Universidade de Coimbra
Faculdade de Ciências e Tecnologia
Departamento de Engenharia Informática

Introdução à Inteligência Artificial
2011/2012 - 2º Semestre

Trabalho Prático 2: *Pac-Man: Back to School*

Data Limite de Entrega: 1/06/2012 – 17 Horas

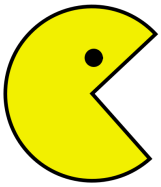
Nota 1: É obrigatório indicar no relatório **por cada aluno**:

- tempo de estudo
- tempo de implementação
- contributo para o trabalho

A omissão destes elementos implica a não consideração do trabalho prático para efeitos de avaliação.

Nota 2: A fraude denota uma grave falta de ética e constitui um comportamento inadmissível num estudante do ensino superior e futuro profissional licenciado. Qualquer tentativa de fraude levará à anulação da componente prática tanto do facilitador como do prevaricador, independentemente de acções disciplinares adicionais a que haja lugar nos termos da legislação em vigor. Caso haja recurso a material não original as **fontes** devem estar explicitamente indicadas.

1 Introdução



Os Pac-men viram-se gregos mas conseguiram sobreviver à difícil crise financeira. Para ultrapassar as dificuldades económicas os Pac-men optaram por uma política de redução de custos. Os grandes investimentos – aeroportos, comboios de alta velocidade, auto-estradas, etc. – foram cancelados e os salários congelados. Infelizmente, para garantir a segurança das ilhas "penhasco-vazio", foram forçados a optar pela compra de submarinos em detrimento do financiamento das instituições de ensino. Para sobreviverem as universidades aumentaram o valor das propinas mas tal apenas agravou a situação. Os alunos, incapazes de suportar os custos, viram-se obrigados a abandonar os estudos e as universidades, desertas, acabaram por abrir falência.

A crise foi ultrapassada mas o preço pode ter sido demasiado elevado. Os fantasmas estão de volta e os Pac-men não têm os conhecimentos teórico-práticos para os evitar. Felizmente ainda há alguns docentes dispostos a voltar ao activo.

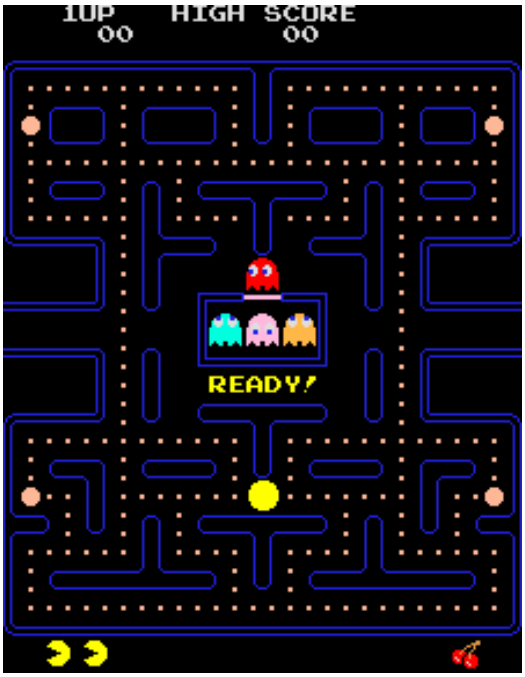


Figura 1: Os fantasmas regressaram.

2 Objectivos Genéricos

Em termos genéricos pretedem-se alcançar os seguintes objectivos:

1. Compreender o funcionamento de uma rede neuronal *feedforward*
2. Compreender o funcionamento do algoritmo de retropropagação
3. Aprender a desenhar uma rede neuronal
4. Apreender a treinar uma rede neuronal
5. Compreender as forças, fraquezas e limitações desta abordagem

Estes objectivos genéricos serão alcançados através do trabalho em grupo e da experimentação, promovendo-se, assim, estas capacidades.

3 Enunciado

O ambiente é semelhante ao do primeiro trabalho prático. Os objectivos do Pac-man também. A diferença é a seguinte: no primeiro trabalho prático desenvolveu um agente reactivo, agora vai desenvolver um agente que aprende a jogar o jogo. Para o efeito seguirá uma abordagem coneccionista, treinando uma rede neuronal que controlará o Pac-man.

4 Desenvolvimento

O trabalho prático divide-se em duas tarefas que se diferenciam pela liberdade que é dada.

4.1 Tarefa Base

Deve seguir os seguintes passos no desenvolvimento deste trabalho prático:

1. Recolha de padrões de treino – Para treinar a sua rede neuronal necessita de um conjunto de padrões de treino. Estes padrões podem ser gerados através de dois métodos:
   (a) Controlando o Pac-man através do teclado
   (b) Usando um agente reactivo para jogar Pac-man, gerando assim os padrões de treino de forma automática

2. Design da rede neuronal – O número de entradas e saídas da rede é fixo. No entanto, existem várias opções a tomar (funções de activação, número de neurónios na camada escondida, etc.)

3. Treino da rede – A rede deve ser treinada através do algoritmo de retropropagação. Deve garantir que a implementação do algoritmo é correcta e parametrizar o algoritmo de forma adequada. Tenha em conta questões tais como:

   (a) Ritmo de aprendizagem

   (b) Critério de paragem

   (c) Inicialização dos pesos

   Estes parâmetros devem ser definidos empiricamente. O relatório deve descrever o processo.

4. Análise e comparação – Deve testar o desempenho da rede neuronal por forma a avaliar se o comportamento desta corresponde ao esperado. Tenha em conta questões como:

   (a) Quando treina a rede com padrões gerados por um agente reactivo consegue reproduzir o comportamento do agente?

   (b) Quando treina a rede com padrões gerados por um humano consegue reproduzir o comportamento?

   (c) O desempenho do agente aprendiz é satisfatório? Quais as forças e fraquezas?

   (d) Como poderia ultrapassar as limitações encontradas?

4.2 Tarefa Avançada

Esta meta diferencia-se da anterior pelo facto das únicas restrições existentes serem as seguintes:

1. O Pacman deve ser controlado por uma rede neuronal;

2. O Pacman apenas pode utilizar informação acessível a um jogador humano;

Esta meta terá uma cotação de 20% sendo privilegiadas soluções elegantes que promovam um comportamento eficiente e eficaz. Os agentes desenvolvidos serão testados contra fantasmas semelhantes aos do jogo de arcade original.

5 O Simulador

Será utilizada uma variante do simulador de Pac-Man maioritariamente desenvolvido por John DeNero na universidade de Berkley (http://inst.eecs.berkeley.edu/~cs188/sp09/pacman.html). Este simulador é constituído por uma série de bibliotecas Python que implementam todas as funcionalidades necessárias, permitindo que se centre no desenvolvimento do agente reactivo. Deve fazer **download da versão disponível no WOC**, visto que esta se encontra preparada para este trabalho prático.

5.1 Camada de Entrada e Saída da Rede Neuronal

A rede neuronal recebe 20 entradas que codificam o valor dos sensores do Pac-man. O agente possui 4 sensores para paredes e para comida:

	s_0	
s_3	P	s_2
	s_1	

Os valores destes sensores determinam as primeiras oito entradas da rede neuronal. Cada sensor corresponde a duas entradas. A codificação é feita da seguinte forma:

$sensor_i$	$entrada_(i*2)$	$entrada_(i*2+1)$
Parede	0	0
Vazio	0	1
Comida	1	0
Cápsula	1	1

O Pac-man possui 12 sensores para fantasmas:

		s_2		
	s_9	s_1	s_{10}	
s_8	s_7	P	s_5	s_6
	s_{12}	s_3	s_{11}	
		s_4		

Os valores destes sensores determinam as restantes doze entradas sendo a codificação efectuada da seguinte forma:

A rede neuronal tem 5 neurónios na camada de saída. O valor destes neurónios determina a acção a executar. A codificação é a seguinte:

sensor	*entrada*
Assustado	-1
Sem fantasma	0
Fantasma	1

sensor	o_0	o_1	o_2	o_3	o_4
Norte	1	0	0	0	0
Sul	0	1	0	0	0
Este	0	0	1	0	0
Oeste	0	0	0	1	0
Stop	0	0	0	0	1

NOTA: Durante a simulação, para determinar a acção do agente, deve utilizar uma abordagem "winner takes all". Ou seja, a acção a executar corresponde ao neurónio com valor de saída mais elevado.

5.2 Correr o Simulador

Para criar padrões de teste deve executar o Learning Keyboard Agent (opção -p Learning Keyboard Agent). Enquanto é executado o código efectua o print das codificaçõese e guarda os padrões de treino em ficheiro No ficheiro learningAgents.py encontra o Neural Network Agent (neste momento aleatório) que deverá implementar.

Seguem-se exemplos da utilização dos métodos disponibilizados.

loadFile

```
from util import loadFile

file = loadFile('training_1.iia')
print "file:", file
for case in file:
print "case:", case


Sample output:

file: [[[0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 1, 0]], (...)
case: [[0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 1, 0]]
(...)
```

loadFilesFromList

```
from util import loadFilesFromList

files = loadFilesFromList(['training_1.iia', 'trainindgasdgadsha', 'training_3.iia', 'agasdgsdg'])
```

```
print files

for file in files:
print "file:", file
        for case in file:
         print "case:", case


Sample output:


file: [[[0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 1, 0]], (...)
case: [[0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 1, 0]]
(..)

file: [[[0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0]], (...)
case: [[0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0]]
(...)}
```

loadFilesUntil

```
from util import loadFilesUntil

files = loadFilesUntil(50)
print files
for file in files:
print "file:", file
        for case in file:
         print "case:", case


Sample output:


file: [[[0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 1, 0]], (...)
case: [[0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 1, 0]]
(..)

file: [[[0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0]], (...)
case: [[0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 1, 0, 0]]
(...)
```

Dispõe ainda dos métodos `getActionRepresentation(action)` e `getStateRepresentation(state)` que devolvem a codificação respectivamente de uma acção e um estado.

6 Plano de Trabalho

O plano de trabalhos aqui proposto visa promover a qualidade dos trabalhos e as probabilidades de sucesso. No entanto, a entrega das metas intermédias é inteiramente opcional não afectando directamente a avaliação. As datas aqui definidas não serão sujeitas a alterações.

6.1 Meta 1 – Recolha e Treino

Corresponde aos passos 1 a 3 da secção 4.1.

Material a entregar:

Um breve documento (max. 3 páginas), em formato pdf, com a seguinte informação:

- Identificação dos elementos do grupo (Nomes, Números de Estudante, e-mails, Turma(s) Prática(s))
- Esforço: tempo de estudo, tempo de implementação, contributo para o trabalho (por aluno)
- Descrição da metodologia seguida.
- Os ficheiros alterados, devidamente comentados;

Modo de Entrega:

Através do inforestudante.

Data Limite: 4 de Maio de 2012

6.2 Meta 2 – Análise e comparação

Corresponde ao passo 4 da secção 4.1.

Material a entregar:

Um breve documento (max. 5 páginas), em formato pdf, com a seguinte informação:

- Identificação dos elementos do grupo (Nomes, Números de Estudante, e-mails, Turma(s) Prática(s))
- Esforço: tempo de estudo, tempo de implementação, contributo para o trabalho (por aluno)
- Descrição dos resultados experimentais obtidos e análise dos resultados experimentais.
- Os ficheiros alterados, devidamente comentados;

Modo de Entrega:

Através do inforestudante.

Data Limite: 18 de Maio de 2012

6.3 Relatório e Entrega Final

Num trabalho desta natureza o relatório assume um papel importante. Deve descrever de forma detalhada e clara todo o trabalho realizado no âmbito deste projecto, dando destaque aos problemas e soluções encontradas.

A experimentação é uma parte essencial do desenvolvimento de aplicações de IA. Assim, deve descrever detalhadamente as experiências realizadas, analisar os resultados, extrair conclusões, e efectuar alterações (caso se justifique) em função dos resultados por forma a melhorar o desempenho do agente.

O relatório deve conter informação relevante tanto da perspectiva do utilizador como do programador. Não deve ultrapassar as 20 páginas, formato A4. Todas as opções tomadas deverão ser devidamente justificadas e explicadas.

A primeira secção do relatório é obrigatoriamente composta pela seguinte informação, para cada elemento do grupo:

- nome completo;
- e-mail;
- o tempo de estudo e de implementação gasto na elaboração do trabalho;
- as tarefas desempenhadas pelo elemento em causa.

Deverá ainda mencionar todos os detalhes relacionados com a divisão do trabalho pelos diferentes elementos do grupo que considere relevantes. O trabalho colaborativo é um aspecto fundamental da aprendizagem.

6.4 Modo de Entrega

Deve através do inforestudante um ficheiro .zip contendo o seguinte material:

- Relatório em formato pdf (max. 20 páginas)
- Os ficheiros alterados, devidamente comentados;

No acto de entrega deve inscrever-se num dos intervalos de tempo disponíveis para a defesa do trabalho prático.

Data Limite: 01 Junho 2012

Bibliografia

- **Inteligência Artificial: Fundamentos e Aplicações**
  Ernesto Costa, Anabela Simões