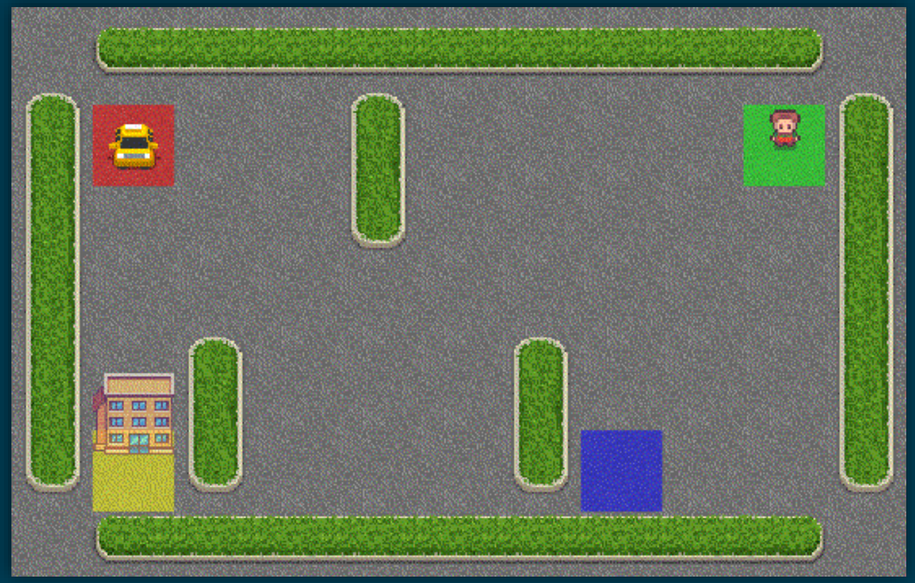
Introdução aos Sistemas Inteligentes e Autónomos

202106775 Guilherme Vaz
202106968 Pedro Campião
202107547 Ricardo Costa

Costumização de ambientes do OpenAi Gym e Implementação de agentes de Reinforcement Learning

Taxi



O ambiente “Taxi” integra-se no conjunto de ambientes “Toy Tex”. Este conjunto de ambientes é caracterizado por ser simples, com um número reduzido de ações e estados discretos.

Para este ambiente específico, existem quatro locais definidos na grid (Vermelho, Verde, Azul e Amarelo).

Quando um episódio começa, o táxi inicia numa posição random da grid e o passageiro numa das quatro posições já definidas. O táxi conduz até a localização do passageiro, apanha-o e dirige até o destino do passageiro (uma das outras três localizações definidas.) e deixa-o. Assim que o passageiro é deixado o episódio termina.

Descrição do ambiente Taxi

Estados

500 espaços discretos
=
25 posições do táxi X 5 posições do passageiro X 4 destinos possíveis

Perceções

Posição atual no mapa
Recompensas e penalizações associadas a cada etapa
Informação sobre o ambiente (existência de paredes, destino, etc.)

Ações

0: mover para sul

1: mover para norte

2: mover para este

3: mover para oeste

4: apanhar passageiro

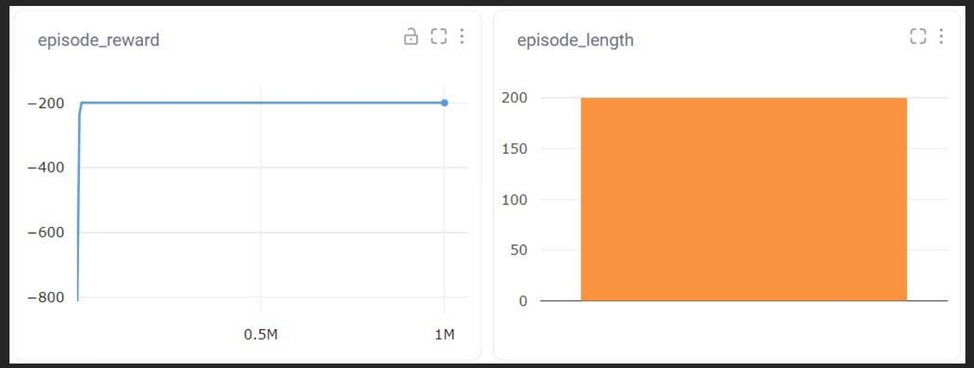
5: deixar passageiro

Recompensas

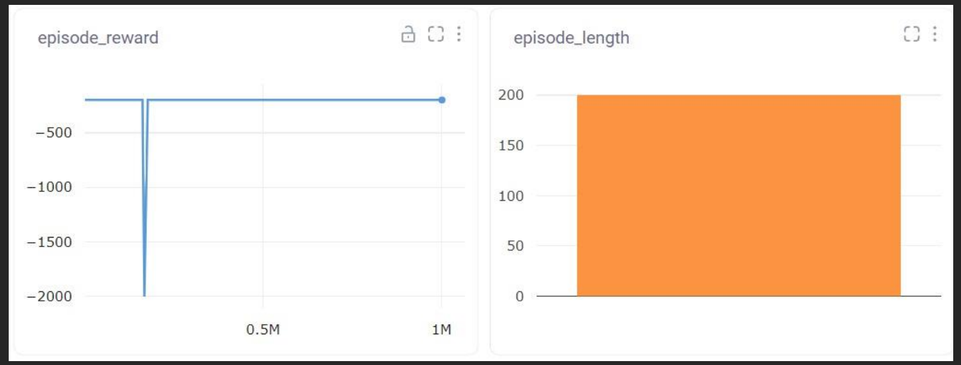
-1: Por etapa, a menos que seja ativada outra recompensa
+20: Deixar o passageiro
-10: Apanhar ou deixar o passageiro quando não é suposto

Algoritmos de Reinforcement Learning

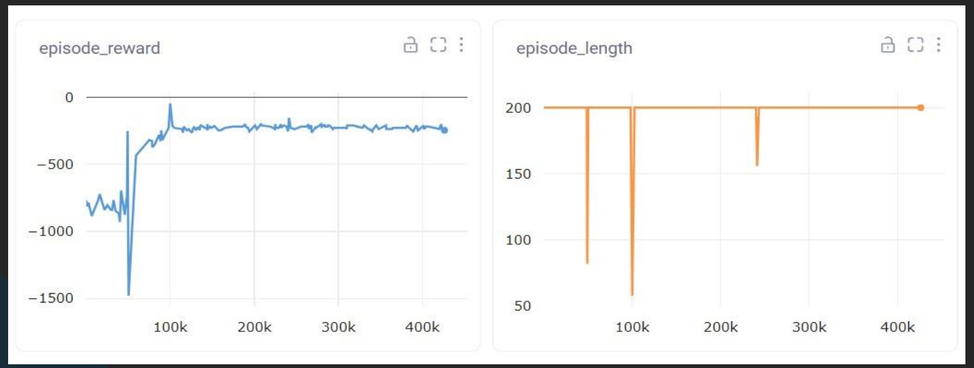
A2C



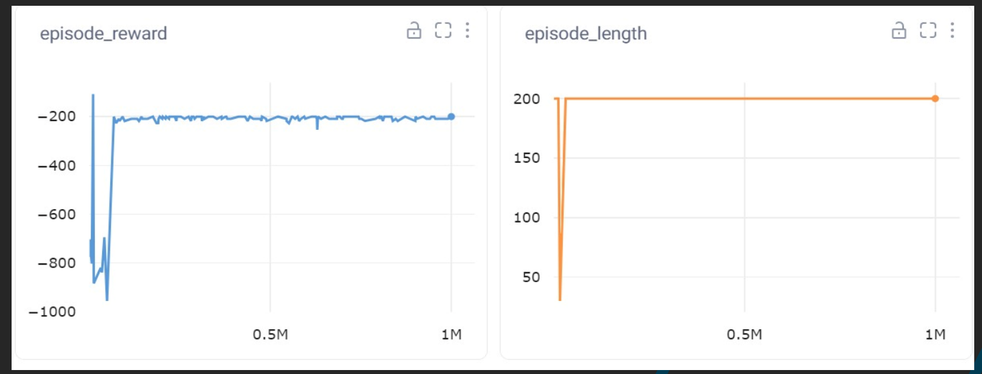
ARS



DQN



QR-DQN

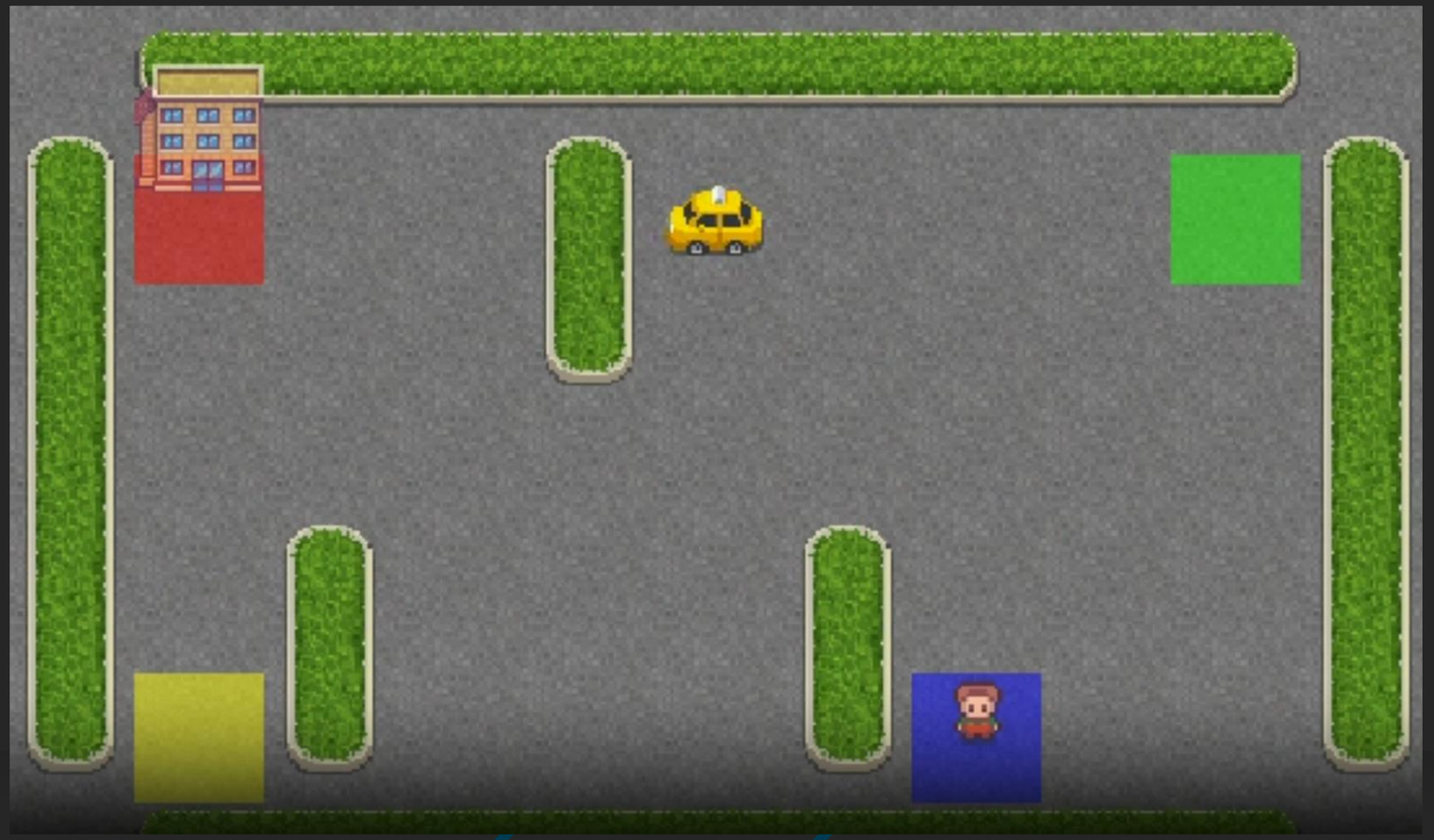


Algoritmos de Reinforcement Learning

PPO – Azul **TRPO – Roxo** **Maskable PPO – Vermelho**

A2C antes das alterações

Nota: vídeos na pasta da submissão



Alterações introduzidas no ambiente

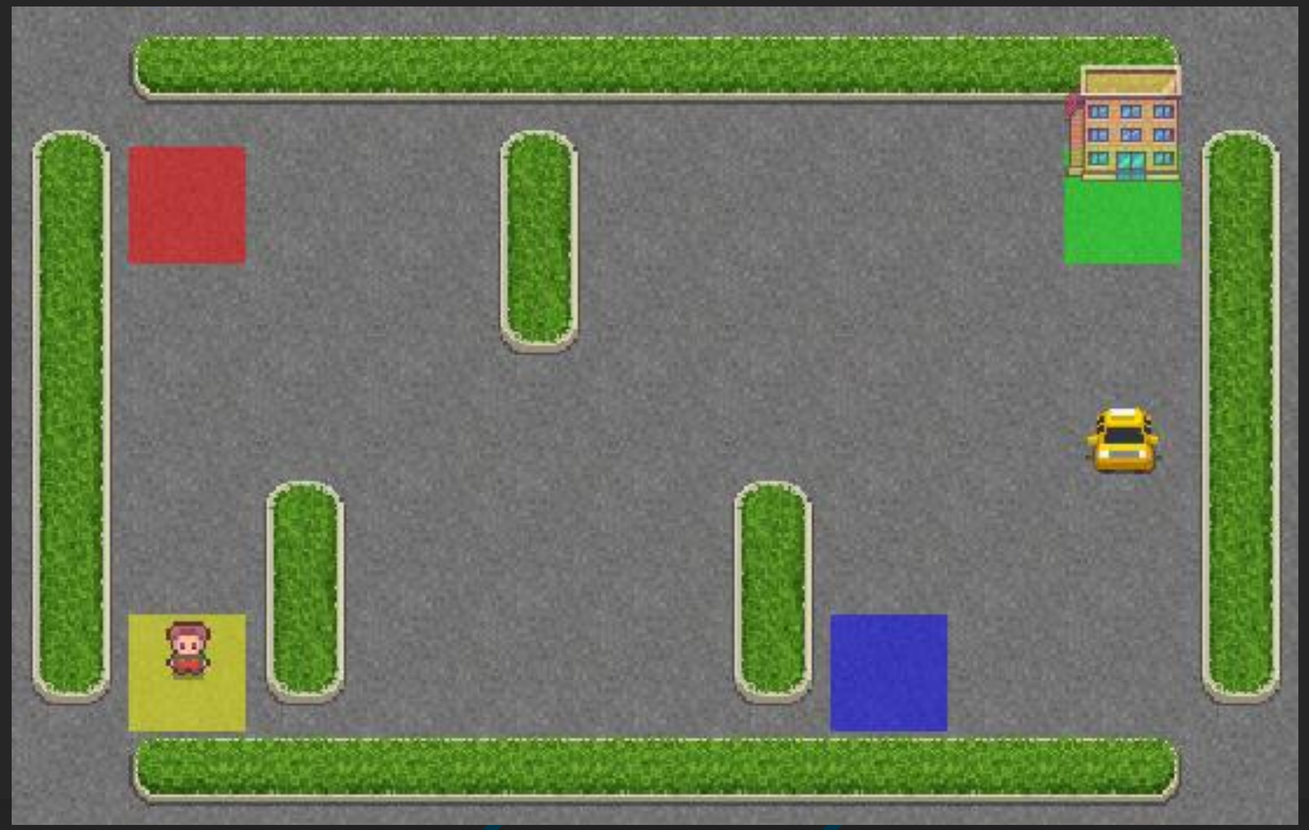
Para os algoritmos A2C e ARS:

Criamos uma classe customizada para o action space baseada na classe de espaços do spaces.discrete do gym.

Dentro dessa classe alteramos a função sample do action space ao fazer com que consiga aceder à mascara para o estado atual.

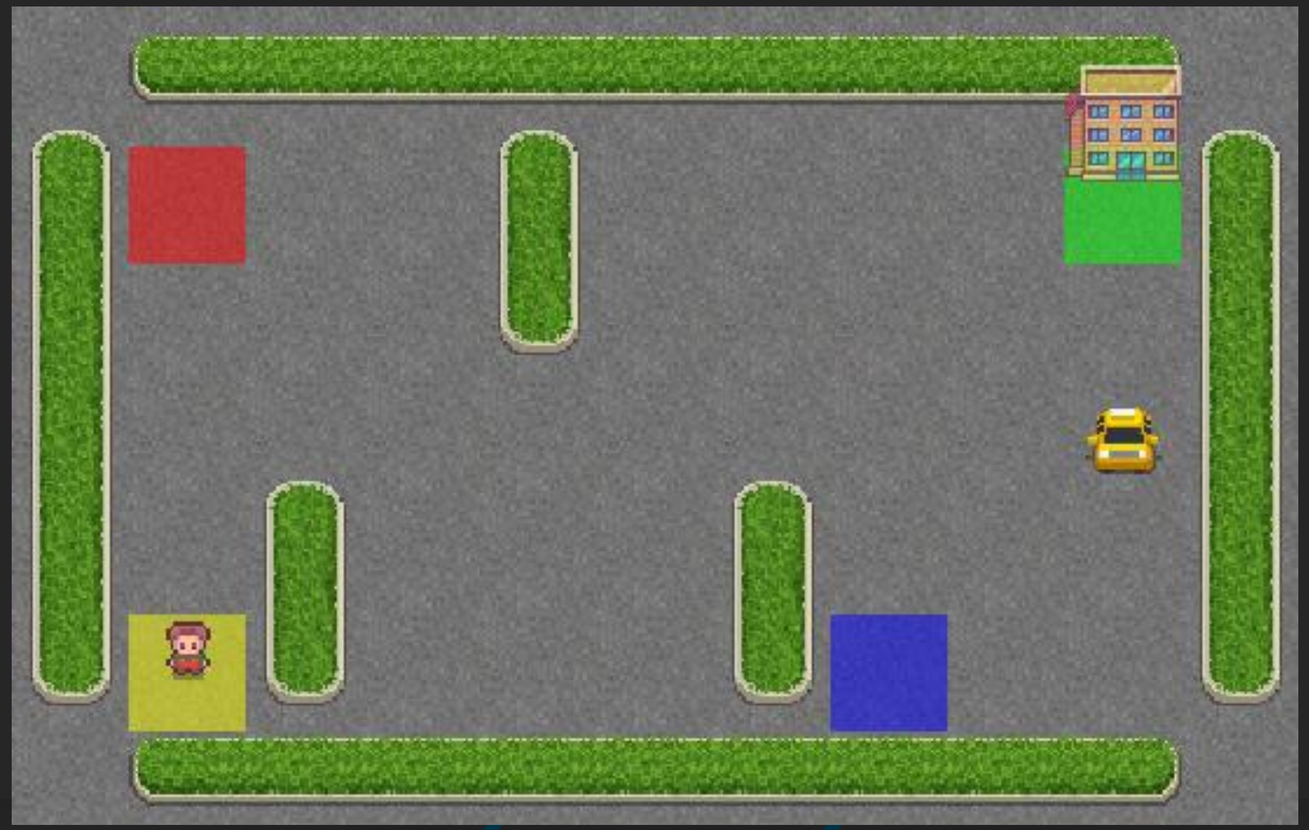
A2C após alterações

Nota: vídeos na pasta da submissão



Maskable PPO antes das alterações

Nota: vídeos na pasta da submissão

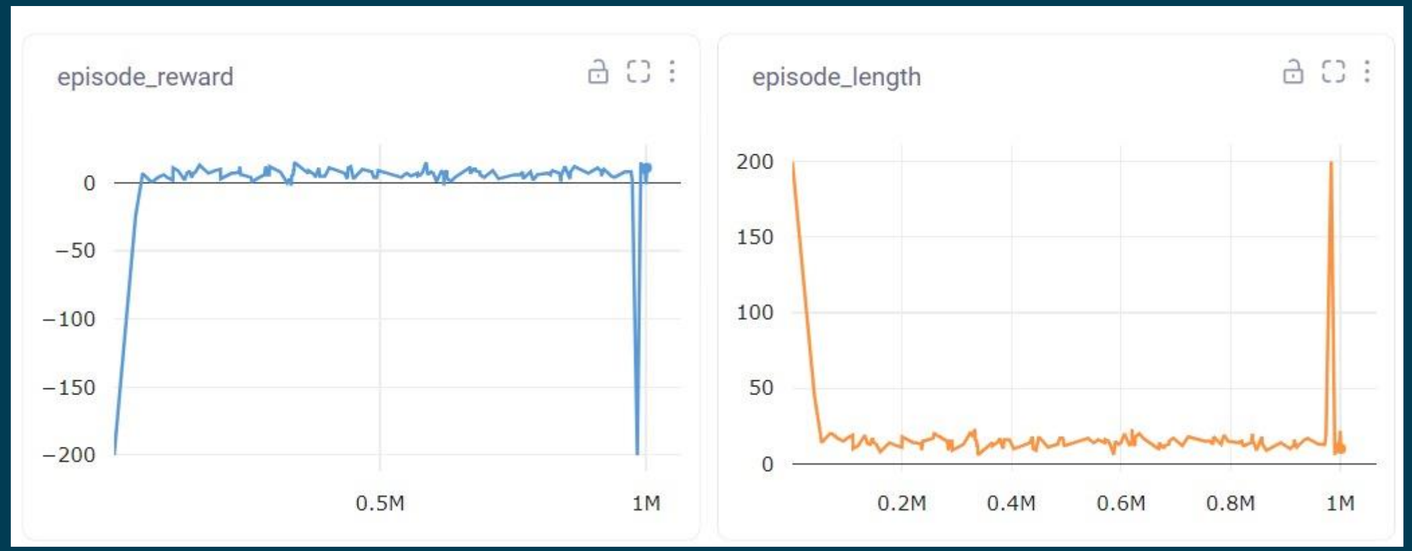


Alterações introduzidas nos algoritmos

Para o algoritmo Maskable - PPO:

Adicionamos alguns hiperparâmetros do algoritmo tais como o Learning Rate e um coeficiente de entropia. Ambos com os valores [0.01,0.001,0.0001,0.00001].

Maskable PPO após alterações



Resultados e Conclusões

Decidimos manter os algoritmos A2C e o ARS, porque apresentavam resultados bastante aquém das espectativas. Depois das modificações que introduzimos, os algoritmos conseguiram chegar a um bom nível de desempenho, como é possível observar no vídeo.

Por sua vez, escolhemos o modelo Maskable-PPO, pois, apesar de apresentar um bom desempenho inicialmente, tínhamos o intuito de perceber de que forma os hiperparamêtros poderiam melhorar o desempenho do modelo.

No final, concluímos que os hiperparamêtros não causaram um impacto significativo no nosso modelo. Porém a inclusão da máscara no A2C e no ARS fez destes modelos muito mais capazes.

A mudança que nós implementamos explora o maior ponto fraco do ambiente original e qualquer algoritmo que usássemos, desde que tivesse a nossa modificação implementada, iria produzir bons resultados.

No nosso ambiente, a alteração que afeta maioritariamente os resultados é evitar que o agente se desloque para posições inválidas.

Mais do que com os hiperparâmetros, quando adicionamos a máscara para corrigir o problema supramencionado todos os algoritmos mostraram ser capazes de aprender.