Reinforcement Learning applied to IEEE's VSS Soccer Strategy

Thiago Filipe de Medeiros

Orientador: Prof. Dr. Marcos R. O. A. Máximo
Co-orientador: Prof. Dr. Takashi Yoneyama
Convidado: Prof. Dr. Carlos H. Q. Forster

Roteiro

- **Introdução**
- ***Supervised Learning***
- ***Reinforcement Learning***
- **Objetivos**
- **Metodologia**
- **Trabalhos Futuros (Cronograma)**

Roteiro

- **Introdução**
- ***Supervised Learning***
- ***Reinforcement Learning***
- **Objetivos**
- **Metodologia**
- **Trabalhos Futuros (Cronograma)**

Introdução: IEEE

Institute of Electrical and Eletronics Engineers (IEEE)

Introdução: IEEE

Institute of Electrical and Eletronics Engineers (IEEE)

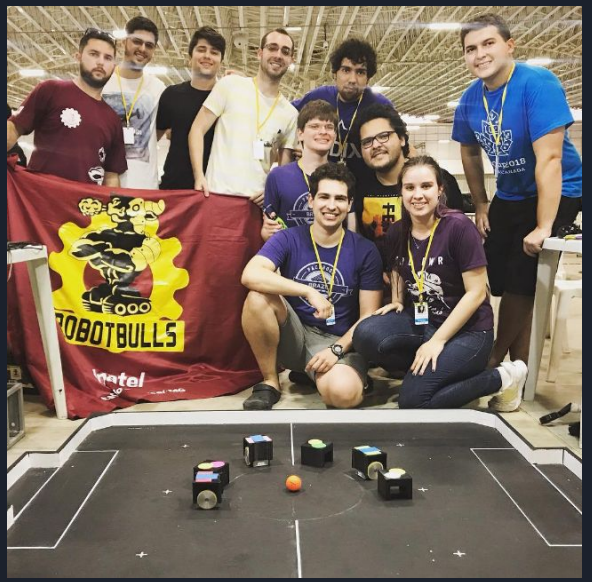
Introdução: VSS Soccer

Institute of Electrical and Eletronics Engineers (IEEE)

Very Small Size (VSS) Soccer

Introdução: VSS Soccer

Nov/2018



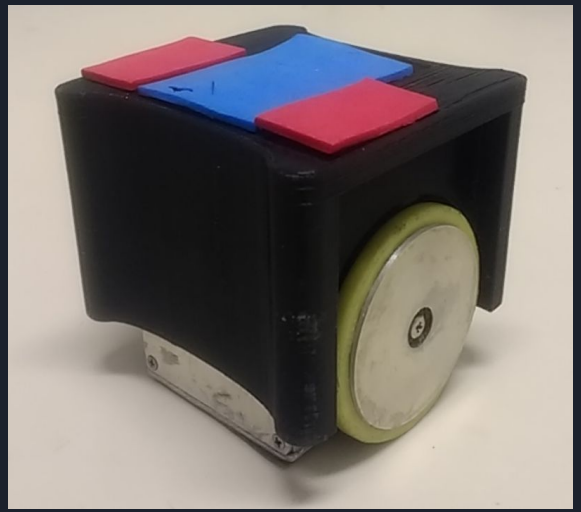
Very Small Size (VSS) Soccer

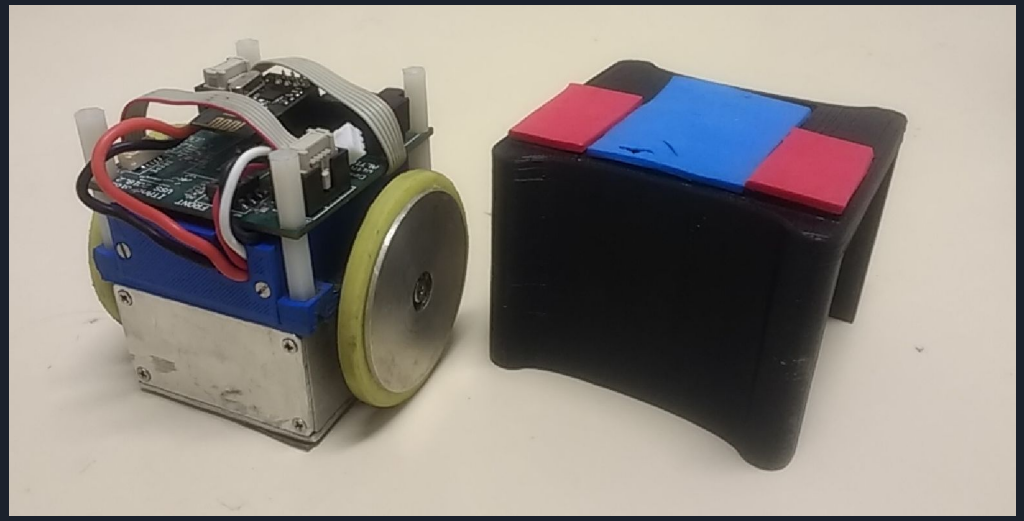
Institute of Electrical and Eletronics Engineers (IEEE)

IEEE Robotics & Automation Society

Latin America Robot Competition (LARC)

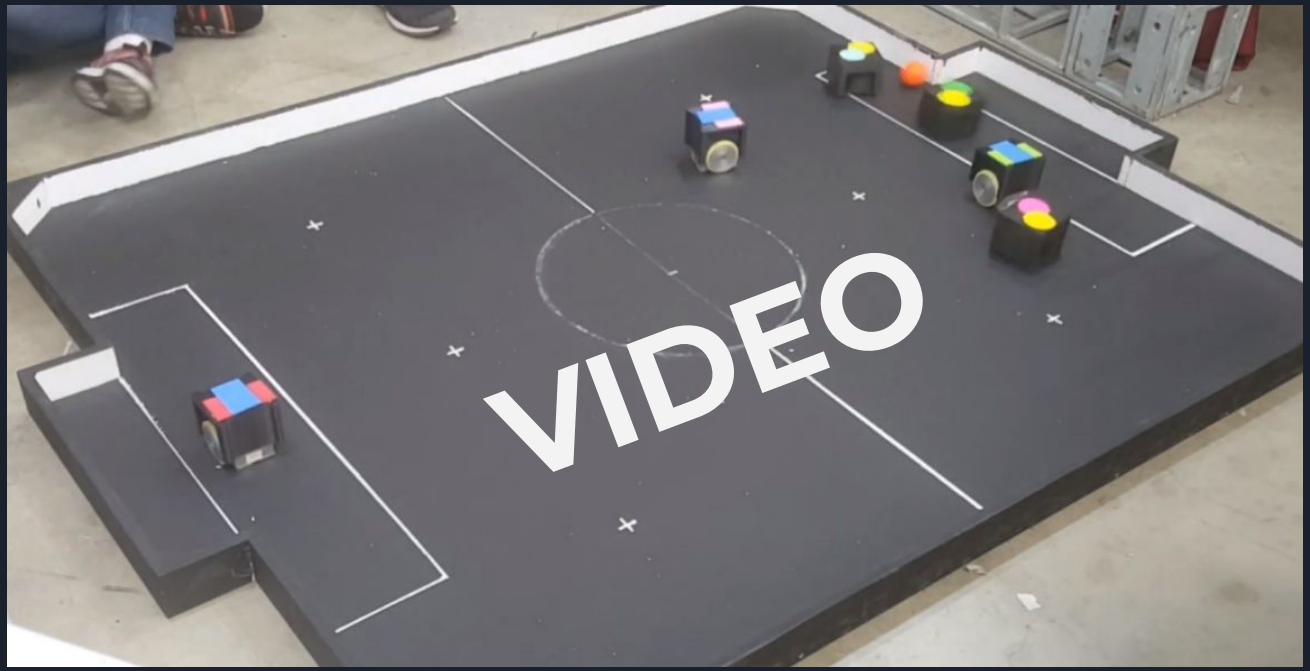
Introdução: VSS Soccer



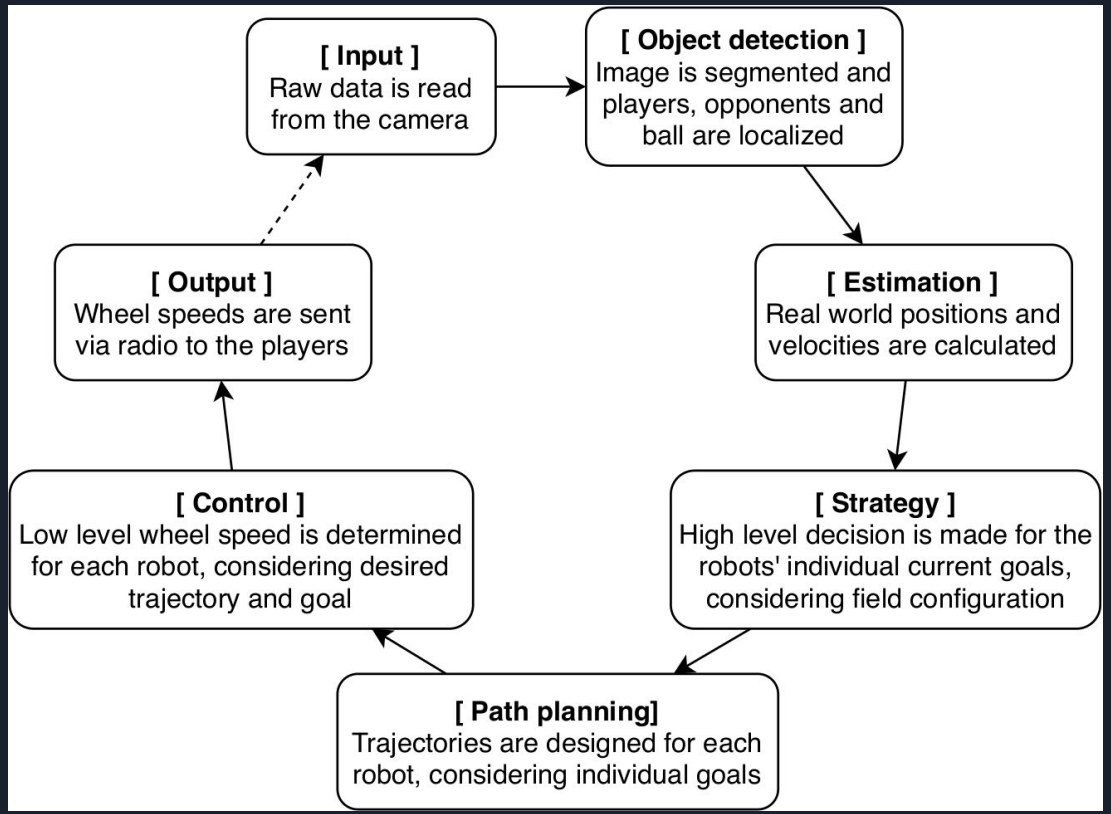


Introdução: VSS Soccer

Introdução: VSS Soccer

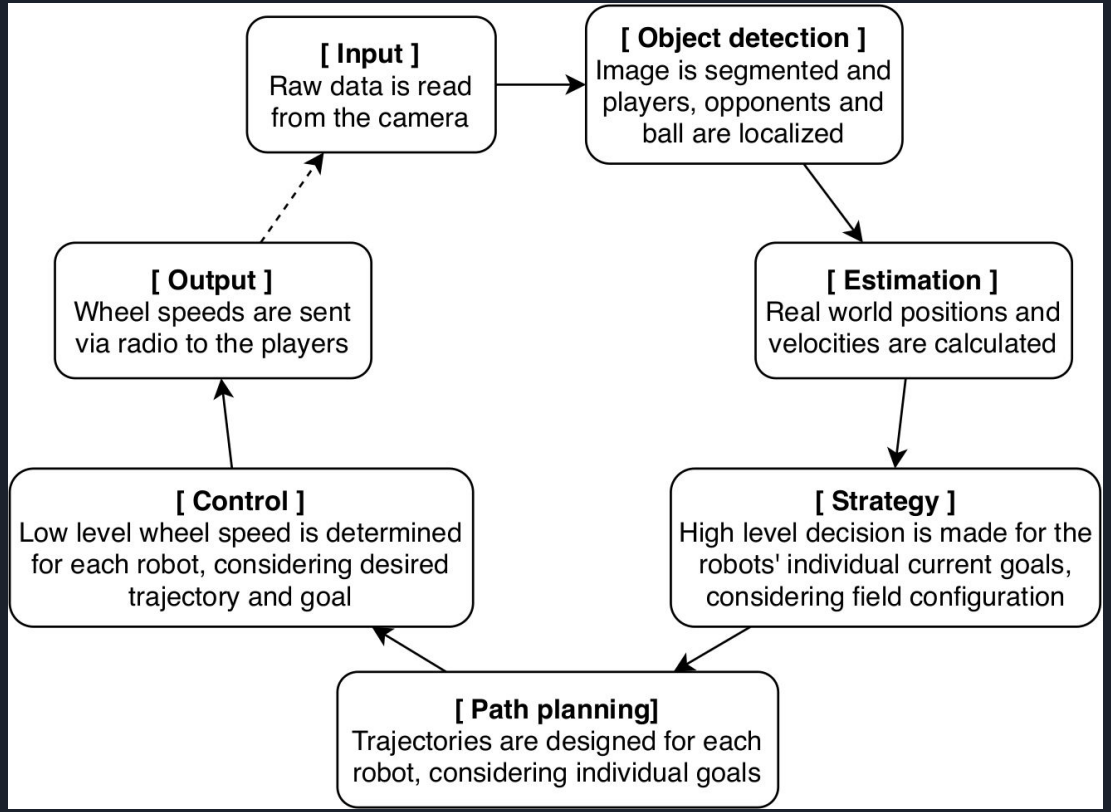


Introdução: VSS Soccer

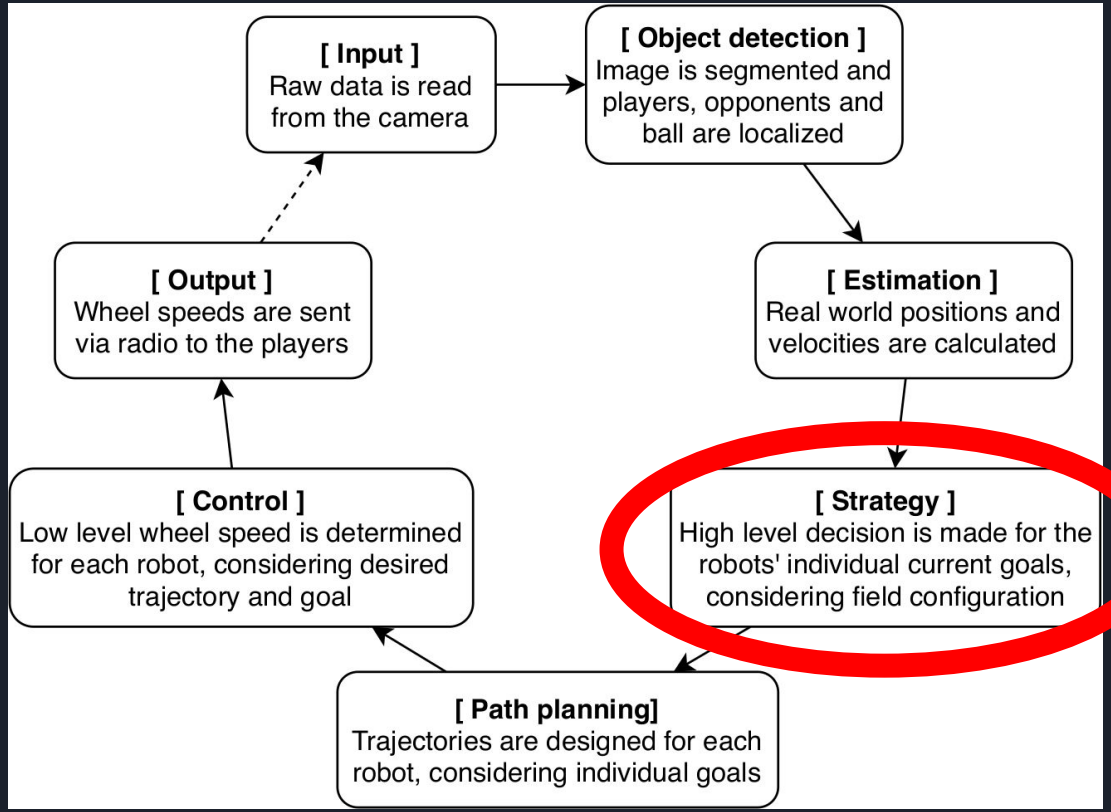


Introdução: VSS Soccer

Ocorre várias vezes por segundo (60 Hz)



Introdução: VSS Soccer

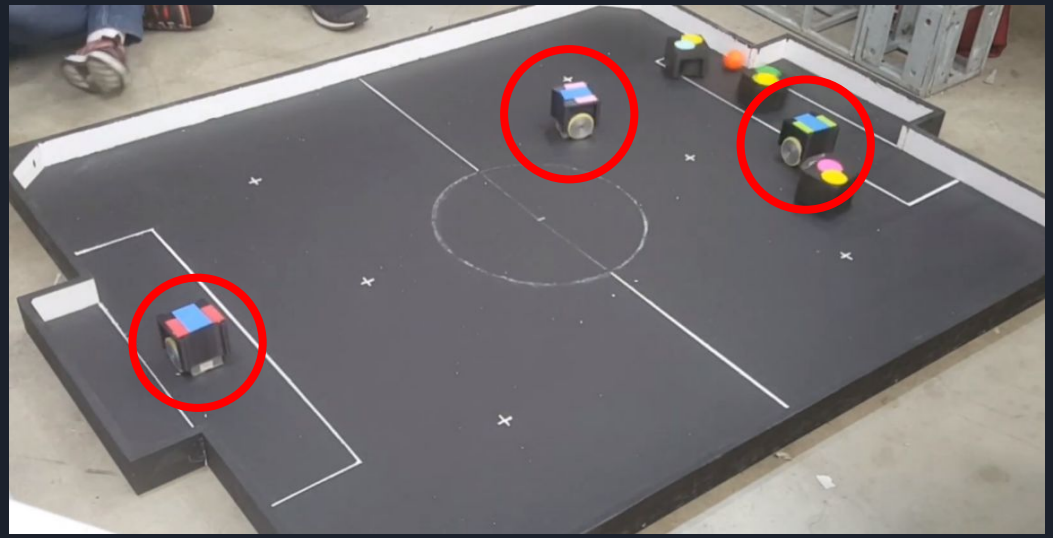


Introdução: VSS Soccer

Configurações (Estados)

- **Posições**
- **Velocidades**

Introdução: VSS Soccer

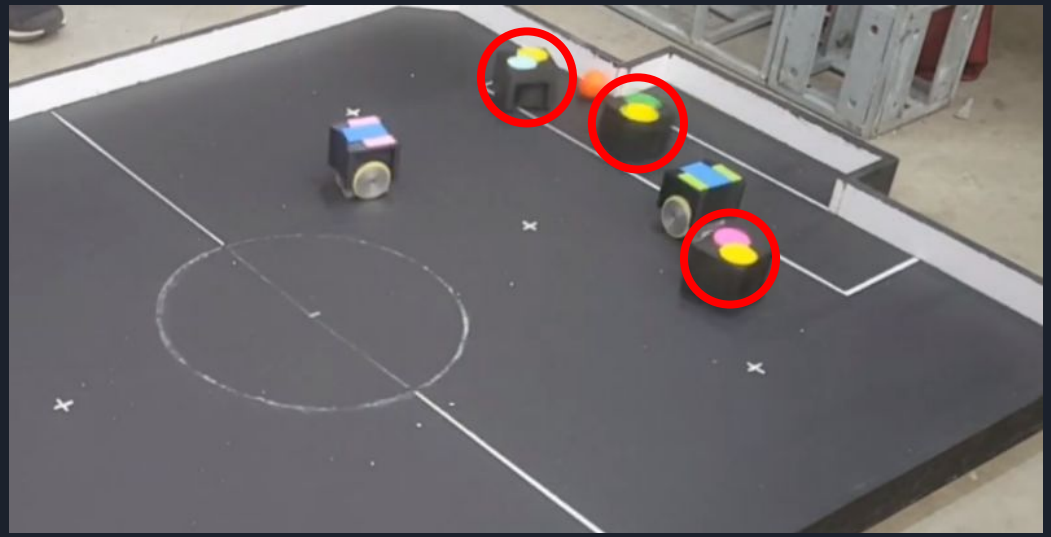


Configurações (Estados)

- **Posições**
- **Velocidades**

- **Jogadores**

Introdução: VSS Soccer



Configurações (Estados)

- **Posições**
- **Velocidades**

- **Jogadores**
- **Oponentes**

Introdução: VSS Soccer

Configurações (Estados)

- **Posições**
- **Velocidades**

- **Jogadores**
- **Oponentes**
- **Bola**

Introdução: VSS Soccer

Ações (Alto Nível):

- **Posições desejadas**
- **Orientações**

Introdução: VSS Soccer

Atualmente:

- **Estratégia baseada em heurísticas, funciona bem para 3 vs 3, mas não escala facilmente para 5 vs 5**

Introdução: VSS Soccer

***Reinforcement Learning*:**

- **Modela o comportamento de um agente e proporciona formas de melhorar tal comportamento, dada uma avaliação objetiva do que é "melhor" (função de recompensa).**

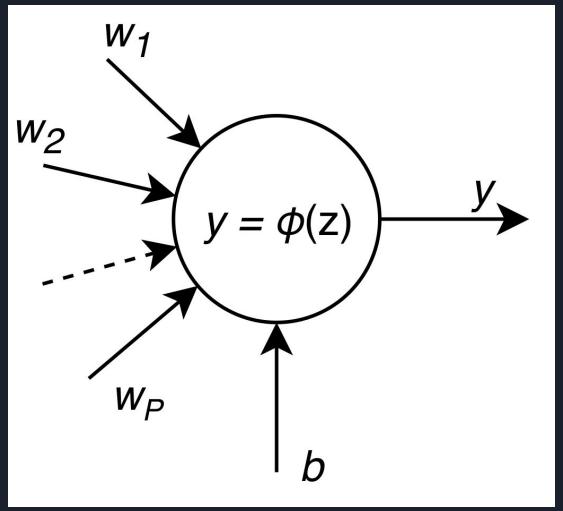
Roteiro

- **Introdução**
- ***Supervised Learning***
- ***Reinforcement Learning***
- **Objetivos**
- **Metodologia**
- **Trabalhos Futuros (Cronograma)**

Roteiro

- Introdução
- ***Supervised Learning***
- ***Reinforcement Learning***
- Objetivos
- Metodologia
- Trabalhos Futuros (Cronograma)

Supervised Learning: Perceptron

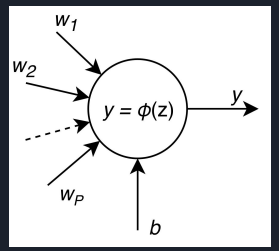


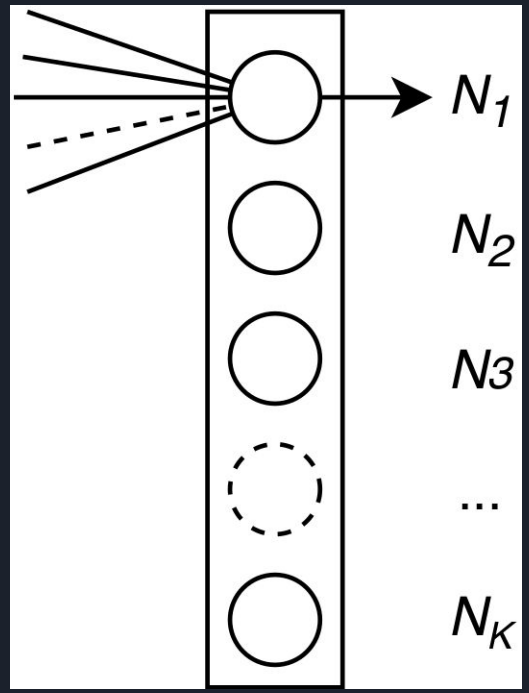
$$z = \left(\sum_i w_i y_i \right) + b$$

$$y = \phi(z)$$

Neurônio Artificial (Perceptron)

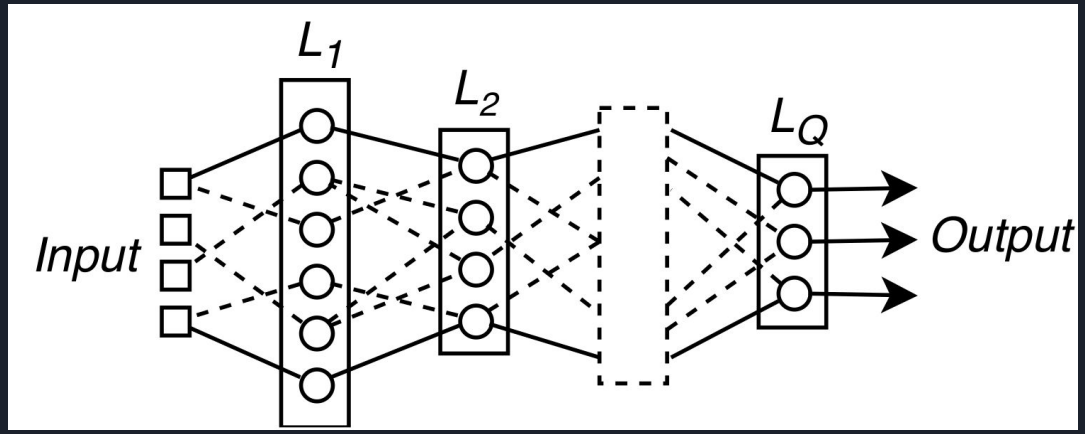
Supervised Learning: Perceptron



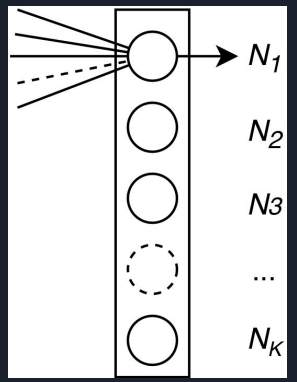


Camada de neurônios

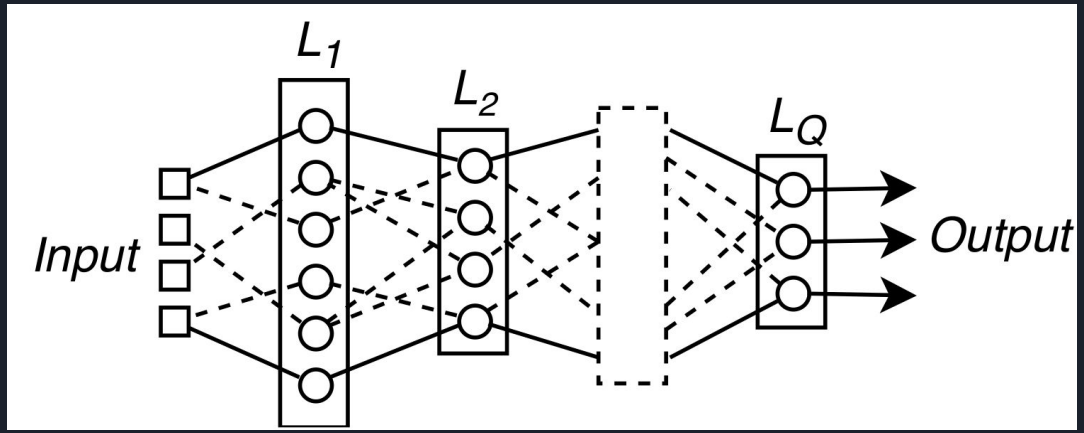
Supervised Learning: M.L.P.



Redes Feedforward
ou
Multi-Layer
Perceptron (MLP)

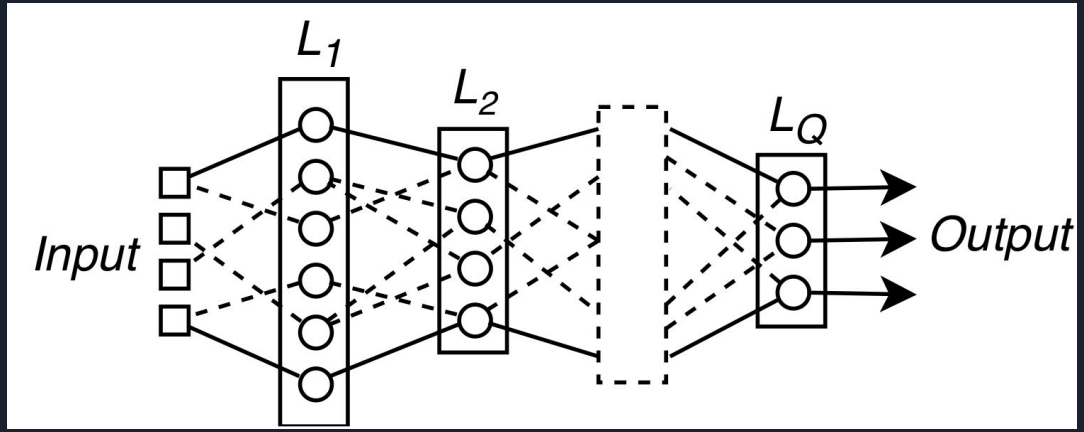


Supervised Learning: Dinâmica



Predição

Supervised Learning: Dinâmica



Otimização (Aprendizagem)

Backpropagation
+
Gradient Descent

Supervised Learning: Treinamento

$J(W)$ **Função de custo (Escalar)**

Supervised Learning: Treinamento

$$J(W)$$

Função de custo (Escalar)

$$W^{(l)} \leftarrow W^{(l)} - \alpha \frac{\partial J(W)}{\partial W^{(l)}}$$

Gradiente Descendente

Supervised Learning: Treinamento

$$J(W) = -\mathbb{E}_{x,y \sim \hat{p}_{\text{data}}} \{log\,(p_{\text{model}}(y|x))\}$$

Max Likelihood Estimation (MLE)

Supervised Learning: Treinamento

$$J(W) = -\mathbb{E}_{x,y \sim \hat{p}_{\text{data}}} \{log\,(p_{\text{model}}(y|x))\}$$

Max Likelihood Estimation (MLE)

$$p_{\text{model}}(y|x) = \mathcal{N}\left(y; \hat{y}(x), \sigma^2 I\right)$$

Distribuição Gaussiana

Supervised Learning: Treinamento

$$J(W) = -\mathbb{E}_{x,y \sim \hat{p}_{\text{data}}} \{log\left(p_{\text{model}}(y|x)\right)\}$$

Max Likelihood Estimation (MLE)

$$p_{\text{model}}(y|x) = \mathcal{N}\left(y; \hat{y}(x), \sigma^2 I\right)$$

Distribuição Gaussiana

$$J(W) = \frac{1}{2}\mathbb{E}_{x,y \sim \hat{p}_{\text{data}}} ||y - \hat{y}(x)||^2$$

Supervised Learning: Treinamento

$$J(W) = -\mathbb{E}_{x,y \sim \hat{p}_{\text{data}}} \{log\left(p_{\text{model}}(y|x)\right)\}$$

Max Likelihood Estimation (MLE)

$$p_{\text{model}}(y|x) = \mathcal{N}\left(y; \hat{y}(x), \sigma^2 I\right)$$

Distribuição Gaussiana

$$J(W) = \frac{1}{2}\mathbb{E}_{x,y \sim \hat{p}_{\text{data}}} ||y - \hat{y}(x)||^2$$

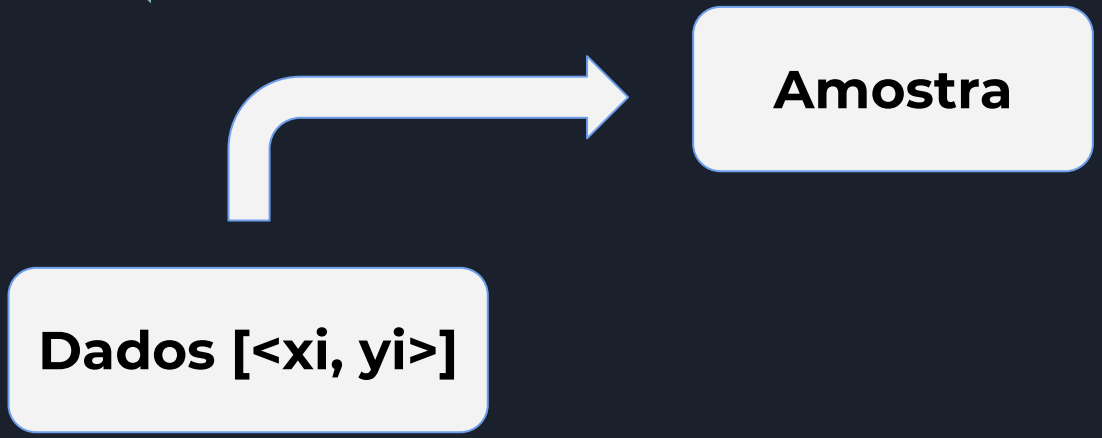
$$J(W) = \frac{1}{2m}\sum_{i}^{m} ||y_i - \hat{y}(x_i)||^2$$

Mean Squared Error (MSE)

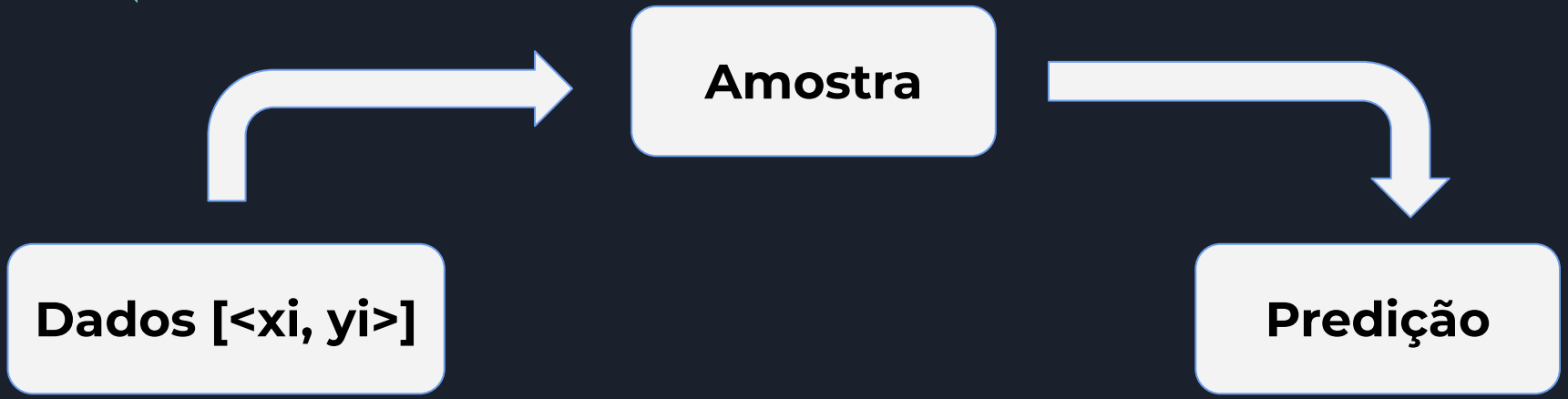
Supervised Learning: Treinamento

Dados [<xi, yi>]

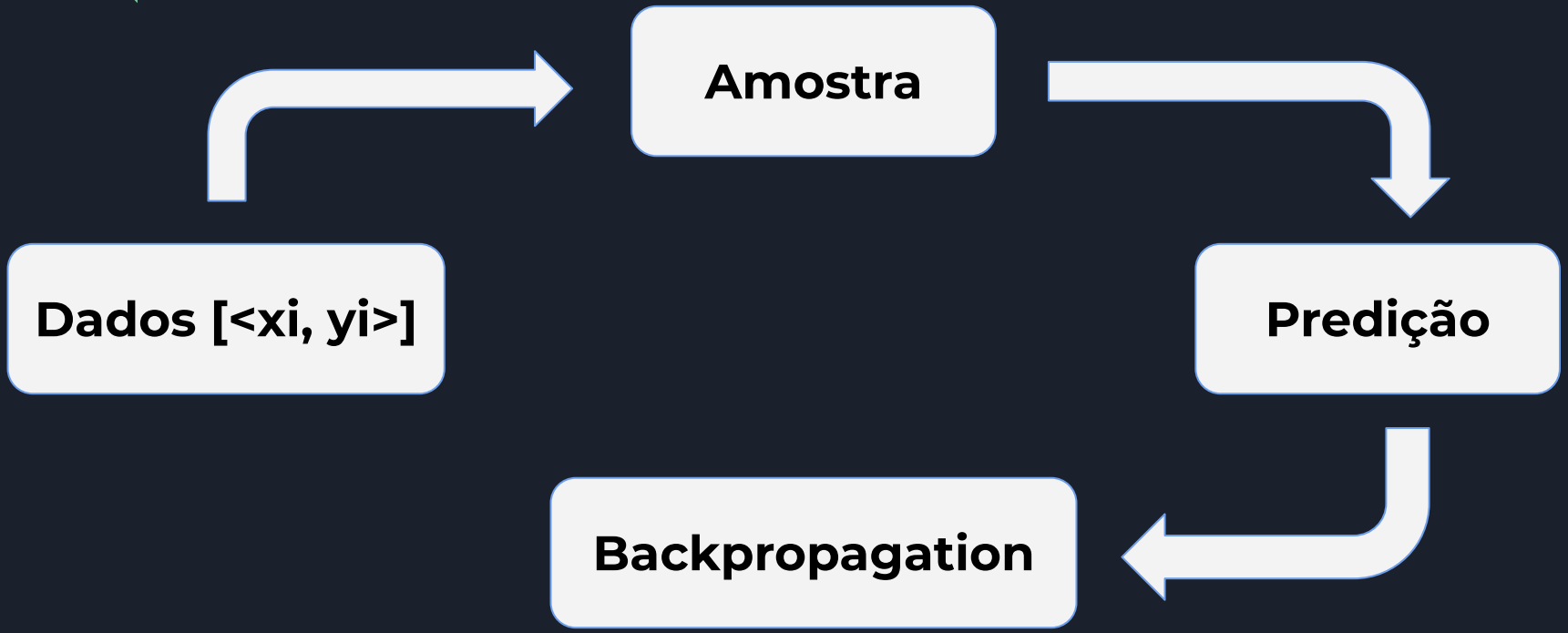
Supervised Learning: Treinamento



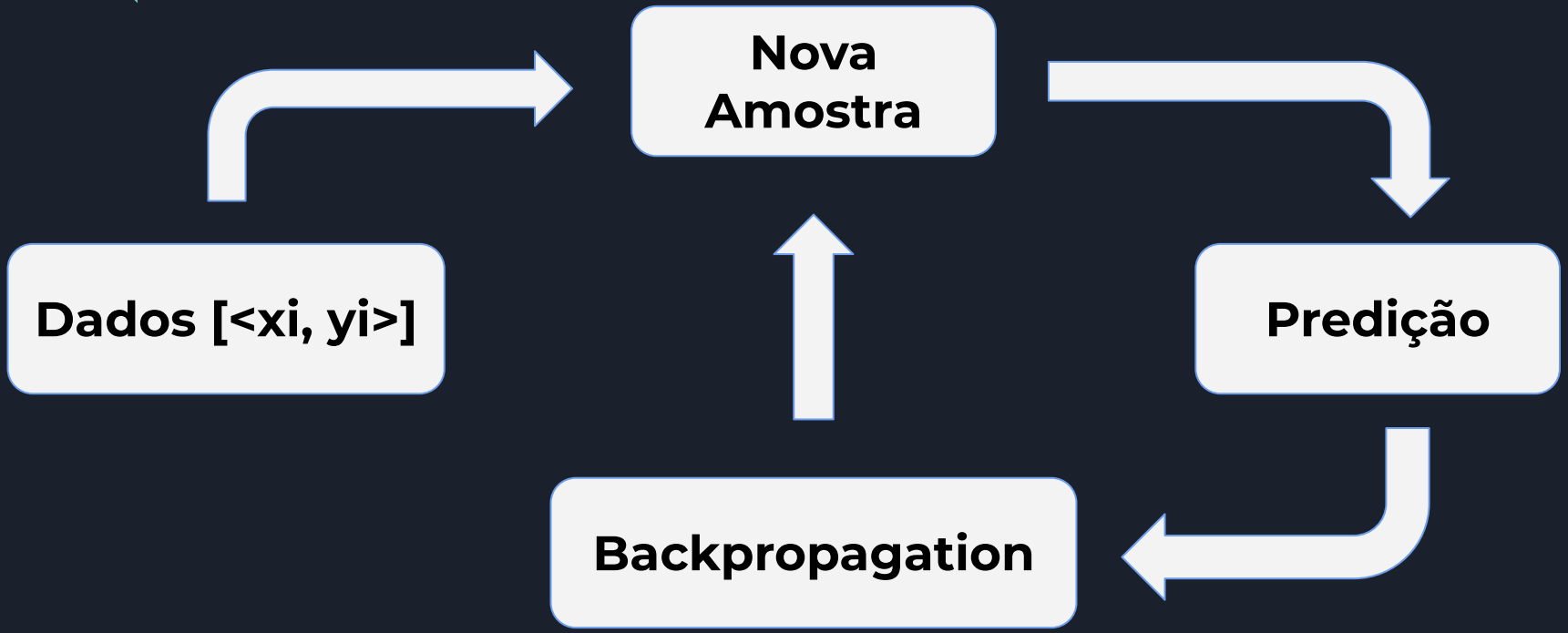
Supervised Learning: Treinamento



Supervised Learning: Treinamento



Supervised Learning: Treinamento



Supervised Learning: D.L.

Maior volume de dados

Para criar o *dataset* é necessário classificação prévia

Supervised Learning: D.L.

Maior volume de dados

Novas técnicas de Machine Learning

Redes com muitas camadas são difíceis de treinar (pesos divergem ou convergem para 0)

Supervised Learning: D.L.

Maior volume de dados + **Novas técnicas de Machine Learning** + **Mais poder computacional**

Processamento paralelo e *hardware* mais poderoso

Supervised Learning: D.L.

Maior volume de dados + **Novas técnicas de Machine Learning** + **Mais poder computacional**

=

Deep Learning (DL)

Supervised Learning: D.L.

Maior volume de dados + **Novas técnicas de Machine Learning** + **Mais poder computacional**

=

Deep Learning (DL)

Mais camadas, maior poder de representação

Roteiro

- Introdução
- ***Supervised Learning***
- ***Reinforcement Learning***
- Objetivos
- Metodologia
- Trabalhos Futuros (Cronograma)

Roteiro

- Introdução
- *Supervised Learning*
- ***Reinforcement Learning***
- Objetivos
- Metodologia
- Trabalhos Futuros (Cronograma)

Reinforcement Learning (RL)

Espaço de estados $\mathcal{S}$

Reinforcement Learning (RL)

Espaço de estados $\mathcal{S}$

Espaço de ações $\mathcal{A}$

Reinforcement Learning (RL)

Espaço de estados $\mathcal{S}$

Espaço de ações $\mathcal{A}$

Função de recompensa $\mathcal{R}$ $r(s, a, s')$

Reinforcement Learning (RL)

Espaço de estados $\mathcal{S}$

Espaço de ações $\mathcal{A}$

Função de recompensa $\mathcal{R}$ $r(s, a, s')$

Dinâmica do sistema $\mathcal{P}$ $p(s'|s, a)$

Reinforcement Learning (RL)

Processo Decisório de Markov (Markov Decision Process, MDP)

Espaço de estados $\mathcal{S}$	
Espaço de ações $\mathcal{A}$	
Função de recompensa $\mathcal{R}$	$r(s, a, s')$
Dinâmica do sistema $\mathcal{P}$	$p(s'\|s, a)$
Fator de desconto $\gamma \in [0, 1]$	

Reinforcement Learning (RL)

Espaço de estados $\mathcal{S}$

Espaço de ações $\mathcal{A}$

Função de recompensa $\mathcal{R}$ $r(s, a, s')$

Dinâmica do sistema $\mathcal{P}$ $p(s'|s, a)$

Fator de desconto $\gamma \in [0, 1]$

Processo Decisório de Markov (Markov Decision Process, MDP)

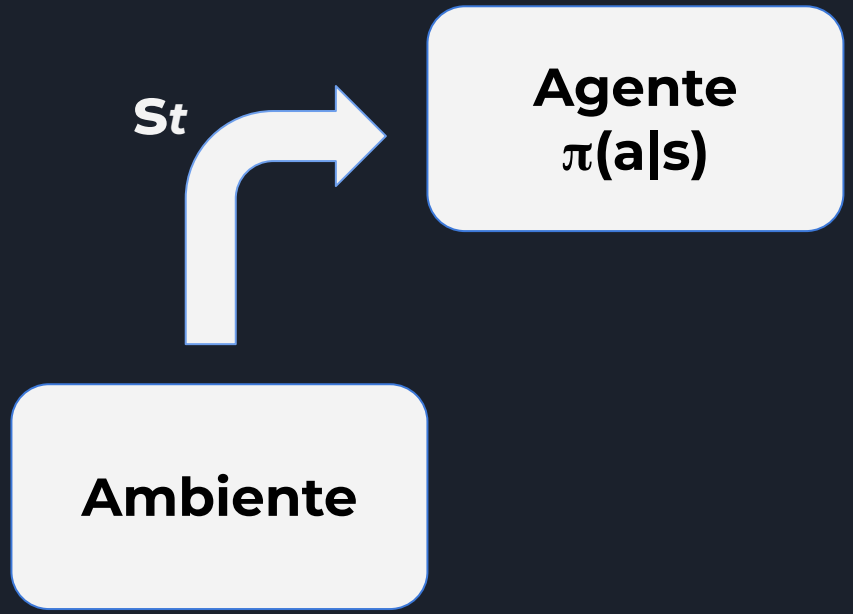
Política $\pi(a|s)$

RL: Dinâmica

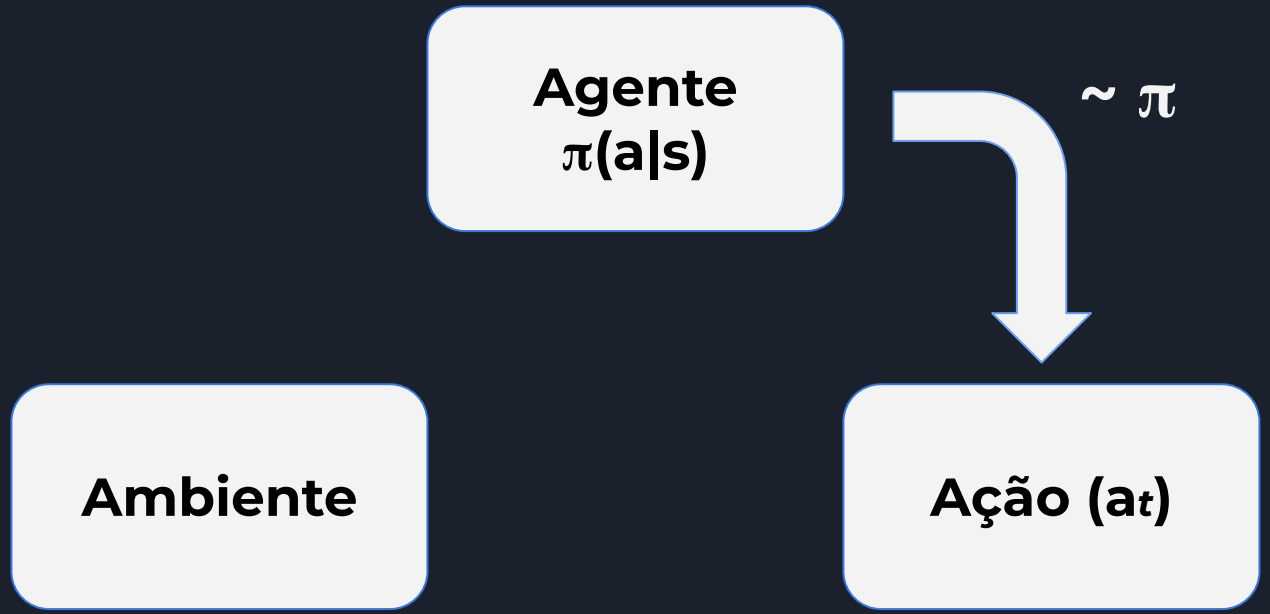
Agente
$\pi(a|s)$

Ambiente

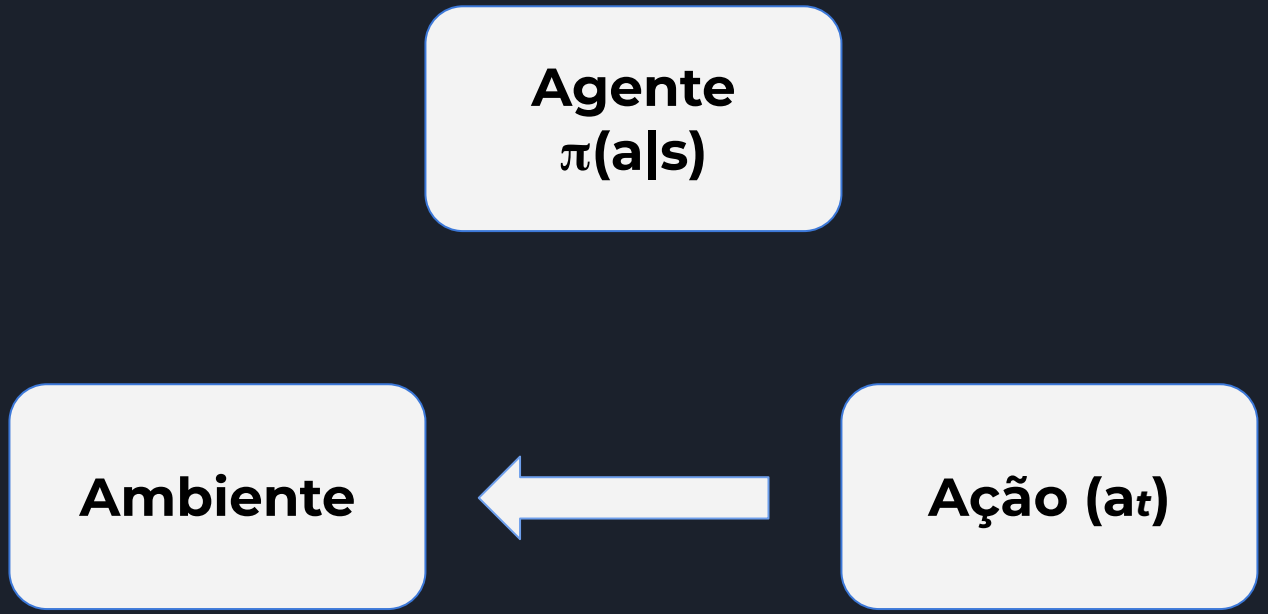
RL: Dinâmica



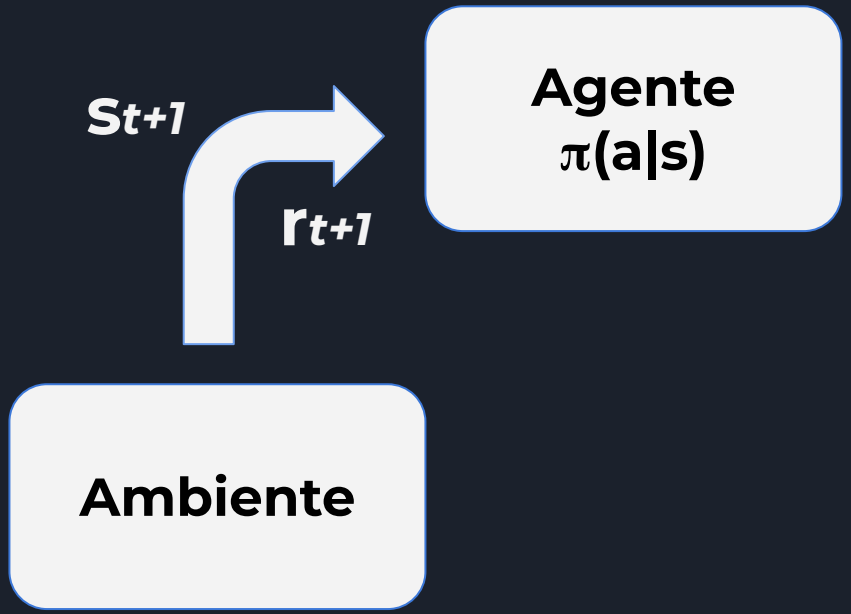
RL: Dinâmica



RL: Dinâmica



RL: Dinâmica



RL: Q-Learning

$$\text{Política } \pi(a|s)$$

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Q-Learning

$$\text{Política } \pi(a|s)$$

$$G_t = R_{t+1} + R_{t+2} + (\ldots) + R_{t+k+1} + (\ldots), k \geq 0$$

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Q-Learning

$$\text{Política } \pi(a|s)$$

$$G_t = R_{t+1} + R_{t+2} + (\ldots) + R_{t+k+1} + (\ldots), k \geq 0$$

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + (\ldots) + \gamma^k R_{t+k+1} + (\ldots)$$

$$\text{Fator de desconto } \gamma \in [0, 1]$$

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Q-Learning

$$\text{Política } \pi(a|s)$$

$$\text{Recompensa acumulada } G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Q-Learning

$$\text{Política } \pi(a|s)$$

$$\text{Recompensa acumulada } G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

$$\text{Recompensa acumulada esperada } \mathbb{E}_\pi \{G_t\}$$

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Q-Learning

$$\text{Recompensa acumulada } G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Q-Learning

$$\text{Recompensa acumulada } G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

$$Q^{\pi}(a_t, s_t) = \mathbb{E}_{\pi}\left\{G_t | s_t, a_t\right\}$$

Função Ação-Valor

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Q-Learning

$$\text{Recompensa acumulada } G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

$$Q^{\pi}(a_t, s_t) = \mathbb{E}_{\pi}\left\{G_t | s_t, a_t\right\}$$

Função Ação-Valor

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r(s_t, a_t, s_{t+1}) + \gamma \cdot \max_a Q(s_{t+1}, a) - Q(s_t, a_t)\right]$$

Aproximar Q através de iterações

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Q-Learning

$$Q^{\pi}(a_t, s_t) = \mathbb{E}_{\pi}\left\{G_t | s_t, a_t\right\}$$

Função Ação-Valor

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r(s_t, a_t, s_{t+1}) + \gamma \cdot \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right]$$

Precisa olhar todos os estados
(encontrar Q de forma iterativa)

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Q-Learning

$$Q^{\pi}(a_t, s_t) = \mathbb{E}_{\pi}\left\{G_t | s_t, a_t\right\}$$

Função Ação-Valor

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[r(s_t, a_t, s_{t+1}) + \gamma \cdot \max_a Q(s_{t+1}, a) - Q(s_t, a_t) \right]$$

Precisa olhar todos os estados

Problema: Espaço de estados grande (ou contínuo)

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Q-Learning

Configurações (Estados)

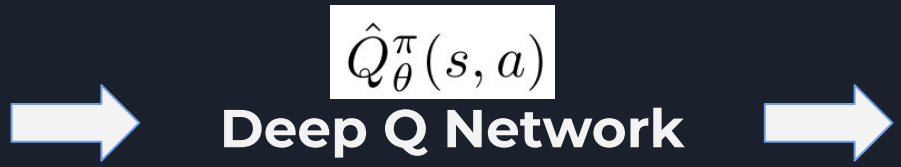
- **Posições**
- **Velocidades**

Problema: Espaço de estados grande (ou contínuo)

RL: Deep Q-Learning (DQN)

$$Q^{\pi}(a_t, s_t) = \mathbb{E}_{\pi}\{G_t | s_t, a_t\}$$

Função Ação-Valor



Alternativa: Aproximar Q com uma rede neural (avaliar sequências S, A, R, S ...)

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Deep Q-Learning (DQN)

$$Q^{\pi}(a_t, s_t) = \mathbb{E}_{\pi}\{G_t | s_t, a_t\}$$

Função Ação-Valor

$\hat{Q}^{\pi}_{\theta}(s, a)$

St → **Deep Q Network** → **Valor de Q para cada ação possível**

Alternativa: Aproximar Q com uma rede neural (... e atualizar a rede de forma supervisionada com a equação de iteração)

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Deep Q-Learning (DQN)

$$Q^{\pi}(a_t, s_t) = \mathbb{E}_{\pi}\{G_t | s_t, a_t\}$$

Função Ação-Valor

St

$$\hat{Q}^{\pi}_{\theta}(s, a)$$

Deep Q Network

Valor de Q para cada ação possível

Alternativa: Aproximar Q com uma rede neural

Problema: Espaço de ações grande (ou contínuo)

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Deep Q-Learning (DQN)

Ações (Alto Nível):

- **Posições desejadas**
- **Orientações**

Problema: Espaço de ações grande (ou contínuo)

RL: Policy Gradient

$$\pi_\theta(a|s)$$

$$J(\theta) = \mathbb{E}_{a \sim \pi_\theta}\{G_t\}$$

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Policy Gradient

$$\pi_\theta(a|s) \qquad J(\theta) = \mathbb{E}_{a \sim \pi_\theta}\{G_t\}$$

$$\theta_{t+1} \leftarrow \theta_t + \alpha^\theta \nabla J(\theta)$$

$$\nabla J(\theta) = \mathbb{E}_{a \sim \pi_\theta}\{\nabla \log(\pi_\theta)\, G_t\}$$

Gradient Ascent

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Policy Gradient

$$\pi_\theta(a|s)$$

$$J(\theta) = \mathbb{E}_{a \sim \pi_\theta}\{G_t\}$$

$$\theta_{t+1} \leftarrow \theta_t + \alpha^\theta \nabla J(\theta)$$

$$\nabla J(\theta) = \mathbb{E}_{a \sim \pi_\theta}\{\nabla \log(\pi_\theta)\, G_t\}$$

Problema: Alta variância e convergência lenta

Gradient Ascent

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic

$$\nabla J(\theta) = \mathbb{E}_{a \sim \pi_\theta} \{\nabla \log (\pi_\theta) G_t\}$$

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic

$$\nabla J(\theta) = \mathbb{E}_{a \sim \pi_\theta} \{\nabla \log (\pi_\theta) G_t\}$$

$$V^\pi(s_t) = \mathbb{E}_\pi \{G_t | s_t\}$$

Função Valor

$$Q^\pi(a_t, s_t) = \mathbb{E}_\pi \{G_t | s_t, a_t\}$$

Função Ação-Valor

$$A^\pi(a_t, s_t) = Q^\pi(a_t, s_t) - V^\pi(s_t)$$

Função Vantagem

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic

$$\nabla J(\theta) = \mathbb{E}_{a \sim \pi_\theta} \{\nabla \log(\pi_\theta) A_t^\pi\}$$

$$A^\pi(a_t, s_t) = Q^\pi(a_t, s_t) - V^\pi(s_t)$$

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic

$$\nabla J(\theta) = \mathbb{E}_{a \sim \pi_\theta} \{\nabla \log(\pi_\theta) A_t^\pi\}$$

$$A^\pi(a_t, s_t) = Q^\pi(a_t, s_t) - V^\pi(s_t)$$

$$Q^\pi(s_t, a_t) = R_t + \gamma V^\pi(s_{t+1})$$

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic

$$\nabla J(\theta) = \mathbb{E}_{a \sim \pi_\theta} \{\nabla \log(\pi_\theta) A_t^\pi\}$$

$$A^\pi(a_t, s_t) = Q^\pi(a_t, s_t) - V^\pi(s_t)$$

$$Q^\pi(s_t, a_t) = R_t + \gamma V^\pi(s_{t+1})$$

$$A_t^\pi = R_t + \gamma V^\pi(s_{t+1}) - V^\pi(s_t)$$

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic

$$A_t^\pi = R_t + \gamma V^\pi(s_{t+1}) - V^\pi(s_t)$$

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic

$$A_t^\pi = R_t + \gamma V^\pi(s_{t+1}) - V^\pi(s_t)$$

$$\hat{A}_t^\pi = R_t + \gamma \hat{V}_w^\pi(s_{t+1}) - \hat{V}_w^\pi(s_t)$$

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic

$$A_t^\pi = R_t + \gamma V^\pi(s_{t+1}) - V^\pi(s_t)$$

$$\hat{A}_t^\pi = R_t + \gamma \hat{V}_w^\pi(s_{t+1}) - \hat{V}_w^\pi(s_t)$$

$$J(w) = \frac{1}{2}\left(R_t + \gamma \hat{V}_w^\pi(s_{t+1}) - \hat{V}_w^\pi(s_t)\right)^2$$

$$w_{t+1} \leftarrow w_t - \alpha^w \nabla J(w)$$

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic (Treinamento)

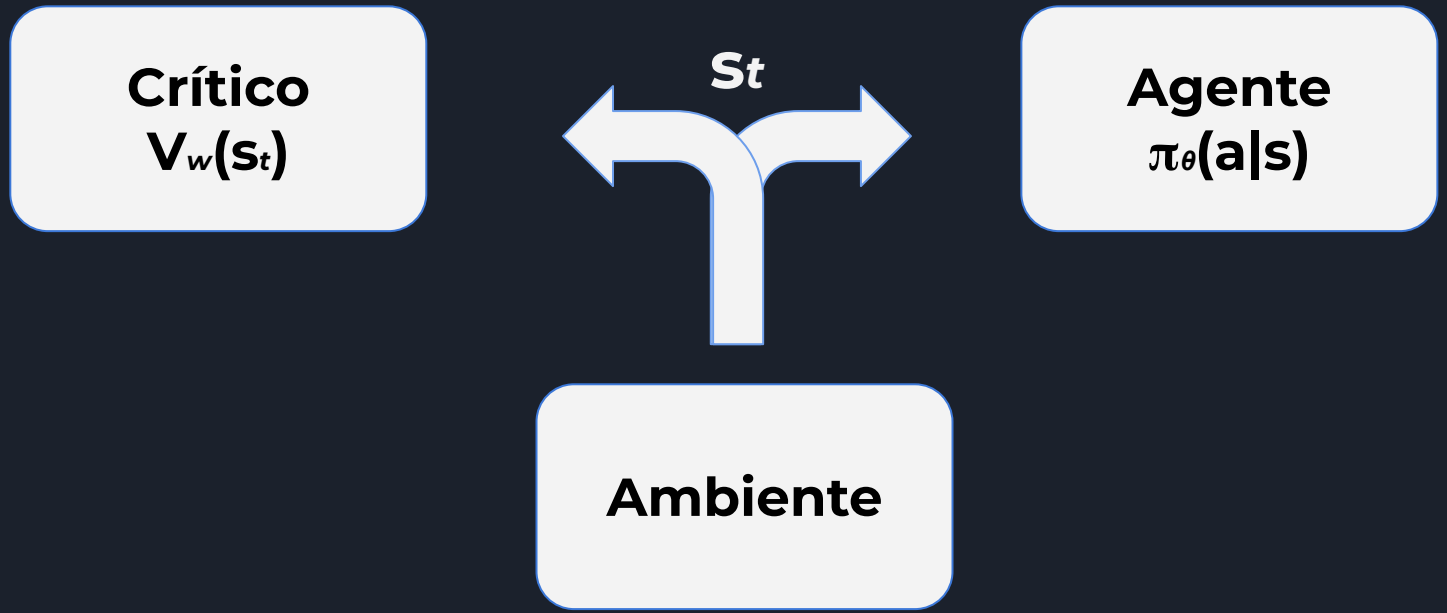
Crítico
$V_w(s_t)$

Agente
$\pi_\theta(a|s)$

Ambiente

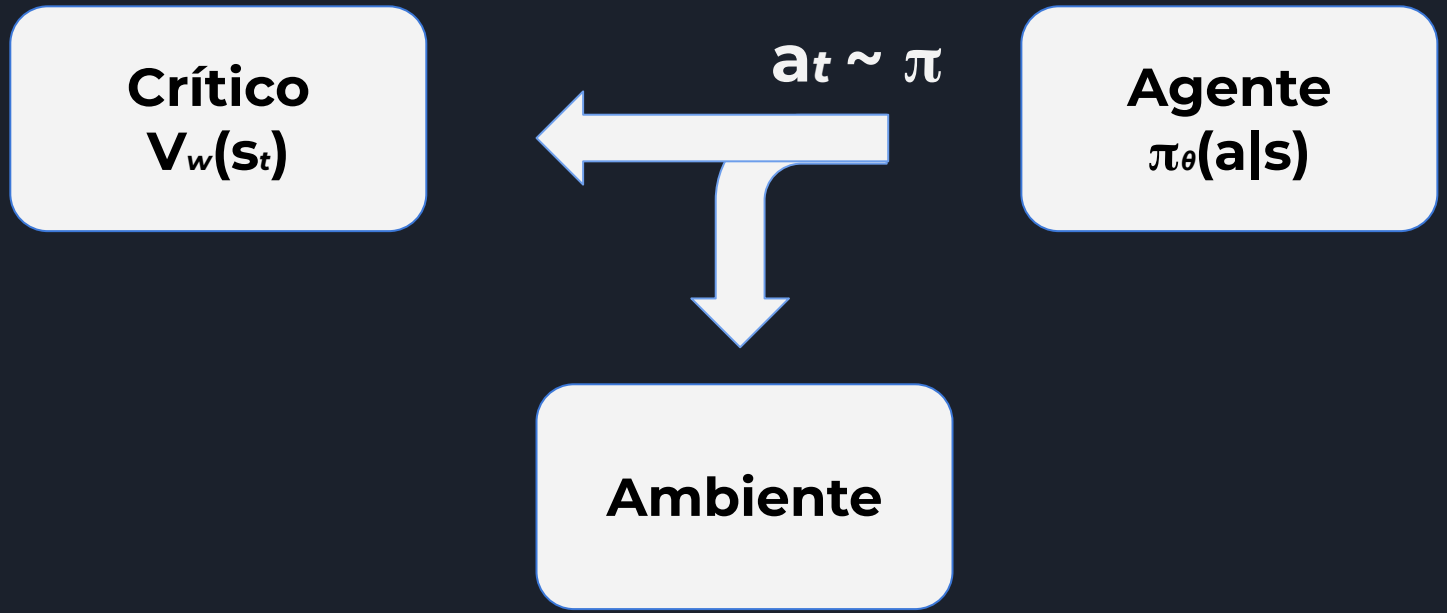
Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic (Treinamento)



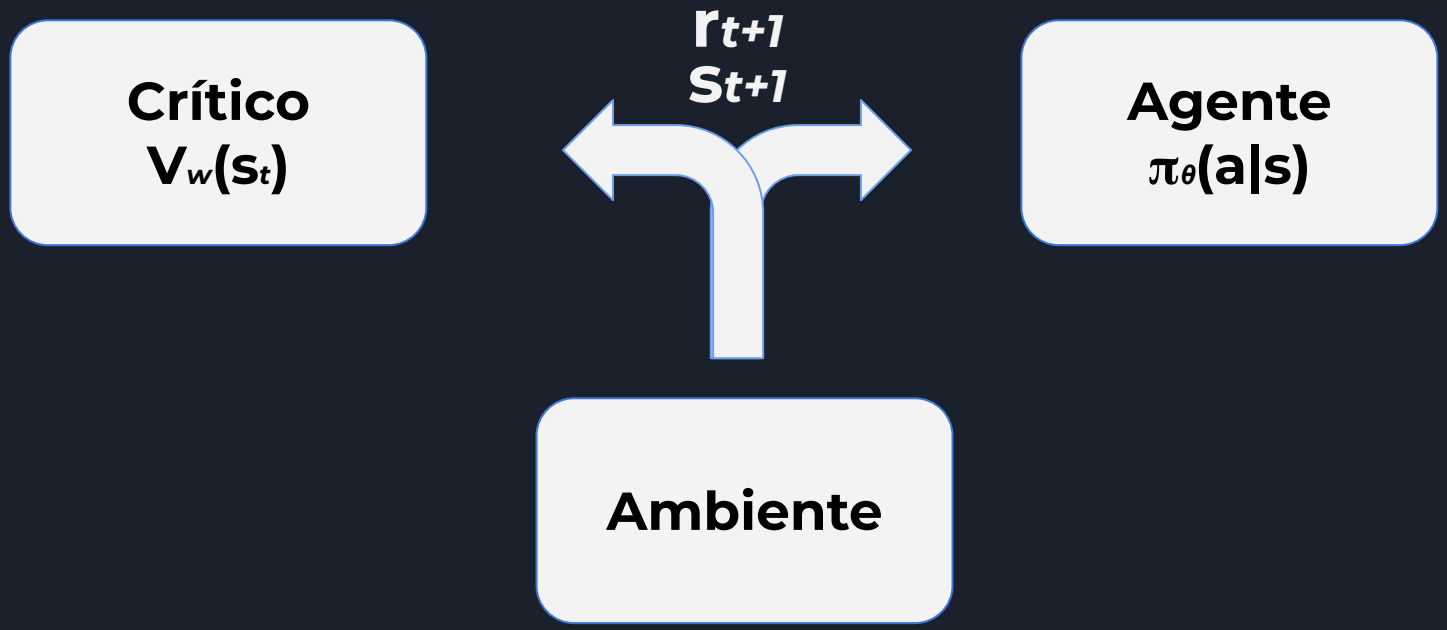
Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic (Treinamento)



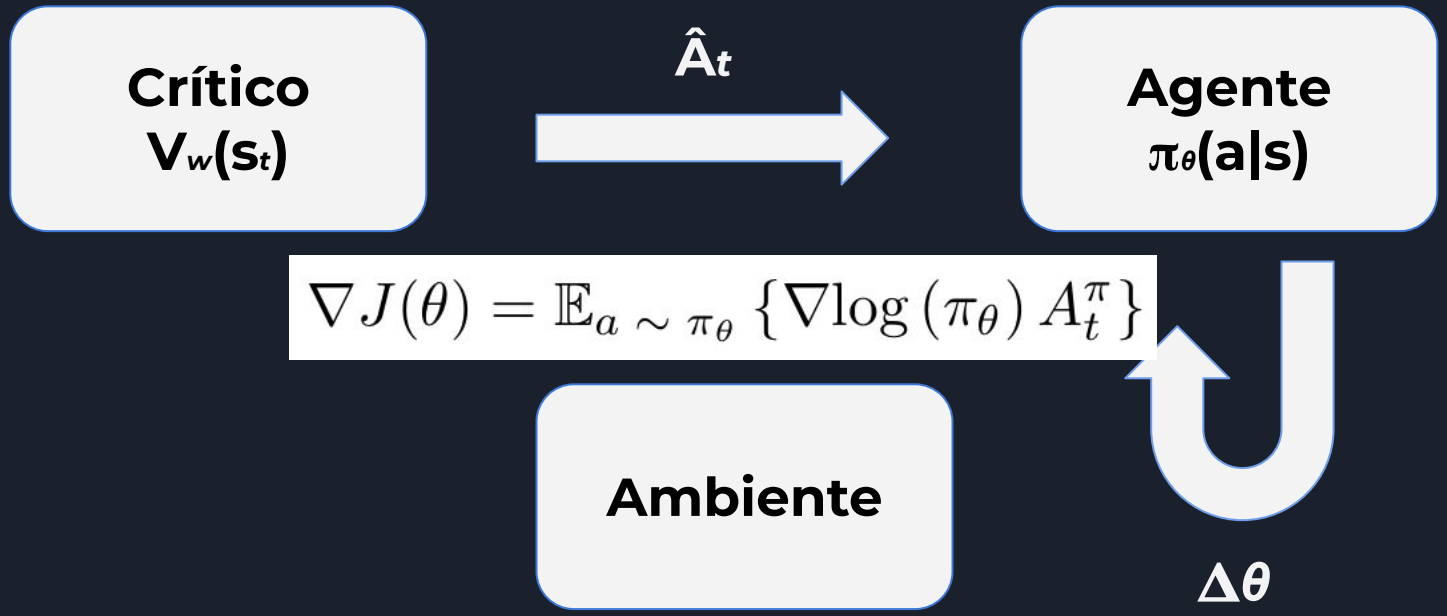
Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic (Treinamento)



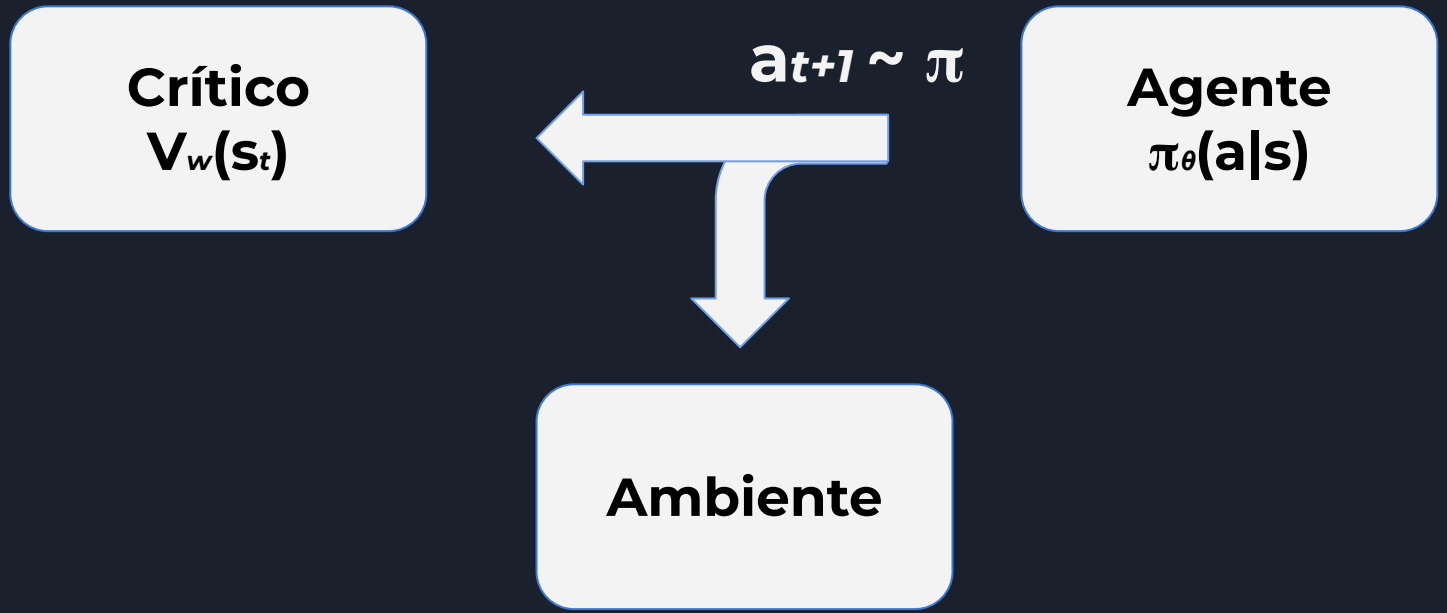
Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic (Treinamento)



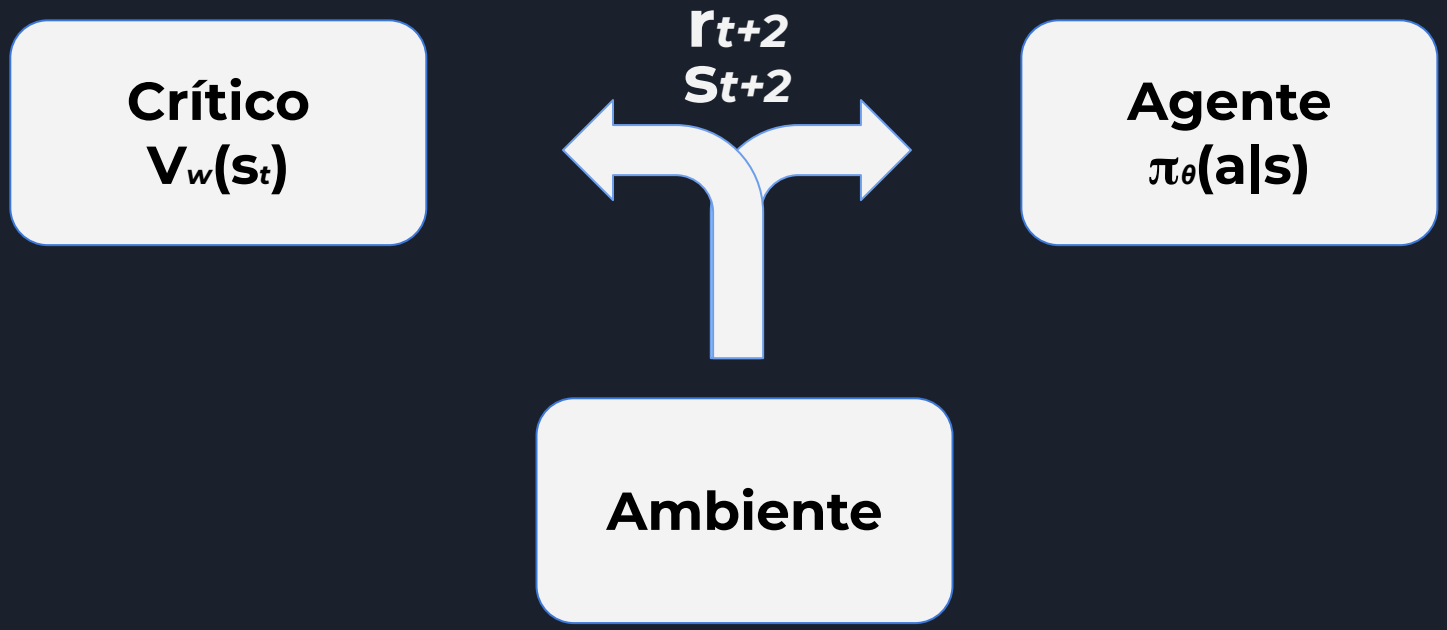
Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic (Treinamento)



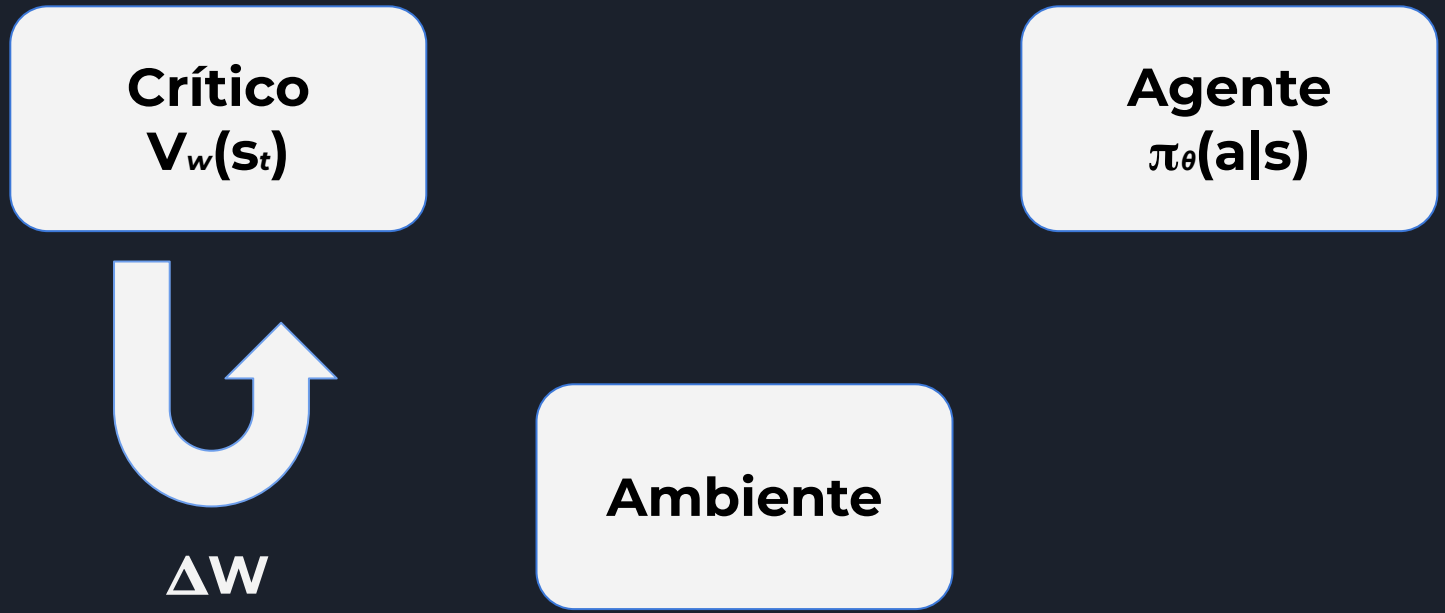
Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic (Treinamento)



Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic (Treinamento)



Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic

Actor

$$\pi_\theta(a|s)$$

Critic

$$\hat{A}_t^\pi = R_t + \gamma \hat{V}_w^\pi(s_{t+1}) - \hat{V}_w^\pi(s_t)$$

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic

Actor

$$\pi_\theta(a|s)$$

$$J(\theta) = \mathbb{E}_{a \sim \pi_\theta}\left\{\hat{A}_t^\pi\right\}$$

Critic

$$\hat{A}_t^\pi = R_t + \gamma \hat{V}_w^\pi(s_{t+1}) - \hat{V}_w^\pi(s_t)$$

$$J(w) = \frac{1}{2}\left(R_t + \gamma \hat{V}_w^\pi(s_{t+1}) - \hat{V}_w^\pi(s_t)\right)^2$$

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic

Actor

$$\pi_\theta(a|s)$$

$$J(\theta) = \mathbb{E}_{a \sim \pi_\theta}\left\{\hat{A}_t^\pi\right\}$$

$$\theta_{t+1} \leftarrow \theta_t + \alpha^\theta \nabla J(\theta)$$

Critic

$$\hat{A}_t^\pi = R_t + \gamma \hat{V}_w^\pi(s_{t+1}) - \hat{V}_w^\pi(s_t)$$

$$J(w) = \frac{1}{2}\left(R_t + \gamma \hat{V}_w^\pi(s_{t+1}) - \hat{V}_w^\pi(s_t)\right)^2$$

$$w_{t+1} \leftarrow w_t - \alpha^w \nabla J(w)$$

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Actor-Critic

Actor

$$\pi_\theta(a|s)$$

$$J(\theta) = \mathbb{E}_{a \sim \pi_\theta}\left\{\hat{A}_t^\pi\right\}$$

$$\theta_{t+1} \leftarrow \theta_t + \alpha^\theta \nabla J(\theta)$$

Critic

$$\hat{A}_t^\pi = R_t + \gamma \hat{V}_w^\pi(s_{t+1}) - \hat{V}_w^\pi(s_t)$$

$$J(w) = \frac{1}{2}\left(R_t + \gamma \hat{V}_w^\pi(s_{t+1}) - \hat{V}_w^\pi(s_t)\right)^2$$

$$w_{t+1} \leftarrow w_t - \alpha^w \nabla J(w)$$

Problema: Difícil de treinar!

Sutton, R. and Barto, A. - Reinforcement Learning.

RL: Proximal Policy Optimization

$$r_t(\theta) = \frac{\pi_\theta(a|s)}{\pi_{\theta\ \mathrm{old}}(a|s)}$$

Schulman *et al*. Proximal Policy Optimization Algorithms. CoRR, abs/1707.06347, 2017.

RL: Proximal Policy Optimization

$$r_t(\theta) = \frac{\pi_\theta(a|s)}{\pi_{\theta\ \text{old}}(a|s)}$$

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

Schulman *et al*. Proximal Policy Optimization Algorithms. CoRR, abs/1707.06347, 2017.

RL: Exemplos (PPO)

openai.com/five

Roteiro

- Introdução
- *Supervised Learning*
- ***Reinforcement Learning***
- Objetivos
- Metodologia
- Trabalhos Futuros (Cronograma)

Roteiro

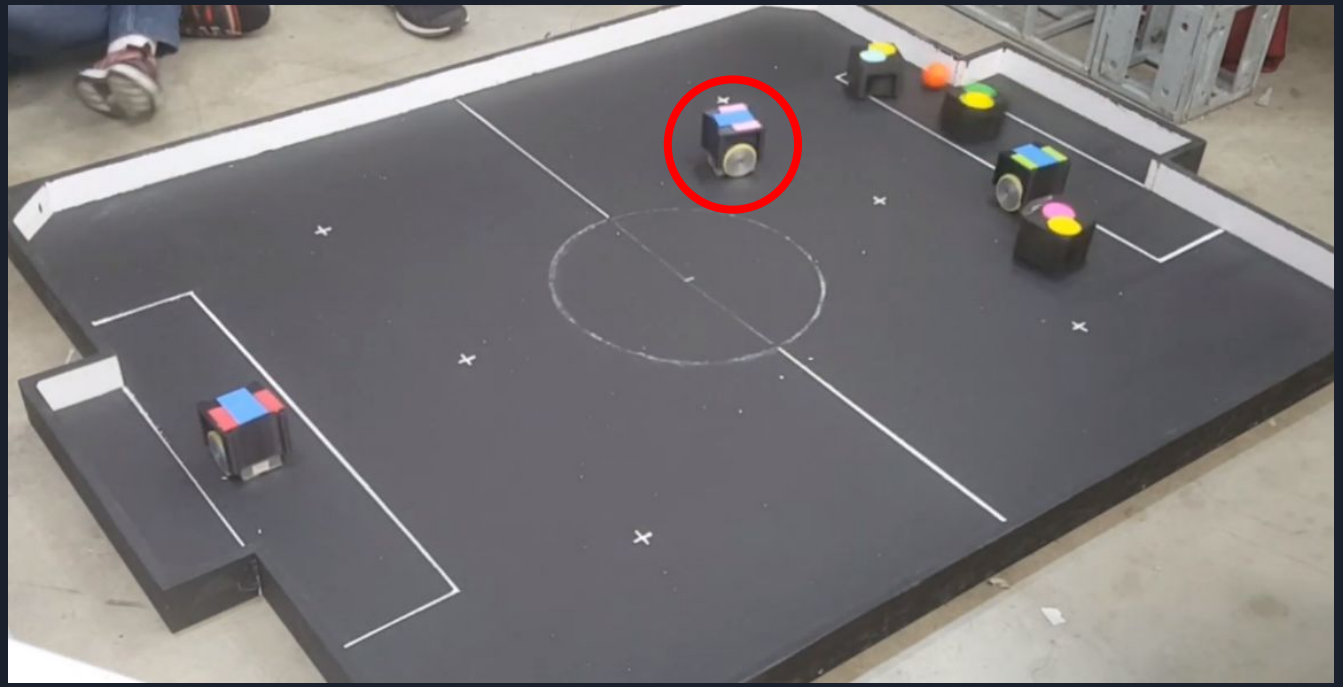
- Introdução
- *Supervised Learning*
- *Reinforcement Learning*
- **Objetivos**
- Metodologia
- Trabalhos Futuros (Cronograma)

Objetivos

- **Treinar uma política para impedir a bola de sair do campo adversário durante o ataque.**
- **Treinar uma política para levar a bola para o campo adversário durante a defesa.**

Objetivos

Objetivos



Objetivos

- Treinar uma política para impedir a bola de sair do campo adversário durante o ataque.

- **Treinar uma política para levar a bola para o campo adversário durante a defesa.**

Objetivos

Objetivos

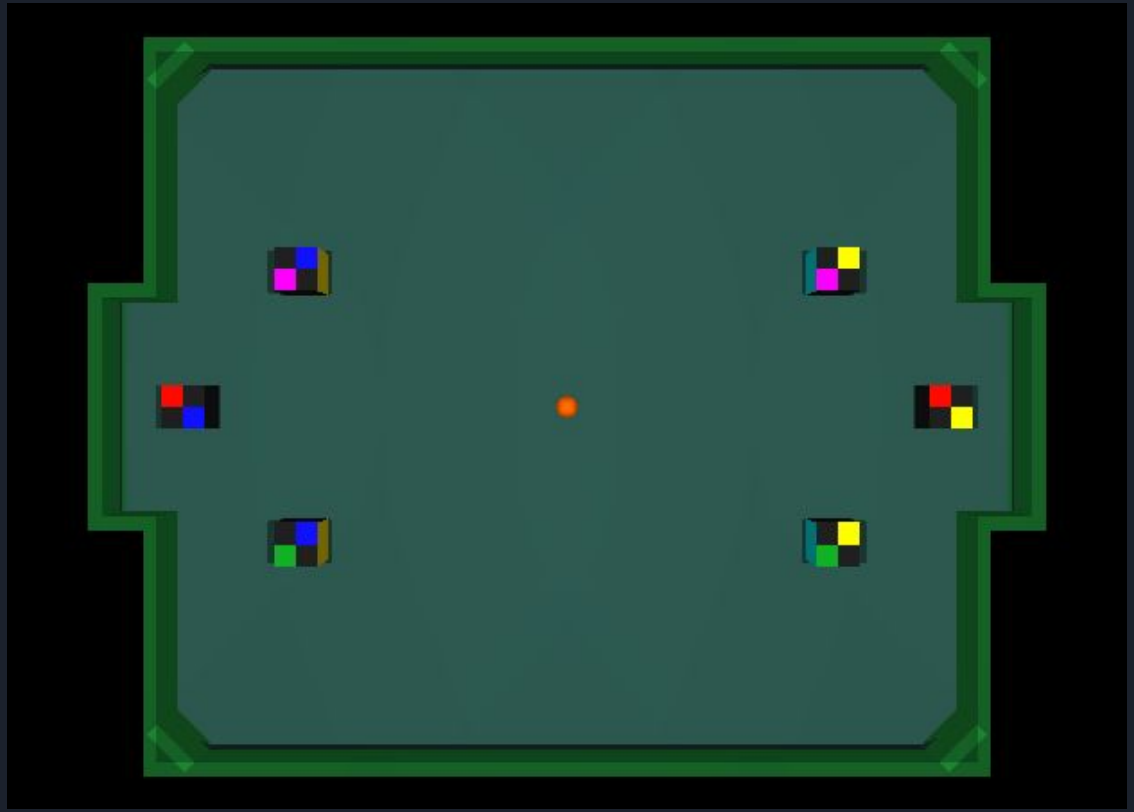
Roteiro

- Introdução
- *Supervised Learning*
- *Reinforcement Learning*
- **Objetivos**
- Metodologia
- Trabalhos Futuros (Cronograma)

Roteiro

- Introdução
- *Supervised Learning*
- *Reinforcement Learning*
- Objetivos
- **Metodologia**
- Trabalhos Futuros (Cronograma)

Metodologia: Simulador



Metodologia: Simulador

[Entrada]
Câmera

Metodologia: Simulador

[Entrada]
Câmera

[Detecção]
Jogadores,
Oponentes,
Bola

Metodologia: Simulador

[Entrada]
Câmera

[Detecção]
Jogadores,
Oponentes,
Bola

[Estimação]
Posições,
Velocidades

Metodologia: Simulador

[Entrada]
Câmera

[Detecção]
Jogadores,
Oponentes,
Bola

[Estimação]
Posições,
Velocidades

[Estratégia]
Definição
de Objetivo

Metodologia: Simulador

[Entrada]
Câmera

[Detecção]
Jogadores,
Oponentes,
Bola

[Estimação]
Posições,
Velocidades

[Estratégia]
Definição
de Objetivo

[Trajetória]
Caminho a
seguir

Metodologia: Simulador

[Entrada]
Câmera

[Detecção]
Jogadores,
Oponentes,
Bola

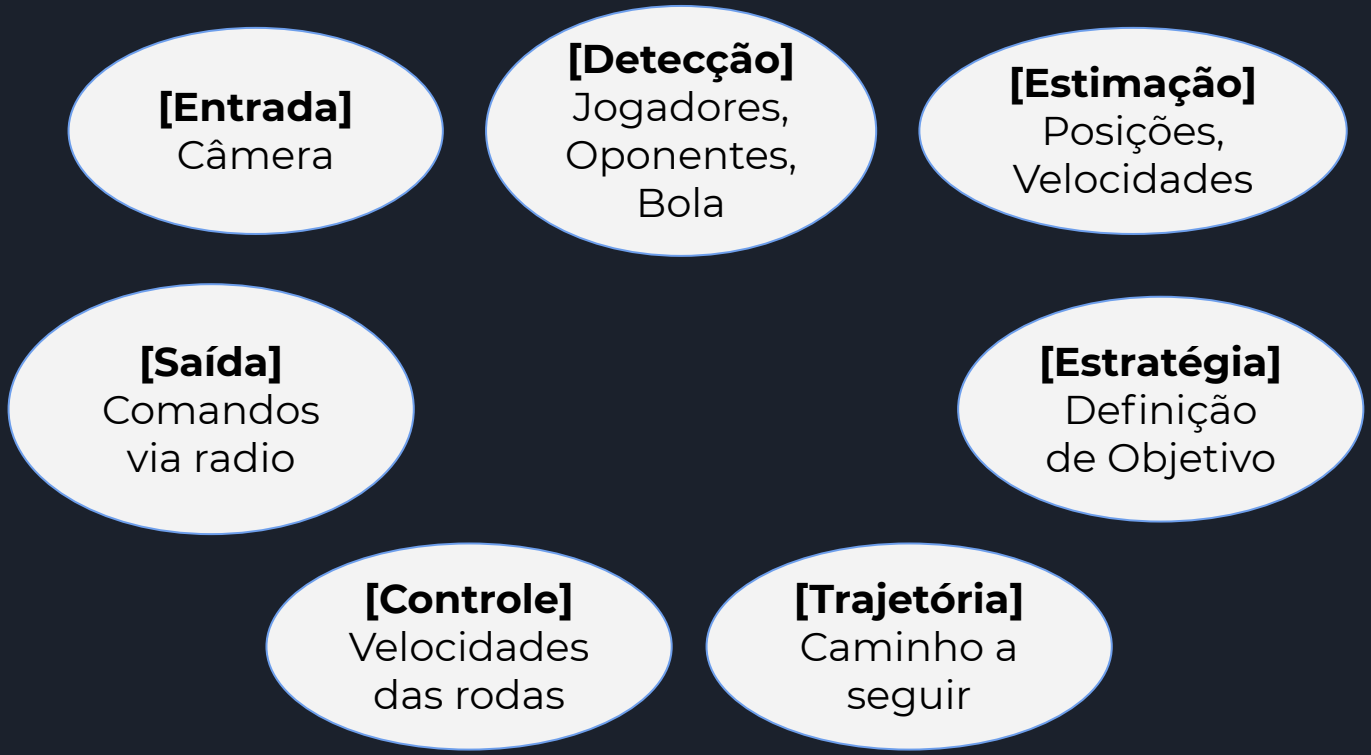
[Estimação]
Posições,
Velocidades

[Estratégia]
Definição
de Objetivo

[Trajetória]
Caminho a
seguir

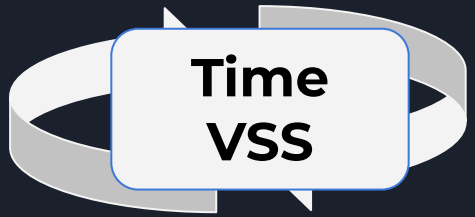
[Controle]
Velocidades
das rodas

Metodologia: Simulador



Metodologia: Simulador

Metodologia: Simulador



Metodologia: Simulador

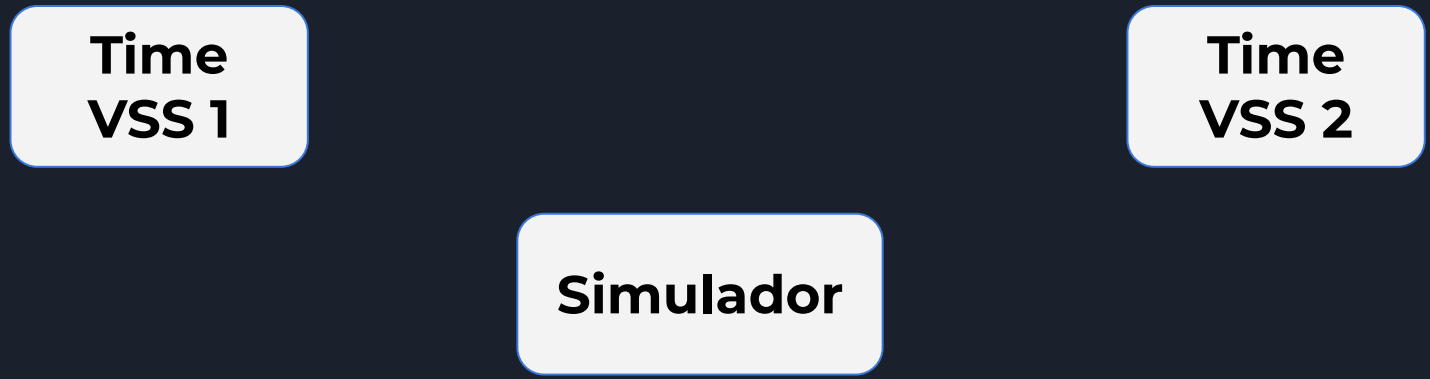
Time
VSS

Metodologia: Simulador

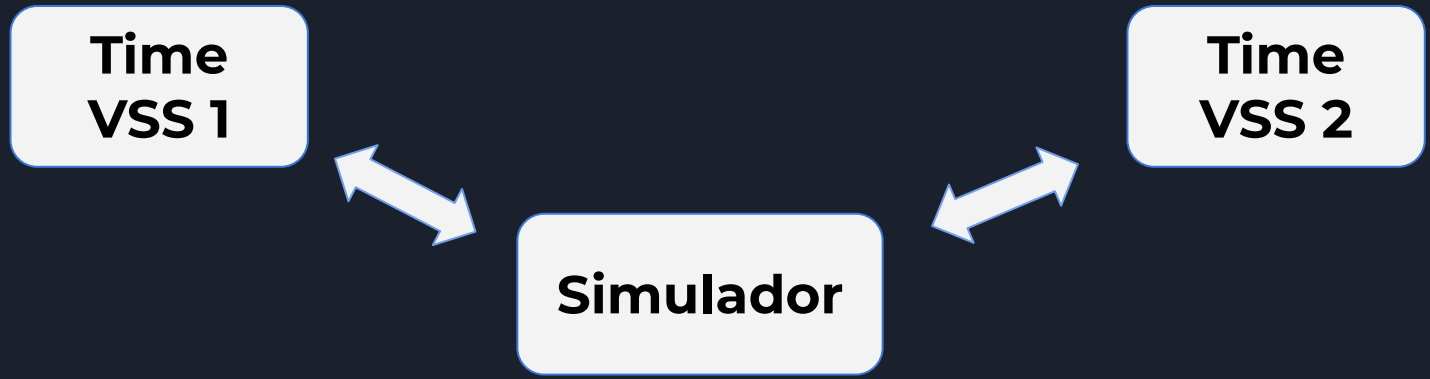
Time
VSS

Simulador

Metodologia: Simulador



Metodologia: Simulador



Metodologia: OpenAI

Organização de pesquisa em IA sem fins lucrativos, financiada pelo Elon Musk

openai.com

Metodologia: OpenAI Baselines

Status: Active (under active development, breaking changes may occur)

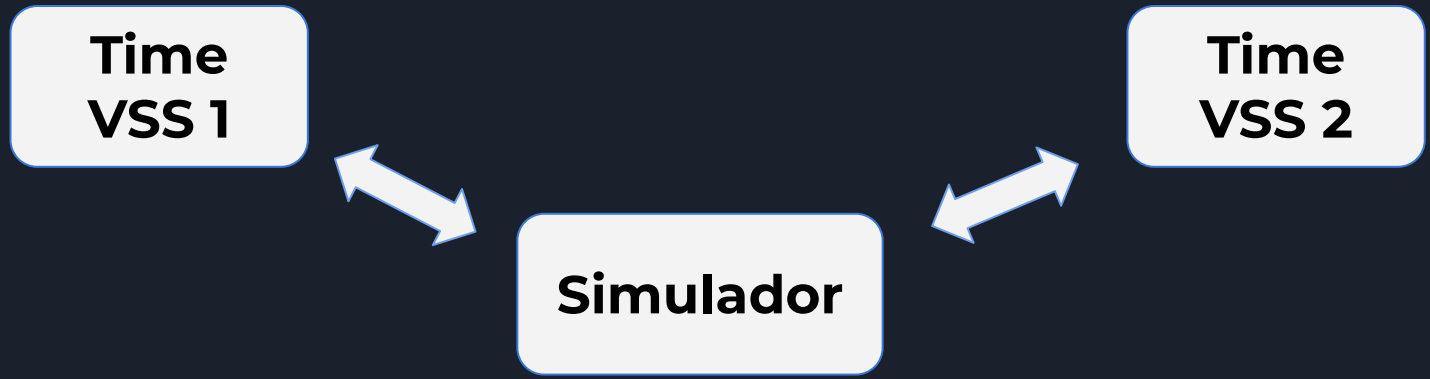
build passing

Baselines

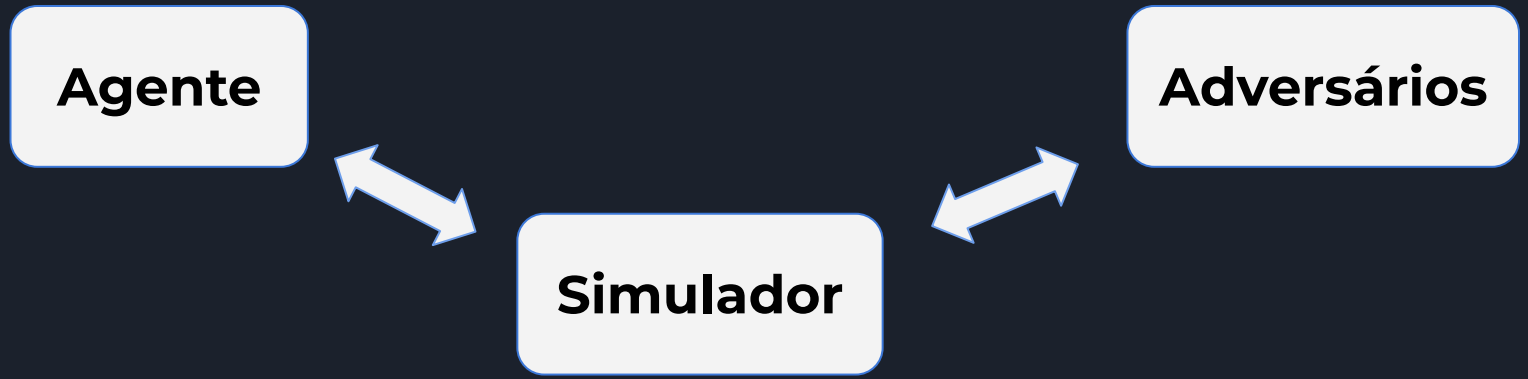
OpenAI Baselines is a set of high-quality implementations of reinforcement learning algorithms.

github.com/openai/baselines

Metodologia: Treinamento



Metodologia: Treinamento



Metodologia: Treinamento

Agente
+
Simulador
+
Adversários

Metodologia: Treinamento

Agente
+
Simulador
+
Adversários

Metodologia: Treinamento

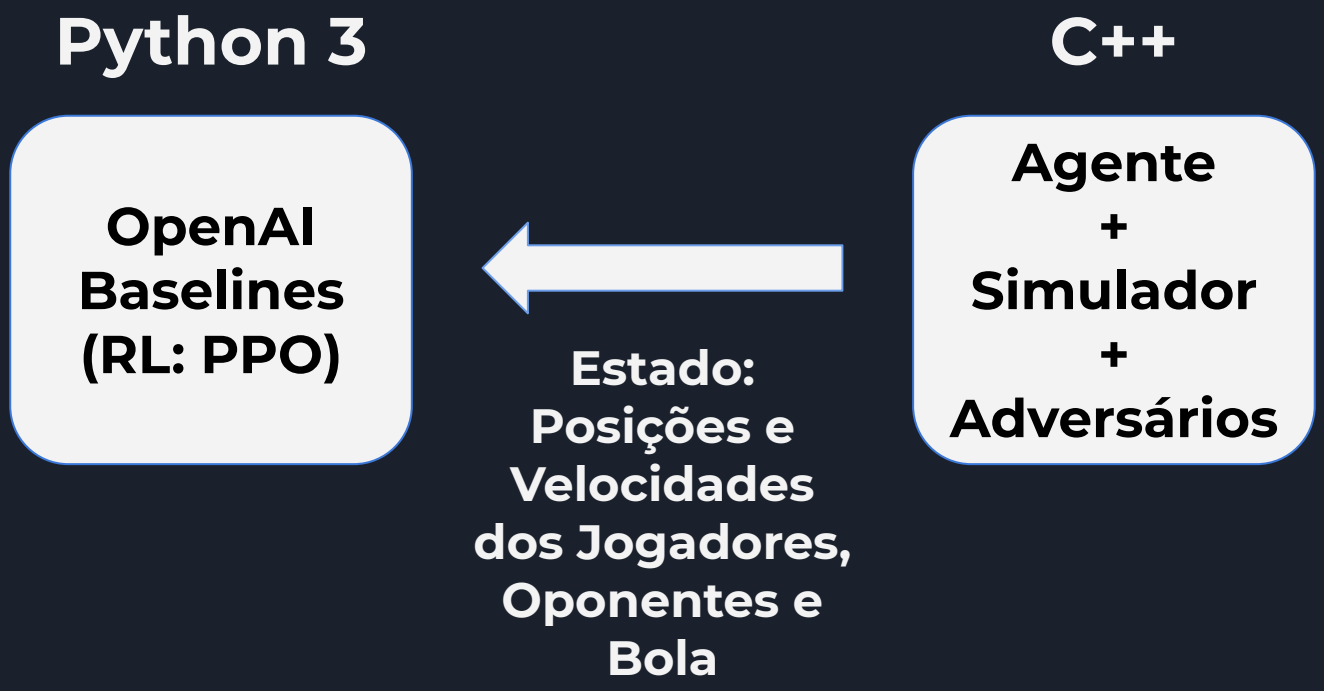
Python 3

OpenAI Baselines (RL: PPO)

C++

Agente + Simulador + Adversários

Metodologia: Treinamento



Metodologia: Treinamento

Python 3

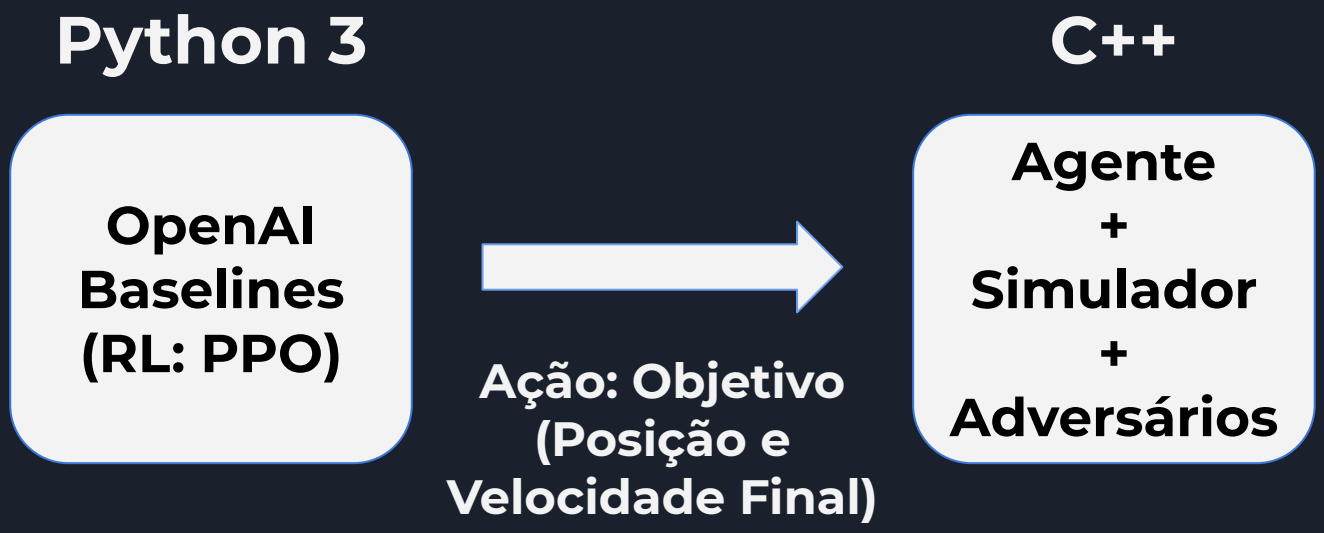
OpenAI Baselines (RL: PPO)

Processamento

C++

Agente + Simulador + Adversários

Metodologia: Treinamento



Metodologia: Treinamento

Python 3

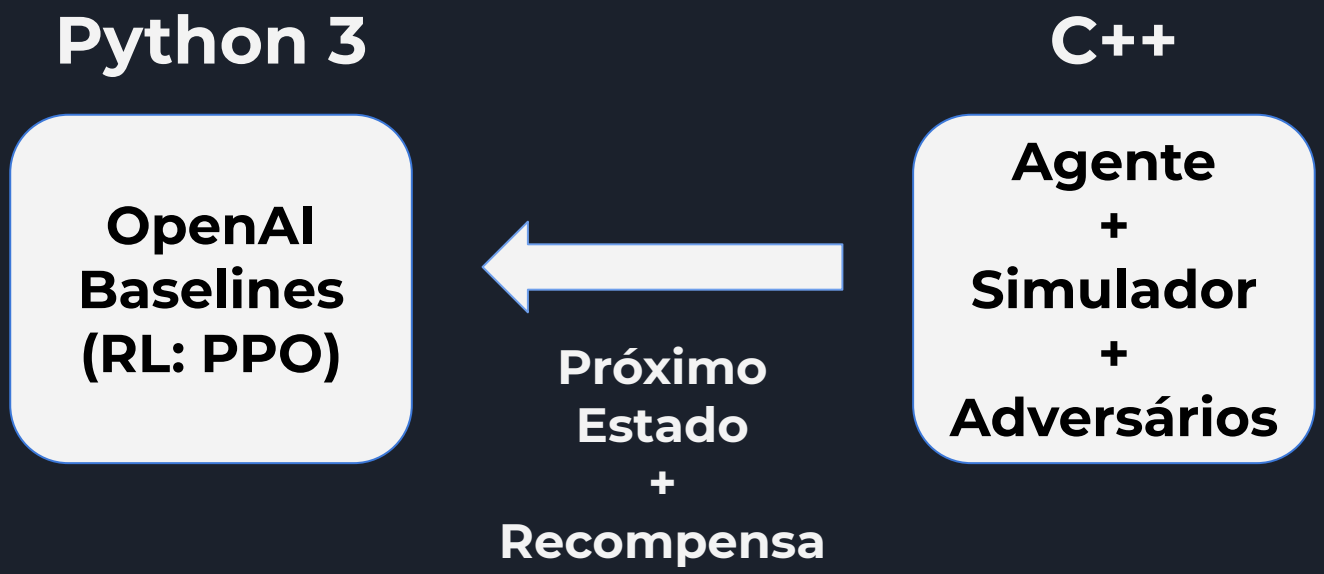
OpenAI Baselines (RL: PPO)

C++

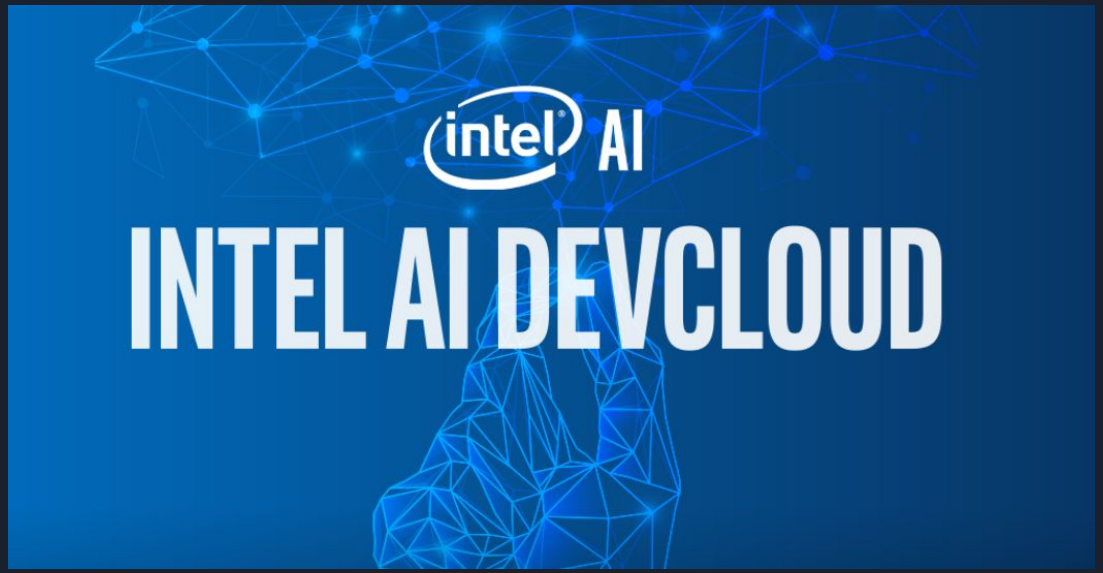
Agente + Simulador + Adversários

Atualização

Metodologia: Treinamento



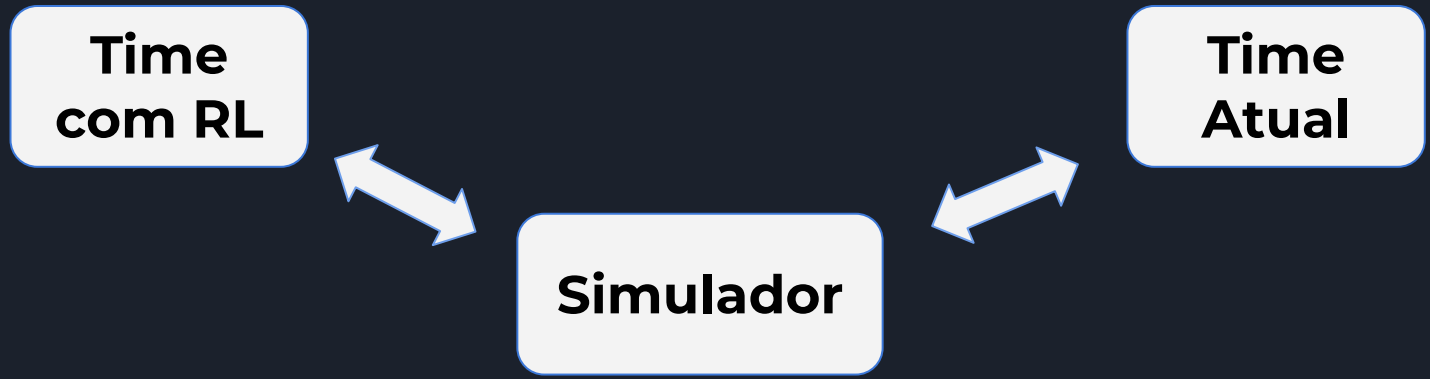
Metodologia: Treinamento



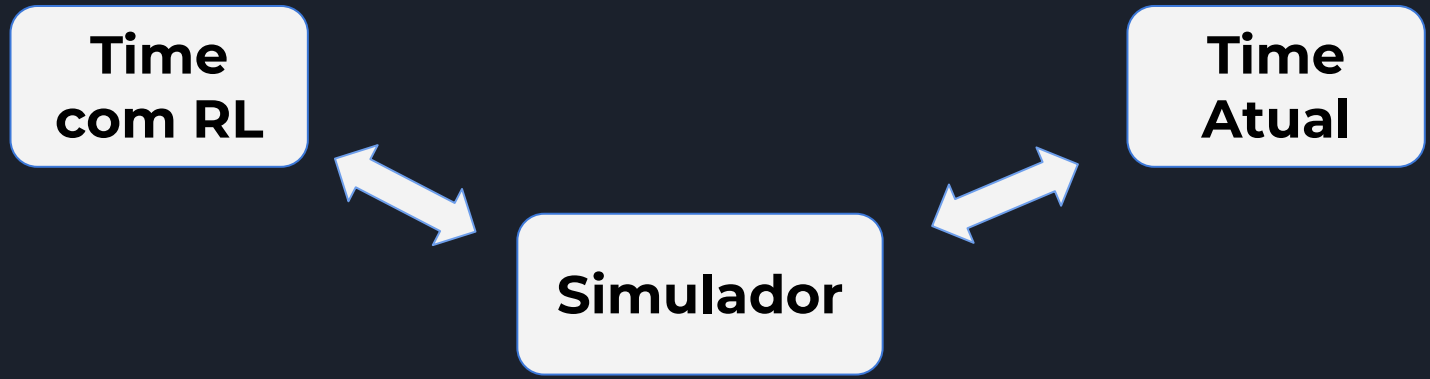
Computação em nuvem gratuita para treinamento de aprendizado de máquina profunda e necessidades de computação de inferência

intel.ai/devcloud

Metodologia: Testes

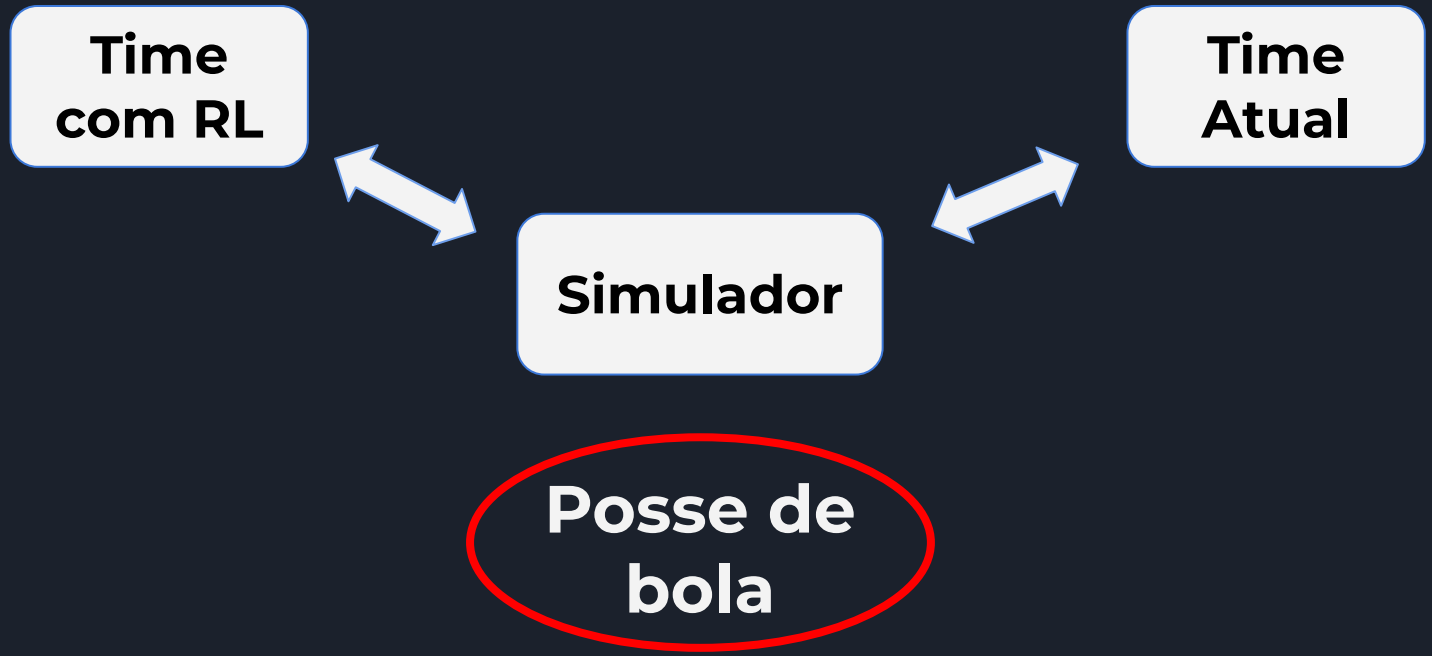


Metodologia: Testes



Posse de bola

Metodologia: Testes



Roteiro

- Introdução
- *Supervised Learning*
- *Reinforcement Learning*
- Objetivos
- **Metodologia**
- Trabalhos Futuros (Cronograma)

Roteiro

- Introdução
- *Supervised Learning*
- *Reinforcement Learning*
- Objetivos
- Metodologia
- **Trabalhos Futuros (Cronograma)**

Trabalhos Futuros (Cronograma)

- Integrar OpenAI baselines ao código do simulador do VSS [até meados de Julho];
- Decidir detalhes da arquitetura da rede e ambiente virtual de treino [Julho];
- Ambientação à ferramenta da Intel AI DevCloud para treinamento distribuído [Julho/Agosto];
- Treinamento [Agosto/meados de Outubro];
- Testes [final de Outubro];
- Inclusão de resultados e confecção do TG2 [Novembro];

Trabalhos Futuros (Cronograma)

- **Integrar OpenAI baselines ao código do simulador do VSS [até meados de Julho];**
- **Decidir detalhes da arquitetura da rede e ambiente virtual de treino [Julho];**
- **Ambientação à ferramenta da Intel AI DevCloud para treinamento distribuído [Julho/Agosto];**
- **Treinamento [Agosto/meados de Outubro];**
- **Testes [final de Outubro];**
- **Inclusão de resultados e confecção do TG2 [Novembro];**

Trabalhos Futuros (Cronograma)

- Integrar OpenAI baselines ao código do simulador do VSS [até meados de Julho];
- **Decidir detalhes da arquitetura da rede e ambiente virtual de treino [Julho];**
- Ambientação à ferramenta da Intel AI DevCloud para treinamento distribuído [Julho/Agosto];
- Treinamento [Agosto/meados de Outubro];
- Testes [final de Outubro];
- Inclusão de resultados e confecção do TG2 [Novembro];

Trabalhos Futuros (Cronograma)

- Integrar OpenAI baselines ao código do simulador do VSS [até meados de Julho];
- Decidir detalhes da arquitetura da rede e ambiente virtual de treino [Julho];
- **Ambientação à ferramenta da Intel AI DevCloud para treinamento distribuído [Julho/Agosto];**
- Treinamento [Agosto/meados de Outubro];
- Testes [final de Outubro];
- Inclusão de resultados e confecção do TG2 [Novembro];

Trabalhos Futuros (Cronograma)

- Integrar OpenAI baselines ao código do simulador do VSS [até meados de Julho];
- Decidir detalhes da arquitetura da rede e ambiente virtual de treino [Julho];
- Ambientação à ferramenta da Intel AI DevCloud para treinamento distribuído [Julho/Agosto];
- **Treinamento [Agosto/meados de Outubro];**
- Testes [final de Outubro];
- Inclusão de resultados e confecção do TG2 [Novembro];

Trabalhos Futuros (Cronograma)

- Integrar OpenAI baselines ao código do simulador do VSS [até meados de Julho];
- Decidir detalhes da arquitetura da rede e ambiente virtual de treino [Julho];
- Ambientação à ferramenta da Intel AI DevCloud para treinamento distribuído [Julho/Agosto];
- Treinamento [Agosto/meados de Outubro];
- **Testes [final de Outubro];**
- Inclusão de resultados e confecção do TG2 [Novembro];

Trabalhos Futuros (Cronograma)

- Integrar OpenAI baselines ao código do simulador do VSS [até meados de Julho];
- Decidir detalhes da arquitetura da rede e ambiente virtual de treino [Julho];
- Ambientação à ferramenta da Intel AI DevCloud para treinamento distribuído [Julho/Agosto];
- Treinamento [Agosto/meados de Outubro];
- Testes [final de Outubro];
- **Inclusão de resultados e confecção do TG2 [Novembro];**

Trabalhos Futuros (Cronograma)

- **Continuidade: Tese no Programa Mestrado na Graduação (PMG) [2020].**