Segundo Trabalho de Inteligência Artificial

Elaine Dias Pires

9 de julho de 2023

Resumo

O presente trabalho visa aprofundar os conceitos de aprendizagem por reforço por meio da criação de um agente inteligente. O agente foi desenvolvido com o objetivo de jogar o jogo Dino do Google. Para criá-lo foi utilizado o classificador K-Nearest Neighbors e a meta heurística Simulated Annealing. Após os treinamentos, o agente inteligente alcançou uma média de 370.775 e uma máxima de 919.75 pontos.

1 Introdução

O reinforcement learning é o treinamento de modelos de machine learning que se baseiam em uma sequência de decisões. Nesse tipo de aprendizado, há um agente inteligente que aprende a tomar melhores decisões em um ambiente complexo e de inceterza. Devido a essas característica, o reinforcement learning vem sendo bastante utilizado em áreas como robótica, simulação e treinamento e principalmente jogos virtuais. Para criar o agente inteligente é bastante comum utilizar um classificador e uma meta heurística.

Uma vez que foram estudados alguns classificadores e metaheurísticas ao longo da disciplina, neste trabalho foi desenvolvido um agente inteligente utilizando esses conhecimentos. Devido ao número de matrícula, o classificador utilizado foi o KNN e a metaheurística Simuated Annealing. Tanto o KNN quanto o Simulated Annealing foram implementados manualmente.

2 Descrição do Classificador

O classificador utilizado foi o KNN. Este classificador é baseado na ideia de que objetos semelhantes tendem a estar mais próximos no espaço das características.

Para utilizá-lo foi necessário gerar uma base de dados com características advindas do jogo e sua respectiva classificação. Assim, para um dado estado atual eram calculados as distâncias entre esse estado atual e todos os outros estados. A ação escolhida era a que mais aparecia nos targets dos k estados mais próximos do estado atual.

Como é necessário calcular todas as distâncias diversas vezes, o KNN pode ser computacionalmente caro, por causa disso foi escolhido apenas quatro atributos do jogo **distance, speed, obHeight, nextObDistance** para a base. A distância utilizada foi a distância euclidiana, com $k = 5$. Os dados da base foram obtidos a partir do teclado jogando o jogo no modo HUMAN_MODE , no total a base possui 1702 linhas. Vale salientar que os resultados obtidos são altamente dependentes da base.

Inicialmente, os dados foram normalizados com StandardScaler, porém os resultados obtidos com a normalização foram inferiores comparados aos dados sem normalização, por isso optou-se por não normalizar a base, apesar de ser uma estratégia incomum.

3 Descrição da Meta Heurística

A meta heurística utilizada foi o Simulated Annealing (Recozimento Simulado). Essa meta heurística se baseia no processo físico do recozimento da metalurgia. O simulated annealing começa com uma solução inicial e a cada iteração gera uma nova solução que é comparada com a atual. Se a nova solução for melhor que a atual, a atual é substituída. Apesar disso, soluções piores também podem ser aceitas dependendo de uma probabilidade que é ajustada a cada iteração. A chance de uma solução

pior ser aceita vai dimuindo a cada iteração, conforme o esquema de resfriamento simulado, pois a cada a iteração, a temperatura diminui. O motivo pelo qual soluções piores podem ser aceitas é porque o algoritmo tenta explorar todo o espaço de soluções e não apenas focar em soluções ótimas de um subespaço.

Na implementação feita, o custo da uma base era avaliado mediante a pontuação obtida no jogo. A cada iteração do simulated annealing eram geradas 3 novas bases e isso era feito até o número de máximo de iterações e enquanto o tempo máximo não era atingido e a temperatura era maior ou igual a 1. A temperatura inicial foi 900, o número máximo de iterações foi 2 e o alfa foi 0.8.

4 Resultados

Os resultados obtidos foram:

	Teacher's Scores	My Scores
Resultado 1	1214.0	677.0
Resultado 2	759.5	82.0
Resultado 3	1164.25	885.5
Resultado 4	977.25	317.0
Resultado 5	1201.0	25.25
Resultado 6	930.0	919.75
Resultado 7	1427.75	314.25
Resultado 8	799.5	68.75
Resultado 9	1006.25	227.25
Resultado 10	783.5	86.75
Resultado 11	728.5	101.75
Resultado 12	419.25	409.75
Resultado 13	1389.5	461.5,
Resultado 14	730.0	49.0,
Resultado 15	1306.25	27.0
Resultado 16	675.5	676.5
Resultado 17	1359.5	880.0
Resultado 18	1000.25	122.25
Resultado 19	1284.5	247.5,
Resultado 20	1350.0	112.0
Resultado 21	751.0	371.25
Resultado 22	1418.75	23.25
Resultado 23	1276.5	607.0
Resultado 24	1645.75	538.25
Resultado 25	860.0	142.5
Resultado 26	745.5	717.25
Resultado 27	1426.25	703.25
Resultado 28	783.5	42.5
Resultado 29	1149.75	883.5
Resultado 30	1482.25	403.75
Mean	1068.18	370.775
Std	304.035	301.16

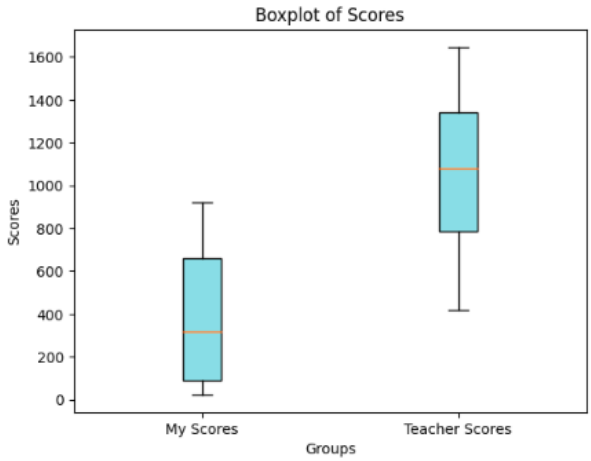

Figura 1: Boxplot

	my_score	teacher_score
my_score	my_score	0.000002
teacher_score	0.000002	teacher_score

Figura 2: Testes

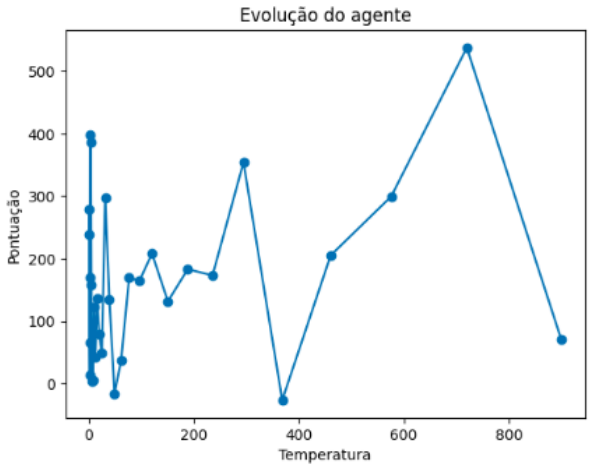

Figura 3: Evolução do agente

5 Conclusões

5.1 Análise geral dos resultados

Pode-se perceber pelos resultados que o agente inteligente implementado apresentou resultados inferiores aos dos professor. A média obtida foi bastante inferior, com alguns resultados semelhantes, como o resultado 6 e o resultado 16. Essa diferentes de resultados também pode ser observada nos testes, pois há diferente estatística a 95% de significância. No boxplot, os resultados obtidos também estão abaixo dos resultados do professor, com a maioria se concentrando entre por volta de 300 e 600 pontos.

Em relação à evolução do agente, o gráfico confunde um pouco, pois no simulated annealing, o algoritmo aceita soluções a medida que a temperatura diminui e não a medida que a temperatura

aumenta. Podemos observar que logo de início foi encontrada uma solução ótima, mas devido a própria natureza da metaheurística, a mesma aceitou soluções piores para tentar explorar todo o espaço de soluções.

5.2 Contribuições do Trabalho

Apesar do KNN e do simulated annealing não serem o classificador e a metaheurística mais utilizados para jogar o jogo do dino, este trabalho pode contribuir como forma de comparação com outros agentes inteligentes e também como estudo para pesquisadores e alunos interessados em aprendizado por reforço, em especial aqueles mais interessados em jogos virtuais.

5.3 Melhorias e trabalhos futuros

Uma melhoria importante a ser feita é utilizar uma base melhor, seja com menor número de linhas, as quais sejam significativas para o classificador e/ou com diferentes atributos do jogo. Também é importante normalizar os dados e manter/aumentar a pontuação, por ser a estratégia mais comum. Além disso, realizar mais testes com diferentes de parâmetros, com temperaturas e números de iterações maiores. Dessa forma, acredita-se que os resultados obtidos serão melhores.

6 Referências Bibliográficas

[1]NORVIG, Peter; Russel, Stuart. Inteligência Artificial. 3a. ed. Rio de Janeiro: Campus, 2013.[2]WITTEN, I. H.; FRANK, Eibe; HALL, Mark A. Data mining: practical machine learning tools and techniques.3rd ed. Burlington, Mass.: Morgan Kaufmann, 2011. xxxiii, 629 p. (Morgan Kaufmann series in data managementsystems).[3] MULLER, Andreas C. and Guido, Sarah. Introduction to Machine Learning with Python: A Guide to DataScientists, 2017.[4] LUGER, George F. Artificial intelligence: structures and strategies for complex problem solving. 6th ed. Boston,Mass.: Pearson Addison Wesley, 2009. xxiii, 754 p.