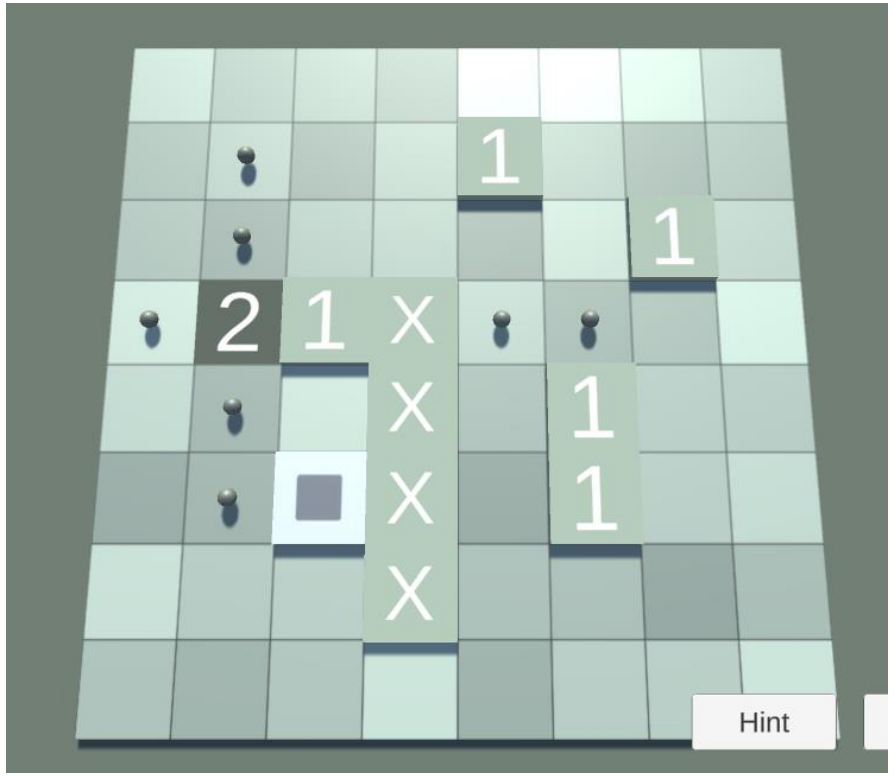
IART – Entrega Final

Daniel Brandão – 201705812
Gaspar Pinheiro – 201704700
Pedro Moás – 201705208

Zhed – Aprendizagem por reforço

- Neste trabalho, continuámos a utilizar o jogo que trabalhámos no primeiro projeto, Zhed, desenvolvido em Unity e C#. Desta vez, utilizámos algoritmos de aprendizagem por reforço, para ajudar o computador a resolver os diferentes puzzles, através da ferramenta ML-agents do Unity.



Abordagens

Abordagem Regular

Observações:

- Peças disponíveis para jogar (coordenadas e valor)
- Coordenadas da peça de destino

Ação:

- Peça a jogar
- Direção da jogada (Cima, baixo, esquerda, direita)

Uso de uma forte função de avaliação

Abordagem Wild

Observações:

- Matriz do tabuleiro

Ação:

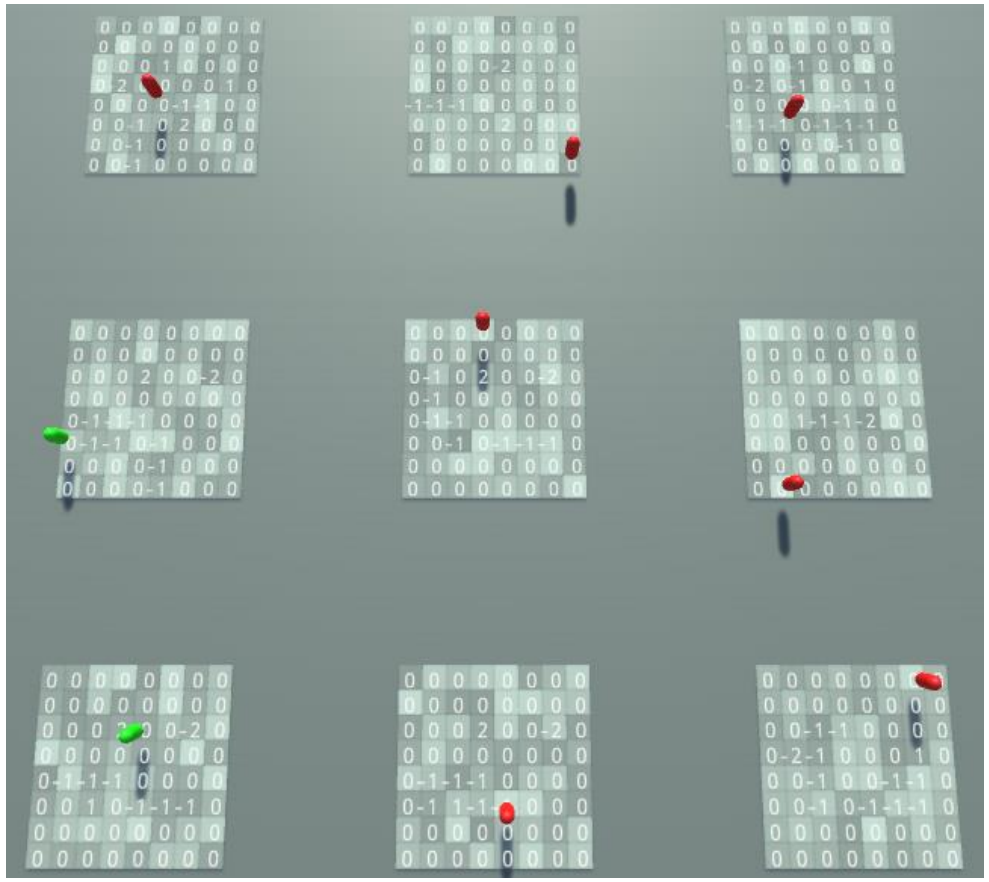
- Coordenadas onde jogar
- Direção da jogada (Cima, baixo, esquerda, direita)

Algoritmos implementados

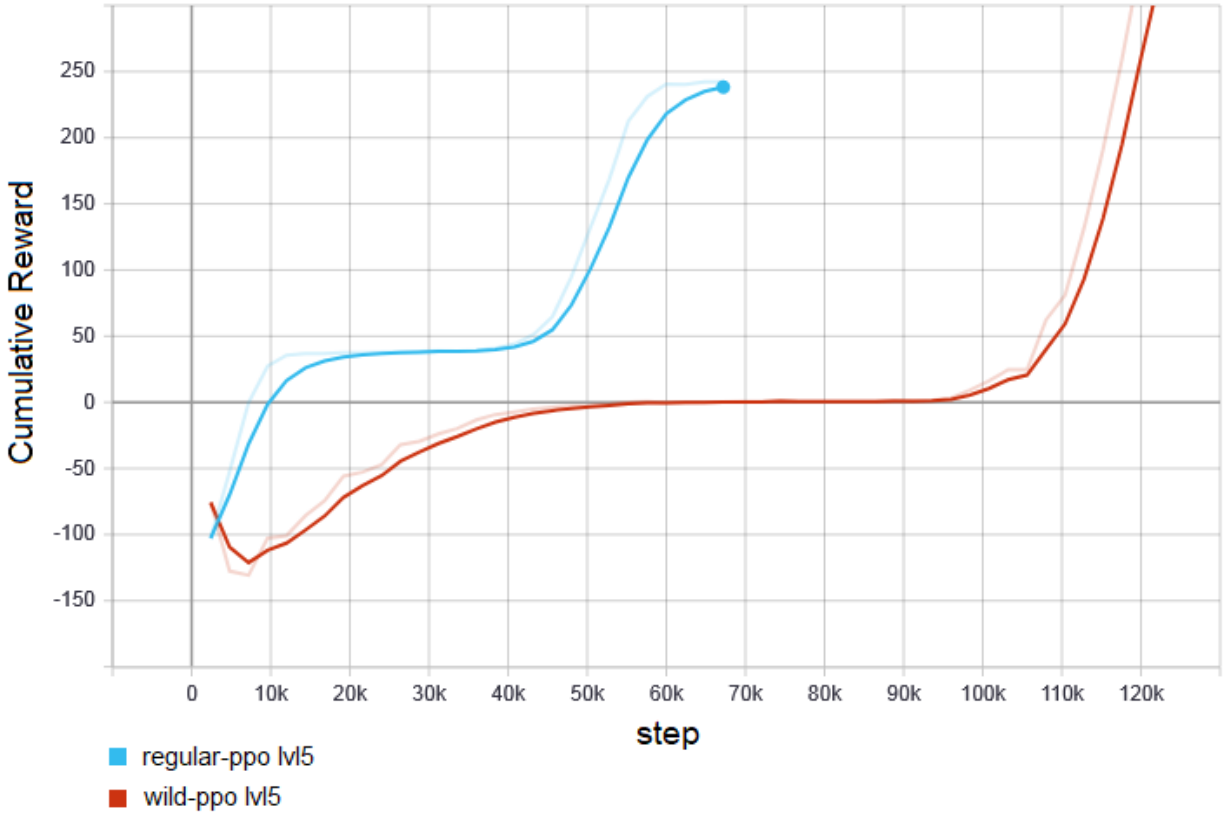
- Tendo em conta que estamos limitados às capacidades do plugin ML-agents, apenas utilizámos os algoritmos que este tem disponibilizado: Proximal Policy Optimization (PPO) e Soft Actor Critic (SAC).
- O algoritmo SAC difere do PPO na medida em que é um algoritmo off-policy, sendo que o computador poderá aprender das experiências obtidas em qualquer episódio anterior.
- Utilizámos ambos os algoritmos para as duas abordagens **Regular** e **Wild**

Desenvolvimento

- Implementação em Unity e adaptação do Zhed para ML-Agents

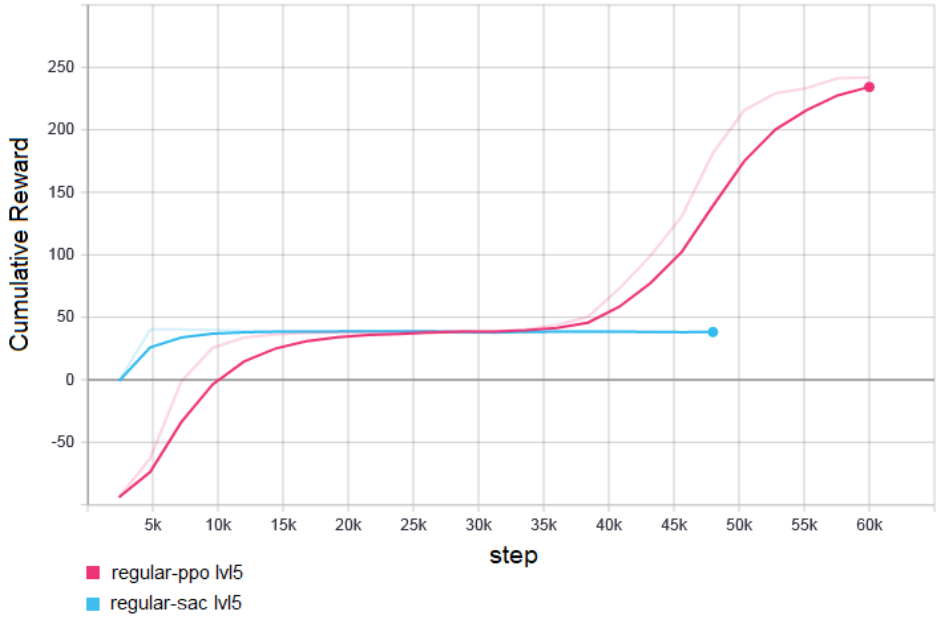


Resultados – Diferenca de abordagens

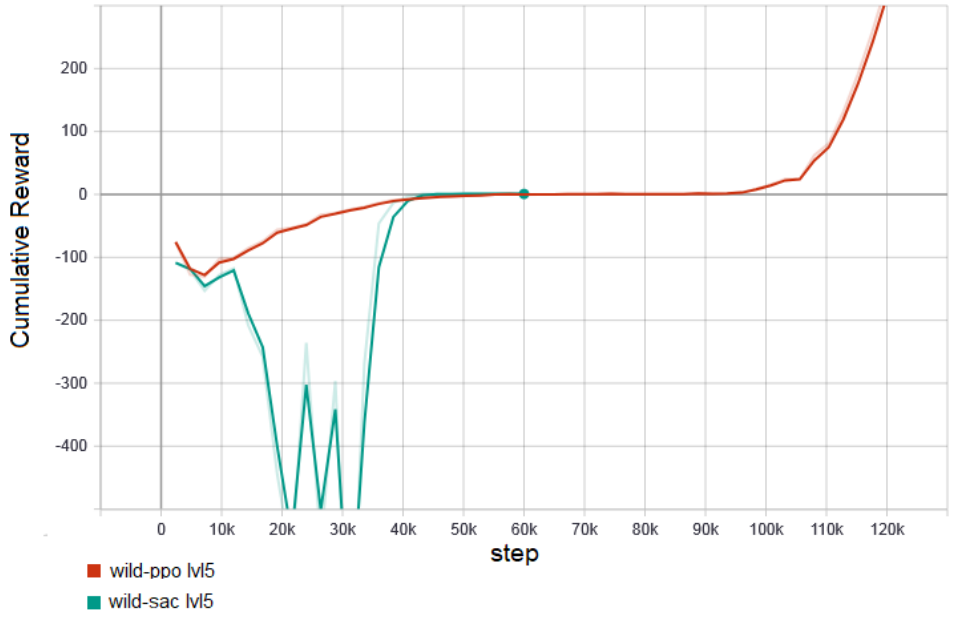


Recompensas acumuladas em função do número de passos, nas abordagens Regular e Wild, nível 5.

Resultados- PPO vs SAC

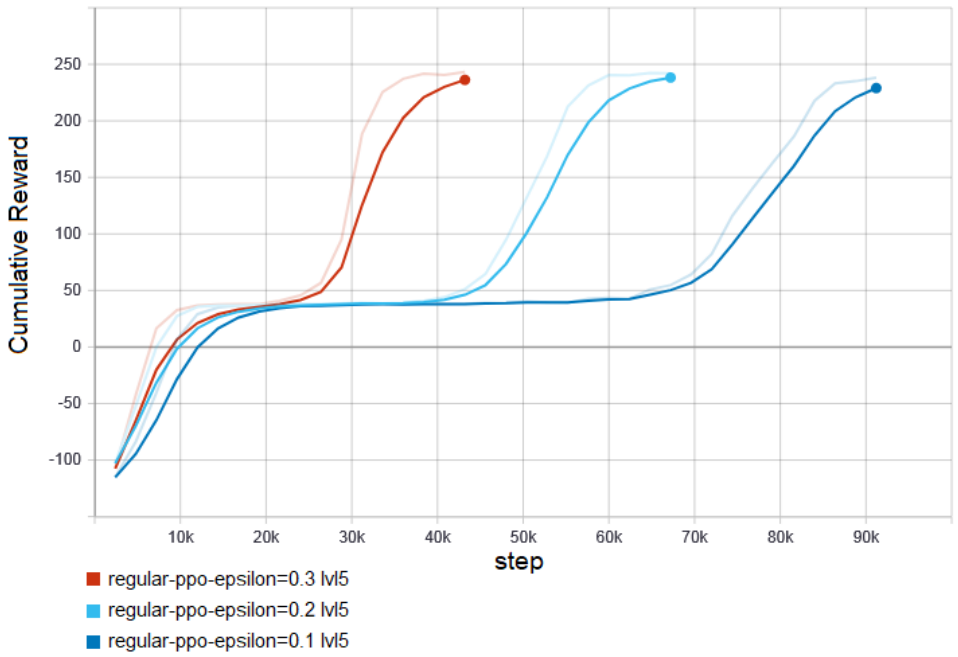


Recompensas acumuladas dos algoritmos PPO e SAC na abordagem Regular, nível 5.

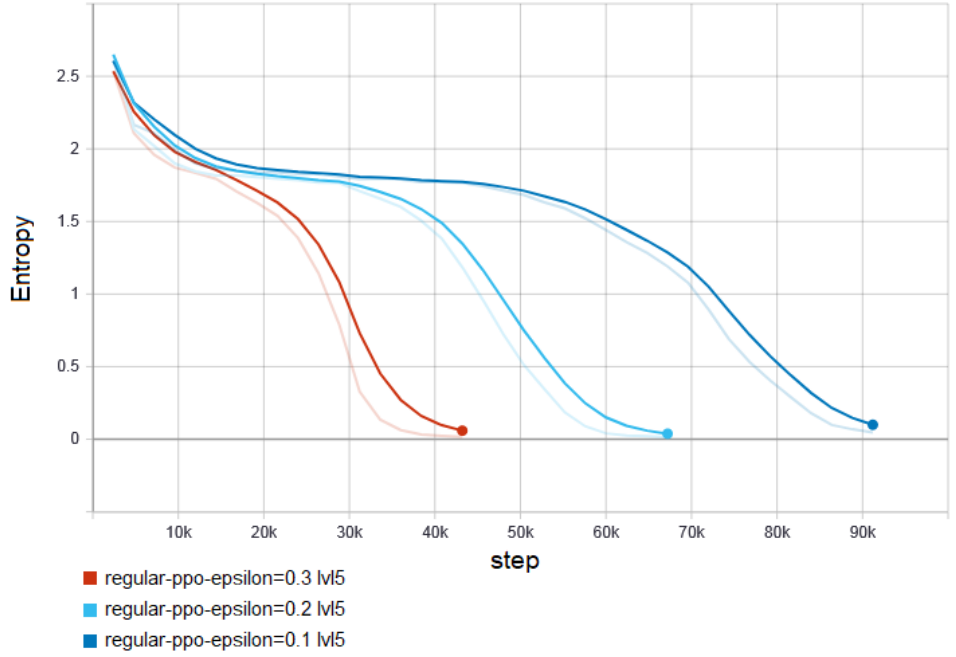


Recompensas acumuladas dos algoritmos PPO e SAC na abordagem Wild, nível 5.

Resultados - Epsilon

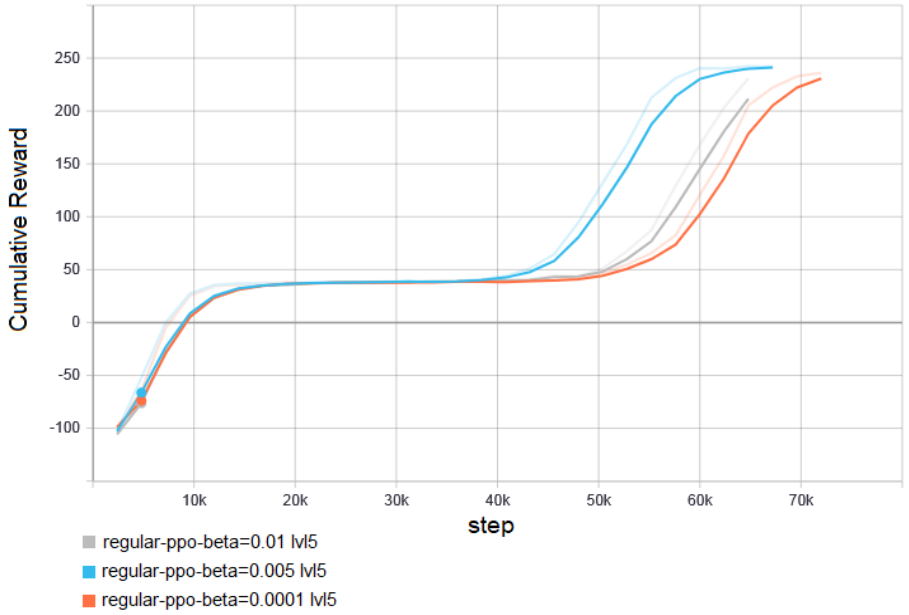


Recompensa acumulada no algoritmo PPO, abordagem Regular, nível 5, variando epsilon

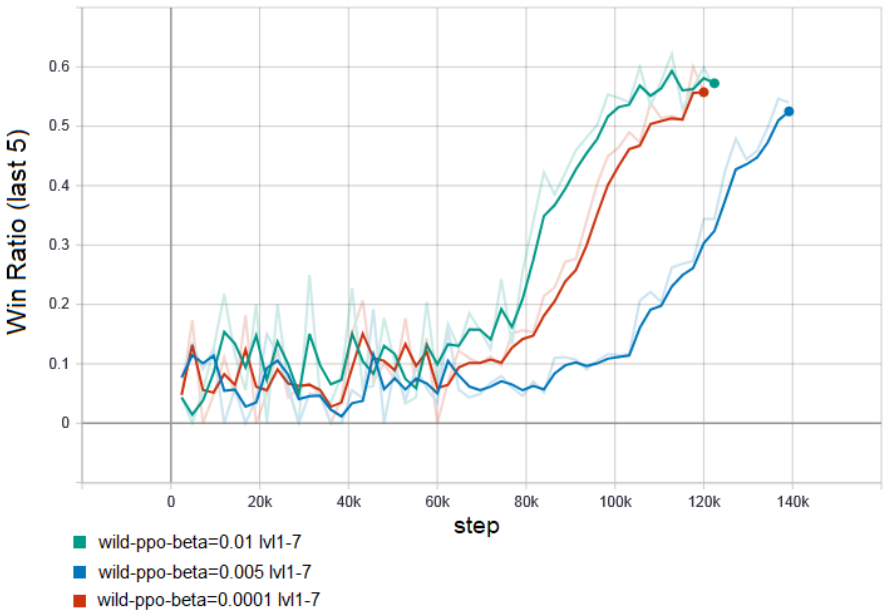


Entropia no algoritmo PPO, abordagem Regular, nível 5, variando epsilon

Resultados - Beta

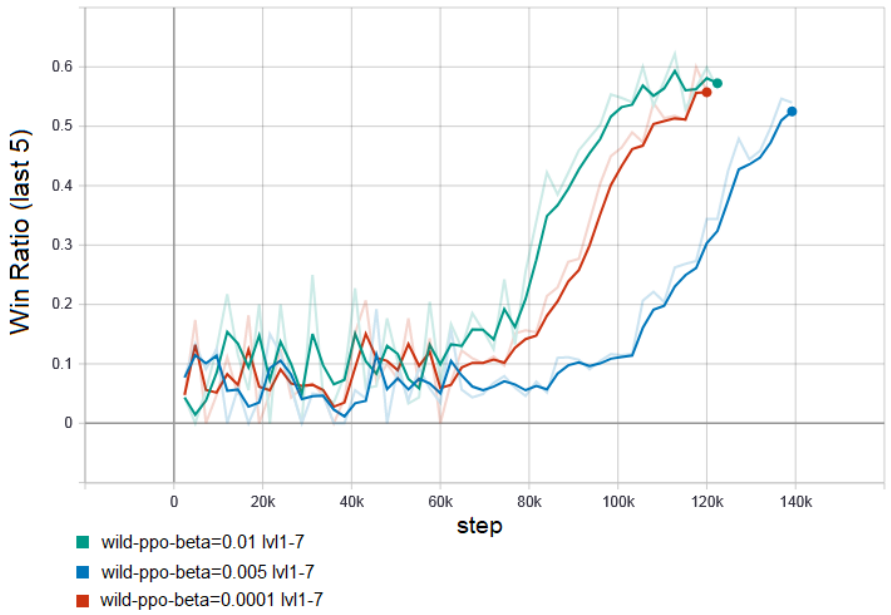


Recompensa acumulada no algoritmo PPO, abordagem Regular, nível 5, variando beta



Recompensa acumulada no algoritmo PPO, abordagem Wild, níveis 1 a 7, variando beta.

Resultados - Múltiplos níveis



Recompensa acumulada no algoritmo PPO, abordagem Regular, nível 5, variando epsilon

- Agentes foram treinados com cerca de 10 puzzles com semelhante nível de dificuldade
- Ambos os métodos conseguiram resolver níveis para o qual não foram treinados
- Nível 4: Wild (≅20%) , Regular(100%)

Conclusões

- Obtivemos resultados positivos com ambas as abordagens.
- Ambas as abordagens ofereceram vantagens e desvantagens em relação à outra.
- O algoritmo PPO teve muito melhor performance do que SAC.
- ML-Agents poderá não ter sido ideal para este tipo de problema-

Trabalho Relacionado

- Deep Reinforcement Learning for 2048 http://www.mit.edu/~amarj/files/2048.pdf
- Improving Generalization Ability in a Puzzle Game Using Reinforcement Learning http://www.cig2017.com/wp-content/uploads/2017/08/paper_71.pdf
- Applying Reinforcement learning to find the logic puzzles solution https://journals.tu-plovdiv.bg/index.php/journal/article/view/16/14
- Deep Reinforcement Learning Bot That Plays Tetris https://github.com/nuno-faria/tetris-ai
- Playing Tetris with Deep Reinforcement Learning http://cs231n.stanford.edu/reports/2016/pdfs/121_Report.pdf
- ML Agents Learning Environment Examples – GridWorld https://github.com/Unity-Technologies/ml-agents/blob/master/docs/Learning-Environment-Examples.md#gridworld