



Aprendizado por Reforço

AULA - 5

Actor-Critic

Retrospectiva do penúltimo episódio

- Diferença Temporal

$$V(s) \leftarrow V(s) + \alpha[r_{t+1} + \gamma V(s_{t+1}) - V(s)]$$

- Deep Q-Network

$$L(\theta) = \left(r + \gamma \max_{a'} Q(s', a'; \phi) - Q(s, a; \theta) \right)^2$$



Retrospectiva do último episódio

- Gradiente de Política

$$\nabla_{\theta} J(\theta) \approx \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_{i,t} | s_{i,t}) R_{i,t}$$

- Com Baseline

$$\nabla_{\theta} J(\theta) \approx \frac{1}{N} \sum_{i=1}^N \sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(a_{i,t} | s_{i,t}) [R_{i,t} - V_{\phi}(s_t)]$$



Actor-Critic

Actor-Critic != Baseline

- Algoritmos de *Actor-Critic* fazem *Bootstrapping* do Retorno

$$R_t = r_{t+1} + \gamma V_{\pi}(s_{t+1})$$

- Chama-se **Crítico** porque o modelo estima o quão bem a política irá performar no futuro, dado também que ele aprende com as experiências coletadas por tal política

n-step Bootstrapping

- Posso fazer Bootstrapping com mais de uma recompensa adquirida?

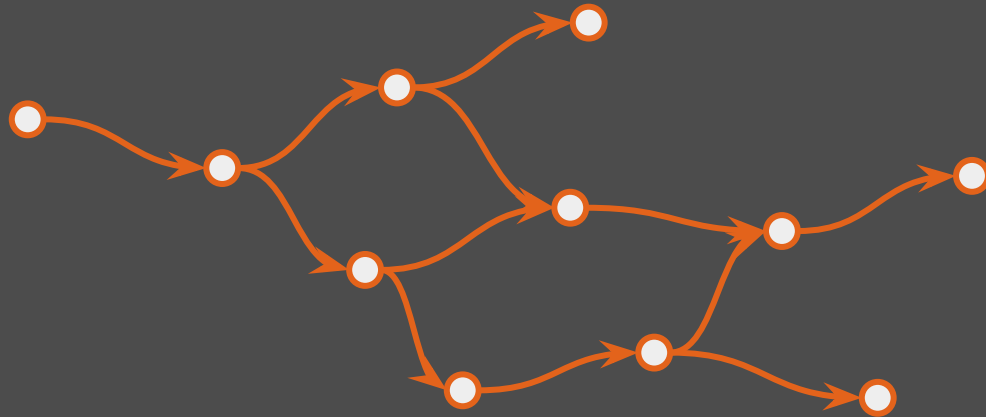
$$R_t = r_{t+1} + \gamma V(s_{t+1})$$

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 V(s_{t+2})$$

$$R_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \gamma^3 V(s_{t+3})$$

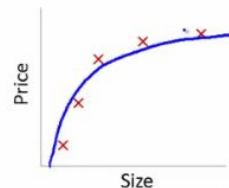
Quanto mais melhor?

- Mais recompensas coletadas
- Mais preciso é o retorno para atualizar a política (e a função de valor)
- Mais variável é o retorno (Mais variância na *loss*)



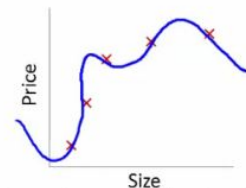
Variância é ruim?

- É mais difícil de aprender uma estimativa se o alvo varia muito
- É mais difícil aprender uma política se o retorno varia muito
- “Overfit” nas experiências coletadas pode não representar a distribuição real



$$\theta_0 + \theta_1 x + \theta_2 x^2$$

“Just right”



$$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$$

High variance
(overfit)

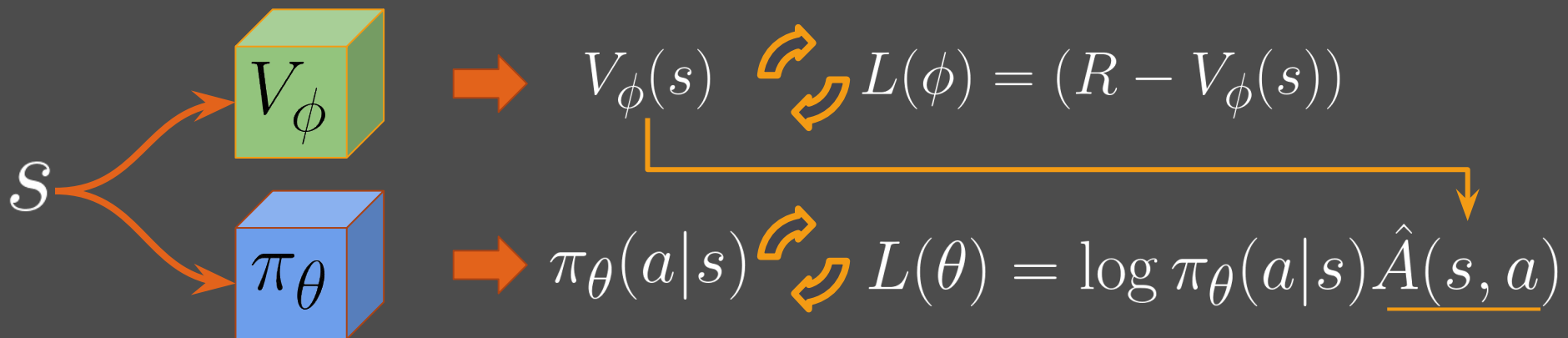
Como balancear?

- Já falamos do *Baseline*
- Advantage Function
 - Retorno completo

$$A(s_t, a_t) = \underline{R_t} - \underline{V(s_t)}$$


- Estimate of the Advantage Function
 - Bootstrapping

Resumindo o Actor-Critic



Loop:

- Coleta de trajetórias
- Cálculo de *Advantages*
- Atualização da Política
- Atualização da Função de Valor
- Descarte de trajetórias (on-policy)

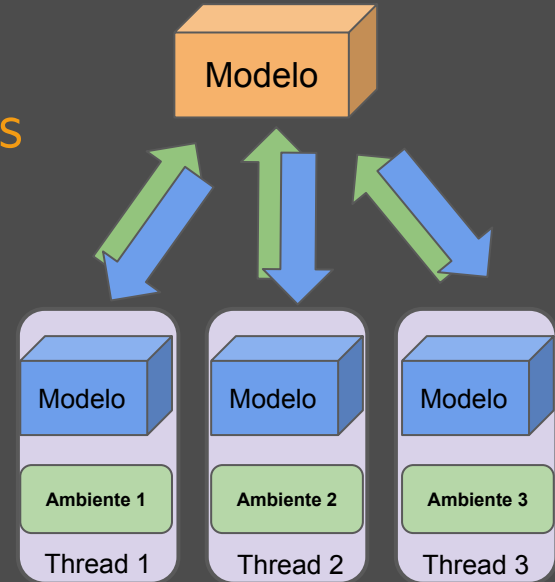


Asynchronous Advantage Actor-Critic (A3C)

Asynchronous Methods for Deep Reinforcement Learning
<https://arxiv.org/pdf/1602.01783.pdf>

Paralelização de Ambientes

- Múltiplas instâncias do ambiente
- Cada uma com uma cópia do modelo
- Experiências coletadas treinam os pesos
 - Experiências diferentes, não correlacionadas
- Modelos são atualizados de forma assíncrona
- Mais rápido e eficiente que o replay de experiências



Resultados do Paper

Method	Training Time	Mean	Median
DQN	8 days on GPU	121.9%	47.5%
Gorila	4 days, 100 machines	215.2%	71.3%
D-DQN	8 days on GPU	332.9%	110.9%
Dueling D-DQN	8 days on GPU	343.8%	117.1%
Prioritized DQN	8 days on GPU	463.6%	127.6%
A3C, FF	1 day on CPU	344.1%	68.2%
A3C, FF	4 days on CPU	496.8%	116.6%
A3C, LSTM	4 days on CPU	623.0%	112.6%

Table 1. Mean and median human-normalized scores on 57 Atari games using the human starts evaluation metric. Supplementary Table SS3 shows the raw scores for all games.

Algoritmo

- Criar Política π
- Criar Função de Valor V
- Instanciar ambientes paralelos com cópias de π e V
- Manter uma cópia de π e V central/mestre/global
- Loop (em cada ambiente, de forma assíncrona):
 - Coletar n passos/experiências no ambiente (ex: 8)
 - Calcular advantages
 - Calcular gradientes para os parâmetros de π e V locais
 - Enviar os gradientes para atualizar π e V globais
 - Atualizar os parâmetros de π e V locais

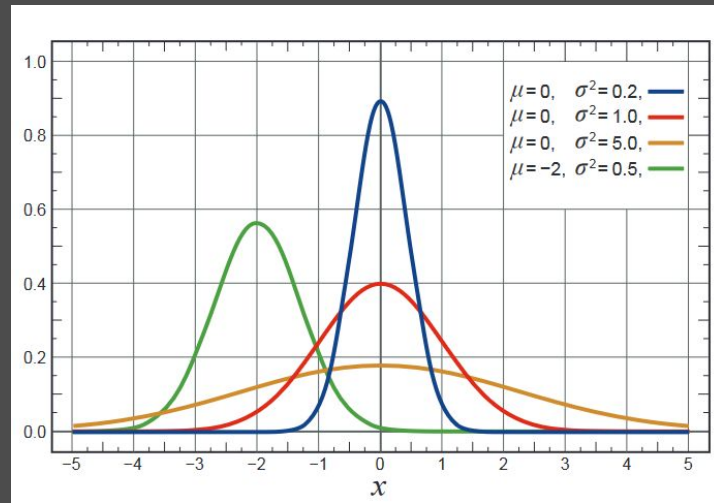
Bônus: Controle Contínuo


Saída da Rede:

- Média
- Desvio Padrão

Monta-se uma distribuição Gaussiana

Ação amostrada a partir da distribuição





Proximal Policy Optimization (PPO)

Proximal Policy Optimization Algorithms
<https://arxiv.org/pdf/1707.06347.pdf>

A Razão de Probabilidade

- Conservative Policy Iteration

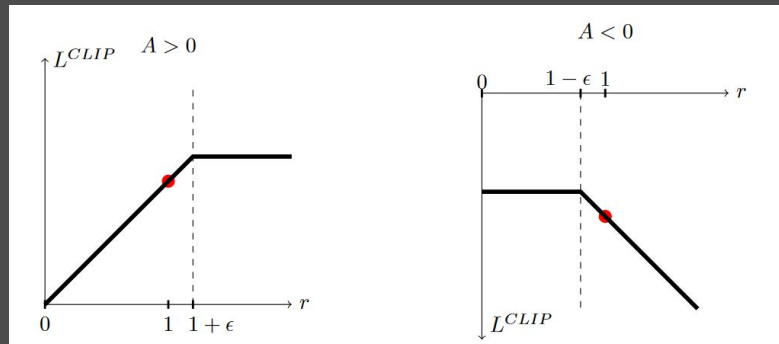
$$L(\theta) = \mathbb{E} \left[\frac{\pi_{\theta}(a_t|s_t)}{\pi_{\theta_{old}}(a_t|s_t)} \hat{A}_t \right] = \mathbb{E} \left[r_t(\theta) \hat{A}_t \right]$$

PPO-clip

- Corte na proporção

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

- Ignora-se o “clip” quando a probabilidade atual está baixa para uma vantagem boa
- Ignora-se o “clip” quando a probabilidade atual está alta para uma vantagem ruim



*Situações que precisam de correção

Generalized Advantage Estimation

- E se a gente pudesse descontar *one-step advantages*?

$$GAE = \hat{A}_t + (\gamma\lambda)\hat{A}_{t+1} + (\gamma\lambda)^2\hat{A}_{t+2}$$

$$\hat{A}_t = r_{t+1} + \gamma V(s_{t+1}) - V(s_t)$$

Bias Variance Balancing

$$0 < \lambda < 1$$

$$GAE = r_{t+1} + \gamma V(s_{t+1}) - V(s_t) + \gamma \lambda r_{t+2} + \gamma^2 \lambda V(s_{t+2}) - \gamma \lambda V(s_{t+1})$$

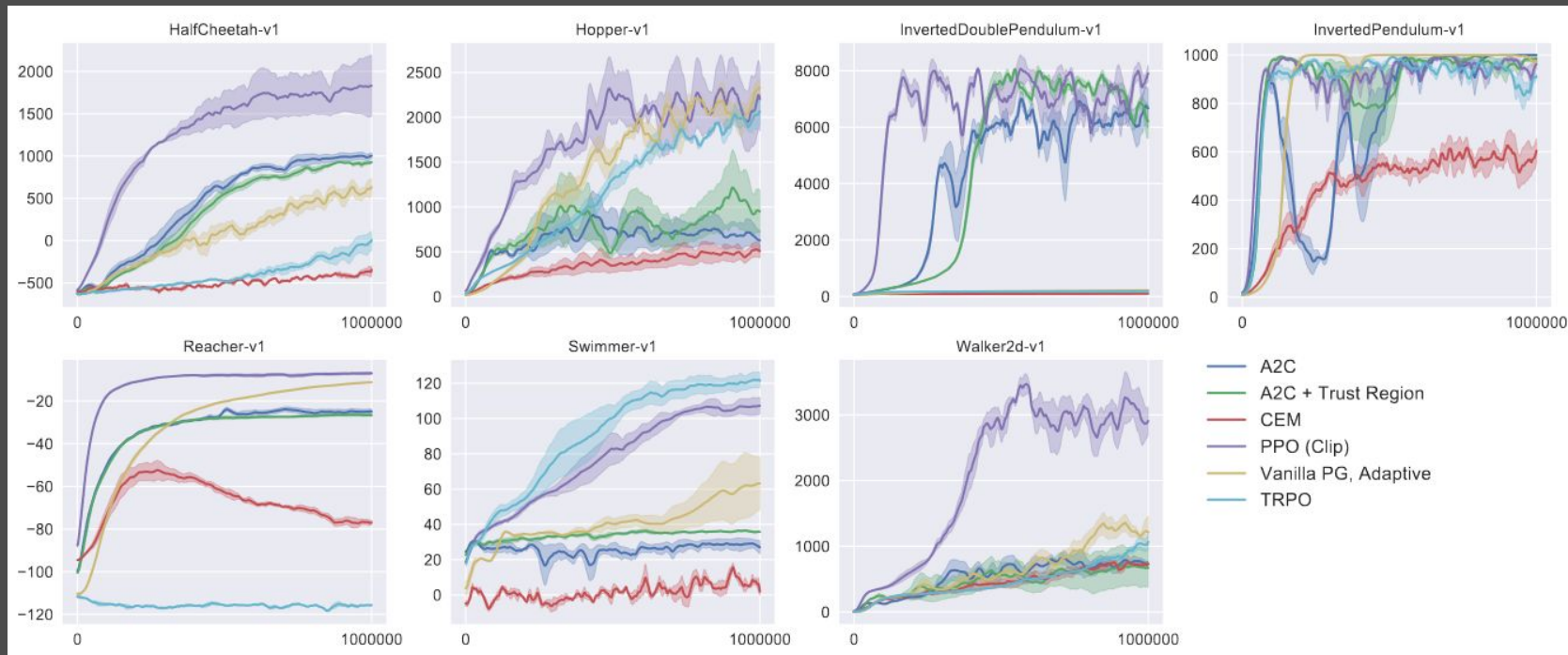
$$\lambda = 1$$

$$GAE = r_{t+1} + \gamma r_{t+2} + \gamma^2 V(s_{t+2}) - V(s_t)$$

$$\lambda = 0$$

$$GAE = r_{t+1} + \gamma V(s_{t+1}) - V(s_t)$$

Resultados do Paper



Algoritmo

- Criar Política π
- Criar Função de Valor V
- Loop:
 - Coletar n passos/experiências no ambiente (ex: 2048)
 - Calcular GAE
 - Para K épocas (ex: 10):
 - Atualizar π e V
 - Descarte experiências



Soft Actor-Critic (SAC)

Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement
Learning with a Stochastic Actor <https://arxiv.org/pdf/1801.01290.pdf>

Aprendendo $Q(s,a)$

- O SAC aprende duas funções Q
- Essas funções também tem suas próprias *target networks*

$$L(\phi_i, \mathcal{D}) = \mathbb{E}_{(s,a,r,s',d) \sim \mathcal{D}} \left[\left(Q_{\phi_i}(s, a) - y(r, s', d) \right)^2 \right]$$

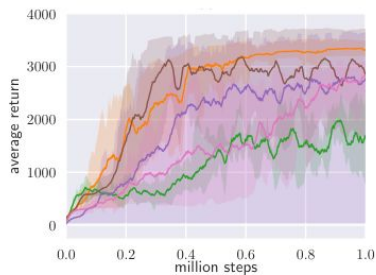
$$y(r, s', d) = r + \gamma(1 - d) \left(\min_{j=1,2} Q_{\phi_{\text{targ},j}}(s', \tilde{a}') - \alpha \log \pi_{\theta}(\tilde{a}'|s') \right), \quad \tilde{a}' \sim \pi_{\theta}(\cdot|s').$$

Aprendendo pi

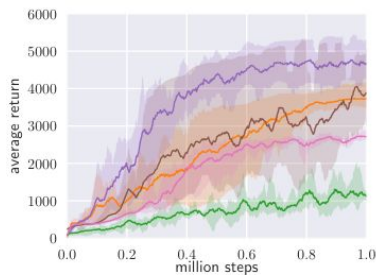
- Usa-se a estimativa Q dada a ação da política atual para dizer o quão boa está a política (loss function)

$$\max_{\theta} \mathbb{E}_{\substack{s \sim \mathcal{D} \\ \xi \sim \mathcal{N}}} \left[\min_{j=1,2} Q_{\phi_j}(s, \tilde{a}_{\theta}(s, \xi)) - \alpha \log \pi_{\theta}(\tilde{a}_{\theta}(s, \xi) | s) \right]$$

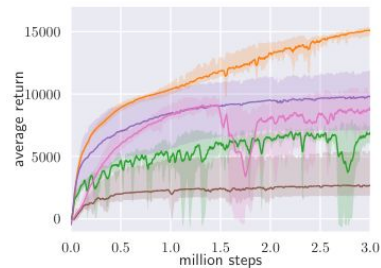
Resultados no Paper



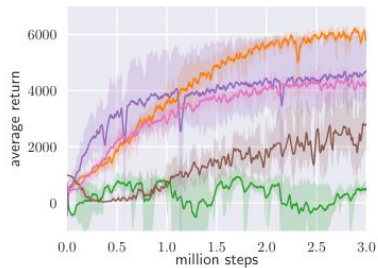
(a) Hopper-v1



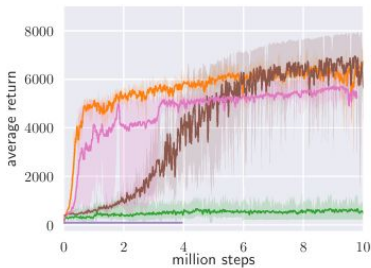
(b) Walker2d-v1



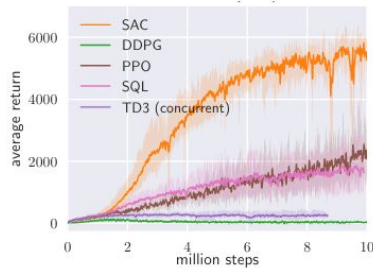
(c) HalfCheetah-v1



(d) Ant-v1



(e) Humanoid-v1



(f) Humanoid (rllab)



Comparações e Insights

Comparativo

A3C

- On-Policy
- Paralelização
- Ineficiente por amostras

PPO

- On-Policy
- Paralelização
- Mais eficiente que o A3C
- Sensível a Hiperparâmetros

SAC

- Off-Policy
- Replay Buffer
- Off-Policy sempre terão melhor eficiência por amostra



**Leiam os
trabalhos e usem
o RAY[rllib]**