Aprendizaje automático

Tutorial 1

Grupo 84

Sabrina Riesgo Reyes 100363834 100363834@alumnos.uc3m.es
Laura Yunta García 100363785 100363785@alumnos.uc3m.es

Índice

1. INTRODUCCIÓN

El desarrollo de la práctica busca poder desarrollar sistemas de control automáticos y explorar algunas capacidades del aprendizaje automático. Para ello se ha utilizado una versión modificada del Pac – Man, el cual debe intentar comerse a todos los fantasmas para poder terminar el juego, tratando de obtener la mejor puntuación posible. Los premios para Pac – Man son 200 puntos por cada fantasma comido y 100 por cada punto de comida. El objetivo de esta práctica es conseguir el mejor comportamiento para el Pac – Man en cada momento, con las dificultades de que los fantasmas se podrán mover y los mapas se pueden modificar.

Finalmente, el documento y con ello la práctica se dividirá en cuatro partes diferenciadas:

- **Ejercicios planteados:** Donde se responderán a las diversas preguntas planteadas en el enunciado del ejercicio.
- **Función implementada para el fichero de información:** En este apartado se explicará el desarrollo de la función pedida por el enunciado de la práctica, sus distintas partes y proceso de ejecución.
- **Comportamiento del agente:** Donde se especificará el algoritmo planteado para el movimiento eficiente del Pac – Man sin el uso de técnicas de aprendizaje automático.
- **Conclusiones y problemas encontrados:** En este apartado se comentarán las conclusiones sacadas con la realización del ejercicio, además los diversos problemas que se han conseguido solventar durante el trascurso de la práctica.

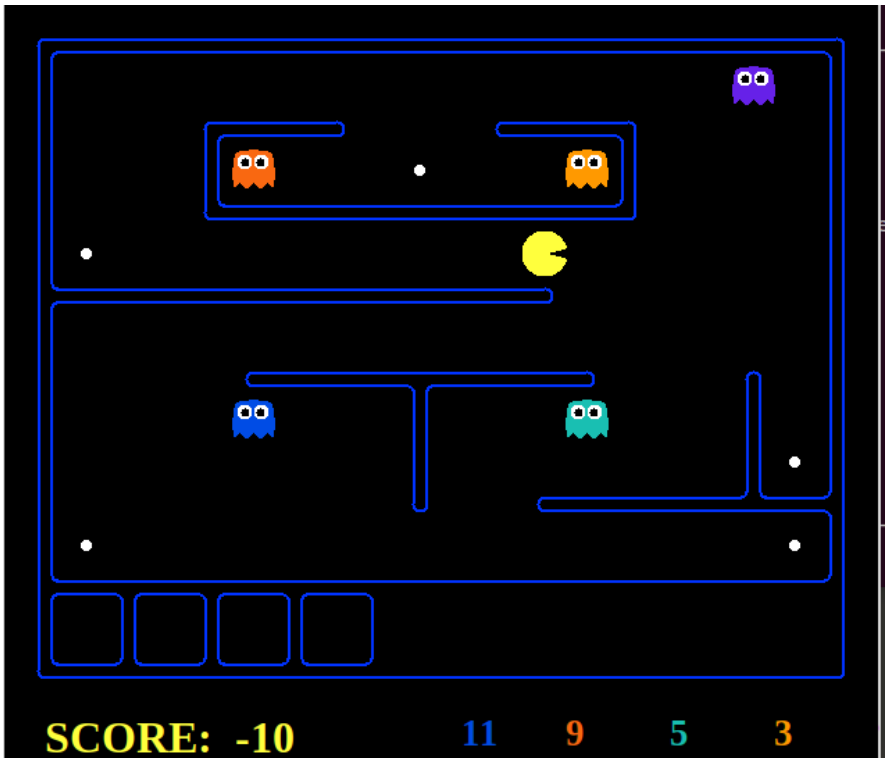
2. EJERCICIOS PLANTEADOS

1. **¿Qué información se muestra en la interfaz? ¿Y en la terminal? ¿Cuál es la posición que ocupa Pac-Man inicialmente?**
   Dada la interfaz del Pac – Man se muestra la representación del tablero con los muros, el agente (Pac – Man), los fantasmas y bolitas a recolectar correspondientes en su interior. Además en la parte inferior izquierda se representa la puntuación del jugador en cada instante (debido a que según van pasando los segundos la puntuación va disminuyendo linealmente) y aumenta según se vayan cazando los fantasmas y recogido las bolitas del mapa. Finalmente en la parte inferior derecha se encuentra la distancia de Manhattan desde la posición de Pac – Man en cada instante hasta la posición de cada uno de los fantasmas (cada número de distancia se identifica con colores, es decir, tendrán el mismo color que el fantasma al que se le está calculando la distancia de Manhattan). Cabe destacar que inicialmente cuando utilizas un método de ejecución básico en la terminal no aparece ningún tipo de información. Finalmente la posición que ocupa Pac – Man al ejecutar inicialmente el programa es: (12 , 10).

2. **Según tu opinión, ¿qué datos podrían ser útiles para decidir lo que tiene que hacer Pac – Man en cada momento?**
   Una información muy útil para determinar el movimiento de Pac – Man sería la posición del agente en cada instante y la distancia de Manhattan a cada uno de los elementos que debe recolectar antes de acabar la ejecución. Finalmente, también

sería bueno tener almacenado el número de elementos que le falta por recolectar y ser capaz de llegar a una condición de parada sin necesidad de cálculos internos.

3. **Revisa la carpeta layouts. ¿Cómo están definidos los laberintos de los ficheros? Diseña un laberinto nuevo, guárdalo y ejecútalo en el juego. Incluye una captura de pantalla del mismo en la memoria.**
   Los laberintos están definidos casilla a casilla por símbolos representativos en una matriz de forma que los símbolos % representan cada uno de los muros del mapa, las G los fantasmas que componen el juego, los o las bolitas que permiten a Pac – Man comerse a los fantasmas, los · representan cada una de las bolitas que se puede comer Pac – Man y P representa a Pac – Man en cada momento.
   El nuevo mapa creado para la ejecución es el siguiente:



4. **Ejecuta el agente BasicAgentAA tal y como se indica a continuación: Python busters.py –p BasicAgentAA. Describe qué información se muestra por pantalla sobre el teclado del juego en cada turno. De esta información, ¿cuál crees que podría ser más útil para decidir automáticamente la siguiente acción de Pac – Man?**
   La información que se muestra por pantalla al ejecutar con "–p BasicAgentAA" es la información por cada movimiento que realiza el agente. Es decir, para cada instante y movimiento realizado se especifica la siguiente información siguiendo este determinado orden:
   - Ancho y largo del mapa que se está ejecutando.
   - La posición de Pac - Man en ese estado.
   - La lista de movimientos permitidos que tiene Pac - Man para la posición en la que está.
   - Movimiento realizado en ese estado.

- Número de fantasmas que hay en el mapa.
- Lista booleana que hace referencia en primer lugar al agente (siempre será False) y a continuación a los fantasmas del mapa, tomando el valor True para los fantasmas que aún no se ha comido Pac-Man y False para los que ya han sido comidos.
- Lista de tuplas con las posiciones de cada uno de los fantasmas.
- Lista de movimientos que han realizado todos fantasmas en ese estado.
- Distancia de Pac – Man a los diferentes fantasmas.
- Número de bolitas que puede coger Pac – Man.
- Distancia a la bolita más cercana a Pac – Man.
- Apariencia del mapa.
- Puntuación de Pac – Man.

La información que puede ser más relevante para decidir la acción de Pac – Man son: la posición de Pac – Man en ese estado, la lista de las distancias del Pac – Man a cada uno de los fantasmas y las acciones legales de Pac – Man en la posición actual.

5. **Programa una función de nombre printLineData() dentro del agente BasicAgentAA del fichero busterAgents.py. Esta función debe devolver una cadena de caracteres con la información que se considere relevante del estado del Pac – Man. Dicha función será llamada desde el bucle principal del juego en Game.py para que se escriba en un fichero. Este paso es de gran importancia ya que servirá como primera versión de la fase de extracción de características y será imprescindible para las siguientes prácticas.**
   Por cada turno de juego, se debe guardar una línea con todos los datos concatenados del estado del juego que se calcula por omisión, separados por el carácter coma (,).
   Además, cada vez que se inicia una nueva partida o se abra el juego de nuevo, las nuevas líneas deben guardarse debajo de las antiguas. No se debe reiniciar el fichero de texto al empezar una nueva partida, por lo que el fichero de texto resultante tendrá tantas líneas como turnos se hayan jugado en todas las partidas.

6. **Programa un comportamiento "inteligente" para el Pac – Man. Para ello, en la clase BasicAgentAA, se pide modificar el método chooseAction() que hasta ahora presentaba un comportamiento aleatorio. Pac-Man debe perseguir y comerse a todos los fantasmas de la pantalla. Compara los resultados ejecutando el juego con los fantasmas estáticos y en movimiento aleatorio (-g RandomGhost). Haz varias ejecuciones y determina cuantos turnos de media suele tardar en finalizar.**

7. **El agente programado en el ejercicio anterior no utiliza ninguna técnica de aprendizaje automático. ¿Qué ventaja crees que puede tener el aprendizaje automático para controlar a Pac – Man?**
   Puesto que no se han utilizado técnicas de aprendizaje automático podemos ver que, en ocasiones, dependiendo de las características del mapa y los movimientos de los fantasmas, Pac - Man puede no realizar los movimientos más eficientes para la ejecución pudiendo realizar recorridos más largos para la solución del mapa. De esta

forma, podemos concluir que el uso de técnicas de aprendizaje automático para la solución de problemas de este tipo resultaría útil, ya que haciendo uso de la información recopilada mediante la ejecución del programa un número determinado de veces se podría mejorar la eficiencia del juego, haciendo que el personaje aprenda aquellos lugares por los que no puede pasar (muros) y aquellos caminos que le llevan a su objetivo (cazar todos los fantasmas) de la mejor manera posible.

3. FUNCIÓN IMPLEMENTADA PARA FICHERO DE INFORMACIÓN

Para la implementación de la función del fichero de información, se han tenido que hacer modificaciones en el código de los ficheros game.py y busters.py de la siguiente forma:

- **game.py:** Desde este fichero se llama a la función que escribe el texto en el fichero correspondiente. Para empezar, se utiliza la función "open", que recibe el nombre de un fichero "info.txt" y el parámetro "a+". El objetivo de dicha función es abrir el fichero "info.txt" para lectura y escritura y en caso de que este no exista, se creará; además, el puntero se posicionará al final del fichero, haciendo que la nueva información se escriba a continuación de la ya existente. A continuación, a través del uso de la función write, que permite escribir en el fichero anteriormente abierto, se llama a la función "printLineData" que se explicará más adelante y que devuelve la información que debemos escribir en el fichero. Finalmente, se cierra el fichero mediante el empleo de la función "close()".

- **bustersAgents.py:** D En el código de este fichero se desarrolla la función printlineData, la cual devuelve los datos relevantes que utiliza la función "write" para que posteriormente sean escritos en el fichero. En su interior simplemente se almacena en variables tipo string la información que se quiere escribir. La información que se ha considerado relevante para el estado es: la posición de Pac - Man, los posibles movimientos legales para Pac - Man y la distancia de Manhattan entre los fantasmas y el mismo.

4. COMPORTAMIENTO DEL AGENTE

El objetivo de la modificación de la función chooseAction() es eliminar el comportamiento totalmente aleatorio que presentaba el agente según el código inicial. La primera parte del código se podría definir como la preparación de las variables para el correcto funcionamiento del algoritmo implementado posteriormente. Esta parte, además de almacenar en variables la información relevante necesaria (posición actual de Pac-Man, acciones legales, posición actual de los fantasmas y distancia de Manhattan entre cada uno de estos y el personaje principal), ordena la lista de distancias para que el agente vaya en busca del fantasma que más cerca está de su posición actual. Para evitar que las posiciones de los fantasmas que ya han sido cazados influyan en la nueva decisión del personaje se almacenará el valor 0 en la distancia de estos, y Pac-Man sólo se fijará en aquellos valores positivos. A continuación, a través de una serie de bucles if anidados se representan las acciones del personaje, dependiendo de la posición del fantasma objetivo (el más cercano). La idea general es:

- Si el personaje se encuentra a la izquierda y debajo del fantasma, las acciones principales que intentará realizar serán EAST o NORTH, si ninguna de estas es posible, se elegirá de forma aleatoria.

- Si el personaje se encuentra a la izquierda y encima del fantasma, las acciones principales que intentará realizar serán EAST o SOUTH, si ninguna de estas es posible, se elegirá de forma aleatoria.
- Si el personaje se encuentra a la izquierda del fantasma pero en la misma fila, la acción principal que intentará realizar será EAST, si no es posible, se elegirá de forma aleatoria.
- Si el personaje se encuentra a la derecha y debajo del fantasma, las acciones principales que intentará realizar serán WEST o NORTH, si ninguna de estas es posible, se elegirá de forma aleatoria.
- Si el personaje se encuentra a la derecha y encima del fantasma, las acciones principales que intentará realizar serán WEST o SOUTH , si ninguna de estas es posible, se elegirá de forma aleatoria.
- Si el personaje se encuentra a la derecha del fantasma pero en la misma fila, la acción principal que intentará realizar será WEST, si no es posible, se elegirá de forma aleatoria.
- Si el personaje se encuentra debajo del fantasma pero en la misma columna, la acción principal que intentará realizar será NORTH, si no es posible, se elegirá de forma aleatoria.
- Si el personaje se encuentra encima del fantasma pero en la misma columna, la acción principal que intentará realizar será SOUTH, si no es posible, se elegirá de forma aleatoria.

La última parte del código consiste en un bucle que cada 30 movimientos generará un nuevo movimiento de forma aleatoria con el objetivo de evitar bucles ya que al no utilizar técnicas de aprendizaje automático resulta imposible hacer que el personaje principal realice los movimientos adecuados. Finalmente, la función devuelve el próximo movimiento a realizar.

5. CONCLUSIONES Y DIFICULTADES ENCONTRADAS

Con el desarrollo de la práctica hemos conseguido mejorar nuestro conocimiento y técnica respecto al uso del lenguaje Python que al no haberlo usado mucho con anterioridad, inicialmente nos creaba algo de dificultad. Finalmente, consideramos que uno de los principales problemas de la práctica es que no es sencillo programar un agente muy eficiente sin utilizar técnicas de aprendizaje automático, ya que al modificar parámetros (los mapas a ejecutar o los movimientos de los fantasmas) pueden hacer recorridos poco eficientes e incluso pudiendo llegar a entrar en bucles. Algo que no ocurriría si se usasen técnicas de aprendizaje automático.

A continuación, se adjunta una tabla comparativa en la que se recopilan datos relevantes recogidos durante las distintas ejecuciones del mapa.

Tipo de mapa / Movimiento de los fantasmas	Finaliza	Ticks	Tiempo
oneHunt/fijo	Sí	25	4,88 s
oneHunt/aleatorio	Sí	51	8,31 s
nuevoMapa/fijo	No		
nuevoMapa/aleatorio	Sí	460	1 m 0,964 s

Gracias a esta tabla se puede visualizar que el algoritmo desarrollado por el Pac – Man programado hace que las ejecuciones sean más eficientes que si se realiza con la ejecución

aleatoria. Aunque en ocasiones y dependiendo del mapa y del número de fantasmas y sus posiciones puede llevar a bucles que se han intentado solucionar con la aleatoriedad adicional tras el número de movimientos.

En el caso del nuevoMapa con los fantasmas fijos debido a que hay paredes y se generan demasiados movimientos aleatorios, entra en un bucle por lo que no finaliza el programa. En el caso de los fantasmas en movimiento, finaliza en el programa en algunos casos ya que el movimiento de los fantasmas muchas veces favorece el encuentro de estos con el agente y le permite salir de los bucles que puedan surgir.