IART - 2º Projeto
1C - Aprendizagem por Reforço - Bubble Blast

João Leite
3MIEIC04
FEUP
up201705312@fe.up.pt

Inês Alves
3MIEIC04
FEUP
up201605335@fe.up.pt

Márcia Teixeira
3MIEIC04
FEUP
up201706065@fe.up.pt

***Abstract*—O objetivo deste projeto é ensinar um agente a resolver vários níveis do jogo 2D Bubble Blast através de aprendizagem por reforço, utilizando os métodos PPO e SAC com a ajuda do ML-Agents Toolkit do Unity.**

***Index Terms*—inteligência artificial, aprendizagem por reforço, mlagents, unity, ppo, sac**

I. Introdução

Este projeto foi desenvolvido na Unidade Curricular de Inteligência Artificial e visa ensinar um agente a solucionar o puzzle 2D Bubble Blast através de aprendizagem por reforço. O presente artigo está estrutrado da seguinte forma:

II. Descrição do Problema: descrição detalhada do problema tratado.

III. Abordagem: apresentação da abordagem utilizada para resolver o problema.

IV. Avaliação Experimental: resultados obtidos no trabalho, representados através de vários gráficos.

V. Conclusões: conclusões retiradas do projeto e análise crítica dos resultados obtidos.

II. Descrição do Problema

Bubble Blast é um puzzle 2D com um tabuleiro com 6 linhas e 5 colunas. As várias células podem ter bolhas, podendo estas bolhas ter diferentes tamanhos e cores (sendo as cores, por ordem crescente de tamanho: azul, amarelo, verde e vermelho). O utilizador pode carregar numa bolha, fazendo-a crescer e passar à cor seguinte ou, caso seja já uma bolha vermelha, rebentá-la. Quando uma bolha rebenta, esta desaparece e dá origem a 4 bolhas pequenas que se deslocam nos dois sentidos, horizontalmente e verticalmente. A colisão de uma bolha pequena com uma bolha normal tem o mesmo efeito que um toque do utilizador. O objetivo final é obter um tabuleiro vazio dentro de um número limitado e pré-definido de toques, que varia de acordo com o nível.

Pretende-se ensinar um agente a resolver este tipo de puzzles utilizando aprendizagem por reforço. Na aprendizagem por reforço, o agente recebe uma recompensa pelas ações tomadas e tenta maximiza-la ao longo do tempo.

III. Abordagem

Como dito anteriormente, a aprendizagem por reforço usa uma política de recompensas para treinar o agente. É dada uma recompensa ao agente por cada ação que toma, e o agente tenta maximizar o valor acumulado das recompensas ao longo do tempo.

No caso da resolução de puzzles Bubble Blast, escolheu-se uma recompensa de 1.0 sempre que se atinge um tabuleiro vazio (vitória), -0.8 quando o número de toques foi esgotado e o tabuleiro não está vazio (derrota), -0.05 quando o agente tenta carregar numa célula vazia e -0.01 por cada toque numa bolha. Dá-se uma recompensa negativa por cada toque de forma a que o agente aprenda a utilizar o mínimo número de toques possíveis.

Para este trabalho, utilizamos o Toolkit ML-Agents do Unity, aproveitando os seus algoritmos PPO (Proximal Policy Optimization) e SAC (Soft Actor Critic). Começou-se por implementar o jogo e a sua lógica em Unity, ficando este jogável por um humano. De seguida criou-se um agente que deve ser treinado para resolver os vários níveis. Este agente realiza periodicamente observações do ambiente, colhendo informação sobre a posição das várias bolhas presentes no jogo e o seu valor (relacionado com a cor e tamanho referidos na secção anterior). Depois de obter as observações o agente toma uma decisão sobre a ação a realizar, recebendo de seguida uma das recompensas referidas acima.

Por fim, treinou-se o agente para resolver vários níveis do Bubble Blast, tendo-se realizado vários treinos para o nível 6, com ambos os algoritmos, fazendo variar metaparâmetros de forma a perceber o impacto destes na aprendizagem.

Treinou-se também um agente para resolver 4 níveis diferentes, de dificuldade relativamente reduzida, utilizando o algoritmo PPO.

IV. Análise Experimental

Para analisarmos os resultados do nosso programa, utilizamos o Tensorboard para gerar gráficos de cada treino.

A. Análise do PPO

Para o PPO, treinamos o agente no nosso nível 6 com metaparâmetros default e fizemos depois variar três metaparâmetros: o beta, o epsilon e o lambda, entre valores máximos e mínimos, de modo a observar como estas mudanças influenciariam os resultados.

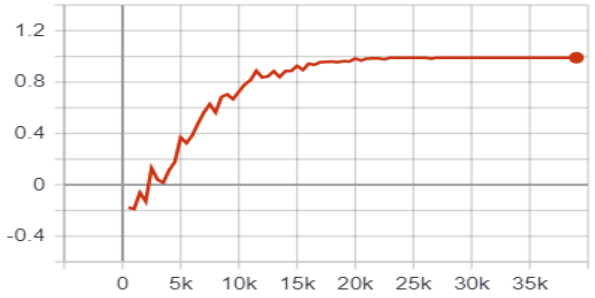

Fig. 1. Gráfico da Recompensa Cumulativa do treino do nível 6 com metaparâmetros default (Beta = 5.0e-3; Lambda = 0.92; Epsilon = 0.1)

No gráfico da figura 1 observa-se que a recompensa só atinge o valor desejado e estabiliza a partir de cerca de 30000 passos.

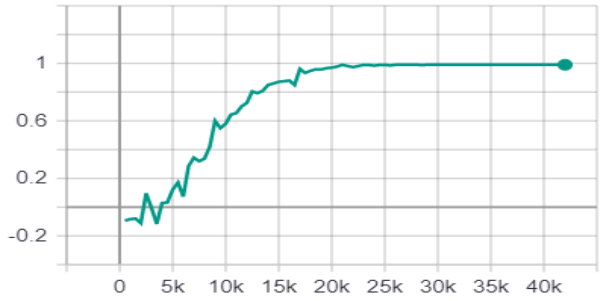

Fig. 2. Gráfico da Recompensa Cumulativa do treino do nível 6 com o valor Beta mínimo (= 1.0e-4)

O metaparâmetro beta do PPO representa a força da regularização da entropia. Isto significa que se o beta tiver um valor mínimo, a entropia baixará muito depressa, o que, comparativamente a um beta de valor máximo, faz com que a curva do gráfico estabilize mais cedo - como a entropia está associada às acções aleatórias por parte do agente, quanto menor ela for, mais rapidamente converge para a solução.

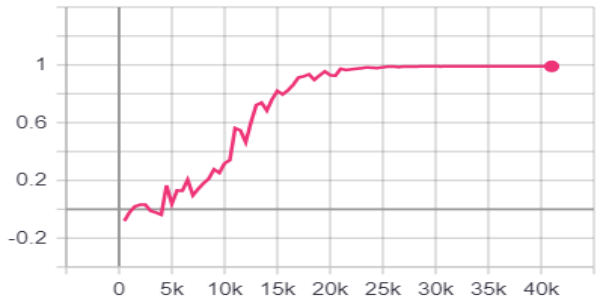

Fig. 3. Gráfico do Recompensa Cumulativa do treino do nível 6 com o valor Beta máximo (= 1.0e-2)

Como explicado anteriormente, o valor de beta está associado à entropia, que por sua vez está ligada ao agente realizar acções aleatórias. Neste caso, em que o beta assume um valor máximo, a entropia subirá significativamente, o que causa uma instabilidade no treino do agente, algo que se verifica pelas oscilações da curva do gráfico e uma maior dificuldade em atingir o valor de recompensa máximo.

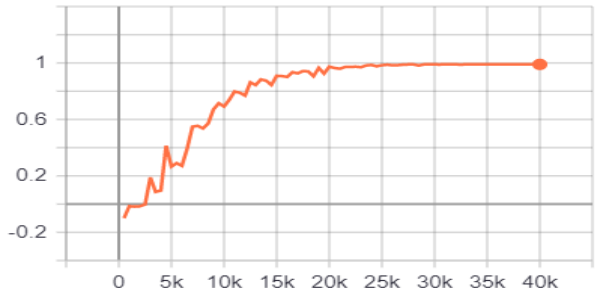

Fig. 4. Gráfico do Recompensa Cumulativa do treino do nível 6 com valor Lambda mínimo (= 0.90)

O valor de lambda influencia o quanto o agente se baseia na sua estimativa atual quando calcula uma nova estimativa atualizada. Altos valores de lambda levam a estimativas mais tendenciosas que se baseiam mais nos valores estimados atuais, podendo-se observar estes resultados no gráfico da figura 5. Por outro lado, valores de lambda mais baixos fazem com que o agente se baseie mais nas recompensas que recebe do ambiente, o que pode levar a um processo de aprendizagem com mais variações, como se pode concluir pelo gráfico da figura 4, em que a curva da recompensa acumulada tem grandes oscilações antes de estabilizar.

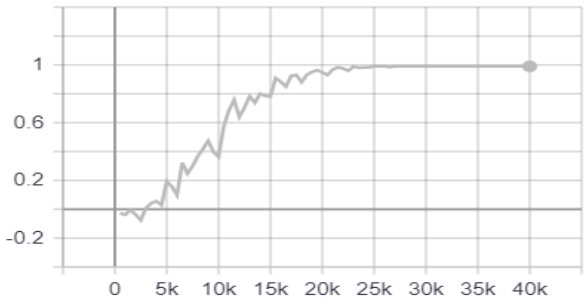

Fig. 5. Gráfico do Recompensa Cumulativa do treino do nível 6 com valor Lambda máximo (= 0.95)

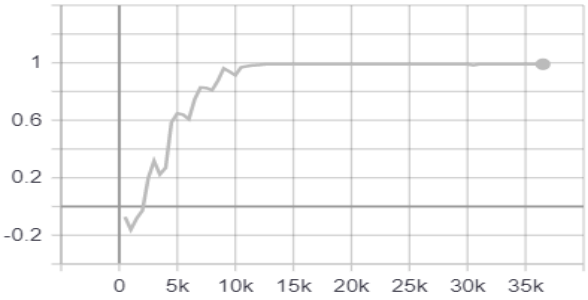

Fig. 6. Gráfico do Recompensa Cumulativa do treino do nível 6 com valor Epsilon máximo (= 0.3)

O valor de Epsilon afeta a velocidade de evolução da política do agente, ajustando o limite de divergência entre

a política anterior e a nova política. Valores mais altos de epsilon aumentam o limite de divergência, resultando numa evolução mais rápida da aprendizagem, que se pode notar pelo facto de no gráfico da figura 6 o valor da recompensa estabilizar mais cedo, entre os 10000 e 15000 passos. No entanto, esta aprendizagem mais rápida resulta também numa maior oscilação dos valores, como se pode observar pela maior oscilação inicial da curva comparativamente aos gráficos anteriores.

B. Análise do SAC

Por sua vez, para o SAC, treinamos de igual modo o nível 6 mas variando apenas o coeficiente de entropia (init entcoef).

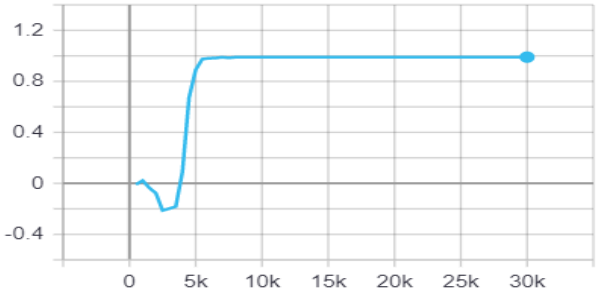

Fig. 7. Gráfico da Recompensa Cumulativa do treino do nível 6 com metaparâmetros default (coeficiente de entropia = 0.25)

No gráfico da figura 7 é possível observar que, utilizando o algoritmo SAC com os valores pré-definidos para os seus metaparâmetros, a recompensa cumulativa atinge o valor desejado em pouco mais de 5000 passos, estabilizando o seu valor a partir desse momento.

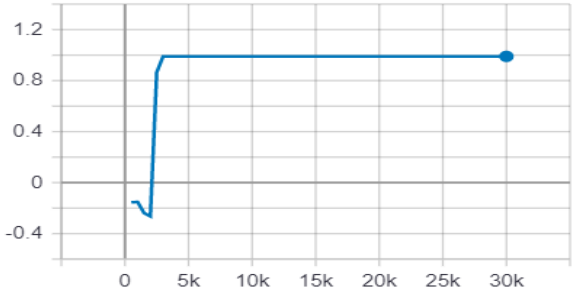

Fig. 8. Gráfico da Recompensa Cumulativa do treino do nível 6 com o coeficiente de entropia mínimo (= 0.05)

No gráfico da figura 8 observa-se que, com a diminuição do coeficiente de entropia inicial, o agente converge para uma solução mais rapidamente, com o valor desejado da recompensa a ser atingido e a estabilizar por volta dos 4000 passos. Há uma menor exploração por parte do agente no início e encontra-se a solução mais rapidamente.

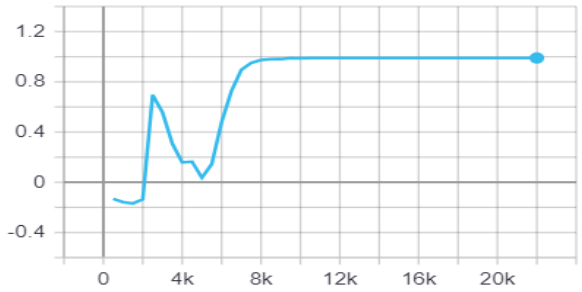

Fig. 9. Gráfico da Recompensa Cumulativa do treino do nível 6 com o coeficiente de entropia máximo (= 0.50)

No gráfico da figura 9 observa-se que, com o aumento do coeficiente de entropia para um valor máximo, comparativamente ao gráfico default (Fig. 4), há um desvio significativo da recompensa acumulada. Isto deve-se ao facto de um coeficiente elevado levar a uma maior exploração por parte do agente no início do treino.

C. Múltiplos Níveis

Treinou-se um agente para resolver 4 níveis diferentes, utilizando-se o algoritmo PPO com os valores de meta-parâmetros *default* definidos das subsecções anteriores, tendo-se obtido os resultado do gráfico da figura 10. Depois de a aprendizagem com estes 4 níveis estar concluída, utilizou-se o agente para resolver outros níveis do Bubble Blast, tendo-se verificado que este sucedia nos níveis de complexidade baixa mas não conseguia resolver outros de maior complexidade.

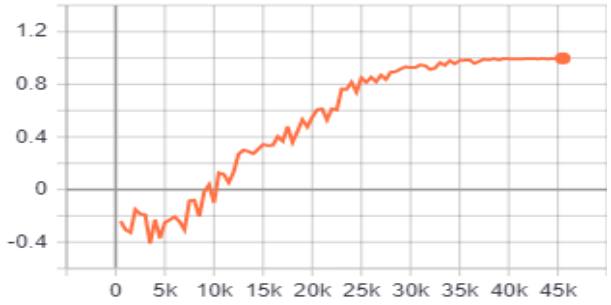

Fig. 10. Gráfico da Recompensa Cumulativa do treino com vários níveis com o algoritmo PPO

V. Conclusão

Através deste projeto foi-nos possível ensinar um agente a jogar um jogo 2D através de aprendizagem por reforço, nomeadamente com os métodos PPO e SAC. Com a análise dos gráficos conseguimos estudar melhor estes algoritmos e concluir que os seus vários metaparâmetros influenciam significativamente os resultados obtidos. Consideramos que conseguimos aprofundar os nossos conhecimentos sobre aprendizagem por reforço e que o resultado do trabalho foi positivo. Apesar disto, achamos que poderíamos ter usado amostras maiores para melhorar as redes neuronais e os resultados obtidos. Também consideramos que o MLAgents poderia ter

sido mais bem aproveitado noutro tipo de projeto que não um jogo de tabuleiro, visto que para problemas de espaços discretos o PPO e o SAC não são os melhores algoritmos, e o próprio toolkit não tem suporte para matrizes de valores.