Uma viagem no tempo: de 1992 até 2017

Participação especial: Reinforce, A2C e PPO

Material adicional, 2025

FABRÍCIO J. BARTH

Algorithm 13 REINFORCE

Initialize policy network π with random parameters ϕ
Repeat for every episode:
for time step $t = 0, 1, 2, \ldots, T-1$ **do**
  Observe current state s^t
  Sample action $a^t \sim \pi(\cdot \mid s^t; \phi)$
  Apply action a^t; observe reward r^t and next state s^{t+1}
Loss $\mathcal{L}(\phi) \leftarrow -\frac{1}{T} \sum_{t=0}^{T-1} \left(\sum_{\tau=t}^{T-1} \gamma^{\tau-t} r^\tau \right) \log \pi(a^t \mid s^t; \phi)$
Update parameters ϕ by minimizing the loss $\mathcal{L}(\phi)$

gerar de uma trajetória $s_0, a_0, r_0, s_1, a_1, r_1, \ldots s_t, a_t, r_t$ seguindo π

$G \leftarrow \sum_{k=t+1}^{T} \gamma^{k-t-1} \times r_k$

Simulação do aprendizado usando Reinforce:

1 ep	s_0	s_1	s_2	...	s_t	trajetória
2 ep 1	1 0	01	01		0	
-	1 0	01	01		0	
-	1 0	01	01		0	
1	1 0	01	01		0	

prob atual = [p_0 p_1 p_2 ... p_t]

Rewards = [0 -1 0 +1]

Discounted Returns = [0+0 -1+1 0+1 1] $\gamma = 1$

$\gamma^3 \cdot 1$ $\gamma^2 \cdot 1$ $\gamma \cdot 1$ $\gamma^0 \cdot 1$

loss = [$-\log(p_0) \cdot 0$ $-\log(p_1) \cdot 0$ $-\log(p_2) \cdot 1$ $-\log(p_t) \cdot 1$]

GA(l) = GD(-l)
loss = -log-prob * G

backpropagation

Pode ter alta variância, o que leva a alta variância dos gradientes e instabilidade no treinamento.

* executar o programa exemplo

Algorithm 14 Simplified advantage actor-critic (A2C)

Initialize actor network π with random parameters ϕ
Initialize critic network V with random parameters θ
Repeat for every episode:
for time step $t = 0, 1, 2, \ldots$ **do**
  Observe current state s^t
  Sample action $a^t \sim \pi(\cdot \mid s^t; \phi)$
  Apply action a^t; observe reward r^t and next state s^{t+1}
  if s^{t+1} is terminal **then**
    Advantage $Adv(s^t, a^t) \leftarrow r^t - V(s^t; \theta)$
    Critic target $y^t \leftarrow r^t$
  else
    Advantage $Adv(s^t, a^t) \leftarrow r^t + \gamma V(s^{t+1}; \theta) - V(s^t; \theta)$
    Critic target $y^t \leftarrow r^t + \gamma V(s^{t+1}; \theta)$
  actor loss $\mathcal{L}(\phi) \leftarrow -Adv(s^t, a^t) \log \pi(a^t \mid s^t; \phi)$
  Critic loss $\mathcal{L}(\theta) \leftarrow \left(y^t - V(s^t; \theta)\right)^2$
  Update parameters ϕ by minimizing the actor loss $\mathcal{L}(\phi)$
  Update parameters θ by minimizing the critic loss $\mathcal{L}(\theta)$

Rede usada p/ o cálculo da vantagem

Rede usada p/ escolher a ação dado um estado

cálculo da vantagem menos um baseline

tentativa para diminuir a variância.

equação similar ao Reinforce, com a adaptação do conceito de Vantagem

Algorithm 15 Simplified proximal policy optimization (PPO)

1: Initialize actor network π with random parameters ϕ
2: Initialize critic network V with random parameters θ
3: Repeat for every episode:
4: **for** time step $t = 0, 1, 2, \ldots$ **do**
5:   Observe current state s^t
6:   Sample action $a^t \sim \pi(\cdot \mid s^t; \phi)$
7:   Apply action a^t; observe reward r^t and next state s^{t+1}
8:   $\pi_\beta(a^t \mid s^t) \leftarrow \pi(a^t \mid s^t; \phi)$
9:   **for** epoch $e = 1, \ldots, N_e$ **do**
(10) 10: $\rho(s^t, a^t) \leftarrow \pi(a^t \mid s^t; \phi) \div \pi_\beta(a^t \mid s^t)$
11:     **if** s^{t+1} is terminal **then**
12:       Advantage $Adv(s^t, a^t) \leftarrow r^t - V(s^t; \theta)$
13:       Critic target $y^t \leftarrow r^t$
14:     **else**
15:       Advantage $Adv(s^t, a^t) \leftarrow r^t + \gamma V(s^{t+1}; \theta) - V(s^t; \theta)$
16:       Critic target $y^t \leftarrow r^t + \gamma V(s^{t+1}; \theta)$
17:     Actor loss $\mathcal{L}(\phi) \leftarrow -\min \begin{pmatrix} \rho(s^t, a^t) Adv(s^t, a^t), \\ \text{clip}\left(\rho(s^t, a^t), 1-\epsilon, 1+\epsilon\right) Adv(s^t, a^t) \end{pmatrix}$
18:     Critic loss $\mathcal{L}(\theta) \leftarrow \left(y^t - V(s^t; \theta)\right)^2$
19:     Update parameters ϕ by minimizing the actor loss $\mathcal{L}(\phi)$
20:     Update parameters θ by minimizing the critic loss $\mathcal{L}(\theta)$

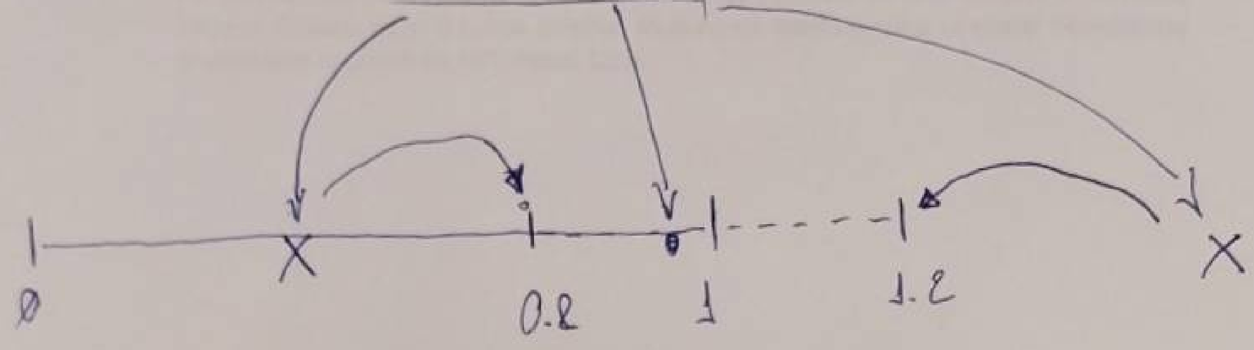
10) $\pi(\theta) = \dfrac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta old}(a_t \mid s_t)}$ como uma forma de estabilizar o treinamento.

No entanto, mesmo assim podemos ter alguns problemas, por exemplo:

$\dfrac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta old}(a_t \mid s_t)} = \dfrac{0.4}{0.0001} = 4000 \times$ GAIN vamos ter um número grande para o $loss(\theta)$?

probabilidade atual ⇒ $\pi_\theta(a_t|s_t)$

probabilidade antiga ⇒ $\pi_{\theta_{old}}(a_t|s_t)$



Probabilidades são valores entre 0 e 1 e neste caso queremos deixar mais próximo 1. pq elas não deveriam mudar muito.

$$\min\left(p(s_t,a_t)\cdot Adv(s_t,a_t)\ ,\ \boxed{clip(p(s_t,a_t), 1-\epsilon, 1+\epsilon)}\cdot Adv(s_t,a_t)\right)$$

⇓

linha 17

if $p(s_t,a_t)$ entre $1-\epsilon$ and $1+\epsilon$, então fica

if menor que $1-\epsilon$, então $\underline{1-\epsilon}$

if maior que $1+\epsilon$, então $\underline{1+\epsilon}$

Referências

Os pseudo-códigos utilizados neste material foram retirados do livro Stefano V. Albrecht, Filippos Christianos, and Lukas Schäfer. Multi-Agent Reinforcement Learning: Foundations and Modern Approaches. MIT Press, 2024.