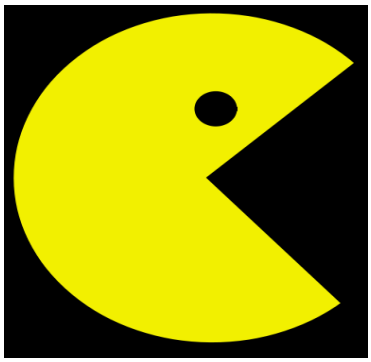


Introdução à Inteligência Artificial
Trabalho Prático nº2- Rede Neuronal

Grupo constituído por:

- Bruno Miguel Oliveira Rolo (2010131200)
- João Artur Ventura Valério Nobre(2010131516)
- Daniel Luís Caetano Pereira (2006124813)

Grupo/Tarefas

Nome: Bruno Rolo

Número de Estudante: 2010131200

Email: brolo@student.dei.uc.pt

Turma Prática: PL5

Esforço Total para a realização deste trabalho: Aproximadamente 10 horas de estudo e 15 de implementação.

Tarefas Desempenhadas: Desenvolvimento do agente reactivo, desenvolvimento da rede neuronal, e testes sobre o desempenho da rede, criação do relatório.

Nome: Daniel Pereira

Número de Estudante 2006124813

E-mail dlcp@student.dei.uc.pt

Turma Prática: PL5

Esforço: 10 horas de estudo, 15 de implementação.

Tarefas Desempenhadas: Desenvolvimento, testes sobre o desempenho de cada Agente, criação do relatório.

Nome: João Nobre

Número de Estudante: 2010131516

E-mail janobre@student.dei.uc.pt

Turma Prática: PL5

Esforço Total para a realização deste trabalho: Aproximadamente 10 horas de estudo, 15 de implementação.

Tarefas Desempenhadas: Desenvolvimento do agente reactivo, desenvolvimento da rede neuronal, e testes sobre o desempenho da rede, criação do relatório.

1. Descrição

1.1-Topologia

Com o objectivo de desenvolver um agente aprendiz para o jogo pacman decidimos criar uma rede neuronal com a seguinte topologia:

- 20 entradas
- 10 neurónios na camada escondida (valor a considerar ao longo dos testes)
- 5 Neurónios na camada de saída

Como dito no enunciado iremos ter 20 entradas, não necessariamente neurónios, onde todas as entradas estão ligadas a todos os neurónios pertencentes à camada escondida. Os 10 neurónios da camada escondida, número obtido de forma empírica, ao longo dos testes e análises foi o valor que melhor desempenho obteve, tendo por fim 5 neurónios na camada de saída.

1.2-Função de Activação

A função de ativação que iremos utilizar será a função sigmoide que é definida como:

$$P(t) = 1/(1 + e^{-t})$$

Visto que a função sigmóide tem características importantes como de ser não linear, contínua, diferenciável, monotonamente não decrescente e ter uma derivada fácil de calcular.

1.3- Inicializações

Para começar com o processo de aprendizagem iremos inicializar os pesos das ligações da rede neuronal com valores aleatórios entre [-0.5, 0.5], como recomendado na documentação da cadeira. Estes mesmos pesos serão melhorados à medida que o algoritmo de retro propagação redimensionar os pesos a cada iteração. Ao longo dos testes chegámos a conclusão que a camada escondida com 10 neurónios seria a melhor escolha, o equilibrio entre bom desempenho e tempo de execução do jogo pareceu-nos a melhor opção a tomar.

Com um ritmo de aprendizagem de 0.1, embora a diferença entre os ritmos testados não era significativa, mas tendo em conta que quanto menor for o ritmo maior número de iterações será necessária para uma melhor aprendizagem da rede.

2. Desenvolvimento

Na implementação começamos por criar um agente reativo, no ficheiro iiAgents.py, de modo a criar padrões de treino para a rede.

No ficheiro LearningAgents.py desenvolvemos a nossa rede neuronal. Começamos por inicializar os valores da rede, com os valores ditos anteriormente. Depois de inicializados treinamos a rede com 500 iterações, foram testados vários números máximos de iterações, possibilitando assim a execução do código em tempo útil. Percebendo ao longo dos testes que com 500 iterações já contém um valor aceitável de erro, já aprendendo o esperado.

Em cada iteração treinávamos a rede, isto incluiu treinar simplesmente a rede, com 70% dos casos de treino obtidos, é de salientar que os casos de treino são filtrados ao inicio excluindo os repetidos. Já que os dados duplicados tendem a ser prejudiciais, fazendo com que veja padrões frequentes muitas vezes e padrões raros poucas vezes, por comparação. Consequentemente aprende o que é usual, e os casos menos frequentes não chega a aprender porque está demasiado ocupada a memorizar o resto.

Depois verificámos o erro quadrático, para isso propagávamos com 10% dos casos de treino, nunca vistos pela rede, calculando de seguida o erro quadrático de todos os casos. No final de todas as iterações validávamos a rede, com 20% dos casos de treino, que são desconhecidos à rede e verificando se o erro difere em relação ao erro calculado no treino.

A implementação das funções que calculam a propagação e a retropropagação da rede foram baseada em fórmulas apresentadas na documentação da cadeira.

Na função getAction que define a ação do pacman no jogo, inicializámos um vector com os 5 movimentos possíveis, propagamos o caso de jogo atual, com a respetiva entrada, a propagação retorna um vetor com as 5 saídas da rede treinada, ordenamos esse vector em conjunto com o vector dos 5 movimentos possíveis, crescentemente. Percorrendo de seguida o vector ordenado, se o movimento em questão for legal, então é executado, seguindo assim uma abordagem “winner takes all”.

Ao longo do desenvolvimento do agente aprendiz, para além da implementação, propriamente dita, fomos analisando os resultados que a rede neuronal nos retornava, questionando se o agente estaria a ter um comportamento esperado, e se deveríamos alterar os

valores pré-definidos da rede, para melhorar o seu rendimento. Com este processo obtivemos assim de forma empírica os valores da rede neuronal.

2.1. Recolha de Padrões de Treino

A partir do jogador humano iremos captar os nossos ficheiros de treino para realizar a aprendizagem do agente, de maneira que o agente tenha o melhor comportamento possível.

Iremos dividir os casos de treino obtidos em 3 grupos:

1. Treinar a rede neuronal
2. Testar a rede treinada
3. Verificar se o erro quadrático não terá uma grande variação comparada com o erro obtido na altura dos treinos.

Criámos padrões de treino tanto com o agente reactivo desenvolvido, como também criámos padrões de treino com um jogador humano. Cerca de 50 ficheiros. Foram desenvolvidos no mapa MediumClassic (default), MediumScaryMaze e o CapsuleClassic.

Tentámos na recolha dos padrões que os nossos casos de treinos fossem mais diversificados possíveis. Dando a rede input's mais variados possíveis, com máximo de situações de jogo possível.

2.2. Treino da Rede Neuronal

Dividimos os padrões de treino recolhidos em 3 grupos, 70% dos casos foram usados para treinar a rede, 10% foram usados para testar a rede em cada iteração, a função dos restantes 20% era verificar depois de treinada se a rede respondia com um erro quadrático próximo do obtido no treino.

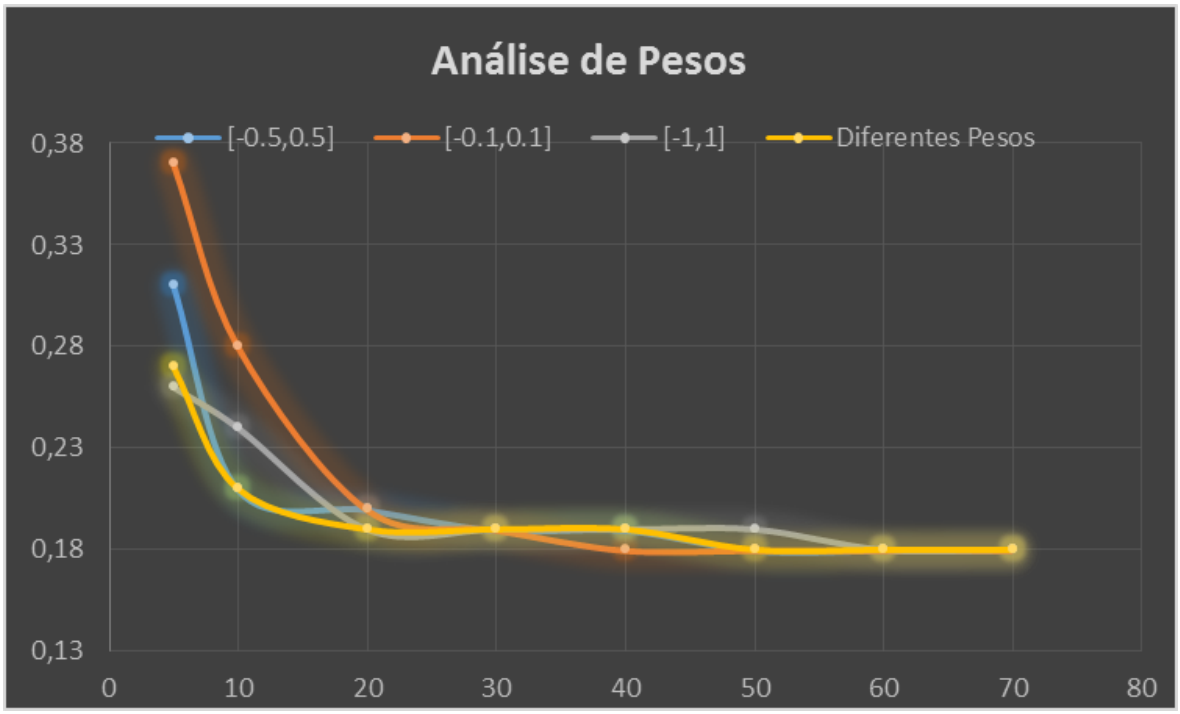
2.3. Critérios de paragem

Um dos critérios é o número de iterações, que são 500. Testámos vários números máximos de iterações, possibilitando assim a execução do código em tempo útil. Acabando essas iteracções obtemos já um valor aceitável de erro. O segundo critério foi definido como o sucessivo decréscimo do erro até este começar a aumentar, isto se já tiverem passado 100 iterações sem existir melhoria, porque este mesmo erro pode oscilar. Por fim o terceiro critério, optámos por incluir um valor de erro aceitável a partir do qual os ganhos e perdas da rede não justificava o tempo perdido no cálculo. Havendo assim um maior equilíbrio entre número de treino e tempo de jogo, valor esse determinado ao longo dos testes, percebendo assim o que seria já um valor aceitável. Ainda referente ao erro esperado, definimos o seu valor com base nos testes recolhidos pelo LearningKeyboardAgent.

2.4. Valores Certos

Ao longo dos testes à rede utilizámos ritmos de aprendizagem com os seguintes valores 0.01, 0.03, 0.1 e 0.3. Chegando à conclusão que quanto menor for o ritmo de aprendizagem, mais iterações são necessárias para atingir um valor de erro aceitável, tendo um o comportamento esperado. Embora tenhamos variado o ritmo de aprendizagem, em termos de resultados, as diferenças não se mostraram significativas.

Para inicializar os pesos optamos pelo intervalo -0.5 a 0.5, não encontrando diferenças significativas, comparadas com outros intervalos, como está demonstrada no seguinte gráfico:



Anexo 1: Análise de Pesos comparando diferentes intervalos

3. Teste e Análise

Para analisar o comportamento do agente, efectuámos vários testes, representados pelas seguintes tabelas, que contém para cada ritmo de aprendizagem e para cada número de neurónios da camada escondida três corridas. Recolhemos os dados mais relevantes, como padrões de jogo que a rede acerta, erra e desconhece em relação aos padrões fornecidos no treino.

Assinalámos em cada número de neurónios na camada escondida a vermelho a percentagem de padrões mais baixa, e a verde a percentagem de padrões mais elevada.

A tabela contém o seguinte template:

PROJECT PERFORMANCE

MAP:<nomedomapa>

	Input			Output						
Caso nº	Tipo de Agente	Neurónios da Camada Escondida	Ritmo de Aprendizagem	Erros	Média Erro	Padrões Acertados	Padrões Errados	Padrões não Conhecidos	Número de Movimentos	Movimentos Ilegais
1	…	…	….	Erro1 Erro2 Erro3	(M1+M2+ M3)/3	PA1 PA2 PA3	PE1 PE2 PE3	PNC1 PNC2 PNC3	NM1 NM2 NM3	Mi1 Mi2 Mi3

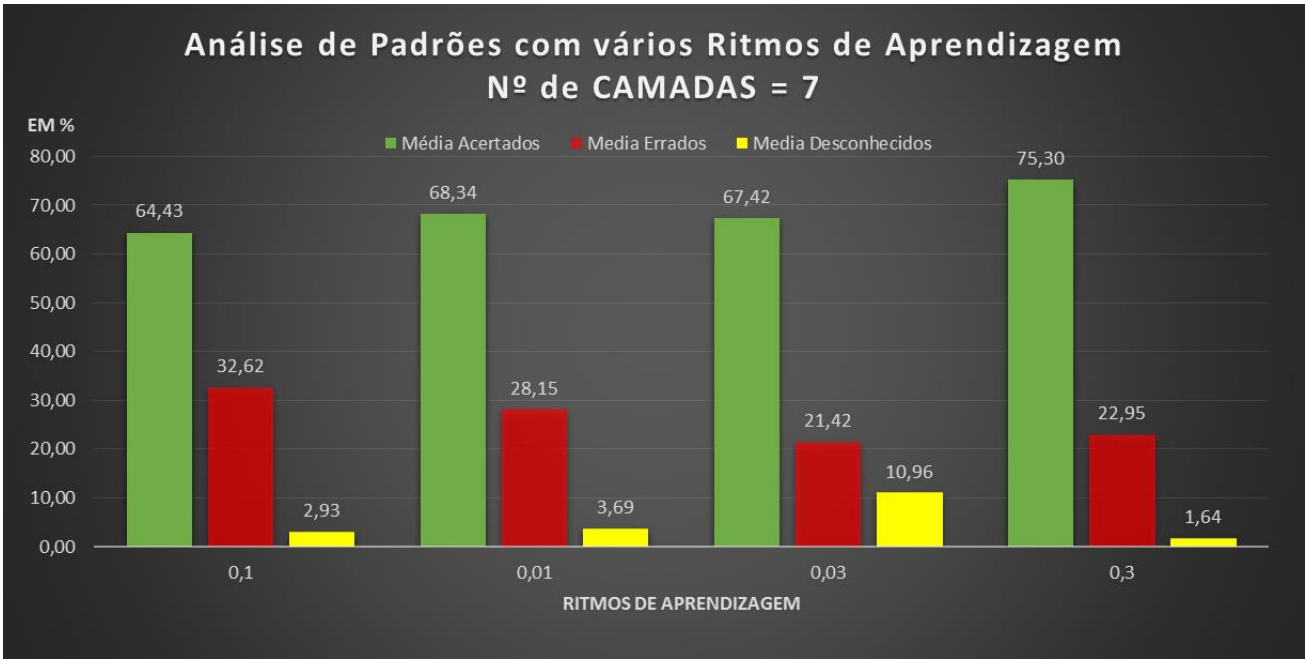
Caso nº: Identificador do caso.
Tipo de Agente: Tipo de agente utilizado (Humano ou Reactivo)
Neurónios da Camada Escondida: Número de neurónios da camada escondida
Ritmo de Aprendizagem: Ritmo de aprendizagem(0.1 ,0.01,0.03 ou 0.3)
Erros: Erro obtido em cada uma das 3 corridas.
Média Erro: Média Obtida em função dos 3 erros obtidos
Padrões Acertados: Percentagem de padrões acertados em cada uma das 3 corridas
Padrões Errados: Percentagem de padrões errados em cada uma das 3 corridas
Padrões não conhecidos: Percentagem de padrões desconhecidos em cada uma das 3 corridas
Número de Movimentos: Número de movimentos efectuados pelo Pacman em cada uma das 3 corridas
Movimentos Ilegais: Percentagem de movimentos Ilegais em cada uma das 3 corridas

PROJECT PERFORMANCE

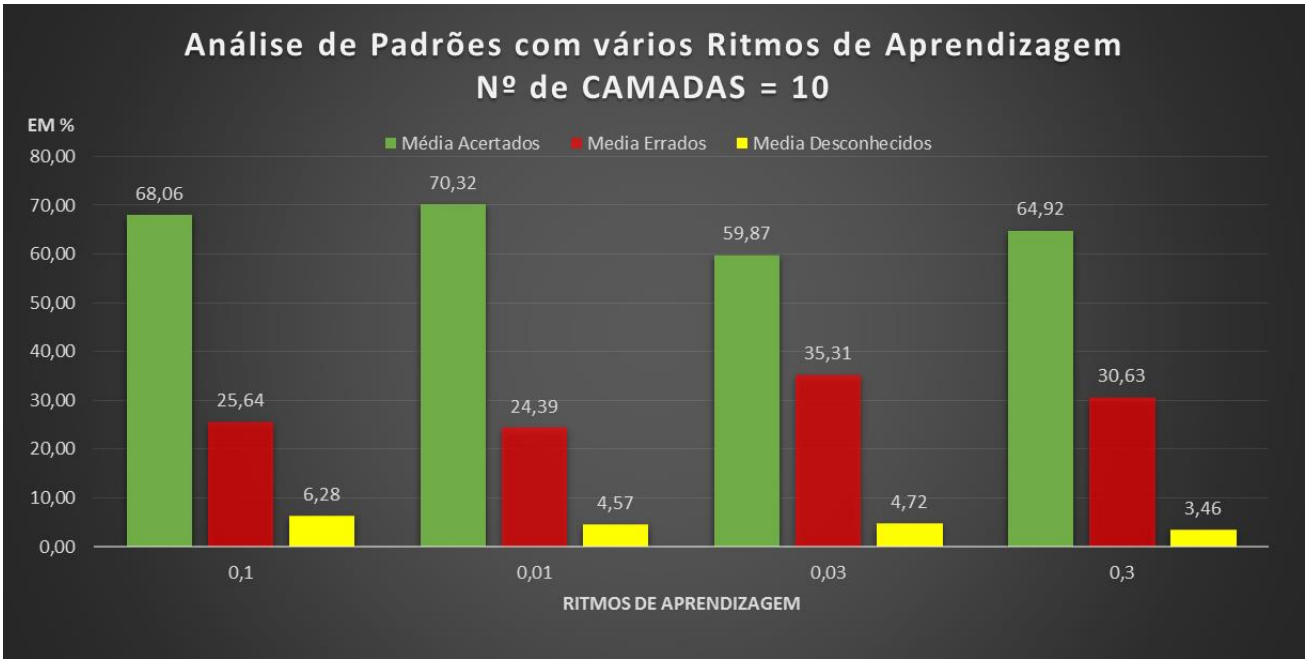
MAP:DEFAULT

	Input			Output						
Caso nº	Tipo de Agente	Neurónios da Camada Escondida	Ritmo de Aprendizagem	Erros	Média Erro	Padrões Acertados	Padrões Errados	Padrões não Conhecidos	Número de Movimentos	Movimentos Ilegais
1	Humano	7	0.1	0.1911 0,1841 0,1662	0,180493	81,81% 63,42% 48,07%	18,18% 33,49% 46,18%	0,0% 3,07% 5,73%	11 812 1429	0,0% 0,49% 2,04%
2	Humano	7	0.01	0,1991 0,1747 0,1408	0,171533	70,0% 53,29% 81,73%	30,0% 42,69% 11,76%	0,0% 4,01% 7,05%	10 349 340	0,0% 0,0% 0,29%
3	Humano	7	0.03	0,1794 0,1904 0,1932	0,187667	64,28% 68,30% 69,69%	21,42% 28,49% 14,34%	14,28% 2,70% 15,90%	14 407 132	7,14% 2,70% 0,75%
4	Humano	7	0.3	0,2195 0,1484 0,2077	0,191733	70% 88,89% 67,61%	30% 8,09% 30,77%	0,0% 3,03% 1,90%	10 99 107	0,0% 1,01% 0,0%
5	Humano	10	0,10	0,2195 0,1945 0,1960	0,203333	66,44% 62,00% 75,75%	28,33% 28,90% 19,69%	5,22% 9,09% 4,54%	1186 737 66	0,0% 0,0% 0,0%
6	Humano	10	0,01	0,1824 0,1911 0,1805	0,184680	72,68% 85,16% 53,11%	19,38% 9,57% 44,22%	7,92% 5,31% 2,66%	227 94 450	0,0% 0,0% 0,223%
7	Humano	10	0,03	0,1881 0,1833 0,1834	0,184933	65,84% 82,19% 31,37%	30,49% 11,43% 64,01%	3,61% 6,36% 4,20%	692 691 214	0,0% 0,43% 0,46%
8	Humano	10	0,30	0,1969 0,1875 0,1791	0,187850	55,82% 70% 68,93%	38,97% 30% 22,93%	5,19% 0,0% 5,19%	1048 10 77	0,54% 0,0% 0,0%
9	Humano	15	0,10	0,1828 0,1833 0,1799	0,182000	61,15% 38,84% 57,27%	38,01% 55,72 41,02%	0,82% 5,42% 1,70%	121 1547 529	0,0% 0,19% 0,0%
10	Humano	15	0,01	0,1816 0,1914 0,1740	0,182347	69,07% 62,25% 31,75%	1,97% 32,85% 64,05%	28,44% 4,88% 4,197%	304 1391 548	0,0% 0,50% 0,18%
11	Humano	15	0,03	0,1856 0,1817 0,1784	0,181930	83,92% 41,30% 70%	10,71% 54,98% 30%	5,35% 4,21% 0,0%	56 1661 10%	0,0% 0,18% 0,07%
12	Humano	15	0,3	0,1952 0,1499 0,2046	0,183263	95,94% 45,75% 60,46%	1,45% 45,50% 36,43%	2,59% 8,73% 3,10%	1308 1248 124	0,076% 0,48% 0,0%

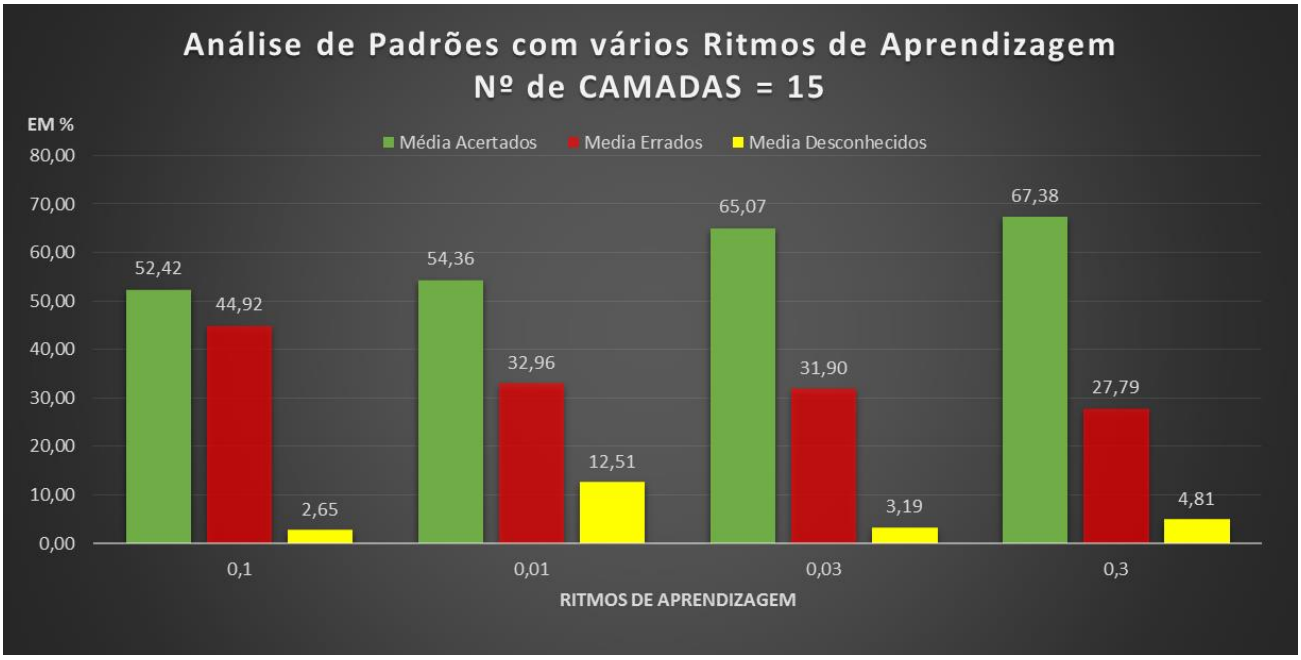
Anexo 2- Tabela de Testes 1



Anexo 3: Gráfico de comparação entre diferentes ritmos de aprendizagem



Anexo 4: Gráfico de comparação entre diferentes ritmos de aprendizagem



Anexo 5: Gráfico de comparação entre diferentes ritmos de aprendizagem

Como pudemos verificar nos testes ilustrados na tabela e nos gráficos auxiliares o agente aprende grande parte dos casos de treino. Em média acerta mais que 60% dos casos, respondendo assim em maior parte das situações de jogo como o esperado. Verificámos empiricamente que os casos que ele realmente erra com mais intensidade são os casos em que o pacman entra em ciclos, quando não existe comida nem fantasma por onde se guiar, e nesse caso ele sente-se "perdido" e começa a errar casos de treino. A dimensão da percentagem depende também do número de movimentos do pacman, isto é, quanto maior número de movimentos maior probabilidade de errar, porque cada vez vai existindo menos comida para ele seguir, havendo mais probabilidade de errar.

Com 10 neurónios existe um maior equilibrio de padrões acertados, com 7 existe igualmente bons resultados, acertando parte significativa dos testes. Com 15, erra com maior frequência, nos testes apresentados.

Destes testes retiramos que a melhor opção, embora não houvesse uma diferença significativa, seria com 10 neurónios, pelo que mantem o melhor número de casos acertados em todos os testes e por computacionalmente ser o mais equilibrado. Com 15 neurónios na camada escondida o treino ficaria mais complexo na propagação e retropropagação, e não teria grande vantagem no desempenho do agente, como dito anteriormente.

PROJECT PERFORMANCE

MAP:capsuleClassic

	Input			Output						
Caso nº	Tipo de Agente	Neurónios da Camada Escondida	Ritmo de Aprendizagem	Erros	Média Erro	Padrões Acertados	Padrões Errados	Padrões não Conhecidos	Número de Movimentos	Movimentos Ilegais
1	Humano	7	0.1	0.1982 0.1917 0.1685	0,1861	55.55% 90.62% 68.18%	0.0% 0.0% 9.09%	44.44% 9.37% 22.72%	18 32 22	11.11% 3.12% 4.81%
2	Humano	7	0.01	0.1812 0.1856 0.1793	0,182	63.63% 25.49% 59.25%	36.36% 1.96% 9.25%	0.0% 72.54% 31.48%	11 102 54	0.0% 3.92% 1.85%
3	Humano	7	0.03	0.1771 0.1875 0.1783	0,1809	65.51% 68.62% 65.0%	13.79% 5.88% 25.0%	20.68% 25.49% 10.0%	29 51 20	0.0% 1.96% 5.0%
4	Humano	7	0.3	0.1982 0.1999 0.2162	0,2047	42.85% 59.09% 65.38%	50.0% 22.72% 15.38%	7.14% 18.18% 19.23%	14 22 26	7.14% 4.54% 0.0%
5	Humano	10	0.1	0.2935 0,1872 0,1969	0,2258	71,42% 77,78% 42,78%	0,0% 11,12% 14,45%	28,57% 11,12% 45,78%	14 9 180	7,14% 0,0% 2,78%
6	Humano	10	0,01	0,1859 0,1844 0,1835	0,1846	77,27% 85,71% 58,34%	4,54% 0,0% 8,34%	18,18% 14,28% 33,34%	66 14 12	0,0% 7,14% 8,33
7	Humano	10	0,03	0,1851 0,1837 0,1848	0,1845	53,03% 61,53% 74,31%	4,54% 7,69% 10,34%	42,42% 30,76% 10,34%	66 26 29	3,03% 3,84% 0,0%
8	Humano	10	0,30	0,1958 0,1943 0,2192	0,2031	51,09% 44,44% 41,11%	22,72% 22,22% 27,94%	18,18% 33,33% 30,88%	22 18 68	0,0454% 0,0% 4,41%
9	Humano	15	0,10	0,1933 0,1938 0,1927	0,1933	57,78% 57,14% 52,38%	31,22% 2,85% 4,76%	11,22% 2,85% 42,85%	45 70 21	0,0% 0,0% 4,76%
10	Humano	15	0,01	0,1782 0,1879 0,1889	0,185	85,71% 61,40% 47,52%	0,0% 21,05% 10,89%	14,28% 17,54% 41,58%	14 57 101	7,14% 0,0% 1,48%
11	Humano	15	0,03	0,1875 0,1895 0,1858	0,1876	77,78% 59,52% 67,24%	14,81% 33,34% 5,17%	7,40% 7,14% 27,5%	27 42 58	0,0% 0,0% 0,0%
12	Humano	15	0,30	0,1743 0,1832 0,1774	0,1783	36,36% 55,0% 58,34%	12,72% 12,5% 5,56%	50,90% 32,5% 36,1%	55 40 72	0,0% 5,0% 2,78%

Anexo 6- Tabela de Testes 2

PROJECT PERFORMANCE

MAP:MediumScaryMaze

	Input			Output						
Caso nº	Tipo de Agente	Neurónios da Camada Escondida	Ritmo de Aprendizagem	Erros	Média Erro	Padrões Acertados	Padrões Errados	Padrões não Conhecidos	Número de Movimentos	Movimentos Ilegais
1	Humano	7	0.1	0,1980 0,1813 0,1872	0,1888	76,67% **39,17%** 42,20%	21,67% 53,18% 50,78%	1,67% 7,64% 7,01%	60 314 1654	0,0% 1,27% 0,24%
2	Humano	7	0,01	0,1829 0,1910 0,1785	**0,1841**	44,4% 66,53% 60,87%	33,3% 30.90% 33,50%	22,2% 2,55% 5,61%	9 1291 570	11% 0,07% 0,0%
3	Humano	7	0,03	0,1889 0,2008 0,1943	0,1946	76,47% 54,77% **86,56%**	14,70% 13,43% 5,97%	8,82% 31,78% 7,46%	102 1771 134	1,96% 0,45% 2,98%
4	Humano	7	0,30	0,2045 0,2043 0,2108	**0,2065**	54,20% 61,11% 56,34%	34,49% 37,60% 37,16%	6,30% 1,28% 6,48%	476 234 339	0,63% 0,42% 0,58%
5	Humano	10	0.1	0.1946 0.1932 0.1897	0,1925	68.60% 60.05% 80.54%	23.25% 29.80% 16.94%	8,13% 10.13% 2.51%	344 681 1033	0.0% 1.61% 0.09%
6	Humano	10	0.01	0.1823 0.1796 0.1919	**0,1846**	57.58% 50.16% 52.72%	22.17% 28.57% 36.36%	20.23% 21.26% 10.90%	257 301 55	0.0% 0.0% 1.81%
7	Humano	10	0.03	0.1840 0.1905 0.1836	0,186	46.29% **15.0%** 36.1%	40.74% 77.5% 58.33%	12.96% 7.50% 5.55%	54 40 36	0.0% 0.0% 0.0%
8	Humano	10	0.3	0.1983 0.1866 0.1977	**0,1942**	**82.92%** 82.0% 59.10%	12.19% 16.0% 35.46%	4.87% 2.0% 5.43%	41 50 313	0.0% 0.0% 0.958%
9	Humano	15	0,10	0,1961 0,1855 0,1960	0,1925	69,51% 61,20% 87,25%	26,05% 20,21% 4,40%	4,42% 18,57% 7,84%	1017 183 102	0,0039% 0,0% 0,0%
10	Humano	15	0,001	0,1811 0,1854 0,1740	0.1801	61,22% 88,37% 83,34%	26,20% 2,32% 9,64%	12,56% 9,30% 7,01%	374 43 114	0,0% 2,32% 0,0%
11	Humano	15	0.03	0.1760 0.1782 0.1840	**0.1794**	57.14% 47.51% 57.87%	42.41% 30.49% 41.21%	0.44% 21.98% 0.90%	224 141 330	0.0% 0.0% 0.30%
12	Humano	15	0.3	0.2034 0.2138 0.1951	**0,2041**	85.41% **92.10%** 57.57%	2.08% 5.26% 31.14%	12.5% 2.63% 11.27%	96 38 594	3.12% 0.0% 0.50%

Anexo 7- Tabela de Testes 3

PROJECT PERFORMANCE

MAP:Default

	Input			Output						
Caso nº	Tipo de Agente	Neurónios da Camada Escondida	Ritmo de Aprendizagem	Erros	Média Erro	Padrões Acertados	Padrões Errados	Padrões não Conhecidos	Número de Movimentos	Movimentos Ilegais
1	Reactivo	10	0,10	0,12712 0,13024 0,11762	0,1249	89,09% 88,57% 85,41%	3,63% 2,85% 10,41%	7,27% 8,57% 4,16%	55 35 48	0,0% 0,0% 0,0%
2	Reactivo	10	0,01	0,11258 0,12412 0,11664	0,1178	74,52% 71,42% 63,03%	16,98% 28,57% 6,81%	8,49% 0,0% 29,54%	106 77 44	0,0% 0,0% 0,0%
3	Reactivo	10	0,03	0,0955 0,10230 0,13048	0,1094	64,28% 65,15% 78,80%	19,64% 34,84% 18,30%	16,07% 0,0% 2,81%	56 132 71	0,0% 0,0% 0,0%
4	Reactivo	10	0,30	0,10467 0,12512 0,10661	0,1121	84,21% 64,06% 73,68%	15,78% 25,39% 25,0%	0,0% 10,54% 1,31%	19 256 152	0,0% 0,0% 0,0%

Anexo 8- Tabela de Testes 3

PROJECT PERFORMANCE

MAP:MediumScaryMaze

	Input			Output						
Caso nº	Tipo de Agente	Neurónios da Camada Escondida	Ritmo de Aprendizagem	Erros	Média Erro	Padrões Acertados	Padrões Errados	Padrões não Conhecidos	Número de Movimentos	Movimentos Ilegais
1	Reactivo	10	0,10	0,0942 0,11659 0,12801	0,1129	50% 97,5% 85,71%	45,80% 0,0% 10,71%	1,19% 2,5% 3,57%	168 40 28	0,0% 0,0% 0,0%
2	Reactivo	10	0,01	0,11569 0,12744 0,12351	0,1221	77,78% 18,18% 34,78%	0,0% 0,0% 52,17%	22,23% 81,18% 13,04%	9 22 25	0,0% 0,0% 0,0%
3	Reactivo	10	0,03	0,10720 0,11217 0,10322	0,1075	90,90% 85,71% 60,0%	0,0% 0,0% 30,0%	9,09% 14,28% 10,0%	11 21 20	0,0% 0,0% 0,0%
4	Reactivo	10	0,30	0,11043 0,11864 0,11095	0,1134	33,34% 66,67% 88,88%	66,67% 27,28% 0,0%	0,0% 5,56% 11,11%	60 18 9	0,0% 0,0% 0,0%

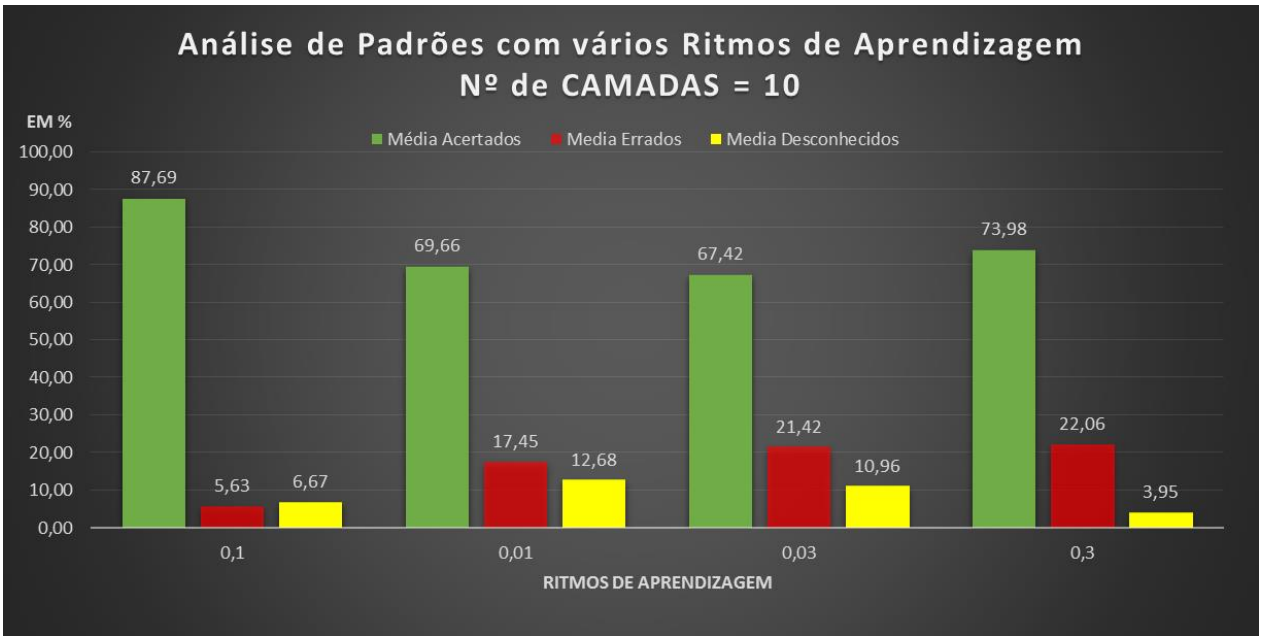
Anexo 9- Tabela de Testes 4

PROJECT PERFORMANCE

MAP:CapsuleClassic

	Input			Output						
Caso nº	Tipo de Agente	Neurónios da Camada Escondida	Ritmo de Aprendizagem	Erros	Média Erro	Padrões Acertados	Padrões Errados	Padrões não Conhecidos	Número de Movimentos	Movimentos Ilegais
1	Reactivo	10	0,10	0,10190 0,09971 0,09829	0,0999	50% 81,57% 22,23%	50% 10,52% 11,12%	0,0% 7,84% 66,67%	6 38 36	0,0% 0,0% 2,78%
2	Reactivo	10	0,01	0,10517 0,10532 0,11314	0,1078	75% 50% 50%	16,67% 50% 40,74%	8,34% 0,0% 9,25%	12 6 54	0,0% 0,0% 0,0%
3	Reactivo	10	0,03	0,10015 0,09732 0,9443	0,3806	75% 50% 55,81%	16,67% 50% 41,86%	8,34% 0,0% 2,32%	12 6 43	0,0% 0,0% 0,0%
4	Reactivo	10	0,30	0,13654 0,12534 0,14974	0,1372	70,0% 54,16% 58,34%	30,0% 45,83% 41,67%	0,0% 0,0% 0,0%	10 48 24	0,0% 0,0% 0,0%

Anexo 10- Tabela de Testes 5



Anexo 6- Gráfico de comparação entre diferentes ritmos de aprendizagem no mapa Default

Analisando o gráfico e a tabela, verificamos que tem um bom desempenho em relação aos padrões acertados. Mas analisando o desempenho empiricamente do Pac-Man verificamos que em determinados casos não têm o resultado esperado. Apesar de em média acertar em mais padrões que a rede treinada pelo agente humano, apresenta um pior rendimento. Explicável pela qualidade dos treinos serem menos elevados que os desenvolvidos pelo jogador humano.

Os erros apresentados pelos testes feitos segundo o agente reactivo são mais baixos que os treinados segundo o agente humano, já que o número de casos de treino depois de filtados é maior que o apresentado pelo agente humano, na mesma situação.

Apresentando um melhor rendimento com o ritmo de aprendizagem 0.1.

4. Conclusões

Ao analisar os testes efectuados verificamos que o agente tende a ter uma taxa de sucesso mais elevada quando se usa os padrões de treino gerados por jogador humano, mesmo no agente reactivo ele segue com rigor o comportamento dos treinos, verificadando elevada taxa de padrões acertados.

Treinando com os padrões do agente reactivo tem, em geral, um comportamento satisfatorio, mas em alguns casos não tem o comportamento esperado, já que os sensores de input não recebem nenhum valor distintivo com o qual se possa guiar, mesmo se analisando os testes realizados contem erros mais baixos que os testes com padrões de jogo humano já que depois de filtrado os treinos, os padrões do agente reactivo contém maior número de casos, motivo pelo qual o erro quadrático é mais baixo. Mas como dito anteriormente, o desempenho não é o esperado, falhando em casos como: com comida à frente e fantasma à frente não fugindo do fantasma em determinadas situações, circuntância que podia ser resolvida com melhores padrões de treino do agente reactivo.

Com o treino realizado pelo jogador humano o pacman têm um melhor comportamento, sendo que os padrões de treino têm maior qualidade. O agente aprendiz tem um nível de desempenho satisfatório, reproduzindo assim o comportameno esperado, tendo limitações, quando não tem comida e não têm fantasma na maior parte dos casos o pacman entra em ciclo, sendo um resultado esperado, com esta arquitectura da rede, como não há movimentos aleatórios, nem memória, o facto da rede depois de treinada ser totalmente determinística e sem memoria leva a este comportamento.Por mais treinos que façamos o agente está condenado entrar em ciclo, basta encontrar-se duas vezes na mesma situação que irá fazer a mesma acção, logo há ciclos. Não querendo dizer que não esteja a aprender, como prova em casos com fastasma assustado, que tenta comê-lo, perseguindo também as capsulas, produzindo assim um comportamento esperado de um Pac-Man num jogo real, apesar de algumas limitações descritas em cima.

Poderiamos ultrapassar as limitações de ciclos alterando a arquitectura da rede, utilizando uma maior camada de input de forma a identificar o caminho de origem e o destino esperado consoante o objectivo estabelicido em cada instante.

Os valores do erro obtidos na fase de validação foram ligeiramente inferiores aos obtidos na fase de treino, apesar do agente aprendiz reproduzir um comportamento bastante semelhante aos agentes de treinos.