Inteligencia Artificial

Taller Aprendizaje por Refuerzo

Abril 2024

1. Marco teórico

El aprendizaje por refuerzo (RL, por su sigla en inglés) [1] es un método de aprendizaje automático basado en psicología del comportamiento. Permite a un agente aprender a realizar nuevas tareas explorando el entorno y observando modificaciones del estado y posibles recompensas. El método está orientado hacia objetivos en los que un agente, ya sea humano o robótico, intenta maximizar la recompensa acumulada a largo plazo mediante interacciones iterativas con el entorno. La Figura 1 muestra el clásico ciclo de interacción entre un agente de RL y el entorno.

Un problema de RL se compone de:

- Política: que define cómo un agente selecciona una acción con el objetivo de maximizar la señal de recompensa obtenida.
- Señal de recompensa: que establece una definición de eventos positivos y/o negativos, es decir, los objetivos a alcanzar por el agente de RL.
- Función de valor: que especifica qué tan buena podría ser la señal de recompensa a lo largo del tiempo desde un estado o un par estado-acción.

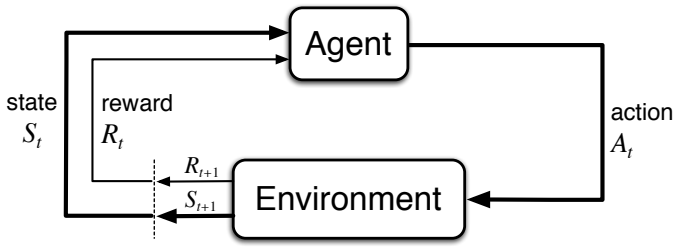

Figura 1: Ciclo de aprendizaje por refuerzo entre un agente y el entorno [1]. En cada iteración, el agente de RL en el estado s_t selecciona una acción a_t para ser ejecutada y recibe desde el entorno un nuevo estado s_{t+1} y una señal de recompensa r_{t+1}.

- Opcionalmente, modelo del entorno: que permite al agente inferir cuál será el próximo estado y recompensa, dado cualquier estado y acción.

Con base en este modelo, un agente de RL puede aprender la política óptima que le permite seleccionar la acción que conduce a la recompensa acumulada más alta dada la función de valor.

Los procesos de decisión de Markov (MDP, por su sigla en inglés) son la base de las tareas de RL. En un MDP, las transiciones y recompensas dependen únicamente del estado actual y de la acción seleccionada por el agente [2]. En otras palabras, un estado de Markov contiene toda la información relacionada con la dinámica de una tarea, es decir, una vez conocido el estado actual, la historia de las transiciones que llevaron al agente a esa posición es irrelevante en términos del problema de toma de decisiones.

Un MDP se caracteriza por la tupla $< S, A, \delta, r >$ donde:

- S es un conjunto finito de estados,
- A es un conjunto de acciones,
- δ es la función de transición $\delta : S \times A \to S$,
- r es la función de recompensa $r : S \times A \to \mathbb{R}$.

En cada iteración t, el agente percibe el estado actual $s_t \in S$ y selecciona la acción $a_t \in A$ para ejecutarla. El entorno devuelve la recompensa $r_t = r(s_t, a_t)$ y el agente transita al estado $s_{t+1} = \delta(s_t, a_t)$. Las funciones r y δ dependen únicamente del estado y la acción actual, es decir, es un proceso sin memoria.

Las acciones se seleccionan de acuerdo con una política π, que en psicología se denomina conjunto de reglas o asociaciones de estímulo-respuesta [3]. Por lo tanto, el valor de realizar una acción a en un estado s bajo una política π se denota $q^\pi(s, a)$, que también se llama función de valor de acción para una política π.

Para encontrar una política, existen diversos métodos de aprendizaje, por ejemplo, SARSA y Q-learning. El método basado en políticas SARSA [4] considera las transiciones de un par estado-acción a un par estado-acción como se muestra en la ecuación (1). En el método Q-learning, los valores de estado-acción se actualizan de acuerdo con la ecuación (2) [5, 6].

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \tag{1}$$

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[r_{t+1} + \gamma \max_{a \in A(s_{t+1})} Q(s_{t+1}, a) - Q(s_t, a_t)] \tag{2}$$

2. Configuración

(a) Usando Python, cree un gridworld de dimensión variable $n \times m$ que permita recompensas positivas y negativas para algunas posiciones.

(b) Cree métodos para devolver el estado actual, la recompensa, el número de estados y el número de acciones.

(c) Cree un método para realizar una acción en el gridworld, es decir, un método que mueve al agente de un estado a otro dada una de las siguientes acciones: derecha, izquierda, arriba, abajo.

(d) Considere solo movimientos válidos, es decir, las acciones que intentan sacar al agente del gridworld no cambian la posición del agente.

(e) Cree un agente de aprendizaje por refuerzo con las siguientes características:

- Utilice una tabla con valores Q inicializados aleatoriamente. Utilice una distribución uniforme entre 0 y $0,01$.
- Parámetros de aprendizaje: tasa de aprendizaje $\alpha = 0{,}7$, factor de descuento $\gamma = 0{,}4$ y método de selección de acción ϵ-greedy con $\epsilon = 0{,}25$.
- Implementar el método de diferencia temporal SARSA para el entrenamiento del agente.

3. Experimentos

(a) Cree un gridworld de 3×4. Consulte la figura 2.

(b) Utilice la cuadrícula $(2, 3)$ como posición objetivo con recompensa 1,0 y la cuadrícula $(1, 1)$ como región aversiva con recompensa $-1{,}0$.

(c) Cree un agente de aprendizaje para navegar por el gridworld creado previamente.

(d) Entrene al agente de aprendizaje por refuerzo utilizando SARSA durante 1000 episodios.

(e) Grafique los valores Q. Observe los pares estado-acción moviendo al agente hacia la posición objetivo como a la posición aversiva.

Referencias

[1] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[2] M. L. Puterman, *Markov decision processes: Discrete stochastic dynamic programming*. John Wiley & Sons, 2014.

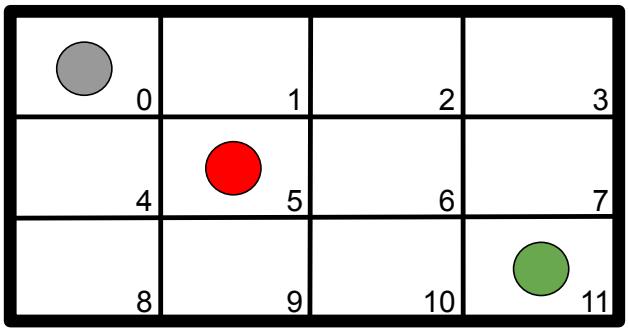

Figura 2: 3×4 gridworld con un estado objetivo y una región aversiva.

[3] S. Kornblum, T. Hasbroucq, and A. Osman, "Dimensional overlap: cognitive basis for stimulus-response compatibility–a model and taxonomy.," *Psychological review*, vol. 97, no. 2, p. 253, 1990.

[4] G. A. Rummery and M. Niranjan, *On-line Q-learning using connectionist systems*, vol. 37. University of Cambridge, Department of Engineering Cambridge, UK, 1994.

[5] C. J. Watkins, *Learning from Delayed Rewards*. Doctoral dissertation, University of Cambridge, 1989.

[6] P. Dayan and C. Watkins, "Q-learning," *Machine learning*, vol. 8, no. 3, pp. 279–292, 1992.