Memoria
Prática 3

Fundamentos de Aprendizaje Automático

Diego Pareja Serrano

INTRODUCCIÓN

En esta práctica se ha explorado principalmente el campo del aprendizaje por refuerzo, una rama del aprendizaje automático donde un agente aprende a tomar decisiones mediante la interacción con un entorno. A diferencia del aprendizaje supervisado, donde se dan ejemplos explícitos de entrada y salida, el aprendizaje por refuerzo se basa principalemente en las recompensas.
El objetivo principal ha sido implementar y comprender distintos algoritmos que permiten al agente aprender políticas óptimas *que maximizan su recompensa acumulada. Para ello, se han trabajado* entornos como Gridworld o el juego de Pacman, y se han implementado desde iteraciones de valor hasta Q-Learning y su versión aproximada.

DESARROLLO DE LA PRACTICA (Análisis propio)

Preguntas 1 a 5: Planificación con modelo conocido

- En la **P1**, se implementó Iteración de valor, actualizando todos los estados con la ecuación de Bellman. Permitió entender la planificación cuando se conoce el modelo.
- **P2 y P3** consistieron en ajustar parámetros de MDP (descuento, ruido, recompensa viva) para inducir comportamientos como evitar riesgos o preferir salidas lejanas.
- En **P4**, se aplicó Iteración de Valor Asincrónica, más eficiente al actualizar estados uno a uno.
- En **P5**, se exploró Prioritized Sweeping, que actualiza estados con mayor impacto. Su implementación fue más compleja por la gestión de predecesores.

Preguntas 6 a 10: Aprendizaje a partir de la experiencia

- En **P6**, se desarrolló desde cero el algoritmo Q-Learning, aprendiendo valores Q mediante exploración y actualización continua.
- En **P7**, se aplicó una estrategia ε-greedy, clave para el equilibrio entre exploración y explotación.
- En **P8**, se experimentó en un entorno complejo (BridgeGrid) donde el aprendizaje dependía mucho de ajustar bien epsilon y alpha.
- En **P9**, se entrenó un agente en Pacman, logrando superar el 80% de victorias tras 2000 episodios de entrenamiento.
- Finalmente, en **P10**, se implementó Q-Learning Aproximado, usando extractores de características y pesos en lugar de tablas, permitiendo generalizar a espacios grandes.

DIFICULTADES APLICADAS Y SOLUCIONES

1. Iteración de valor asincrónica y estados terminales

Problema: Dentro de la implementación del agente asincrónico (Q4), intentaba que los estados terminales se actualizaran, pero esto me generó errores.
Solución: Inserte un condicional que se dedica a omitir cualquier estado terminal antes de calcular valores (if self.mdp.isTerminal(state): continue). Esto me soluciono el problema y mejoró la eficiencia.

2. Gestión de predecesores y cola de prioridad (Q5)

Problema: La implementación de Prioritized Sweeping se me complico personalmente. En primer momento, la cola se llenaba incorrectamente o se olvidaban algunos de los estados, lo que daba como resultado que el agente no aprendiera correctamente.
Solución: Lo que hice fue implementaron estructuras set para evitar duplicados. Pero lo que me llevo realmente a la solución final fue descubrir que la clave era actualizar solo si diff > theta, y con prioridad negativa.

3. Ajuste de parámetros en Q-Learning

Problema: Cuando comencé a probarlo, el agente aprendía de manera lenta o se estancaba en políticas subóptimas.
Solución: Se realizaron múltiples experimentos actualizando alpha, epsilon y gamma. Se identificó que un epsilon moderado (0.1 a 0.3) era crucial para la exploración, y que la tasa de aprendizaje debía reducirse si el entorno tenía demasiado ruido

4. Exploración insuficiente y tasa de aprendizaje inapropiada

Problema: A pesar de múltiples intentos y combinaciones de parámetros, no se consiguió obtener una puntuación satisfactoria ni alcanzar la política óptima de forma consistente. El autograder penalizaba con una puntuación baja (0/1 o 1/1 parcial), lo que indica que el agente no aprendía la política correcta en todos los casos, la conclusión que saque a pesar de todo es que este ejercicio sirvió para entender que no siempre es posible aprender óptimamente en entornos limitados si no se ajustan los parámetros de forma precisa. Además, muestra que algunos entornos requieren mayor capacidad de generalización o más episodios para obtener buenos resultados, lo cual es una lección importante en el diseño de agentes de RL.

CONCLUSION Y VALORACIÓN PERSONAL

Esta práctica ha sido sin duda la más larga y compleja hasta ahora en la asignatura. A lo largo de los ejercicios se ha pasado de la teoría más básica como por ejemplo la iteración de valor hasta la implementación de un agente inteligente capaz de jugar al Pacman o generalizar en espacios de estado grandes mediante Q-learning aproximado.

Desde mi punto de vista, lo mejor de la práctica no fue solo implementar los algoritmos, sino entender de manera un poco más profunda como funcionaban, poder observar cómo se comportan en entornos reales, y cómo pequeñas decisiones de diseño (como la forma del reward o la política de exploración) pueden cambiar de manera drástica el resultado.

En definitiva, esta práctica me ha hecho sentir que he podido llegar a desarrollar algo más parecido a un agente inteligente realista. También debo destacar que el al ser de ingeniería informática biomédica y no tener demasiado conocimiento sobre código, me ha costado mil veces más que a mis compañeros de informática, lo cual ha sido algo frustrante. En resumen esta práctica me ha sido de ayuda para comprender mejor el funcionamiento de los algoritmos estudiados y ver de primera mano como se comportan en entornos realistas.