Departamento de Informática
Universidad Carlos III de Madrid

Grado en Ingeniería Informática

Aprendizaje Automático

Curso 2015-2016

Práctica 3

Aprendizaje por refuerzo

13 de abril de 2016

1. Introducción

En esta práctica se aplicará el algoritmo *Q-learning* en el dominio del juego Pac-Man para construir un agente que funcione de forma automática. Para ello se utilizarán los datos que proporciona el simulador del Pac-man.

El algoritmo *Q-learning* necesita tener un conjunto de estados finitos para construir la tabla Q. Una vez obtenida la tabla, el funcionamiento del agente consistirá en determinar a qué estado pertenece y elegir la mejor acción según la tabla Q. El objetivo de la práctica será desarrollar un agente que maximice el número de fantasmas comidos por el Pac-Man.

2. Tareas a realizar en la práctica

Las tareas a realizar en esta práctica se dividen en distintas fases (recogida, aprendizaje, construcción y evaluación) que se detallan a continuación:

Fase 1. Recogida de información y generación del espacio de estados

1. Decidir que estructura elegir para almacenar los estados y las acciones del juego.
2. Decidir que información define un estado del juego.
3. Almacenar un conjunto suficientemente grande de tuplas de experiencia del tipo (*estado_tick*, *acción*, *estado_tick_n*, *refuerzo*).
4. *estado_tick* contiene información del estado, (por ejemplo: si una casilla posee un fantasma, etc.)
5. *acción* se trata de la acción que ejecuta el agente para transitar del *estado_tick* al estado *estado_tick_n*.
6. *estado_tick_n* contiene la misma información que *estado_tick_n* pero en el *tick* de tiempo $+n$.
7. *refuerzo* cualquier función que potencie comerse los fantasmas.
8. Limitar el número de pasos de agente por partida a una constante.

Fase 2. Aprendizaje: generación de la tabla Q

1. Construir una tabla Q que tendrá tantas filas como número de estados se haya decidido emplear, y tantas columnas como acciones.
2. Programar el agente *Q-learning*:
3. Generar una tabla Q realizando actualizaciones a partir de las tuplas generadas en el punto anterior utilizando el algoritmo *Q-Learning*.
   - Cargar la tabla Q inicial.
   - Seleccionar la acción del agente mediante el método ϵ-*greedy*, lo que significa que ejecuta la acción que tenga un valor $maxQ(s, a)$ con probabilidad ϵ y una acción aleatoria con probabilidad $1 - \epsilon$.
   - Actualizar la tabla Q en cada movimiento.
   - Escribir en fichero la tabla al final del proceso de aprendizaje.

NOTA:
Cambiar los parámetros de aprendizaje para ver cómo afectan en las acciones tomadas por el agente.

Fase 3. Construcción del agente automático

Construir un agente que funcione automáticamente con la política aprendida. Para ello:

1. Cargar la tabla Q aprendida en la Fase 2.
2. En esta fase, fase de explotación, los valores de ϵ y α tienen que ser 0.
3. Dado un nuevo estado definir cuál es su estado actual.
4. Determinar qué acción ejecutar utilizando la tabla Q.

Fase 4. Evaluación del agente

1. Jugar diez partidas en el mismo mapa con el agente desarrollado en esta práctica, cambiando las tablas aprendidas en la fase 1 (valores de ϵ, α y número de iteraciones).
2. Evaluar el progreso del agente.

3. Directrices para la documentación

Uno de los miembros del grupo deberá entregar una memoria en formato **PDF** que deberá contener:

- Breve descripción explicando los contenidos del documento.
- Diagrama con los diferentes pasos de la tarea de aprendizaje realizada en la práctica.
- Justificación del conjunto de atributos final elegido y su rango para la definición de los estados. Se debe indicar la evolución histórica de los agentes implementados hasta llegar al agente final, donde cada uno de estos agentes puede tener un conjunto de atributos diferente y destacar las diferencias con el conjunto final.
- Descripción de cualquier tratamiento sobre los datos que se lleve a cabo y de todos los pasos realizados.
- Descripción del código generado para llevar a cabo el aprendizaje de la política de comportamiento.
- Descripción de asignación de estado empleada.
- Descripción del agente final implementado.
- Descripción y análisis de los resultados producidos por el agente final implementado tras la evaluación. En este apartado es importante describir por qué se cree que ha funcionado bien el agente seleccionado (si es éste el caso). En cualquier caso, se deberán describir posibles mejoras que se pueden hacer para aumentar su rendimiento. Además, es importante responder a la siguiente cuestión:

- ¿El agente final desarrollado es capaz de superar en rendimiento a los agentes utilizados en la generación de las tuplas de experiencia? Justificar la respuesta tanto en caso afirmativo como en caso contrario.

- Conclusiones:
  - Conclusiones sobre la tarea a realizar.
  - Apreciaciones más generales sobre las prácticas de la asignatura como: para qué pueden ser útiles los modelos obtenidos y si se os ocurren otros dominios en los que aplicar aprendizaje automático, etc.
  - Descripción de los problemas encontrados a la hora de realizar esta práctica.
  - Comentarios personales. Opinión acerca de la práctica. Dificultades encontradas, críticas, etc.

Se valorará la claridad de la memoria, la exposición de las distintas alternativas y algoritmos utilizados, la justificación de los algoritmos elegidos, la presentación y el análisis de los resultados obtenidos y las conclusiones aportadas.

4. Evaluación

La evaluación de la práctica se realizará sobre 10 puntos. Los criterios que vamos a considerar son los siguientes.

- Fase 1. Implementación del código.
- Fase 2. Proceso de aprendizaje.
- Fase 3. Construcción del agente automático.
- Fase 4. Evaluación del agente.

En la puntuación anterior también se tendrá en cuenta la claridad de la memoria en la descripción de cada uno de los apartados descritos en el apartado 3. El peso de esta práctica sobre la nota final de la asignatura es de 1.5 puntos.

5. Normas de entrega

La práctica se deberá realizar en grupos de dos personas y podrá ser entregada a través del enlace que se publicará en Aula Global hasta las 23:55 horas del día 11 de Mayo de 2016. El nombre del fichero debe contener los 6 últimos dígitos del NIA de los alumnos (ej practica3-387633-209339.zip). El fichero deberá incluir:

- Una memoria en formato **PDF**, que deberá contener al menos los contenidos descritos en la sección 3.
- El código fuente del agente desarrollado en esta práctica, así como las instrucciones necesarias para su compilación y ejecución. El nombre del agente debe contener los 6 últimos dígitos del NIA de los alumnos (e.g., Practica3AgentAA2014_387633_209339.java).

6. Competición de Agentes Automáticos

La competición entre agentes automáticos (sólo basados en aprendizaje) se realizará una vez se hayan entregado todos los agentes de los dos campus. Los agentes entregados competirán entre sí en base a dos principales criterios: fantasmas eliminados y tiempo. Adicionalmente en caso de producirse un empate, se considerarán otros aspectos relacionados con el comportamiento del agente, como por ejemplo cuando éste se encuentre entre tres muros del tablero y sea capaz de salir.

En esta competición se desarrollarán dos versiones de la misma. La primera de ellas consistirá en conseguir que un agente automático sea capaz de completar un mismo mapa, un número determinado de veces. La segunda modalidad de la competición consistirá en conseguir un agente automático capaz de jugar en distintos mapas de complejidad similar.

En la puntuación de la competición será: 1 punto para el mejor agente de los dos campus, 0.75, para el segundo, 0.5 para el tercero y 0.25 para el cuarto. Los resultados de la competición se publicarán en Aula Global.

Normas de entrega para la competición

- El código fuente generado para el agente y todos los ficheros necesarios para poder ejecutarlo en el simulador, así como un README con las instrucciones necesarias para la ejecución del agente. Los ficheros entregados no tienen que afectar al comportamiento del simulador.
- Un documento de una página explicando la técnica de aprendizaje utilizada y los parámetros utilizados.
- El nombre del agente final, deberá tener un nombre distinto a los básicos y preferiblemente distinto a los otros grupos. La "clase" implementada se copiara al fichero *bustersAgents.py* sin tener dependencias a otras clases.
- Se habilitará un entregador especial para la competición.