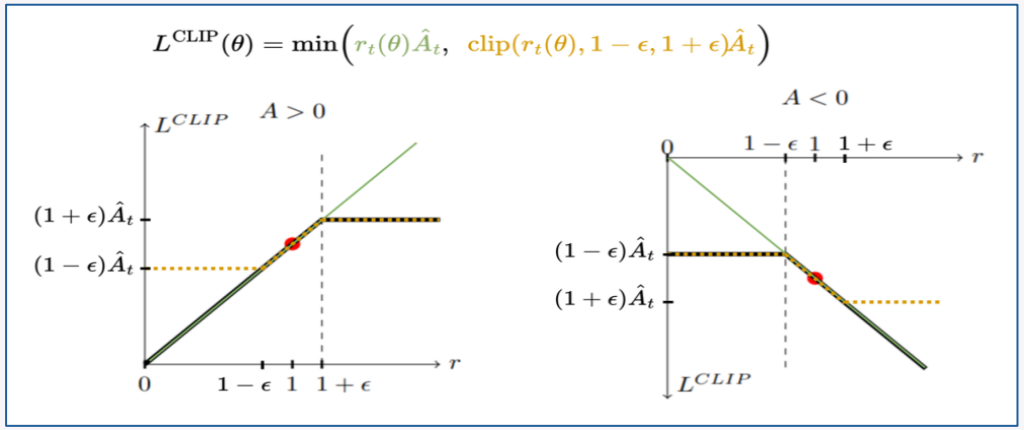
☞ 안정성과 단순함 동시 만족 정책 최적화 방법 필요

해결방안(PPO 제안)

- Clipped Surrogate Objective
  - $L^{CLIP}(\theta) = \hat{\mathbb{E}}_t\left[\min(r_t(\theta)\hat{A}_t, \text{clip}(r_t(\theta), 1-\epsilon, 1+\epsilon)\hat{A}_t)\right]$
  - 급격한 정책 변화(ratio 변화)를 clip하여 안정성 확보
    - ☞ 새로운 Policy가 너무 급격히 변하지 않도록 Ratio (확률비)가 일정범위(1 ± ε) 벗어나면 Clip하여 “보수적 업데이트“ 강제
  - TRPO처럼 복잡한 KL constraint 없이 SGD로 최적화 가능
- Multiple Epochs on Minibatches
  - 온폴리시 데이터 여러번 업데이트 가능 (효율 향상)
- Implementation-Friendly Design
  - Adam optimizer, simple MLP, no second-order optimization
  - 구현 매우 간단하며, reproducible

재현 논문 요약(2/2) – 실험결과 / 논문기여

실험 결과



[Advantage 부호에 따른 ratio와 surrogate objective의 관계]

- MuJoCo 연속 제어
  - PPO는 *TRPO보다 빠르게 수렴*
  - A2C, CEM 등 기존 방법보다 높은 reward
  - ε = 0.2 설정이 가장 안정적
- Atari 49개 게임
  - A2C 대비 더 높은 sample efficiency
  - ACER와 유사 반면 구현은 훨씬 단순

논문 기여

- Stable (안정성)
  - clipped update로 policy collapse 방지
- Simple (단순성)
  - TRPO 대비 구현 난이도 대폭 감소(VPG 수준)
- Effective (효율성)
  - 다양한 작업(예 : MuJoCo, Atari)환경에서 기존 알고리즘보다 우수한 성능

논문 재현 환경 및 State/Action/Reward 정의

논문 재현 환경

- 개발 환경
  - Gymnasium CartPole-v1
- 개발환경 선정 사유
  - 논문의 대규모 환경(Atari 49게임, Mujoco연속 제어) 재현은 많은 계산 자원/시간 및 GPU/병렬환경 요구됨
  - CartPole은 CPU로 재연가능 → 보상 구조 및 상태 차원 단순
    - ☞ PPO 수렴 특성 및 파라미터 영향 명확化
- 목적
  - 막대를 500 step동안 쓰러뜨리지 않고 유지
- Preprocessing
  - CartPole은 이미 normalized 형태 제공
  - 추가 scaling 불필요
  - reward shaping 없음
  - action은 정수형 그대로 사용 (one-hot 불필요)

State/Action/Reward

- State
  - cart position (약 -4.8 ~ 4.8), cart velocity, pole angle, pole angular velocity
- Action
  - 0 : 왼쪽(left)으로 힘 적용
  - 1 : 오른쪽(right)으로 힘 적용
- Reward
  - 매 step : +1
  - episode reward = 살아남은 step 수
- Episode Termination
  - pole angle > threshold (막대 각도가 일정각도 이상 기울어짐)
  - cart position > threshold (카트가 화면범위 벗어남)
  - 500 step 도달 (성공)

논문 재현 실험 설계(Baseline) & 구현(1/2) – 알고리즘 구성요소 및 하이퍼 파라미터

논문 알고리즘 구성요소

① Policy Ratio

$$r_t(\theta) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}$$

② Clipped Objective

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

③ Advantage Estimatior : GAE(λ)

$$\hat{A}_t = \delta_t + (\gamma\lambda)\delta_{t+1} + \cdots + \cdots + (\gamma\lambda)^{T-t+1}\delta_{T-1},$$
$$\text{where} \quad \delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

④ Value Function Loss

$$(V_\theta(s_t) - V_t^{\text{targ}})^2$$

⑤ PPO 업데이트

- Rollout trajectory 수집
- Minibatch SGD방식 반복 업데이트
- policy _old ← policy ($\theta_{\text{old}} \leftarrow \theta$)

논문 기반 Hyperparameter 설정

- **Table 1**

algorithm	avg. normalized score
No clipping or penalty	-0.39
Clipping, $\epsilon = 0.1$	0.76
Clipping, $\epsilon = 0.2$	**0.82**
Clipping, $\epsilon = 0.3$	0.70
Adaptive KL $d_{\text{targ}} = 0.003$	0.68
Adaptive KL $d_{\text{targ}} = 0.01$	0.74
Adaptive KL $d_{\text{targ}} = 0.03$	0.71
Fixed KL, $\beta = 0.3$	0.62
Fixed KL, $\beta = 1.$	0.71
Fixed KL, $\beta = 3.$	0.72
Fixed KL, $\beta = 10.$	0.69

- **Table 3(Appendix A)**

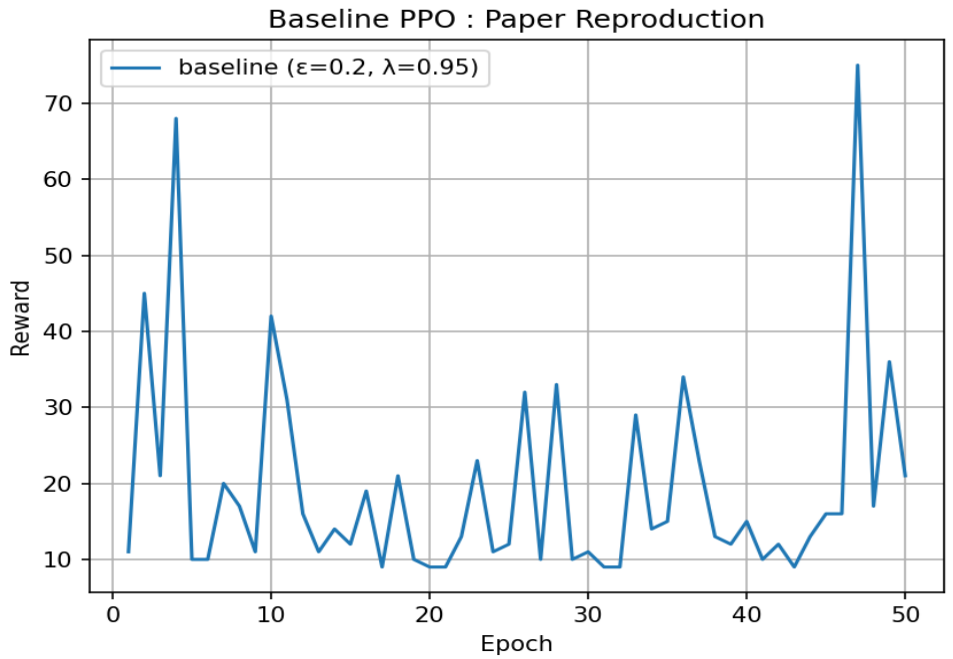
Hyperparameter	Value
Horizon (T)	2048
Adam stepsize	3×10^{-4}
Num. epochs	10
Minibatch size	64
Discount (γ)	0.99
GAE parameter (λ)	0.95

논문 재현 실험 설계(Baseline) & 구현(2/2) – 실험 구성요소 및 결과 그래프

논문 재현 실험 구성요소/결과구현

- 환경 : Gymnasium CartPole-v1
- 구현언어 : Python, PyTorch
- Seed : 0 단일 실행 (baseline_lam095_seed0.csv 사용)
- 파라미터
  - epochs : 50, Clip ε : 0.2, λ = 0.95
- 실험
  - train.py 실행 : λ = 0.95 → 총 3개 CSV 생성(Baseline)
    ☞ 개선실험 포함 일괄 생성(λ 4개, 총 12개 CSV)
- 학습(Train.py) 로그 (CSV 파일 저장)
  - epoch, mean reward, policy loss, value loss, clip epsilon
- 결과 시각화
  - plot_baseline.py 실행

재현 실험 결과 그래프(plot_baseline.py)



- Baseline PPO(ε=0.2, λ=0.95)는 초기에 변동 있으나, epoch 진행될수록 reward 안정적 증가
  → PPO 논문의 전형적 수렴 패턴 재현
- clipped update로 정책 과도한 변함없이 학습 안정적 진행

논문 개선 실험 설계 & 구현(1/2) – 실험 구성요소 및 결과 구현

논문 개선 실험 구성요소/결과구현

- 환경 : Gymnasium CartPole-v1
- 구현언어 : Python, PyTorch
- Seed : 0, 42, 999 총 12회(λ당 3회) 반복수행 → 평균 분석(신뢰도 ↑)
- 파라미터
  - epochs : 50, Clip ε : 0.2, λ = 0.95
- 실험
  - train.py 실행 : PPO λ 확장 실험(λ = 0.80, 0.90, 0.95, 0.99)
    → 총 12개 CSV 생성
- 학습(Train.py) 로그 (CSV 파일 저장)
  - epoch, mean reward, policy loss, value loss, clip epsilon
- 결과 시각화
  - plot_lambda.py 실행

Hyperparameter 설정

- **Table 1**

algorithm	avg. normalized score
No clipping or penalty	-0.39
Clipping, $\epsilon = 0.1$	0.76
Clipping, $\epsilon = 0.2$	**0.82**
Clipping, $\epsilon = 0.3$	0.70
Adaptive KL $d_{\text{targ}} = 0.003$	0.68
Adaptive KL $d_{\text{targ}} = 0.01$	0.74
Adaptive KL $d_{\text{targ}} = 0.03$	0.71
Fixed KL, $\beta = 0.3$	0.62
Fixed KL, $\beta = 1.$	0.71
Fixed KL, $\beta = 3.$	0.72
Fixed KL, $\beta = 10.$	0.69

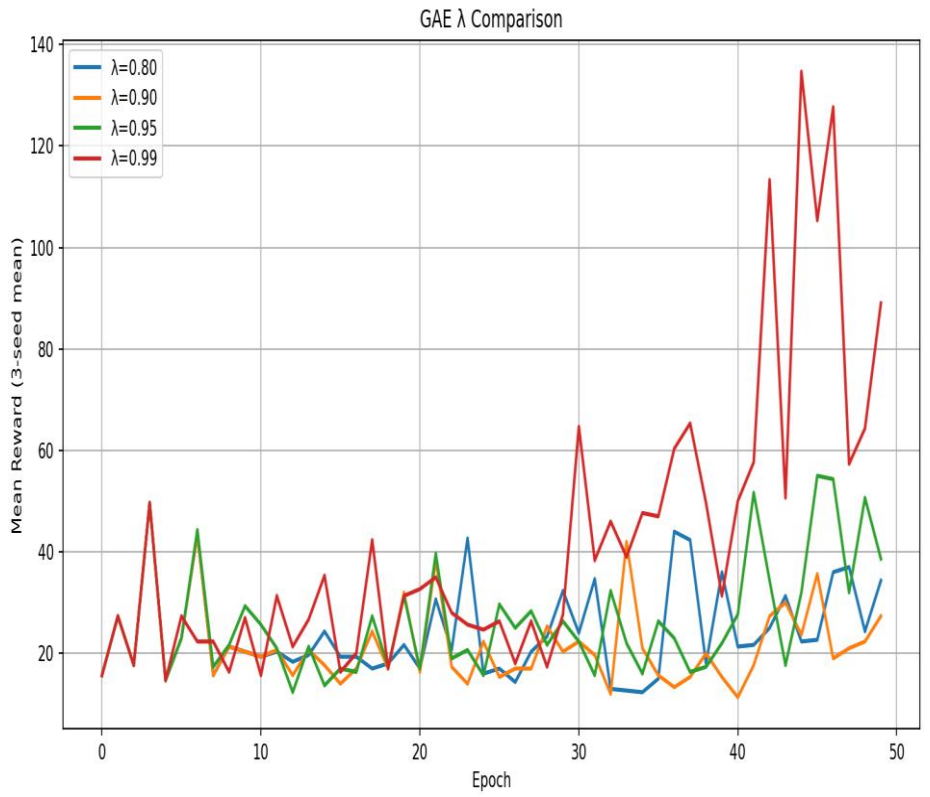
- **Table 3(Appendix A)**

Hyperparameter	Value
Horizon (T)	2048
Adam stepsize	3×10^{-4}
Num. epochs	10
Minibatch size	64
Discount (γ)	0.99

★ 논문에서 다루지 않은 파라미터 λ 확장 실험
- λ ∈ {0.80, 0.90, 0.95, 0.99} ☜ 비교/분석 성능 평가

논문 개선 실험 설계 & 구현(2/2) – 결과 그래프 (λ 비교, 논문 미포함 파라미터)

4가지 λ 비교 그래프(plot_lambda.py)



※ λ = 0.95 재현실험(Baseline)과 개선실험 그래프가 다른 사유?
: 개선실험은 평균(3개 seed의 mean) 사용 → 변동성 완화

개선 실험 결과 (lambda_summary.py)

No	λ	최종 Mean Reward	Variance	해석
1	0.80	34.33	169.56	안정성 낮고 reward도 낮음
2	0.90	27.33	206.89	Reward 낮고 variance 큼
3	0.95	33.66	156.22	Baseline, 비교적 안정
4	0.99	89.00	1152.67	Reward 높지만, 불안정

- λ = 0.99 : reward높지만 variance 매우큼 → 불안정
- λ = 0.95 : reward높고, variance 낮은 편 → 비교적 안정
- λ = 0.80, 0.90 : CartPole에서 좋지 않은 성능

결론

재현 및 개선 실험 결론 (lambda_summary.py)

No	λ	최종 Mean Reward	Variance	해석
1	0.80	낮음	높음	느리고 불안정
2	0.90	낮음	중간 ~ 높음	성능 낮음
3	0.95	적당히 높음	낮음	가장 안정적
4	0.99	매우 높음	매우 높음	불안정, 진폭 큼

- λ 는 PPO 안정성에 큰영향 주는 요소
- CartPole-v1에서는 λ = 0.95가 가장 안정적, 논문 기본값을 그대로 따르는 것이 최적
  → PPO 논문 설정과도 정확하게 일치하는 결과임

결론 요약(lambda_summary.py)

- λ가 작은 경우(0.8 근처)
  - advantage의 variance는 작아지지만 bias 증가
    : 학습 느리고 reward 값 낮게 유지
- λ가 큰 경우(0.99 근처)
  - reward 매우 높지만, variance가 매우 크게 나타남(불안정)
    → CartPole(단순 환경)에서 과도한 variance 초래
- λ=0.95인 경우(두 특성의 균형)
  - reward 좋고, variance도 낮음 → 균형 최적화

☞ λ = 0.95는 bias와 variance의 균형이 가장 잘 맞는 값